

YoctoHub-GSM-2G

Mode d'emploi

---



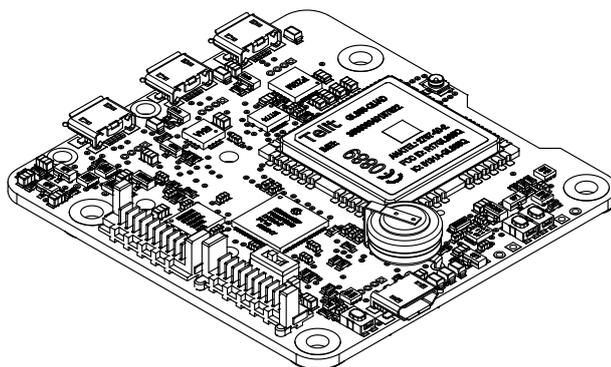
# Table des matières

<b>1. Introduction</b> .....	<b>1</b>
1.1. <i>Accessoires optionnels</i> .....	2
<b>2. Présentation</b> .....	<b>5</b>
2.1. <i>Les éléments du YoctoHub-GSM-2G</i> .....	5
<b>3. Premiers pas</b> .....	<b>9</b>
3.1. <i>Configuration manuelle</i> .....	9
3.2. <i>Fenêtre d'état du hub</i> .....	12
3.3. <i>Configuration automatisée</i> .....	14
3.4. <i>Connexions</i> .....	14
<b>4. Montage</b> .....	<b>17</b>
4.1. <i>Fixation</i> .....	17
4.2. <i>Fixation d'un sous-module</i> .....	18
<b>5. Interactions avec l'extérieur</b> .....	<b>19</b>
5.1. <i>Configuration</i> .....	19
5.2. <i>Emoncms</i> .....	20
5.3. <i>Valarm.net</i> .....	21
5.5. <i>InfluxDB</i> .....	21
5.6. <i>PRTG</i> .....	21
5.7. <i>MQTT</i> .....	21
5.8. <i>Yocto-API callback</i> .....	21
5.9. <i>User defined callback</i> .....	22
<b>6. Programmation</b> .....	<b>25</b>
6.1. <i>Accès aux modules connectés</i> .....	25
6.2. <i>Contrôle du YoctoHub-GSM-2G</i> .....	25
<b>7. Mise en sommeil</b> .....	<b>27</b>
7.1. <i>Configuration manuelle du système de réveil</i> .....	27
7.2. <i>Paramétrage du système de réveil par logiciel</i> .....	28

<b>8. Référence de l'API de haut niveau</b>	<b>31</b>
8.1. La classe <i>YHubPort</i>	32
8.2. La classe <i>YCellular</i>	80
8.3. La classe <i>YNetwork</i>	164
8.4. La classe <i>YFiles</i>	278
8.5. La classe <i>YRealTimeClock</i>	330
8.6. La classe <i>YWakeUpMonitor</i>	379
8.7. La classe <i>YWakeUpSchedule</i>	437
<b>9. Problèmes courants</b>	<b>503</b>
9.1. Par où commencer ?	503
9.2. Linux et USB	503
9.3. Plateformes ARM: HF et EL	504
9.4. Les exemples de programmation n'ont pas l'air de marcher	504
9.5. Module alimenté mais invisible pour l'OS	504
9.6. Another process named xxx is already using yAPI	504
9.7. Déconnexions, comportement erratique	505
9.8. RegisterHub d'un VirtualHub déconnecte le précédent	505
9.9. Commandes ignorées	505
9.10. Impossible de contacter les sous-devices par USB	505
9.11. Network Readiness coincé à	505
9.12. Module endommagé	505
<b>10. Caractéristiques</b>	<b>507</b>

# 1. Introduction

Le YoctoHub-GSM-2G est un module électronique de 60x58mm qui permet de contrôler d'autres modules Yoctopuce à travers une connection cellulaire de type GSM 2G (au standard GPRS EDGE, aussi appelé 2.75G). Le module radio supporte les quatre bandes de fréquences GSM les plus utilisées, à savoir les bandes 900 Mhz et 1800 Mhz utilisées en Europe, Moyen-Orient, Afrique, Asie et Pacifique, et les bandes 850 Mhz et 1900 Mhz utilisés en Amérique du Nord, dans les Caraïbes et l'Amérique Latine<sup>1</sup>



*Le YoctoHub-GSM-2G*

Le YoctoHub-GSM-2G a été conçu pour être déployé facilement et ne pas demander de maintenance particulière. Contrairement à un mini-PC, il n'utilise pas un système d'exploitation complexe. Les réglages peuvent être effectués manuellement ou de manière automatisée, par USB. Il convient de ce fait beaucoup mieux à une industrialisation qu'un mini-PC. En revanche, il ne permet pas l'exécution de programmes supplémentaires écrits par l'utilisateur.

Le YoctoHub-GSM-2G n'est pas un hub USB standard avec accès réseau. Bien qu'utilisant du câblage USB, ses ports descendants utilisent un protocole propriétaire, plus simple qu'USB. Il n'est par conséquent pas possible de contrôler, ni même d'alimenter, des périphériques USB standards avec un YoctoHub-GSM-2G.

Yoctopuce vous remercie d'avoir fait l'acquisition de ce YoctoHub-GSM-2G et espère sincèrement qu'il vous donnera entière satisfaction. Les ingénieurs Yoctopuce se sont donnés beaucoup de mal pour que votre YoctoHub-GSM-2G soit facile à installer n'importe où et soit facile à utiliser en toutes circonstances. Néanmoins, si ce module venait à vous décevoir, n'hésitez pas à contacter le support Yoctopuce<sup>2</sup>.

---

<sup>1</sup> Pour une liste détaillée des bandes de fréquence supportées par pays, consultez la page Wikipedia [http://en.wikipedia.org/wiki/GSM\\_frequency\\_bands](http://en.wikipedia.org/wiki/GSM_frequency_bands) .

<sup>2</sup> [support@yoctopuce.com](mailto:support@yoctopuce.com)

## 1.1. Accessoires optionnels

Il existe un certain nombre d'accessoires qui vous aideront à tirer le meilleur parti de votre YoctoHub-GSM-2G.

### Vis et entretoises

Pour fixer le module YoctoHub-GSM-2G à un support, vous pouvez placer des petites vis de 3mm avec une tête de 8mm au maximum dans les trous prévus ad-hoc. Il est conseillé de les visser dans des sujet entretoises filetés, que vous pourrez fixer sur le support. Vous trouverez plus de détail à ce sujet dans le chapitre concernant le montage et la connectique. Micro-hub USB

### Câble USB MicroB-MicroB

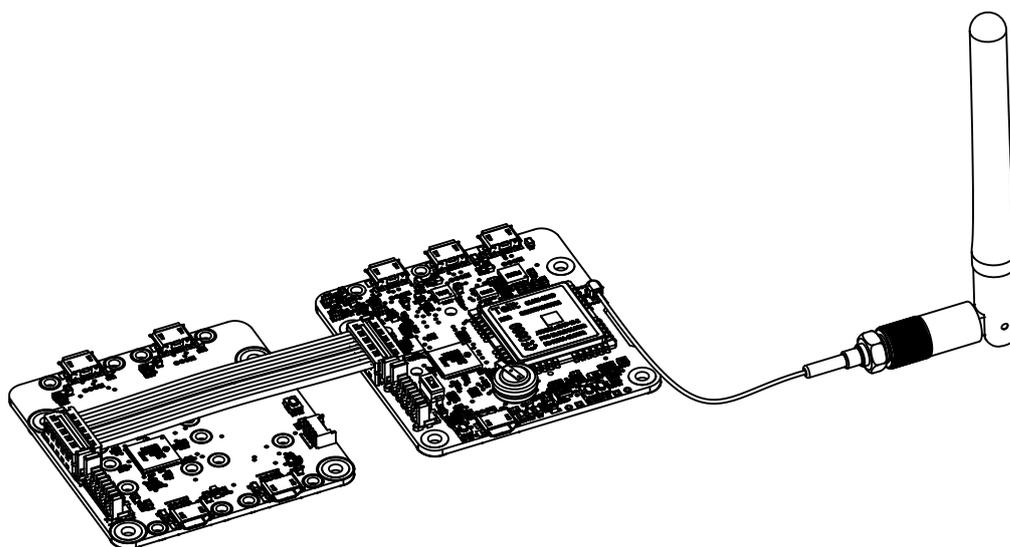
Les ports descendant du YoctoHub-GSM-2G sont au format micro-B, tous les produits Yoctopuce sont équipés de connecteur micro-B. Cela signifie que vous aurez besoin d'un câble se terminant par une prise Micro-B à chaque extrémité. Ces câbles ne sont pas très répandus dans le commerce, mais vous pourrez en trouver sur le magasin en ligne de Yoctopuce<sup>3</sup>.

### Connecteur au pas 1.27mm

L'utilisation de câbles USB pour connecter les sous modules est très pratique, mais cela reste une solution assez volumineuse. Sur le PCB du YoctoHub-GSM-2G, vous trouverez à proximité de chaque connecteur USB une empreinte permettant de souder un connecteur au pas 1.27 ou 1.25mm. Ce qui permet d'interconnecter les modules avec un câblage beaucoup compact. Ce genre de connecteur est très standard, et peut être acheté dans à peu près n'importe que magasin d'électronique. Yoctopuce commercialise une ensemble connecteur + câble de 11cm sous la référence 1.27-1.27-11.

### YoctoHub-Shield

Le YoctoHub-GSM-2G dispose de trois ports permettant de brancher trois sous-modules. Il est possible d'augmenter significativement cette capacité en utilisant des extension nommées *YoctoHub-Shield*. Chacun de ces shields ajoute quatre ports supplémentaires, et il est possible d'en chainer jusqu'à dix. Consulter la documentation du YoctoHub-Shield pour plus de détails.



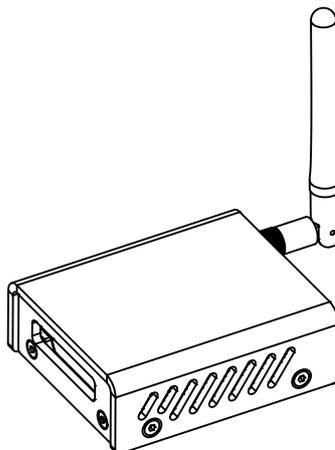
Le YoctoHub-Shield ajoute des ports à votre YoctoHub-GSM-2G.

### Boîtier

Votre YoctoHub-GSM-2G a été conçu pour pouvoir être installé tel quel dans votre projet. Néanmoins Yoctopuce commercialise des boîtiers spécialement conçus pour les modules

<sup>3</sup> USB-OTG-MicroB-MicroB-20 et USB-OTG-MicroB-MicroB-100

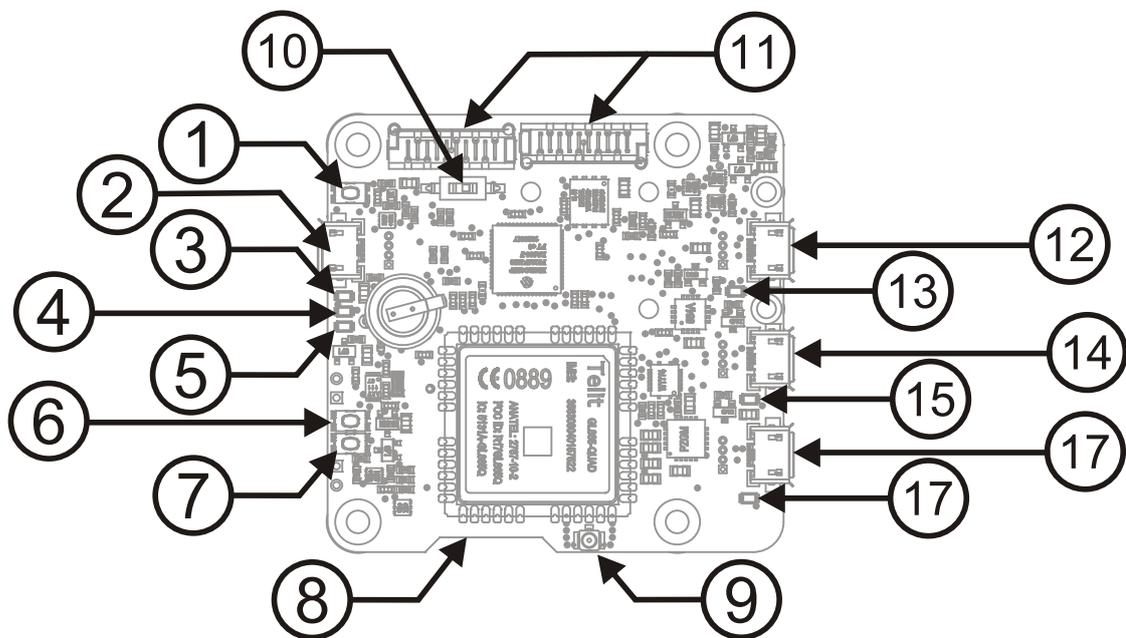
Yoctopuce. Vous trouverez plus d'informations à propos de ces boîtiers sur le site de Yoctopuce. Le boîtier recommandé pour votre YoctoHub-GSM-2G est le modèle YoctoBox-HubWlan-Transp.



*Votre YoctoHub-GSM-2G peut être installé dans un boîtier.*



## 2. Présentation



- |  |  |
|--|--|
| 1: Yocto-bouton                        | 10: Neutralisation de la mise en sommeil |
| 2: Port USB de contrôle + alimentation | 11: Connexion dorsale                    |
| 3: Yocto-Led                           | 12: Port descendant 1                    |
| 4: Indicateur de sur-consommation      | 13: Indicateur port descendant 1         |
| 5: Indicateur de transfert réseau      | 14: Port descendant 2                    |
| 6: Touche réveil                       | 15: Indicateur port descendant 2         |
| 7: Touche mise en sommeil              | 16: Port descendant 3                    |
| 8: Support pour carte SIM (dessous)    | 17: Indicateur port descendant 3         |
| 9: Connecteur d'antenne                |  |

### 2.1. Les éléments du YoctoHub-GSM-2G

#### Le numéro de série

Chaque Yocto-module a un numéro de série unique attribué en usine, pour les modules YoctoHub-GSM-2G ce numéro commence par YHUBGSM1. Le module peut être piloté par logiciel en utilisant ce numéro de série. Ce numéro de série ne peut pas être changé.

### Le nom logique

Le nom logique est similaire au numéro de série, c'est une chaîne de caractères sensée être unique qui permet référencer le module par logiciel. Cependant, contrairement au numéro de série, le nom logique peut être modifié à volonté. L'intérêt est de pouvoir fabriquer plusieurs exemplaires du même projet sans avoir à modifier le logiciel de pilotage. Il suffit de programmer les mêmes noms logiques dans chaque exemplaire. Attention, le comportement d'un projet devient imprévisible s'il contient plusieurs modules avec le même nom logique et que le logiciel de pilotage essaye d'accéder à l'un de ces modules à l'aide de son nom logique. A leur sortie d'usine, les modules n'ont pas de nom logique assigné, c'est à vous de le définir.

### Le Yocto-bouton

Le Yocto-bouton a deux fonctions. Premièrement, il permet d'activer la Yocto-balise (voir la Yocto-Led ci-dessous). Deuxièmement, si vous branchez un Yocto-module en maintenant ce bouton appuyé, il vous sera possible de reprogrammer son firmware avec une nouvelle version. Notez qu'il existe une méthode plus simple pour mettre à jour le firmware depuis l'interface utilisateur, mais cette méthode-là peut fonctionner même lorsque le firmware chargé sur le module est incomplet ou corrompu.

### La Yocto-Led

En temps normal, la Yocto-Led sert à indiquer le bon fonctionnement du module: elle émet alors une faible lumière bleue qui varie lentement mimant ainsi une respiration. La Yocto-Led cesse de respirer lorsque le module ne communique plus, par exemple s'il est alimenté par un hub sans connexion avec un ordinateur allumé.

Lorsque vous appuyez sur le Yocto-bouton, la Led passe en mode Yocto-balise: elle se met alors à flasher plus vite et beaucoup plus fort, dans le but de permettre une localisation facile d'un module lorsqu'on en a plusieurs identiques. Il est en effet possible de déclencher la Yocto-balise par logiciel, tout comme il est possible de détecter par logiciel une Yocto-balise allumée.

La Yocto-Led a une troisième fonctionnalité moins plaisante: lorsque le logiciel interne qui contrôle le module rencontre une erreur fatale, elle se met à flasher SOS en morse<sup>1</sup>. Dans ce cas, débranchez puis re-branchez le module. Si le problème venait à se reproduire, vérifiez que le module contient bien la dernière version du firmware et, dans l'affirmative, contactez le support Yoctopuce<sup>2</sup>.

### Le connecteur de contrôle et d'alimentation (Power / Control port)

Ce connecteur permet d'alimenter le YoctoHub-GSM-2G et les modules qui lui sont connectés à l'aide d'un simple chargeur USB. Ce connecteur permet aussi de prendre le contrôle du YoctoHub-GSM-2G par USB, exactement comme on pourrait le faire avec un module Yoctopuce classique. C'est particulièrement utile lorsque que l'on désire configurer le YoctoHub-GSM-2G sans connaître son adresse IP.

### Les ports descendants

Vous pouvez connecter jusqu'à trois modules Yoctopuce sur ces ports. Ils seront alors accessibles comme s'ils étaient branchés à un ordinateur faisant tourner un VirtualHub. Attention, le protocole entre le YoctoHub-GSM-2G et le module Yoctopuce n'est pas de l'USB mais un protocole propriétaire plus léger. De ce fait le YoctoHub-GSM-2G ne peut pas gérer des périphériques autres que des modules Yoctopuce. Un hub USB standard ne fonctionnera pas non plus<sup>3</sup>. Si vous désirez brancher plus de trois modules Yoctopuce, utilisez le connecteur dorsal pour y connecter un ou plusieurs YoctoHub-Shield<sup>4</sup>.

Attention, les connecteurs USB du YoctoHub-GSM-2G sont simplement soudés en surface et peuvent être arrachés si la prise USB venait à faire fortement levier. Si les pistes sont restées en place, le connecteur peut être ressoudé à l'aide d'un bon fer et de flux. Alternativement, vous pouvez

---

<sup>1</sup> court-court-court long-long-long court-court-court

<sup>2</sup> support@yoctopuce.com

<sup>3</sup> Le Micro-USB-Hub fabriqué par Yoctopuce est un hub USB standard et ne fonctionnera pas avec le YoctoHub-GSM-2G.

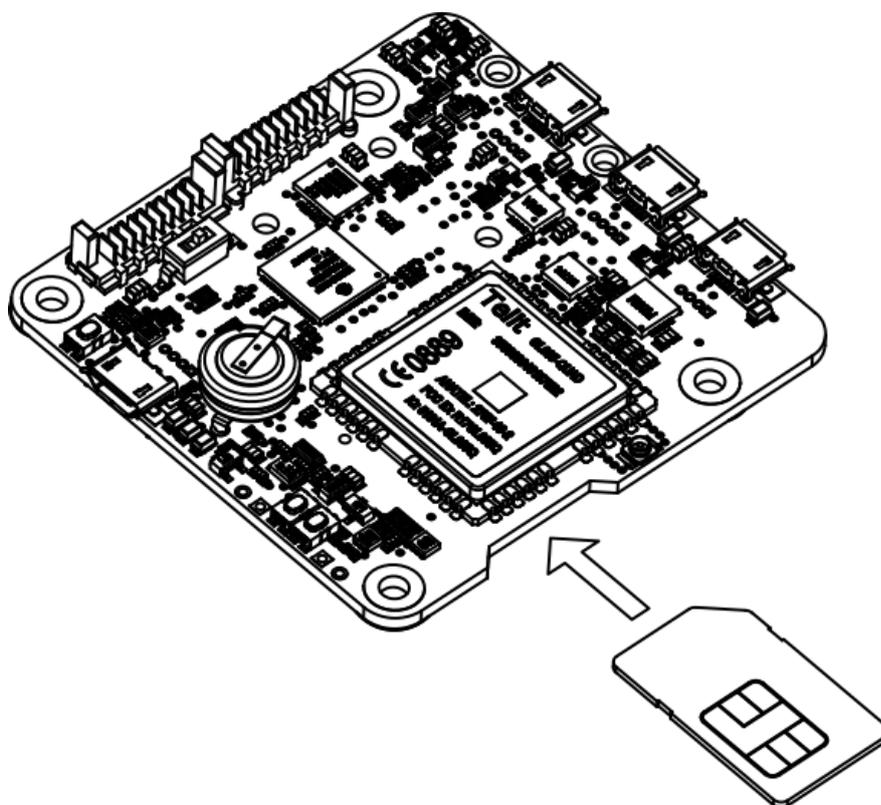
<sup>4</sup> www.yoctopuce.com/FR/products/yoctohub-shield

souder un fil USB directement dans les trous espacés de 1.27mm prévus à cet effet, près du connecteur.

### Le support pour la carte SIM

Pour vous connecter à un réseau cellulaire GSM, vous devrez insérer dans votre YoctoHub-GSM-2G une carte SIM autorisant la connexion au réseau cellulaire, associée à un abonnement permettant le transfert de données. Le support à carte SIM est prévu pour le format mini-SIM le plus standard, aussi appelé 2FF. Il existe des adaptateurs permettant l'utilisation des Micro-SIM ou Nano-SIM, ces adaptateurs peuvent être achetés dans n'importe quel magasin de téléphone portable. Avec le YoctoHub-GSM-2G. Le support SIM est de type *push-push*: pressez pour insérer la SIM jusqu'à ce qu'elle soit en position et produise un petit clic. Re-pressez une deuxième fois éjecter la SIM de son support.

Vous devez insérer la carte SIM avec les contacts métalliques contre le circuit imprimé.



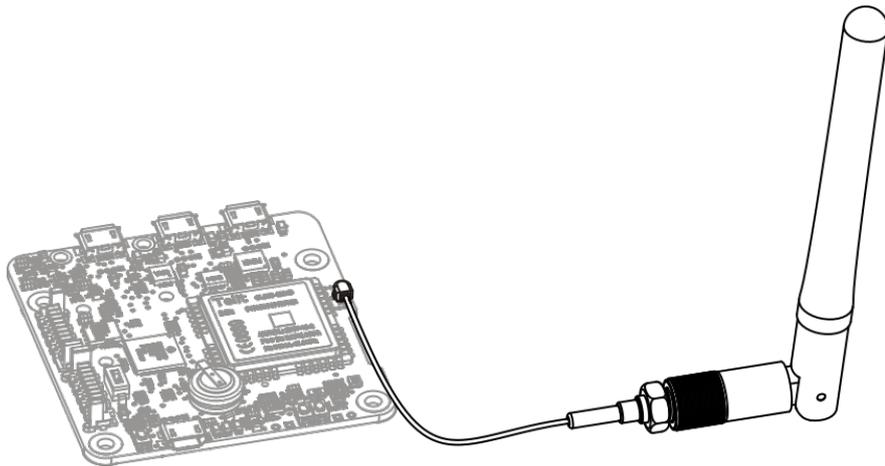
Sens d'insertion de la carte SIM dans le YoctoHub-GSM-2G.

### Le connecteur d'antenne

Le YoctoHub-GSM-2G dispose d'un connecteur d'antenne coaxial ultra miniature (UFL). Prenez grand soin du connecteur UFL, il est fragile et n'est pas conçu pour supporter beaucoup de cycles de connexion/déconnexion. Le YoctoHub-GSM-2G est livré en standard avec un petit câble UFL vers SMA femelle<sup>5</sup> et une antenne correspondante SMA mâle<sup>6</sup>. Vous pouvez utiliser une autre antenne de votre choix, pour autant qu'elle soit conçue pour la gamme de fréquence utilisée dans votre pays pour le GSM et qu'elle ait le bon connecteur. Prenez garde aussi au fait que l'utilisation d'antennes à fort gain peut vous amener à émettre un signal supérieur à la norme autorisée dans votre pays.

<sup>5</sup> filetage extérieur et tube femelle au centre

<sup>6</sup> filetage intérieur et pin mâle au centre



Connexion de l'antenne.

### Indicateur de sur-consommation

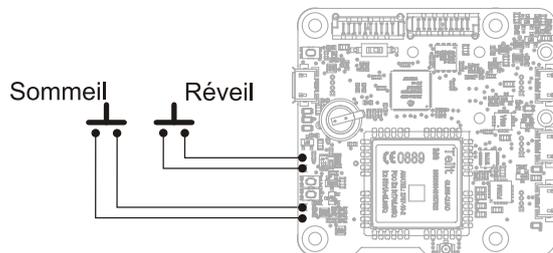
Le YoctoHub-GSM-2G analyse en permanence sa consommation. S'il détecte une consommation globale de plus de 2A suite à une surcharge sur un des ports descendants par exemple, il va automatiquement désactiver tous les ports descendants et allumer l'indicateur de sur-consommation. Pour isoler la source du problème, vous pouvez réactiver les ports un à un, en surveillant l'augmentation de la consommation. Alternativement, si connaissez la source du problème de sur-consommation et savez l'avoir résolu, vous pouvez redémarrer le YoctoHub-GSM-2G pour réactiver tous les ports.

Notez que l'indicateur de sur-consommation est une mesure de protection qui peut éviter la surchauffe, mais ce n'est pas une garantie de protection contre les court-circuits.

### Mise en sommeil

En moyenne, le YoctoHub-GSM-2G consomme environ 0,5 Watt (100mA), auquel il faut ajouter la consommation des modules qui lui sont connectés. Mais il est capable de se mettre en sommeil pour réduire sa consommation d'énergie au strict minimum, et de se réveiller à une heure précise ou lorsqu'un contact extérieur est fermé. Cette fonctionnalité est très utile pour construire des installations de mesure fonctionnant sur batterie. Lorsque que le YoctoHub-GSM-2G est en sommeil, la quasi totalité de l'électronique du module ainsi que les modules Yoctopuce connectés sont hors tension, ce qui réduit sa consommation totale à 75  $\mu$ W (15  $\mu$ A).

La mise en sommeil et le réveil peuvent être soit programmés sur base horaire, soit contrôlés par logiciel, soit contrôlés manuellement à l'aide de deux boutons poussoirs présents sur le circuit du YoctoHub-GSM-2G. Vous y trouverez aussi deux paires de contacts qui permettent de dériver ces deux boutons.



Dérivation des boutons de mise en sommeil et de réveil.

Le YoctoHub-GSM-2G dispose d'un interrupteur qui permet de désactiver au niveau hardware la fonctionnalité de mise en sommeil. Cette fonctionnalité est utile en particulier durant les phases de développement/déverminage de votre projet, ainsi que pour effectuer les mises à jour du firmware.

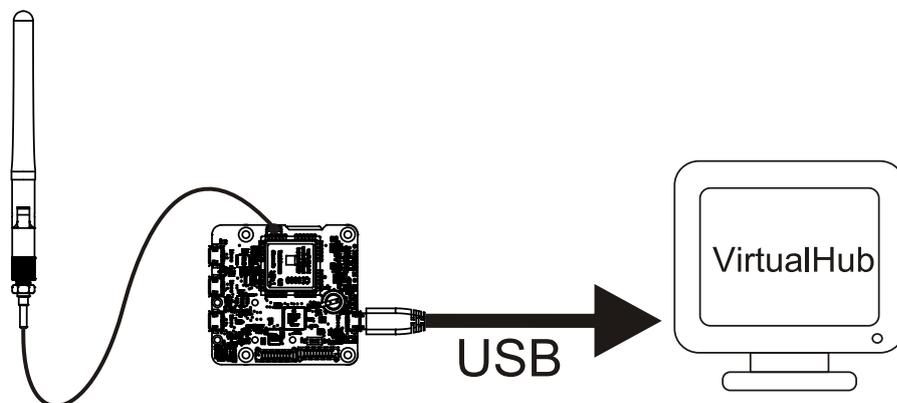
## 3. Premiers pas

Ce chapitre a pour but de vous aider à connecter et configurer votre YoctoHub-GSM-2G pour la première fois

### 3.1. Configuration manuelle

Vous pouvez configurer votre YoctoHub-GSM-2G via son port de contrôle USB, en utilisant le *VirtualHub*<sup>1</sup>.

Lancez un *VirtualHub* sur votre ordinateur favori et raccordez votre ordinateur au port *power / control port* du YoctoHub-GSM-2G. Vous aurez besoin d'un câble USB A-MicroB.



Configuration: raccordez par USB votre YoctoHub-GSM-2G à un ordinateur

Lancez alors votre browser favori sur l'url de votre *VirtualHub*. Il s'agit généralement `http://127.0.0.1:4444`. Vous obtiendrez la liste des modules Yoctopuce connectés par USB, dont votre YoctoHub-GSM-2G

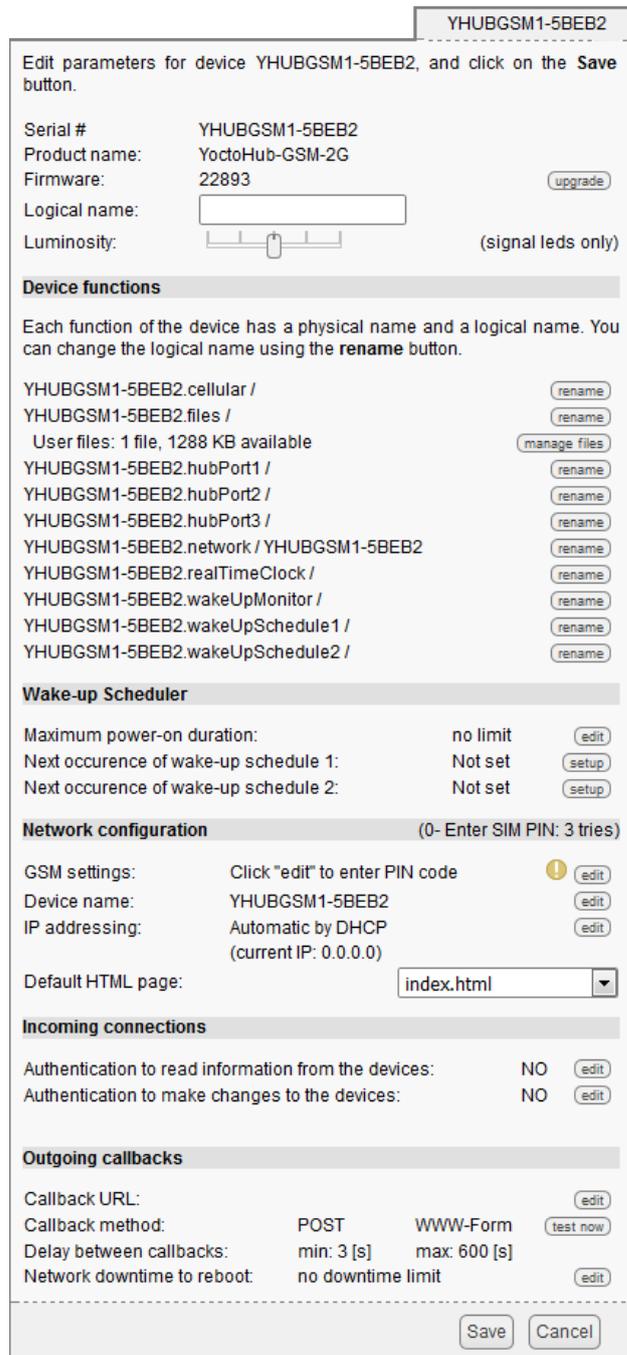
Serial	Logical Name	Description	Action
VIRTHUB0-1521ca755		VirtualHub	<input type="button" value="configure"/> <input type="button" value="view log file"/>
YHUBGSM1-5BEB2		YoctoHub-GSM-2G	<input type="button" value="configure"/> <input type="button" value="view log file"/> <input type="button" value="beacon"/>

Search:

Liste des modules Yoctopuce raccordés par USB à votre ordinateur, dont votre YoctoHub-GSM-2G

<sup>1</sup> <http://www.yoctopuce.com/FR/virtualhub.php>

Cliquez sur le bouton **configure** correspondant à votre YoctoHub-GSM-2G, vous obtiendrez la fenêtre de configuration du module. Cette fenêtre comporte une section **Network configuration**.



Fenêtre de configuration du module YoctoHub-GSM-2G

### Connexion au réseau cellulaire GSM

La première chose à faire consiste à configurer votre YoctoHub-GSM-2G pour qu'il se connecte à votre réseau GSM. Pour cela cliquez sur le bouton **edit** correspondant à **GSM configuration** dans la section **Network configuration**, et la fenêtre de configuration du réseau cellulaire GSM apparaît:

GSM configuration

Specify the desired GSM configuration then click on the **Ok** button.

**SIM PIN code**

Enter current PIN code:

**Cellular Network**

Automatic operator selection  
 Manual operator selection  
 Stay disconnected

**IP connectivity**

Disable IP connectivity (keep only SMS)  
 Enable IP connectivity on home network (no roaming)  
 Enable IP connectivity even when roaming

Use custom APN configuration  
 APN host:  
 Credentials:

**Diagnostics**

Current operator:	auto-select (not connected)
Network readiness:	0- Enter SIM PIN: 3 tries
Link quality:	0%
Current IP:	

Fenêtre de configuration du réseau cellulaire GSM.

Vous pouvez alors entrer le code PIN correspondant à la carte SIM introduite dans le YoctoHub-GSM-2G si nécessaire, et choisir avec quel opérateur cellulaire vous désirez travailler.

Dans la plupart des cas, la carte SIM "sait" pour quel opérateur elle est prévue, et il est possible de garder simplement la sélection automatique. Toutefois, en zone frontalière, il se peut que la carte soit tentée d'utiliser un émetteur étranger plus puissant, mais aussi plus cher à l'utilisation. Pour éviter cela, vous pouvez effectuer un choix d'opérateur manuel de votre réseau domestique.

Vous pouvez aussi spécifier dans votre YoctoHub-GSM-2G le contexte quel cas vous désirez activer la connexion IP (transfert de données). Vous pouvez soit la désactiver complètement si vous n'êtes intéressé qu'à l'utilisation de SMS<sup>2</sup>, soit l'activer sur le réseau propre de la carte SIM, soit autoriser l'utilisation des données sur les réseaux tiers (roaming). Prenez garde si vous activez cette dernière option, car elle peut engendrer rapidement des coûts importants !

Selon votre SIM et votre opérateur cellulaire, il est possible que vous deviez configurer un APN (*Access Point Name*), qui correspond à la passerelle vers internet sur votre réseau cellulaire. Il s'agit d'un nom arbitraire, décidé par votre opérateur, et auquel s'ajoute parfois un nom d'utilisateur et un mot de passe. Il est impossible de lister dans ce mode d'emploi le nom de l'APN à utiliser avec chaque opérateur, mais si vous faites une recherche sur internet avec le nom de votre opérateur cellulaire et le mot "APN", vous trouverez très facilement le nom de l'APN à utiliser ainsi que le nom d'utilisateur et mot de passe éventuel. Le site [apnchanger.org](http://apnchanger.org) contient ces informations pour les principaux opérateurs de nombreux pays.

Attention, vous n'avez que trois essais pour configurer le bon code PIN. Passé ces trois essais vous devrez réinitialiser la SIM à l'aide de son code PUK.

Après avoir entré les paramètres de réseau sans fil et éventuellement les avoir testés, vous pouvez cliquer sur le bouton **Ok** pour fermer cette fenêtre de configuration et retourner à la fenêtre de configuration générale.

<sup>2</sup> La fonctionnalité SMS n'est pas encore disponible

## Différence entre un réseau GSM et un réseau usuel

**Important:** Le réseau cellulaire GSM auquel est connecté votre YoctoHub-GSM-2G n'est pas strictement équivalent à une connexion internet usuelle. En effet, l'adresse IP qui attribuée aux modems cellulaires en quasiment toujours une adresse IP *non routée*. Le YoctoHub-GSM-2G voit tout le réseau Internet, mais Internet ne dispose pas d'adresse publique pour le contacter. Ceci signifie que vous ne pourrez pas vous connecter à distance sur votre YoctoHub-GSM-2G depuis n'importe quel ordinateur, juste en tapant son adresse IP dans un navigateur internet.

Une des solution qui permet de palier à ce problème consiste à obtenir de votre opérateur cellulaire une carte SIM permettant une connexion directe à travers un réseau privé virtuel.

## Utilisation sur un réseau privé virtuel

Certains opérateurs cellulaires peuvent fournir sous condition une connexion GSM avec un lien internet sur une plage d'adresse privée, qui vous est dédiée. Ce type de connexion, dédié aux services *machine-to-machine*, est généralement réservé aux entreprises. Il permet de se connecter à distance (via un réseau privé virtuel) sur votre YoctoHub-GSM-2G, ce qui n'est autrement pas possible car tout téléphone cellulaire est normalement implicitement isolé derrière un filtre NAT.

Dans le cas ou vous disposez d'une telle connexion, vous pouvez configurer quelle adresse IP doit être attribuée au YoctoHub-GSM-2G, et quelle adresse IP est autorisée à le contacter (fonction *firewall*). Pour ce faire, cliquez sur le bouton **edit** en face de la ligne **IP addressing**. Il est indispensable de configurer correctement ces paramètres pour pouvoir contacter votre module par son adresse IP, car sinon le firewall bloquera toute connexion entrante.

## 3.2. Fenêtre d'état du hub

Il est possible de connaitre l'état général du hub c'est à dire don nom logique, l'état du réseau, l'état des ports etc. Pour cela retournez simplement à la liste des modules.

Cliquez sur le numéro de série correspondant à votre YoctoHub-GSM-2G. Cela ouvrira la fenêtre détails de votre module:

YHUBGSM1-5BEB2



YHUBGSM1-5BEB2 is a 58x60mm Quad-band 2G GSM host for Yoctopuce modules, including a timer-based power saving function.

**Kernel**

Serial #	YHUBGSM1-5BEB2
Product name:	YoctoHub-GSM-2G
Logical name:	
Firmware:	22893
Consumption:	35 mA
Beacon:	Inactive <span style="float: right;"><input type="button" value="turn on"/></span>
Luminosity:	50%

**Hub Ports**

Port 1:	ON	<input type="button" value="turn off"/>
Port 2:	ON	<input type="button" value="turn off"/>
Port 3:	ON	<input type="button" value="turn off"/>

**Network**

Operator:	auto-select (not connected)
Readiness:	0- Enter SIM PIN: 3 tries <span style="float: right;">!</span>
IP address:	0.0.0.0 <span style="float: right;"><input type="button" value="ping test"/></span>
Device name:	YHUBGSM1-5BEB2

**Power saving timer**

RTC time:	2016/01/29 10:11:55	<input type="button" value="Set now"/>
Next wake up:	N/A	
Power saving:	sleep not configured	<input type="button" value="Sleep now"/>
WakeUp schedule 1, next occurrence:	not configured	
WakeUp schedule 2, next occurrence:	not configured	

**Misc**

Open API browser (pop-up)  
Get user manual from [yoctopuce.com](http://yoctopuce.com)

*Les propriétés du YoctoHub-GSM-2G*

Cette fenêtre comporte une section qui relate l'état de la partie réseau du YoctoHub-GSM-2G. Vous y trouverez son adresse MAC, adresse IP courante et nom de réseau. Cette section donne aussi l'état de la connexion réseau. Ces états peuvent être:

- 0- Le module ne trouve pas de réseau GSM (2G). Dans ce cas, vérifiez que:
  - vous avez introduit une carte SIM
  - vous avez configuré le code PIN de la carte SIM dans le YoctoHub-GSM-2G
  - vous n'avez pas désactivé la radio (mode avion)
  - vous avez bien connecté une antenne GSM
  - vous êtes à un endroit où l'on peut recevoir du réseau 2G
- 1- network exists: un réseau cellulaire GSM a été détecté, mais le module n'y est pas encore accepté. Si cet état persiste, vérifiez que votre SIM est valable et que l'opérateur cellulaire choisi correspond.
- 2- network linked: le YoctoHub-GSM-2G a pu se connecter au réseau GSM, mais n'a pas encore établi de connexion IP. Si cet état persiste, vérifiez vos réglages APN.
- 3- LAN ready: le réseau local est opérationnel (connexion IP avec l'opérateur de téléphonie)
- 4- WWW ready: le module a pu vérifier la connectivité à Internet en se connectant à un serveur de temps (NTP).

Si votre YoctoHub-GSM-2G passe bien à l'état *WWW Ready*, cela signifie que votre connexion internet cellulaire fonctionne correctement. Vous pouvez alors passer aux étapes suivantes: connecter des modules Yoctopuce désirés (capteurs et/ou actuateurs), et configurer les interactions avec l'extérieur.

### 3.3. Configuration automatisée

Il est possible d'industrialiser la configuration réseau du YoctoHub-GSM-2G. Vous trouverez dans les chapitres suivants de cette documentation la description des fonctions de programmation permettant de relire l'adresse Ethernet d'un module (adresse MAC), et de configurer tous ses paramètres réseau.

Les fonctions de configuration réseau sont aussi accessibles par ligne de commande, en utilisant l'utilitaire `YNetwork` disponible dans la librairie de programmation en ligne de commande<sup>3</sup>.

Après avoir effectué un changement de réglage par programmation, prenez garde à appeler la fonction `saveToFlash()` pour vous assurez que les réglages soient sauvés de manière permanente dans la mémoire flash du module.

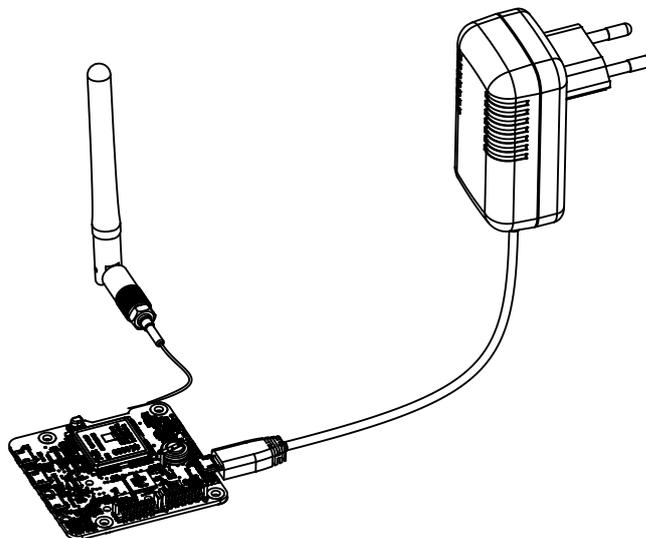
### 3.4. Connexions

#### Alimentation

Le YoctoHub-GSM-2G doit être alimenté par la prise USB de contrôle.

#### USB

Branchez simplement un chargeur USB dans le port *power / control port*, assurez-vous tout de même que le chargeur soit d'une puissance électrique suffisante: le YoctoHub-GSM-2G consomme environ 100mA, auxquels il faudra ajouter la consommation de chaque sous-module. Le YoctoHub-GSM-2G est conçu pour gérer 2A au maximum, c'est pourquoi un chargeur USB capable de délivrer au moins 2A est recommandé. Par ailleurs, vous devrez veiller à ce que la consommation totale de l'ensemble hub + sous-modules ne dépasse pas cette limite.

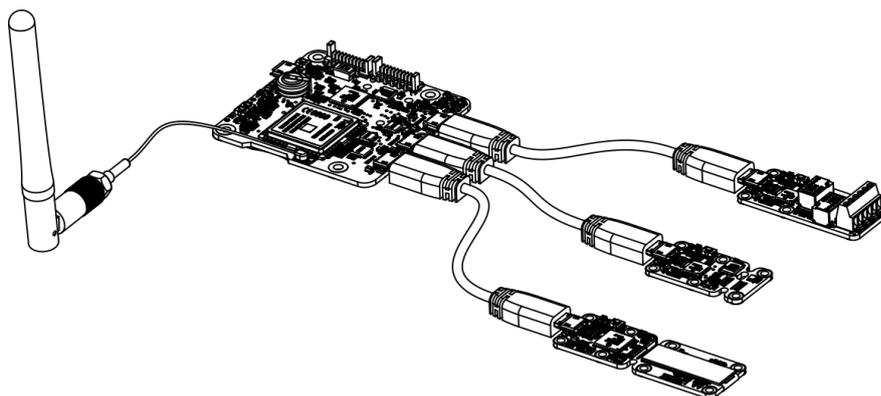


*Le YoctoHub-GSM-2G peut être alimenté par un chargeur USB*

#### Sous-modules

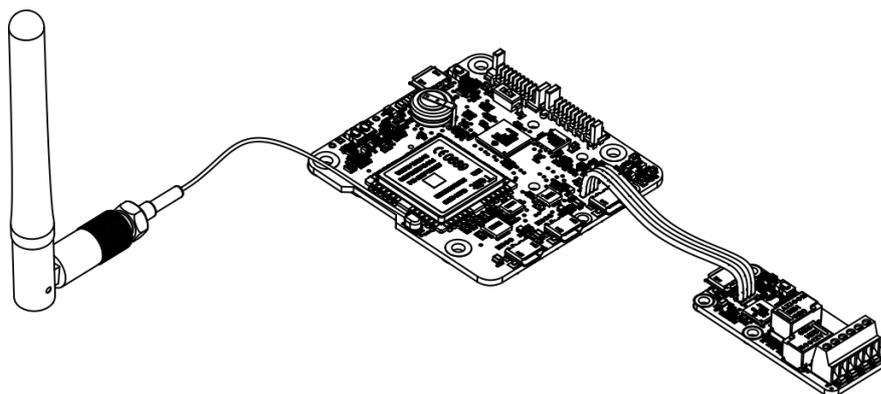
Le YoctoHub-GSM-2G est capable de piloter tous les modules Yoctopuce de la gamme *Yocto*. Ces modules peuvent être connectés directement aux ports descendants, ils seront détectés automatiquement. Vous aurez besoin pour cela de câbles USB MicroB-MicroB. Vous pouvez utiliser des câbles OTG ou non, cela n'a pas d'importance.

<sup>3</sup> <http://www.yoctopuce.com/FR/libraries.php>



*Connexion des sous-modules à l'aide de câbles USB*

Alternativement, vous pouvez connecter vos modules de manière plus compacte à l'aide de câbles au pas 1.27mm: tous les modules Yoctopuce disposent en effet de contacts à cet effet. Vous pouvez soit souder des connecteurs 1.27mm sur les modules et utiliser des câbles avec connecteurs 1.27mm, soit souder directement du câble plat au pas 1.27mm. Si vous choisissez cette dernière option, il est recommandé d'utiliser du câble plat mono-brin, moins souple que le multi-brin mais beaucoup plus facile à souder. Soyez particulièrement attentif aux polarités: Le YoctoHub-GSM-2G, tout comme l'ensemble de modules de la gamme Yoctopuce, n'est pas protégé contre les inversions de polarité. Une telle inversion a toutes les chances de détruire vos équipements. Assurez-vous que la position du contact carré de part et d'autre du câble correspondent.

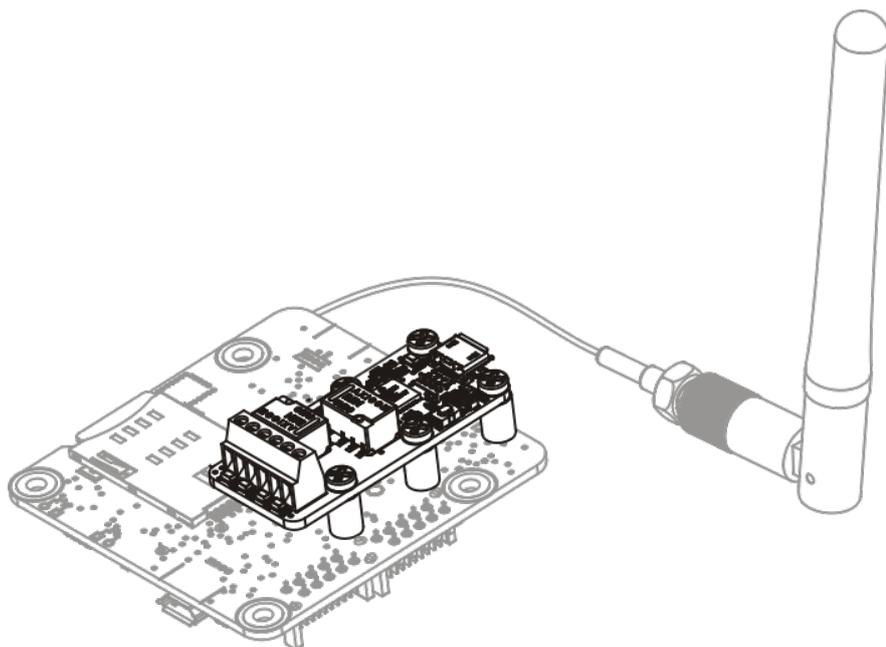


*Connexion des sous-modules à l'aide de câble nappe*

Le YoctoHub-GSM-2G est conçu pour que vous puissiez fixer un module simple largeur directement dessus. Vous aurez besoin de vis, d'entretoises<sup>4</sup> et d'un connecteur au pas 1.27mm<sup>5</sup>. Vous pouvez ainsi transformer un module Yoctopuce USB en en module réseau tout en gardant un format très compact.

<sup>4</sup> <http://www.yoctopuce.com/FR/products/accessoires-et-connectique/fix-2-5mm>

<sup>5</sup> <http://www.yoctopuce.com/FR/products/accessoires-et-connectique/board2board-127>



*Fixation d'un module directement sur le hub*

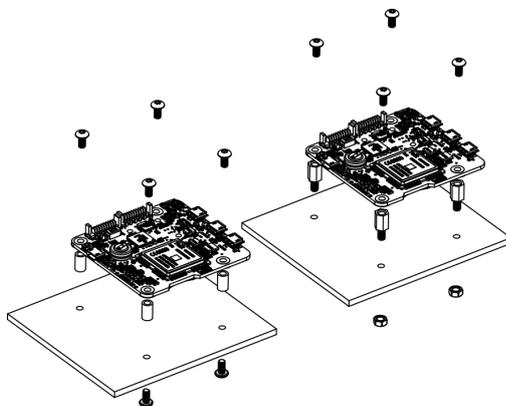
Attention, le YoctoHub-GSM-2G est conçu pour piloter des modules Yoctopuce uniquement. En effet le protocole utilisé entre le YoctoHub-GSM-2G et les sous-modules n'est pas de l'USB mais un protocole propriétaire, beaucoup plus léger. Si d'aventure vous branchez un périphérique autre qu'un module Yoctopuce sur un des ports descendants du YoctoHub-GSM-2G, le port en question sera automatiquement désactivé pour éviter d'endommager le périphérique.

## 4. Montage

Ce chapitre fournit des explications importantes pour utiliser votre module YoctoHub-GSM-2G en situation réelle. Prenez soin de le lire avant d'aller trop loin dans votre projet si vous voulez éviter les mauvaises surprises.

### 4.1. Fixation

Pendant la mise au point de votre projet, vous pouvez vous contenter de laisser le hub se promener au bout de son câble. Veillez simplement à ce qu'il ne soit pas en contact avec quoi que soit de conducteur (comme vos outils). Une fois votre projet pratiquement terminé, il faudra penser à faire en sorte que vos modules ne puissent pas se promener à l'intérieur.



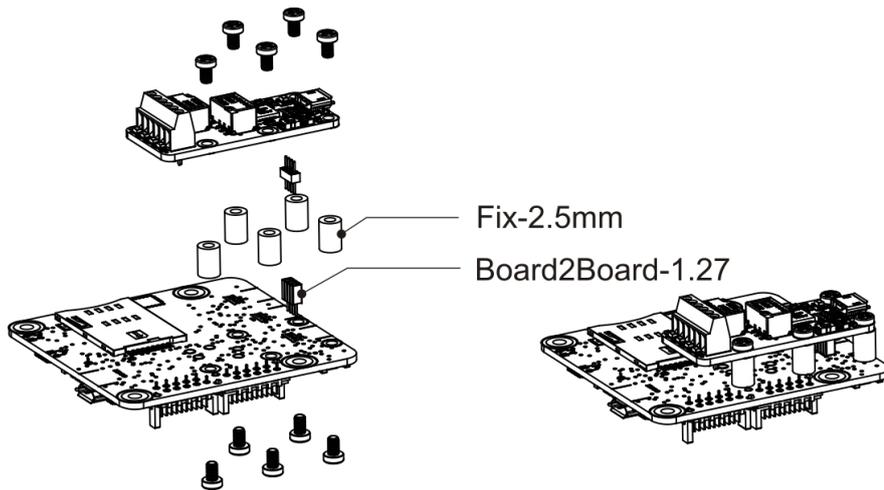
*Exemples de montage sur un support.*

Le module YoctoHub-GSM-2G dispose de trous de montage 3mm. Vous pouvez utiliser ces trous pour y passer des vis. Le diamètre de la tête de ces vis ne devra pas dépasser 8mm, sous peine d'endommager les circuits du module.

Veillez à que l'électronique module ne soit pas en contact avec le support. La méthode recommandée consiste à utiliser des entretoises. Vous pouvez monter le module dans le sens qui vous convient: mais vous devez conscient du fait que les composants électroniques du YoctoHub-GSM-2G, la partie réseau en particulier, dégagent de la chaleur. Vous devrez donc faire en sorte que la chaleur ne puisse pas s'accumuler.

## 4.2. Fixation d'un sous-module

Le YoctoHub-GSM-2G est conçu pour que vous puissiez visser un module simple largeur directement dessus. Par simple largeur, on entend les modules de 20 mm de large. Tous les modules simple largeur ont leurs 5 trous de fixation et le connecteur USB au même endroit. Le sous-module peut être fixé à l'aide de vis et d'entretoises. Il y a derrière les connecteurs USB du YoctoHub-GSM-2G et du sous-module un ensemble de 4 contacts qui permettent d'effectuer la connexion électrique entre le hub et le sous-module. Si vous ne vous sentez pas suffisamment à l'aise avec un fer à souder, vous pouvez aussi utiliser un simple câble USB MicroB-MicroB, OTG ou non.



*Fixation d'un module directement sur le hub*

Prenez garde à bien monter le module sur la face prévue, comme illustré ci-dessus. Les 5 trous du module doivent correspondre aux 5 trous du YoctoHub-GSM-2G, et le contact carré sur le module doit être connecté au contact carré sur le port descendant du YoctoHub-GSM-2G. Si vous montez un module sur l'autre face ou d'une autre manière, la polarité du connecteur sera inversée et vous risquez fort d'endommager définitivement votre matériel.

Tous les accessoires nécessaires à la fixation d'un module sur votre YoctoHub-GSM-2G sont relativement courants. Vous pourrez les trouver sur le site de Yoctopuce tout comme sur la plupart des sites vendant du matériel électronique. Attention cependant, la tête des vis servant à fixer le sous-module devra avoir un diamètre maximum de 4.5 millimètres, sous peine d'endommager les composants électroniques.

## 5. Interactions avec l'extérieur

Le YoctoHub-GSM-2G est capable de poster sur le site web de votre choix l'état des modules qu'il voit. Les valeurs sont postées à intervalles réguliers et à chaque fois qu'une valeur change de manière significative. Cette fonctionnalité, appelée *Callback HTTP*, vous permettra d'interfacer vos modules Yoctopuce avec divers services web.

### 5.1. Configuration

Pour utiliser cette fonctionnalité, cliquez simplement sur le bouton **Configure** de la ligne correspondant au YoctoHub-GSM-2G dans l'interface, puis cliquez sur le bouton **edit** de la section **Outgoing callback**.

Serial	Logical Name	Description	Action
VIRTHUB0-1521ca755		VirtualHub	<a href="#">configure</a> <a href="#">view log file</a>
YHUBGSM1-5BEB2		YoctoHub-GSM-2G	<a href="#">configure</a> <a href="#">view log file</a> <a href="#">beacon</a>

[Show device functions](#)

*Cliquez sur le bouton "Configure" de la première ligne*

YHUBGSM1-5BEB2

Edit parameters for device YHUBGSM1-5BEB2, and click on the **Save** button.

Serial #: YHUBGSM1-5BEB2  
 Product name: YoctoHub-GSM-2G  
 Firmware: 22893   
 Logical name:   
 Luminosity:  (signal leds only)

**Device functions**

Each function of the device has a physical name and a logical name. You can change the logical name using the **rename** button.

YHUBGSM1-5BEB2.cellular /   
 YHUBGSM1-5BEB2.files /   
 User files: 1 file, 1288 KB available   
 YHUBGSM1-5BEB2.hubPort1 /   
 YHUBGSM1-5BEB2.hubPort2 /   
 YHUBGSM1-5BEB2.hubPort3 /   
 YHUBGSM1-5BEB2.network / YHUBGSM1-5BEB2   
 YHUBGSM1-5BEB2.realTimeClock /   
 YHUBGSM1-5BEB2.wakeUpMonitor /   
 YHUBGSM1-5BEB2.wakeUpSchedule1 /   
 YHUBGSM1-5BEB2.wakeUpSchedule2 /

**Wake-up Scheduler**

Maximum power-on duration: no limit   
 Next occurrence of wake-up schedule 1: Not set   
 Next occurrence of wake-up schedule 2: Not set

**Network configuration** (0- Enter SIM PIN: 3 tries)

GSM settings: Click "edit" to enter PIN code   
 Device name: YHUBGSM1-5BEB2   
 IP addressing: Automatic by DHCP   
 (current IP: 0.0.0.0)  
 Default HTML page:

**Incoming connections**

Authentication to read information from the devices: NO   
 Authentication to make changes to the devices: NO

**Outgoing callbacks**

Callback URL:    
 Callback method: POST WWW-Form   
 Delay between callbacks: min: 3 [s] max: 600 [s]  
 Network downtime to reboot: no downtime limit

*Puis éditez la section **Outgoing callbacks**.*

La fenêtre de configuration des callbacks HTTP apparaît. Cette fenêtre va vous permettre de définir comment votre YoctoHub-GSM-2G va interagir avec un serveur Web externe. Vous avez plusieurs type d'interactions a votre disposition. Pour chaque type, un wizard vous guidera pour fournir les paramètres adéquats.

## 5.2. Emoncms

Emoncms.org est un service de cloud open-source gratuit où vous pouvez vous inscrire pour poster les données de vos capteurs, et qui vous permettra de les visualiser en temps réel par Internet et de tracer des graphiques d'historique, et ce sans écrire la moindre ligne de code. Il suffit d'indiquer dans le fenêtre de configuration votre clé d'API fournie par Emoncms, ainsi que le numéro de noeud (arbitraire) que vous désirez attribuer à votre YoctoHub-GSM-2G.

Il est aussi possible d'installer Emoncms sur votre propre serveur, afin de garder le contrôle sur vos données. Vous trouverez des explications plus détaillées sur le blog de Yoctopuce<sup>1</sup>.

Yoctopuce n'est en aucune manière affilié à Emoncms.org.

<sup>1</sup> <http://www.yoctopuce.com/FR/article/utiliser-emoncms-sur-un-serveur-prive>

### 5.3. Valarm.net

Valarm est un service de cloud professionnel où vous pouvez vous inscrire pour poster les données de vos capteurs, et qui offre des fonctions avancées notamment pour la configuration à distance des modules Yoctopuce et la géolocalisation des mesures.

Valarm est revendeur des produits Yoctopuce, mais Yoctopuce n'est pas autrement affilié à Valarm.

### 5.4. Xively (anciennement Cosm)

Xively est un service de cloud payant où vous pourriez poster les données de vos capteurs. Notez que depuis la fin 2015, Xively concentre son activité sur les grosses entreprises et les clients OEM. De ce fait, le service Xively n'est plus forcément disponible pour n'importe qui. Pour plus de détails, consultez [xively.com](http://xively.com).

Yoctopuce n'est en aucune manière affilié à Xively.

### 5.5. InfluxDB

InfluxDB est une base de données dédiée spécifiquement à stocker des séries temporelles de mesures et d'événements. Elle est très efficace pour retrouver des séries de mesures pour une plage de temps donnée, y compris en les rassemblant pour en faire un résumé. Vous pouvez facilement l'installer sur votre propre ordinateur pour enregistrer vos données, et dessiner des graphiques. Vous trouverez un guide pas-à-pas pour configurer InfluxDB et Grafana avec les modules Yoctopuce sur le blog de Yoctopuce.<sup>2</sup>

Yoctopuce n'est en aucune manière affilié à InfluxData ni à Grafana.

### 5.6. PRTG

PRTG est une solution commerciale développée par PAESSLER, destinée à la supervision des systèmes et des applications. Vous pouvez facilement l'installer sur Windows pour enregistrer les mesures et obtenir des graphiques de vos capteurs. Pour plus de détails, voir [fr.paessler.com/prtg](http://fr.paessler.com/prtg). Vous trouverez un guide pas-à-pas pour configurer PRTG avec les modules Yoctopuce sur le blog de Yoctopuce<sup>3</sup>.

Yoctopuce n'est en aucune manière affilié à PAESSLER.

### 5.7. MQTT

MQTT est un protocole de l'Internet des Objets permettant à des capteurs de publier en temps réel des valeurs vers un serveur central, appelé broker MQTT. Pour plus de détails, voir [mqtt.org](http://mqtt.org). Vous trouverez plusieurs articles sur le blog de Yoctopuce décrivant l'utilisation de MQTT.

### 5.8. Yocto-API callback

Avec certains langages de programmation, l'API Yoctopuce est capable de fonctionner en mode *callback HTTP*. Dans ce mode un script sur le serveur web peut prendre le contrôle de vos modules à travers un filtre NAT sans que vous ayez à ouvrir un port. Typiquement cela permet de contrôler depuis un site Web public des modules Yoctopuce installés derrière un router DSL privé. Le YoctoHub-GSM-2G sert alors de passerelle. Vous avez simplement à définir l'URL du script de contrôle sur le serveur HTTP et éventuellement les identifiants nécessaires pour y accéder. Dans le script serveur, vous devez alors initialiser la librairie Yoctopuce avec l'appel suivant:

---

<sup>2</sup> <http://www.yoctopuce.com/FR/article/utiliser-les-capteurs-yoctopuce-avec-influxdb-et-grafana>

<sup>3</sup> <http://www.yoctopuce.com/FR/article/nouveaute-le-support-prtg-dans-les-yoctohub>

```
RegisterHub("http://callback");
```

Il existe deux possibilités pour utiliser l'API Yoctopuce en mode callback. La première, disponible en PHP, Java et Node.JS est basée sur des connections en pur HTTP. Le YoctoHub-GSM-2G poste l'intégralité de son état au serveur, et reçoit en retour des commandes du script sur le serveur. Ce fonctionnement implique quelques limitations: les interactions complexes, comme la récupération de données depuis l'enregistreur de données sur les capteurs, ne sont pas possible.

Le deuxième mode fonctionnement de l'API par callback est l'utilisation des callbacks WebSocket. Ce mode est disponible uniquement avec les serveurs Java et Node.JS. Les WebSockets sont une extension standard de HTTP, qui permet l'établissement d'une liaison bidirectionnelle complète sur une liaison HTTP. Lorsqu'un script serveur est connecté à un YoctoHub-GSM-2G au travers d'un callback WebSocket, l'API Yoctopuce peut être utilisée sans la moindre limitation.

L'application web **GatewayHub**, disponible sur le site de Yoctopuce, utilise cette technologie de callback WebSocket pour permettre un accès à distance au YoctoHub-GSM-2G, même à travers un filtre NAT ou un pare-feu. Pour plus d'informations, référez-vous au blog de Yoctopuce <sup>4</sup>.

## 5.9. User defined callback

Les "*User defined callback*" vous permettent de personnaliser la manière dont votre YoctoHub-GSM-2G va interagir avec un site Web externe. Vous avez besoin de définir l'URL du serveur Web sur lequel le YoctoHub-GSM-2G va poster l'état de ses devices. Notez que seul le protocole HTTP est supporté (pas de HTTPS).

La fenêtre de configurations des callbacks

Si vous désirez protéger votre script de callback, vous pouvez configurer un contrôle d'accès HTTP standard sur le serveur Web. Le YoctoHub-GSM-2G sait comment gérer les méthodes standard d'identification de HTTP: indiquez simplement le nom d'utilisateur et le mot de passe nécessaires pour accéder à la page. Il est possible d'utiliser la méthode "Basic" aussi bien que la méthode "Digest", mais il est recommandé d'utiliser la méthode "Digest", car elle est basée sur un protocole de question-réponse qui évite la transmission du mot de passe sur le réseau et évite aussi les copies d'autorisation.

Le YoctoHub-GSM-2G poste avec la méthode POST les valeurs notifiées<sup>5</sup> des modules à intervalle régulier, et à chaque fois qu'une de ces valeurs change de manière significative. Vous pouvez changer les délais d'attente entre les posts.

<sup>4</sup> <http://www.yoctopuce.com/FR/article/une-passerelle-pour-acceder-aux-yoctohubs-a-distance>

<sup>5</sup> Les valeurs notifiées sont celles que vous voyez quand vous cliquez sur *show functions* dans l'interface principale du YoctoHub-GSM-2G .

## Tests

Afin de vous permettre de déboguer le processus, le YoctoHub-GSM-2G vous permet de visualiser la réponse au callback envoyé par le serveur Web. Cliquez simplement sur le bouton **test** une fois que vous avez renseigné tous les champs. Si le résultat vous paraît satisfaisant, fermez la fenêtre de debug, et cliquez sur **Ok**.

## Formats

Les valeurs sont postées sous une des formes suivantes:

1. Si un nom logique a été défini pour une fonction:

```
NOM_LOGIQUE_DE_LA_FONCTION = VALEUR
```

2. Si un nom logique a été défini pour le module, mais pas pour la fonction:

```
NOM_DU_MODULE#NOM_HARDWARE = VALUE
```

3. Si aucun nom logique n'a été attribué:

```
NUMERO_DE_SERIE#NOM_HARDWARE = VALEUR
```

Voici un script PHP qui vous permettra de visualiser le contenu des données postées par le callback, suivi du résultat dans la fenêtre de debug.

```
<?php
Print(Date('H:i:s')." \r\n");
foreach ($_POST as $key=>$value) {
    Print("$key=$value\r\n");
}
?>
```



*Le résultat du test de callback avec un Yocto-PowerRelay et un Yocto-Temperature.*



## 6. Programmation

### 6.1. Accès aux modules connectés

Le YoctoHub-GSM-2G se comporte exactement comme un ordinateur faisant tourner un *VirtualHub*. La seule différence entre un programme utilisant l'API Yoctopuce utilisant des modules en USB natif et ce même programme utilisant des modules Yoctopuce connecté à un YoctoHub-GSM-2G se situe au niveau de l'appel à *registerHub*. Pour utiliser des modules USB connectés en natif, le paramètre de *RegisterHub* est *usb*, pour utiliser des modules connectés à un YoctoHub-GSM-2G, il suffit de remplacer ce paramètre par l'adresse IP du YoctoHub-GSM-2G. Par exemple, en Delphi:

```
YRegisterHub("usb",errmsg);
```

devient

```
YRegisterHub("192.168.0.10",errmsg); // l'adresse IP du hub est 192.168.0.10
```

### 6.2. Contrôle du YoctoHub-GSM-2G

Du point de vue API de programmation, le YoctoHub-GSM-2G est un module comme les autres. Il est parfaitement contrôlable depuis l'API Yoctopuce. Pour ce faire, vous aurez besoin des classes suivantes.

#### Module

Cette classe, commune à tous les modules Yoctopuce permet de contrôler le module en temps que tel. Elle vous permettra de contrôler la Yocto-Led, de connaître la consommation sur USB du YoctoHub-GSM-2G, etc.

#### Cellular

Cette classe permet de contrôler la configuration du réseau cellulaire du YoctoHub-GSM-2G, en particulier le nom de l'opérateur cellulaire, le code PIN pour utiliser la carte SIM et les paramètres de l'APN.

#### Network

Cette classe permet de contrôler la partie réseau du YoctoHub-GSM-2G, vous pourrez contrôler l'état du link, lire l'adresse MAC, changer l'adresse IP du YoctoHub-GSM-2G, connaître la consommation sur PoE, etc.

## HubPort

Cette classe permet de contrôler la partie hub. Vous pourrez activer ou désactiver les ports du YoctoHub-GSM-2G, vous pourrez aussi savoir quel module est connecté à quel port.

## Files

Cette classe permet d'accéder aux fichiers stockés dans la mémoire flash du YoctoHub-GSM-2G. Le YoctoHub-GSM-2G dispose en effet d'un petit système de fichiers qui vous permet de stocker par exemple une Web App contrôlant les modules connectés au YoctoHub-GSM-2G.

## WakeUpMonitor

Cette classe permet de contrôler la mise en sommeil du YoctoHub-GSM-2G.

## WakeUpSchedule

Cette classe permet d'agender un ou plusieurs réveils du YoctoHub-GSM-2G.

Vous trouverez quelques exemples de contrôle du YoctoHub-GSM-2G par programmation dans les bibliothèques Yoctopuce, disponibles gratuitement sur le site de Yoctopuce.

## 7. Mise en sommeil

Le YoctoHub-GSM-2G dispose d'une horloge en temps réel (RTC) alimentée par un super condensateur, qui se recharge automatiquement lorsque le module est sous tension mais permet de maintenir l'heure sans aucune alimentation pendant plusieurs jours. Ce RTC est utilisé pour piloter un système de mise en sommeil afin d'économiser l'énergie. Le système de mise en sommeil peut être configuré manuellement à l'aide d'une interface, ou piloté par logiciel.

### 7.1. Configuration manuelle du système de réveil

Les conditions de réveil peuvent être configurées manuellement en vous connectant sur l'interface du YoctoHub-GSM-2G. Dans la section **Wake-up scheduler** de la fenêtre de configuration générale, cliquez sur le bouton setup correspondant à l'un des "wake-up-schedule". Une fenêtre qui permet d'agender des réveils plus ou moins réguliers s'ouvre. Il suffit de sélectionner les cases correspondant aux occurrences désirées. Les sections laissées vides seront ignorées.

WakeUp schedule 1

Define wake up times: each button toggles a condition. A wake-up will occur when at least one condition per section is true. Sections without any condition defined are ignored.

**Days in the week**

Mon Tue Wed Thu Fri Sat Sun

**Days in the month**

1 2 3 4 5 6 7 8 9 10 11 12  
13 14 15 16 17 18 19 20 21 22 23 24  
25 26 27 28 29 30 31

set all every 2 every 3 clear

**Months**

Jan Feb Mar Avr May Jun Jul Aug Sep Oct Nov Dec

set all every 2 every 3 clear

**Hours**

0 1 2 3 4 5 6 7 8 9 10 11  
12 13 14 15 16 17 18 19 20 21 22 23

set all every 2 every 3 every 4 every 6 clear

**Minutes**

0 1 2 3 4 5 6 7 8 9 10 11 12 13 14  
15 16 17 18 19 20 21 22 23 24 25 26 27 28 29  
30 31 32 33 34 35 36 37 38 39 40 41 42 43 44  
45 46 47 48 49 50 51 52 53 54 55 56 57 58 59

set all every 2 every 3 every 5 every 10 clear

Ok Close

Fenêtre de configuration des réveils, ici toutes les 10 minutes entre 9h et 17h

De même, vous pouvez configurer directement sur l'interface du YoctoHub-GSM-2G le temps d'éveil maximal désiré, après lequel le module retournera automatiquement en sommeil profond. Si vous utilisez votre YoctoHub-GSM-2G sur batteries, ceci vous assurera de ne pas vider les batteries même si aucun ordre de mise en sommeil explicite n'est reçu.

## 7.2. Paramétrage du système de réveil par logiciel

Au niveau de l'interface de programmation, le système de réveil est implémenté à l'aide de deux types de fonction : La fonction *wakeUpMonitor* et la fonction *WakeUpSchedule*.

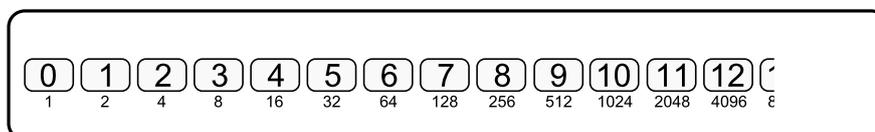
### wakeUpMonitor

La fonction *wakeUpMonitor* gère l'éveil et la mise en sommeil proprement dits. Elle met à disposition toutes les fonctionnalités de contrôle immédiate: éveil immédiat, mise en sommeil immédiate, calcul de la date du prochain réveil etc... Le fonction *wakeUpMonitor* permet aussi de définir le temps maximum pendant lequel le YoctoHub-GSM-2G peut rester éveil avant de se mettre automatiquement en sommeil.

### wakeUpSchedule

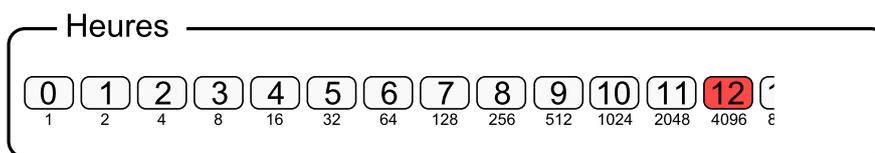
La fonction *wakeUpSchedule* permet de programmer une condition de réveil. Elle dispose de cinq variables qui permet de définir des correspondance sur les minutes, heure, jour de la semaine, jour dans le mois, et mois. Ces variables sont des variables entières dont chaque bit défini une correspondance. Schématiquement, chaque ensemble de minutes, jours, heures est représenté sous la forme d'un ensemble de case avec chacune un coefficient qui est une puissance de deux, exactement comme dans l'interface correspondante du YoctoHub-GSM-2G.

Par exemple le bit 0 des heures correspond à l'heure zéro, le bit 1 correspond à l'heure une, le bit 2 correspond à l'heure 2 etc.



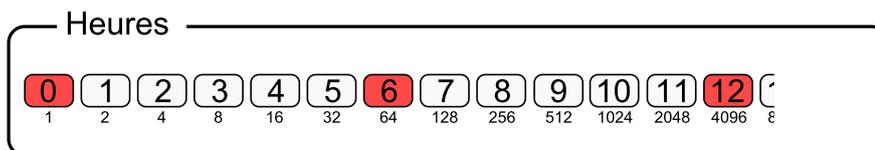
Chaque case se voit affecter une puissance de deux

Ainsi pour programmer le YoctoHub-GSM-2G pour qu'il se réveille tout les jours a midi, il mettre le bit 12 à 1, ce qui correspond à la valeur  $2^{12} = 4096$ .



Exemple de réveil à 12 H

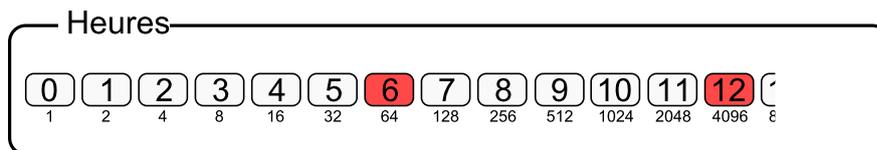
Pour que le module se reveille à 0 heure, 6 heures et 12 heures, il faut mettre les bit 0,6,et 12 à un, ce qui correspondant à la valeur  $2^0 + 2^6 + 2^{12} = 1 + 64 + 4096 = 4161$



$$1 + 64 + 4096 = 4151$$

Exemple de réveil à 0, 6 et 12 H

Les variables peuvent être combinées, pour qu'un réveil ait lieu tous les jours à 6H05, 6h10, 12h05 et 12h10 il suffit de mettre les heures à  $2^6 + 2^{12} = 4060$  et les minutes à  $2^5 + 2^{10} = 1056$ . Les variables laissée à zéro sont ignorées.

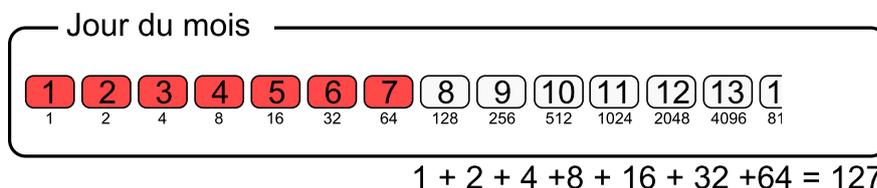
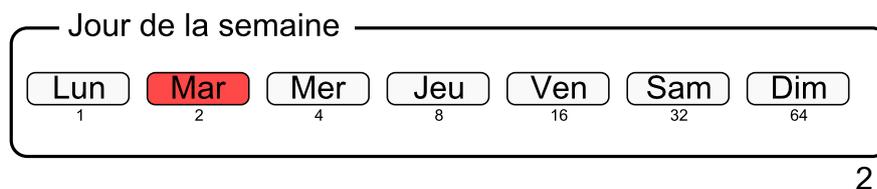


$$64 + 4096 = 4060$$

Exemple de réveil à 6H05, 6h10, 12h05 et 12h10

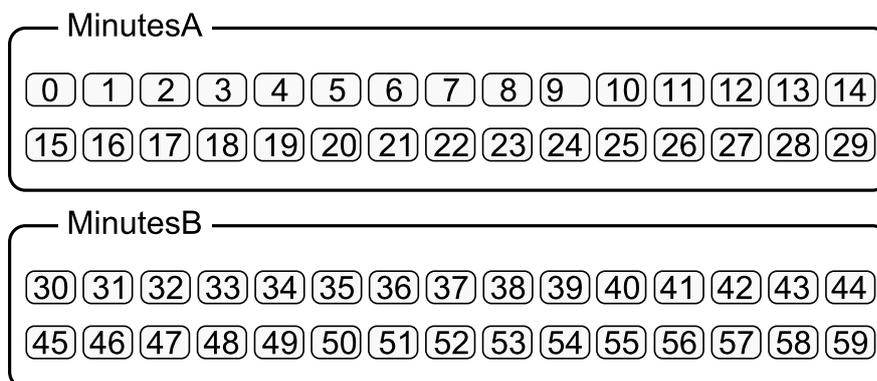
Notez que si vous désirez programmer un réveil à 6H05 et 12h10, mais pas 6h10 et 12h10, vous aurez besoin d'utiliser deux fonctions *wakeUpSchedule* différentes.

Ce paradigme permet de programmer des réveils assez complexe. Ainsi pour programmer un réveil tous les premiers mardis du mois, faut mettre à un le deuxième bit des jours de la semaine et les sept premiers bit des jours du mois.



Exemple de réveil tous les premiers mardi du mois

Certains langages de programmation, dont Javascript et Python, ne supportent pas les entiers 64 bits, ce qui pose un problème pour encoder les minutes. C'est pourquoi les minutes sont à la fois accessibles via un entier 64 bits *minutes* et deux entiers 32 bits, *minutesA* et *minutesB*, qui eux sont disponibles dans tous les langages actuels.



Les minutes sont aussi disponibles sous forme de deux entiers 32 bits.

La fonction *wakeUpSchedule* dispose d'une variable supplémentaire qui permet de définir le temps, en secondes, durant lequel le module restera éveillé après un réveil. Si cette variable est mise à zéro, le module restera éveillé.

Le YoctoHub-GSM-2G dispose de deux fonctions *wakeUpSchedule* ce qui permet de programmer jusqu'à deux types de réveils indépendants.



## 8. Référence de l'API de haut niveau

Ce chapitre résume les fonctions de l'API de haut niveau pour commander votre YoctoHub-GSM-2G. La syntaxe et les types précis peuvent varier d'un langage à l'autre mais, sauf avis contraire toutes sont disponibles dans chaque langage. Pour une information plus précise sur les types des arguments et des valeurs de retour dans un langage donné, veuillez vous référer au fichier de définition pour ce langage (`yocto_api.*` ainsi que les autres fichiers `yocto_*` définissant les interfaces des fonctions).

Dans les langages qui supportent les exceptions, toutes ces fonctions vont par défaut générer des exceptions en cas d'erreur plutôt que de retourner la valeur d'erreur documentée pour chaque fonction, afin de faciliter le débogage. Il est toutefois possible de désactiver l'utilisation d'exceptions à l'aide de la fonction `yDisableExceptions()`, si l'on préfère travailler avec des valeurs de retour d'erreur.

Ce chapitre ne reprend pas en détail les concepts de programmation des modules Yoctopuce. Vous trouverez des explications plus détaillées dans la documentation des modules que vous souhaitez raccorder à votre YoctoHub-GSM-2G.

## 8.1. La classe YHubPort

Interface pour interagir avec les ports de YoctoHub, disponibles par exemple dans le YoctoHub-Ethernet, le YoctoHub-GSM-4G, le YoctoHub-Shield et le YoctoHub-Wireless-n

La classe `YHubPort` permet de contrôler l'alimentation des ports descendants d'un YoctoHub. Il permet de détecter si un module `y` est raccordé et lequel. Un `YHubPort` reçoit toujours automatiquement comme nom logique le numéro de série unique du module Yoctopuce qui y est connecté.

Pour utiliser les fonctions décrites ici, vous devez inclure:

es	in HTML: <code>&lt;script src='../lib/yocto_hubport.js'&gt;&lt;/script&gt;</code> in node.js: <code>require('yoctolib-es2017/yocto_hubport.js');</code>
js	<code>&lt;script type='text/javascript' src='yocto_hubport.js'&gt;&lt;/script&gt;</code>
cpp	<code>#include "yocto_hubport.h"</code>
m	<code>#import "yocto_hubport.h"</code>
pas	<code>uses yocto_hubport;</code>
vb	<code>yocto_hubport.vb</code>
cs	<code>yocto_hubport.cs</code>
java	<code>import com.yoctopuce.YoctoAPI.YHubPort;</code>
uwp	<code>import com.yoctopuce.YoctoAPI.YHubPort;</code>
py	<code>from yocto_hubport import *</code>
php	<code>require_once('yocto_hubport.php');</code>
ts	in HTML: <code>import { YHubPort } from '../dist/esm/yocto_hubport.js';</code> in Node.js: <code>import { YHubPort } from 'yoctolib-cjs/yocto_hubport.js';</code>
dnp	<code>import YoctoProxyAPI.YHubPortProxy</code>
cp	<code>#include "yocto_hubport_proxy.h"</code>
vi	<code>YHubPort.vi</code>
ml	<code>import YoctoProxyAPI.YHubPortProxy</code>

### Fonction globales

#### **YHubPort.FindHubPort(func)**

Permet de retrouver un port de YoctoHub d'après un identifiant donné.

#### **YHubPort.FindHubPortInContext(yctx, func)**

Permet de retrouver un port de YoctoHub d'après un identifiant donné dans un Context YAPI.

#### **YHubPort.FirstHubPort()**

Commence l'énumération des ports de YoctoHub accessibles par la librairie.

#### **YHubPort.FirstHubPortInContext(yctx)**

Commence l'énumération des ports de YoctoHub accessibles par la librairie.

#### **YHubPort.GetSimilarFunctions()**

Enumère toutes les fonctions de type `HubPort` disponibles sur les modules actuellement joignables par la librairie, et retourne leurs identifiants matériels uniques (`hardwareId`).

### Propriétés des objets YHubPortProxy

#### **hubport→AdvertisedValue** [lecture seule]

Courte chaîne de caractères représentant l'état courant de la fonction.

#### **hubport→Enabled** [modifiable]

Vrai si le port du YoctoHub est alimenté, faux sinon.

#### **hubport→FriendlyName** [lecture seule]

Identifiant global de la fonction au format `NOM_MODULE . NOM_FONCTION`.

**hubport**→**FunctionId** [*lecture seule*]

Identifiant matériel du port de YoctoHub, sans référence au module.

**hubport**→**HardwareId** [*lecture seule*]

Identifiant matériel unique de la fonction au format `SERIAL . FUNCTIONID`.

**hubport**→**IsOnline** [*lecture seule*]

Vérifie si le module hébergeant la fonction est joignable, sans déclencher d'erreur.

**hubport**→**LogicalName** [*modifiable*]

Nom logique de la fonction.

**hubport**→**PortState** [*lecture seule*]

état actuel du port de YoctoHub.

**hubport**→**SerialNumber** [*lecture seule*]

Numéro de série du module, préprogrammé en usine.

### Méthodes des objets `YHubPort`

**hubport**→**clearCache()**

Invalide le cache.

**hubport**→**describe()**

Retourne un court texte décrivant de manière non-ambigüe l'instance du port de YoctoHub au format `TYPE ( NAME ) = SERIAL . FUNCTIONID`.

**hubport**→**get\_advertisedValue()**

Retourne la valeur courante du port de YoctoHub (pas plus de 6 caractères).

**hubport**→**get\_baudRate()**

Retourne la vitesse de transfert utilisée par le port de YoctoHub, en kbps.

**hubport**→**get\_enabled()**

Retourne vrai si le port de YoctoHub est alimenté, faux sinon.

**hubport**→**get\_errorMessage()**

Retourne le message correspondant à la dernière erreur survenue lors de l'utilisation du port de YoctoHub.

**hubport**→**get\_errorType()**

Retourne le code d'erreur correspondant à la dernière erreur survenue lors de l'utilisation du port de YoctoHub.

**hubport**→**get\_friendlyName()**

Retourne un identifiant global du port de YoctoHub au format `NOM_MODULE . NOM_FONCTION`.

**hubport**→**get\_functionDescriptor()**

Retourne un identifiant unique de type `YFUN_DESCR` correspondant à la fonction.

**hubport**→**get\_functionId()**

Retourne l'identifiant matériel du port de YoctoHub, sans référence au module.

**hubport**→**get\_hardwareId()**

Retourne l'identifiant matériel unique du port de YoctoHub au format `SERIAL . FUNCTIONID`.

**hubport**→**get\_logicalName()**

Retourne le nom logique du port de YoctoHub.

**hubport**→**get\_module()**

Retourne l'objet `YModule` correspondant au module Yoctopuce qui héberge la fonction.

**hubport**→**get\_module\_async(callback, context)**

Retourne l'objet `YModule` correspondant au module Yoctopuce qui héberge la fonction.

**hubport**→**get\_portState()**

	Retourne l'état actuel du port de YoctoHub.
<b>hubport</b> → <b>get_serialNumber()</b>	Retourne le numéro de série du module, préprogrammé en usine.
<b>hubport</b> → <b>get_userData()</b>	Retourne le contenu de l'attribut <code>userData</code> , précédemment stocké à l'aide de la méthode <code>set_userData</code> .
<b>hubport</b> → <b>isOnline()</b>	Vérifie si le module hébergeant le port de YoctoHub est joignable, sans déclencher d'erreur.
<b>hubport</b> → <b>isOnline_async(callback, context)</b>	Vérifie si le module hébergeant le port de YoctoHub est joignable, sans déclencher d'erreur.
<b>hubport</b> → <b>isReadOnly()</b>	Test si la fonction est en lecture seule.
<b>hubport</b> → <b>load(msValidity)</b>	Met en cache les valeurs courantes du port de YoctoHub, avec une durée de validité spécifiée.
<b>hubport</b> → <b>loadAttribute(attrName)</b>	Retourne la valeur actuelle d'un attribut spécifique de la fonction, sous forme de texte, le plus rapidement possible mais sans passer par le cache.
<b>hubport</b> → <b>load_async(msValidity, callback, context)</b>	Met en cache les valeurs courantes du port de YoctoHub, avec une durée de validité spécifiée.
<b>hubport</b> → <b>muteValueCallbacks()</b>	Désactive l'envoi de chaque changement de la valeur publiée au hub parent.
<b>hubport</b> → <b>nextHubPort()</b>	Continue l'énumération des ports de YoctoHub commencée à l'aide de <code>yFirstHubPort()</code> . Attention, vous ne pouvez faire aucune supposition sur l'ordre dans lequel les ports de YoctoHub sont retournés.
<b>hubport</b> → <b>registerValueCallback(callback)</b>	Enregistre la fonction de callback qui est appelée à chaque changement de la valeur publiée.
<b>hubport</b> → <b>set_enabled(newval)</b>	Modifie le mode d'activation du port du YoctoHub.
<b>hubport</b> → <b>set_logicalName(newval)</b>	Modifie le nom logique du port de YoctoHub.
<b>hubport</b> → <b>set_userData(data)</b>	Enregistre un contexte libre dans l'attribut <code>userData</code> de la fonction, afin de le retrouver plus tard à l'aide de la méthode <code>get_userData</code> .
<b>hubport</b> → <b>unmuteValueCallbacks()</b>	Réactive l'envoi de chaque changement de la valeur publiée au hub parent.
<b>hubport</b> → <b>wait_async(callback, context)</b>	Attend que toutes les commandes asynchrones en cours d'exécution sur le module soient terminées, et appelle le callback passé en paramètre.

## YHubPort.FindHubPort() YHubPort.FindHubPort()

YHubPort

Permet de retrouver un port de YoctoHub d'après un identifiant donné.

js	function <b>yFindHubPort</b> ( <b>func</b> )
cpp	YHubPort* <b>FindHubPort</b> ( string <b>func</b> )
m	+(YHubPort*) <b>FindHubPort</b> : (NSString*) <b>func</b>
pas	TYHubPort <b>yFindHubPort</b> ( <b>func</b> : string): TYHubPort
vb	function <b>FindHubPort</b> ( ByVal <b>func</b> As String) As YHubPort
cs	static YHubPort <b>FindHubPort</b> ( string <b>func</b> )
java	static YHubPort <b>FindHubPort</b> ( String <b>func</b> )
uwp	static YHubPort <b>FindHubPort</b> ( string <b>func</b> )
py	<b>FindHubPort</b> ( <b>func</b> )
php	function <b>FindHubPort</b> ( <b>\$func</b> )
ts	static <b>FindHubPort</b> ( <b>func</b> : string): YHubPort
es	static <b>FindHubPort</b> ( <b>func</b> )
dnp	static YHubPortProxy <b>FindHubPort</b> ( string <b>func</b> )
cp	static YHubPortProxy * <b>FindHubPort</b> ( string <b>func</b> )

L'identifiant peut être spécifié sous plusieurs formes:

- NomLogiqueFonction
- NoSerieModule.IdentifiantFonction
- NoSerieModule.NomLogiqueFonction
- NomLogiqueModule.IdentifiantMatériel
- NomLogiqueModule.NomLogiqueFonction

Cette fonction n'exige pas que le port de YoctoHub soit en ligne au moment où elle est appelée, l'objet retourné sera néanmoins valide. Utiliser la méthode `YHubPort.isOnline()` pour tester si le port de YoctoHub est utilisable à un moment donné. En cas d'ambiguïté lorsqu'on fait une recherche par nom logique, aucune erreur ne sera notifiée: la première instance trouvée sera renvoyée. La recherche se fait d'abord par nom matériel, puis par nom logique.

Si un appel à la méthode `is_online()` de cet objet renvoie FAUX alors que vous êtes sûr que le module correspondant est bien branché, vérifiez que vous n'avez pas oublié d'appeler `registerHub()` à l'initialisation de de l'application.

### Paramètres :

**func** une chaîne de caractères qui référence le port de YoctoHub sans ambiguïté, par exemple `YHUBETH1.hubPort1`.

### Retourne :

un objet de classe `YHubPort` qui permet ensuite de contrôler le port de YoctoHub.

## YHubPort.FindHubPortInContext() YHubPort.FindHubPortInContext()

YHubPort

Permet de retrouver un port de YoctoHub d'après un identifiant donné dans un Context YAPI.

```
java static YHubPort FindHubPortInContext( YAPIContext yctx, String func)
uwp static YHubPort FindHubPortInContext( YAPIContext yctx, string func)
ts static FindHubPortInContext( yctx: YAPIContext, func: string): YHubPort
es static FindHubPortInContext( yctx, func)
```

L'identifiant peut être spécifié sous plusieurs formes:

- NomLogiqueFonction
- NoSerieModule.IdentifiantFonction
- NoSerieModule.NomLogiqueFonction
- NomLogiqueModule.IdentifiantMatériel
- NomLogiqueModule.NomLogiqueFonction

Cette fonction n'exige pas que le port de YoctoHub soit en ligne au moment où elle est appelée, l'objet retourné sera néanmoins valide. Utiliser la méthode `YHubPort.isOnline()` pour tester si le port de YoctoHub est utilisable à un moment donné. En cas d'ambiguïté lorsqu'on fait une recherche par nom logique, aucune erreur ne sera notifiée: la première instance trouvée sera renvoyée. La recherche se fait d'abord par nom matériel, puis par nom logique.

### Paramètres :

**yctx** un contexte YAPI

**func** une chaîne de caractères qui référence le port de YoctoHub sans ambiguïté, par exemple `YHUBETH1.hubPort1`.

### Retourne :

un objet de classe `YHubPort` qui permet ensuite de contrôler le port de YoctoHub.

## YHubPort.FirstHubPort() YHubPort.FirstHubPort()

YHubPort

Commence l'énumération des ports de YoctoHub accessibles par la librairie.

js	function <b>yFirstHubPort</b> ( )
cpp	YHubPort * <b>FirstHubPort</b> ( )
m	+(YHubPort*) <b>FirstHubPort</b>
pas	TYHubPort <b>yFirstHubPort</b> ( ): TYHubPort
vb	function <b>FirstHubPort</b> ( ) As YHubPort
cs	static YHubPort <b>FirstHubPort</b> ( )
java	static YHubPort <b>FirstHubPort</b> ( )
uwp	static YHubPort <b>FirstHubPort</b> ( )
py	<b>FirstHubPort</b> ( )
php	function <b>FirstHubPort</b> ( )
ts	static <b>FirstHubPort</b> ( ): YHubPort   null
es	static <b>FirstHubPort</b> ( )

Utiliser la fonction `YHubPort.nextHubPort( )` pour itérer sur les autres ports de YoctoHub.

**Retourne :**

un pointeur sur un objet `YHubPort`, correspondant au premier port de YoctoHub accessible en ligne, ou `null` si il n'y a pas de ports de YoctoHub disponibles.

## YHubPort.FirstHubPortInContext() YHubPort.FirstHubPortInContext()

YHubPort

Commence l'énumération des ports de YoctoHub accessibles par la librairie.

```
java static YHubPort FirstHubPortInContext( YAPIContext yctx)
uwp static YHubPort FirstHubPortInContext( YAPIContext yctx)
ts static FirstHubPortInContext( yctx: YAPIContext): YHubPort | null
es static FirstHubPortInContext( yctx)
```

Utiliser la fonction `YHubPort.nextHubPort()` pour itérer sur les autres ports de YoctoHub.

### Paramètres :

`yctx` un contexte YAPI.

### Retourne :

un pointeur sur un objet `YHubPort`, correspondant au premier port de YoctoHub accessible en ligne, ou `null` si il n'y a pas de ports de YoctoHub disponibles.

## YHubPort.GetSimilarFunctions() YHubPort.GetSimilarFunctions()

YHubPort

Enumère toutes les fonctions de type HubPort disponibles sur les modules actuellement joignables par la librairie, et retourne leurs identifiants matériels uniques (hardwareId).

```
dnf static new string[] GetSimilarFunctions( )
```

```
cp static vector<string> GetSimilarFunctions( )
```

Chaque chaîne retournée peut être passée en argument à la méthode `YHubPort.FindHubPort` pour obtenir un objet permettant d'interagir avec le module correspondant.

**Retourne :**

un tableau de chaînes de caractères, contenant les identifiants matériels de chaque fonction disponible trouvée.

## hubport→AdvertisedValue

YHubPort

---

Courte chaîne de caractères représentant l'état courant de la fonction.

`dnsp` string **AdvertisedValue**

---

**hubport**→**Enabled****YHubPort**

---

Vrai si le port du YoctoHub est alimenté, faux sinon.

`dnsp` `int Enabled`

**Modifiable.** Modifie le mode d'activation du port du YoctoHub. Si le port est actif, il sera alimenté. Sinon, l'alimentation du module est coupée.

## hubport→FriendlyName

YHubPort

---

Identifiant global de la fonction au format NOM\_MODULE . NOM\_FONCTION.

`dnsp` string **FriendlyName**

Le chaîne retournée utilise soit les noms logiques du module et de la fonction si ils sont définis, soit respectivement le numéro de série du module et l'identifiant matériel de la fonction (par exemple: `MyCustomName.relay1`)

---

**hubport**→**FunctionId****YHubPort**

---

Identifiant matériel du port de YoctoHub, sans référence au module.

`dnsp` `string` **FunctionId**

Par exemple `relay1`.

## hubport→HardwareId

YHubPort

---

Identifiant matériel unique de la fonction au format `SERIAL.FUNCTIONID`.

`dnsp` string **HardwareId**

L'identifiant unique est composé du numéro de série du module et de l'identifiant matériel de la fonction (par exemple `RELAYLO1-123456.relay1`).

---

**hubport→IsOnline****YHubPort**

---

Vérifie si le module hébergeant la fonction est joignable, sans déclencher d'erreur.

`dnsp` `bool IsOnline`

Si les valeurs des attributs en cache de la fonction sont valides au moment de l'appel, le module est considéré joignable. Cette fonction ne cause en aucun cas d'exception, quelle que soit l'erreur qui pourrait se produire lors de la vérification de joignabilité.

## hubport→LogicalName

YHubPort

---

Nom logique de la fonction.

`dnf` `string LogicalName`

**Modifiable.** Vous pouvez utiliser `yCheckLogicalName()` pour vérifier si votre paramètre est valide. N'oubliez pas d'appeler la méthode `saveToFlash()` du module si le réglage doit être préservé.

---

**hubport**→**PortState****YHubPort**

---

état actuel du port de YoctoHub.

`dnsp` `int PortState`

**hubport**→**SerialNumber**

**YHubPort**

---

Numéro de série du module, préprogrammé en usine.

`dnsp` string **SerialNumber**

**hubport**→**clearCache()****YHubPort**

Invalide le cache.

js	function <b>clearCache</b> ( )
cpp	void <b>clearCache</b> ( )
m	-(void) <b>clearCache</b>
pas	<b>clearCache</b> ( )
vb	procedure <b>clearCache</b> ( )
cs	void <b>clearCache</b> ( )
java	void <b>clearCache</b> ( )
py	<b>clearCache</b> ( )
php	function <b>clearCache</b> ( )
ts	async <b>clearCache</b> ( ): Promise<void>
es	async <b>clearCache</b> ( )

Invalide le cache des valeurs courantes du port de YoctoHub. Force le prochain appel à une méthode `get_xxx()` ou `loadxxx()` pour charger les les données depuis le module.

**hubport**→**describe()****YHubPort**

Retourne un court texte décrivant de manière non-ambigüe l'instance du port de YoctoHub au format `TYPE (NAME) =SERIAL.FUNCTIONID`.

js	function <b>describe</b> ( )
cpp	string <b>describe</b> ( )
m	-(NSString*) <b>describe</b>
pas	string <b>describe</b> ( ): string
vb	function <b>describe</b> ( ) As String
cs	string <b>describe</b> ( )
java	String <b>describe</b> ( )
py	<b>describe</b> ( )
php	function <b>describe</b> ( )
ts	async <b>describe</b> ( ): Promise<string>
es	async <b>describe</b> ( )

Plus précisément, `TYPE` correspond au type de fonction, `NAME` correspond au nom utilisé lors du premier accès a la fonction, `SERIAL` correspond au numéro de série du module si le module est connecté, ou "unresolved" sinon, et `FUNCTIONID` correspond à l'identifiant matériel de la fonction si le module est connecté. Par exemple, La methode va retourner `Relay(MyCustomName.relay1)=RELAYLO1-123456.relay1` si le module est déjà connecté ou `Relay(BadCustomeName.relay1)=unresolved` si le module n'est pas déjà connecté. Cette methode ne declenche aucune transaction USB ou TCP et peut donc être utilisé dans un debugueur.

**Retourne :**

une chaîne de caractères décrivant le port de YoctoHub (ex:  
`Relay(MyCustomName.relay1)=RELAYLO1-123456.relay1`)

**hubport**→**get\_advertisedValue()****YHubPort****hubport**→**advertisedValue()**

Retourne la valeur courante du port de YoctoHub (pas plus de 6 caractères).

js	function <b>get_advertisedValue</b> ( )
cpp	string <b>get_advertisedValue</b> ( )
m	-(NSString*) advertisedValue
pas	string <b>get_advertisedValue</b> ( ): string
vb	function <b>get_advertisedValue</b> ( ) As String
cs	string <b>get_advertisedValue</b> ( )
java	String <b>get_advertisedValue</b> ( )
uwp	async Task<string> <b>get_advertisedValue</b> ( )
py	<b>get_advertisedValue</b> ( )
php	function <b>get_advertisedValue</b> ( )
ts	async <b>get_advertisedValue</b> ( ): Promise<string>
es	async <b>get_advertisedValue</b> ( )
dnp	string <b>get_advertisedValue</b> ( )
cp	string <b>get_advertisedValue</b> ( )
cmd	YHubPort <b>target</b> <b>get_advertisedValue</b>

**Retourne :**

une chaîne de caractères représentant la valeur courante du port de YoctoHub (pas plus de 6 caractères).

En cas d'erreur, déclenche une exception ou retourne `YHubPort.ADVERTISEDVALUE_INVALID`.

**hubport**→**get\_baudRate()****YHubPort****hubport**→**baudRate()**

Retourne la vitesse de transfert utilisée par le port de YoctoHub, en kbps.

js	function <b>get_baudRate</b> ( )
cpp	int <b>get_baudRate</b> ( )
m	-(int) baudRate
pas	LongInt <b>get_baudRate</b> ( ): LongInt
vb	function <b>get_baudRate</b> ( ) As Integer
cs	int <b>get_baudRate</b> ( )
java	int <b>get_baudRate</b> ( )
uwp	async Task<int> <b>get_baudRate</b> ( )
py	<b>get_baudRate</b> ( )
php	function <b>get_baudRate</b> ( )
ts	async <b>get_baudRate</b> ( ): Promise<number>
es	async <b>get_baudRate</b> ( )
dnp	int <b>get_baudRate</b> ( )
cp	int <b>get_baudRate</b> ( )
cmd	YHubPort <b>target</b> <b>get_baudRate</b>

La valeur par défaut est 1000 kbps, une valeur inférieure révèle des problèmes de communication.

**Retourne :**

un entier représentant la vitesse de transfert utilisée par le port de YoctoHub, en kbps

En cas d'erreur, déclenche une exception ou retourne `YHubPort.BAUDRATE_INVALID`.

**hubport**→**get\_enabled()****YHubPort****hubport**→**enabled()**

Retourne vrai si le port du YoctoHub est alimenté, faux sinon.

js	function <b>get_enabled</b> ( )
cpp	Y_ENABLED_enum <b>get_enabled</b> ( )
m	-(Y_ENABLED_enum) enabled
pas	Integer <b>get_enabled</b> ( ): Integer
vb	function <b>get_enabled</b> ( ) As Integer
cs	int <b>get_enabled</b> ( )
java	int <b>get_enabled</b> ( )
uwp	async Task<int> <b>get_enabled</b> ( )
py	<b>get_enabled</b> ( )
php	function <b>get_enabled</b> ( )
ts	async <b>get_enabled</b> ( ): Promise<YHubPort_Enabled>
es	async <b>get_enabled</b> ( )
dnp	int <b>get_enabled</b> ( )
cp	int <b>get_enabled</b> ( )
cmd	YHubPort <b>target get_enabled</b>

**Retourne :**

soit `YHubPort.ENABLED_FALSE`, soit `YHubPort.ENABLED_TRUE`, selon vrai si le port du YoctoHub est alimenté, faux sinon

En cas d'erreur, déclenche une exception ou retourne `YHubPort.ENABLED_INVALID`.

**hubport**→**get\_errorMessage()****YHubPort****hubport**→**errorMessage()**

Retourne le message correspondant à la dernière erreur survenue lors de l'utilisation du port de YoctoHub.

js	function <b>get_errorMessage</b> ( )
cpp	string <b>get_errorMessage</b> ( )
m	-(NSString*) <b>errorMessage</b>
pas	string <b>get_errorMessage</b> ( ): string
vb	function <b>get_errorMessage</b> ( ) As String
cs	string <b>get_errorMessage</b> ( )
java	String <b>get_errorMessage</b> ( )
py	<b>get_errorMessage</b> ( )
php	function <b>get_errorMessage</b> ( )
ts	<b>get_errorMessage</b> ( ): string
es	<b>get_errorMessage</b> ( )

Cette méthode est principalement utile lorsque la librairie Yoctopuce est utilisée en désactivant la gestion des exceptions.

**Retourne :**

une chaîne de caractères correspondant au message de la dernière erreur qui s'est produit lors de l'utilisation du port de YoctoHub.

**hubport**→**get\_errorType()****YHubPort****hubport**→**errorType()**

Retourne le code d'erreur correspondant à la dernière erreur survenue lors de l'utilisation du port de YoctoHub.

js	function <b>get_errorType</b> ( )
cpp	YRETCODE <b>get_errorType</b> ( )
m	-(YRETCODE) errorType
pas	YRETCODE <b>get_errorType</b> ( ): YRETCODE
vb	function <b>get_errorType</b> ( ) As YRETCODE
cs	YRETCODE <b>get_errorType</b> ( )
java	int <b>get_errorType</b> ( )
py	<b>get_errorType</b> ( )
php	function <b>get_errorType</b> ( )
ts	<b>get_errorType</b> ( ): number
es	<b>get_errorType</b> ( )

Cette méthode est principalement utile lorsque la librairie Yoctopuce est utilisée en désactivant la gestion des exceptions.

**Retourne :**

un nombre correspondant au code de la dernière erreur qui s'est produit lors de l'utilisation du port de YoctoHub.

**hubport**→**get\_friendlyName()****YHubPort****hubport**→**friendlyName()**

Retourne un identifiant global du port de YoctoHub au format `NOM_MODULE.NOM_FONCTION`.

js	function <b>get_friendlyName</b> ( )
cpp	string <b>get_friendlyName</b> ( )
m	-(NSString*) friendlyName
cs	string <b>get_friendlyName</b> ( )
java	String <b>get_friendlyName</b> ( )
py	<b>get_friendlyName</b> ( )
php	function <b>get_friendlyName</b> ( )
ts	async <b>get_friendlyName</b> ( ): Promise<string>
es	async <b>get_friendlyName</b> ( )
dnp	string <b>get_friendlyName</b> ( )
cp	string <b>get_friendlyName</b> ( )

Le chaîne retournée utilise soit les noms logiques du module et du port de YoctoHub si ils sont définis, soit respectivement le numéro de série du module et l'identifiant matériel du port de YoctoHub (par exemple: `MyCustomName.relay1`)

**Retourne :**

une chaîne de caractères identifiant le port de YoctoHub en utilisant les noms logiques (ex: `MyCustomName.relay1`)

En cas d'erreur, déclenche une exception ou retourne `YHubPort.FRIENDLYNAME_INVALID`.

**hubport**→**get\_functionDescriptor()****YHubPort****hubport**→**functionDescriptor()**

Retourne un identifiant unique de type YFUN\_DESCR correspondant à la fonction.

js	function <b>get_functionDescriptor</b> ( )
cpp	YFUN_DESCR <b>get_functionDescriptor</b> ( )
m	-(YFUN_DESCR) functionDescriptor
pas	YFUN_DESCR <b>get_functionDescriptor</b> ( ): YFUN_DESCR
vb	function <b>get_functionDescriptor</b> ( ) As YFUN_DESCR
cs	YFUN_DESCR <b>get_functionDescriptor</b> ( )
java	String <b>get_functionDescriptor</b> ( )
py	<b>get_functionDescriptor</b> ( )
php	function <b>get_functionDescriptor</b> ( )
ts	async <b>get_functionDescriptor</b> ( ): Promise<string>
es	async <b>get_functionDescriptor</b> ( )

Cet identifiant peut être utilisé pour tester si deux instance de YFunction référencent physiquement la même fonction sur le même module.

**Retourne :**

un identifiant de type YFUN\_DESCR.

Si la fonction n'a jamais été contactée, la valeur retournée sera Y\$CLASSNAME\$.FUNCTIONDESCRIPTOR\_INVALID

**hubport**→**get\_functionId()****hubport**→**functionId()**

Retourne l'identifiant matériel du port de YoctoHub, sans référence au module.

js	function <b>get_functionId</b> ( )
cpp	string <b>get_functionId</b> ( )
m	-(NSString*) <b>functionId</b>
vb	function <b>get_functionId</b> ( ) As String
cs	string <b>get_functionId</b> ( )
java	String <b>get_functionId</b> ( )
py	<b>get_functionId</b> ( )
php	function <b>get_functionId</b> ( )
ts	async <b>get_functionId</b> ( ): Promise<string>
es	async <b>get_functionId</b> ( )
dnp	string <b>get_functionId</b> ( )
cp	string <b>get_functionId</b> ( )

Par exemple `relay1`.

**Retourne :**

une chaîne de caractères identifiant le port de YoctoHub (ex: `relay1`)

En cas d'erreur, déclenche une exception ou retourne `YHubPort.FUNCTIONID_INVALID`.

**hubport**→**get\_hardwareId()****YHubPort****hubport**→**hardwareId()**

Retourne l'identifiant matériel unique du port de YoctoHub au format `SERIAL.FUNCTIONID`.

js	function <b>get_hardwareId</b> ( )
cpp	string <b>get_hardwareId</b> ( )
m	-(NSString*) <b>hardwareId</b>
vb	function <b>get_hardwareId</b> ( ) As String
cs	string <b>get_hardwareId</b> ( )
java	String <b>get_hardwareId</b> ( )
py	<b>get_hardwareId</b> ( )
php	function <b>get_hardwareId</b> ( )
ts	async <b>get_hardwareId</b> ( ): Promise<string>
es	async <b>get_hardwareId</b> ( )
dnp	string <b>get_hardwareId</b> ( )
cp	string <b>get_hardwareId</b> ( )

L'identifiant unique est composé du numéro de série du module et de l'identifiant matériel du port de YoctoHub (par exemple `RELAYLO1-123456.relay1`).

**Retourne :**

une chaîne de caractères identifiant le port de YoctoHub (ex: `RELAYLO1-123456.relay1`)

En cas d'erreur, déclenche une exception ou retourne `YHubPort.HARDWAREID_INVALID`.

## hubport→get\_logicalName()

YHubPort

## hubport→logicalName()

Retourne le nom logique du port de YoctoHub.

js	function <b>get_logicalName</b> ( )
cpp	string <b>get_logicalName</b> ( )
m	-(NSString*) logicalName
pas	string <b>get_logicalName</b> ( ): string
vb	function <b>get_logicalName</b> ( ) As String
cs	string <b>get_logicalName</b> ( )
java	String <b>get_logicalName</b> ( )
uwp	async Task<string> <b>get_logicalName</b> ( )
py	<b>get_logicalName</b> ( )
php	function <b>get_logicalName</b> ( )
ts	async <b>get_logicalName</b> ( ): Promise<string>
es	async <b>get_logicalName</b> ( )
dnp	string <b>get_logicalName</b> ( )
cp	string <b>get_logicalName</b> ( )
cmd	YHubPort <b>target</b> <b>get_logicalName</b>

**Retourne :**

une chaîne de caractères représentant le nom logique du port de YoctoHub.

En cas d'erreur, déclenche une exception ou retourne `YHubPort.LOGICALNAME_INVALID`.

**hubport**→**get\_module()****YHubPort****hubport**→**module()**

Retourne l'objet `YModule` correspondant au module Yoctopuce qui héberge la fonction.

js	function <b>get_module</b> ( )
cpp	<code>YModule *</code> <b>get_module</b> ( )
m	-( <code>YModule*</code> ) module
pas	<code>TYModule</code> <b>get_module</b> ( ): <code>TYModule</code>
vb	function <b>get_module</b> ( ) As <code>YModule</code>
cs	<code>YModule</code> <b>get_module</b> ( )
java	<code>YModule</code> <b>get_module</b> ( )
py	<b>get_module</b> ( )
php	function <b>get_module</b> ( )
ts	async <b>get_module</b> ( ): <code>Promise&lt;YModule&gt;</code>
es	async <b>get_module</b> ( )
dnp	<code>YModuleProxy</code> <b>get_module</b> ( )
cp	<code>YModuleProxy *</code> <b>get_module</b> ( )

Si la fonction ne peut être trouvée sur aucun module, l'instance de `YModule` retournée ne sera pas joignable.

**Retourne :**

une instance de `YModule`

**hubport**→**get\_module\_async()****YHubPort****hubport**→**module\_async()**

Retourne l'objet `YModule` correspondant au module Yoctopuce qui héberge la fonction.

```
js function get_module_async( callback, context)
```

Si la fonction ne peut être trouvée sur aucun module, l'instance de `YModule` retournée ne sera pas joignable.

Cette version asynchrone n'existe qu'en Javascript. Elle utilise une fonction de callback plutôt qu'une simple valeur de retour, pour éviter de bloquer la VM Javascript de Firefox, qui n'implémente pas le passage de contrôle entre threads durant les appels d'entrée/sortie bloquants.

**Paramètres :**

**callback** fonction de callback qui sera appelée dès que le résultat sera connu. La fonction callback reçoit trois arguments: le contexte fourni par l'appelant, l'objet fonction concerné et l'instance demandée de `YModule`

**context** contexte fourni par l'appelant, et qui sera passé tel-quel à la fonction de callback

**Retourne :**

rien du tout : le résultat sera passé en paramètre à la fonction de callback.

**hubport**→**get\_portState()****YHubPort****hubport**→**portState()**

Retourne l'état actuel du port de YoctoHub.

js	function <b>get_portState</b> ( )
cpp	Y_PORTSTATE_enum <b>get_portState</b> ( )
m	-(Y_PORTSTATE_enum) portState
pas	Integer <b>get_portState</b> ( ): Integer
vb	function <b>get_portState</b> ( ) As Integer
cs	int <b>get_portState</b> ( )
java	int <b>get_portState</b> ( )
uwp	async Task<int> <b>get_portState</b> ( )
py	<b>get_portState</b> ( )
php	function <b>get_portState</b> ( )
ts	async <b>get_portState</b> ( ): Promise<YHubPort_PortState>
es	async <b>get_portState</b> ( )
dnp	int <b>get_portState</b> ( )
cp	int <b>get_portState</b> ( )
cmd	YHubPort <b>target</b> <b>get_portState</b>

**Retourne :**

une valeur parmi `YHubPort.PORTSTATE_OFF`, `YHubPort.PORTSTATE_OVRLD`, `YHubPort.PORTSTATE_ON`, `YHubPort.PORTSTATE_RUN` et `YHubPort.PORTSTATE_PROG` représentant l'état actuel du port de YoctoHub

En cas d'erreur, déclenche une exception ou retourne `YHubPort.PORTSTATE_INVALID`.

**hubport**→**get\_serialNumber()****YHubPort****hubport**→**serialNumber()**

Retourne le numéro de série du module, préprogrammé en usine.

js	function <b>get_serialNumber</b> ( )
cpp	string <b>get_serialNumber</b> ( )
m	-(NSString*) <b>serialNumber</b>
pas	string <b>get_serialNumber</b> ( ): string
vb	function <b>get_serialNumber</b> ( ) As String
cs	string <b>get_serialNumber</b> ( )
java	String <b>get_serialNumber</b> ( )
uwp	async Task<string> <b>get_serialNumber</b> ( )
py	<b>get_serialNumber</b> ( )
php	function <b>get_serialNumber</b> ( )
ts	async <b>get_serialNumber</b> ( ): Promise<string>
es	async <b>get_serialNumber</b> ( )
dnp	string <b>get_serialNumber</b> ( )
cp	string <b>get_serialNumber</b> ( )
cmd	YHubPort <b>target</b> <b>get_serialNumber</b>

**Retourne :**

: une chaîne de caractères représentant le numéro de série du module, préprogrammé en usine.

En cas d'erreur, déclenche une exception ou retourne YFunction.SERIALNUMBER\_INVALID.

**hubport**→**get\_userData()****YHubPort****hubport**→**userData()**

Retourne le contenu de l'attribut `userData`, précédemment stocké à l'aide de la méthode `set_userData`.

js	function <b>get_userData</b> ( )
cpp	void * <b>get_userData</b> ( )
m	-(id) userData
pas	Tobject <b>get_userData</b> ( ): Tobject
vb	function <b>get_userData</b> ( ) As Object
cs	object <b>get_userData</b> ( )
java	Object <b>get_userData</b> ( )
py	<b>get_userData</b> ( )
php	function <b>get_userData</b> ( )
ts	async <b>get_userData</b> ( ): Promise<object null>
es	async <b>get_userData</b> ( )

Cet attribut n'es pas utilisé directement par l'API. Il est à la disposition de l'appelant pour stocker un contexte.

**Retourne :**

l'objet stocké précédemment par l'appelant.

## hubport→isOnline()

YHubPort

Vérifie si le module hébergeant le port de YoctoHub est joignable, sans déclencher d'erreur.

js	function <b>isOnline</b> ( )
cpp	bool <b>isOnline</b> ( )
m	-(BOOL) <b>isOnline</b>
pas	boolean <b>isOnline</b> ( ): boolean
vb	function <b>isOnline</b> ( ) As Boolean
cs	bool <b>isOnline</b> ( )
java	boolean <b>isOnline</b> ( )
py	<b>isOnline</b> ( )
php	function <b>isOnline</b> ( )
ts	async <b>isOnline</b> ( ): Promise<boolean>
es	async <b>isOnline</b> ( )
dnp	bool <b>isOnline</b> ( )
cp	bool <b>isOnline</b> ( )

Si les valeurs des attributs en cache du port de YoctoHub sont valides au moment de l'appel, le module est considéré joignable. Cette fonction ne cause en aucun cas d'exception, quelle que soit l'erreur qui pourrait se produire lors de la vérification de joignabilité.

**Retourne :**

`true` si le port de YoctoHub est joignable, `false` sinon

**hubport**→**isOnline\_async()****YHubPort**

Vérifie si le module hébergeant le port de YoctoHub est joignable, sans déclencher d'erreur.

```
js fonction isOnline_async( callback, context)
```

Si les valeurs des attributs en cache du port de YoctoHub sont valides au moment de l'appel, le module est considéré joignable. Cette fonction ne cause en aucun cas d'exception, quelle que soit l'erreur qui pourrait se produire lors de la vérification de joignabilité.

Cette version asynchrone n'existe qu'en Javascript. Elle utilise une fonction de callback plutôt qu'une simple valeur de retour, pour éviter de bloquer la machine virtuelle Javascript avec une attente active.

**Paramètres :**

**callback** fonction de callback qui sera appelée dès que le résultat sera connu. La fonction callback reçoit trois arguments: le contexte fourni par l'appelant, l'objet fonction concerné et le résultat booléen

**context** contexte fourni par l'appelant, et qui sera passé tel-quel à la fonction de callback

**Retourne :**

rien du tout : le résultat sera passé en paramètre à la fonction de callback.

Test si la fonction est en lecture seule.

cpp	bool <b>isReadOnly</b> ( )
m	-(bool) <b>isReadOnly</b>
pas	boolean <b>isReadOnly</b> ( ): boolean
vb	function <b>isReadOnly</b> ( ) As Boolean
cs	bool <b>isReadOnly</b> ( )
java	boolean <b>isReadOnly</b> ( )
uwp	async Task<bool> <b>isReadOnly</b> ( )
py	<b>isReadOnly</b> ( )
php	function <b>isReadOnly</b> ( )
ts	async <b>isReadOnly</b> ( ): Promise<boolean>
es	async <b>isReadOnly</b> ( )
dnp	bool <b>isReadOnly</b> ( )
cp	bool <b>isReadOnly</b> ( )
cmd	YHubPort <b>target isReadOnly</b>

Retourne vrais si la fonction est protégé en ecriture ou que la fontion n'est pas disponible.

**Retourne :**

true si la fonction est protégé en ecriture ou que la fontion n'est pas disponible

**hubport→load()****YHubPort**

Met en cache les valeurs courantes du port de YoctoHub, avec une durée de validité spécifiée.

js	function <b>load</b> ( <b>msValidity</b> )
cpp	YRETCODE <b>load</b> ( int <b>msValidity</b> )
m	-(YRETCODE) <b>load</b> : (u64) <b>msValidity</b>
pas	YRETCODE <b>load</b> ( <b>msValidity</b> : u64): YRETCODE
vb	function <b>load</b> ( ByVal <b>msValidity</b> As Long) As YRETCODE
cs	YRETCODE <b>load</b> ( ulong <b>msValidity</b> )
java	int <b>load</b> ( long <b>msValidity</b> )
py	<b>load</b> ( <b>msValidity</b> )
php	function <b>load</b> ( <b>\$msValidity</b> )
ts	async <b>load</b> ( <b>msValidity</b> : number): Promise<number>
es	async <b>load</b> ( <b>msValidity</b> )

Par défaut, lorsqu'on accède à un module, tous les attributs des fonctions du module sont automatiquement mises en cache pour la durée standard (5 ms). Cette méthode peut être utilisée pour marquer occasionnellement les données cachées comme valides pour une plus longue période, par exemple dans le but de réduire le trafic réseau.

**Paramètres :**

**msValidity** un entier correspondant à la durée de validité attribuée aux les paramètres chargés, en millisecondes

**Retourne :**

YAPI . SUCCESS si l'opération se déroule sans erreur.

En cas d'erreur, déclenche une exception ou retourne un code d'erreur négatif.

## hubport→loadAttribute()

YHubPort

Retourne la valeur actuelle d'un attribut spécifique de la fonction, sous forme de texte, le plus rapidement possible mais sans passer par le cache.

js	function <b>loadAttribute</b> ( <b>attrName</b> )
cpp	string <b>loadAttribute</b> ( string <b>attrName</b> )
m	-(NSString*) <b>loadAttribute</b> : (NSString*) <b>attrName</b>
pas	string <b>loadAttribute</b> ( <b>attrName</b> : string): string
vb	function <b>loadAttribute</b> ( ByVal <b>attrName</b> As String) As String
cs	string <b>loadAttribute</b> ( string <b>attrName</b> )
java	String <b>loadAttribute</b> ( String <b>attrName</b> )
uwp	async Task<string> <b>loadAttribute</b> ( string <b>attrName</b> )
py	<b>loadAttribute</b> ( <b>attrName</b> )
php	function <b>loadAttribute</b> ( <b>\$attrName</b> )
ts	async <b>loadAttribute</b> ( <b>attrName</b> : string): Promise<string>
es	async <b>loadAttribute</b> ( <b>attrName</b> )
dnp	string <b>loadAttribute</b> ( string <b>attrName</b> )
cp	string <b>loadAttribute</b> ( string <b>attrName</b> )

**Paramètres :**

**attrName** le nom de l'attribut désiré

**Retourne :**

une chaîne de caractères représentant la valeur actuelle de l'attribut.

En cas d'erreur, déclenche une exception ou retourne un chaîne vide.

**hubport**→**load\_async()****YHubPort**

Met en cache les valeurs courantes du port de YoctoHub, avec une durée de validité spécifiée.

```
js function load_async( msValidity, callback, context)
```

Par défaut, lorsqu'on accède à un module, tous les attributs des fonctions du module sont automatiquement mises en cache pour la durée standard (5 ms). Cette méthode peut être utilisée pour marquer occasionnellement les données cachées comme valides pour une plus longue période, par exemple dans le but de réduire le trafic réseau.

Cette version asynchrone n'existe qu'en Javascript. Elle utilise une fonction de callback plutôt qu'une simple valeur de retour, pour éviter de bloquer la machine virtuelle Javascript avec une attente active.

**Paramètres :**

- msValidity** un entier correspondant à la durée de validité attribuée aux les paramètres chargés, en millisecondes
- callback** fonction de callback qui sera appelée dès que le résultat sera connu. La fonction callback reçoit trois arguments: le contexte fourni par l'appelant, l'objet fonction concerné et le code d'erreur (ou `YAPI . SUCCESS`)
- context** contexte fourni par l'appelant, et qui sera passé tel-quel à la fonction de callback

**Retourne :**

rien du tout : le résultat sera passé en paramètre à la fonction de callback.

## hubport→muteValueCallbacks()

YHubPort

Désactive l'envoi de chaque changement de la valeur publiée au hub parent.

js	function <b>muteValueCallbacks</b> ( )
cpp	int <b>muteValueCallbacks</b> ( )
m	-(int) <b>muteValueCallbacks</b>
pas	LongInt <b>muteValueCallbacks</b> ( ): LongInt
vb	function <b>muteValueCallbacks</b> ( ) As Integer
cs	int <b>muteValueCallbacks</b> ( )
java	int <b>muteValueCallbacks</b> ( )
uwp	async Task<int> <b>muteValueCallbacks</b> ( )
py	<b>muteValueCallbacks</b> ( )
php	function <b>muteValueCallbacks</b> ( )
ts	async <b>muteValueCallbacks</b> ( ): Promise<number>
es	async <b>muteValueCallbacks</b> ( )
dnp	int <b>muteValueCallbacks</b> ( )
cp	int <b>muteValueCallbacks</b> ( )
cmd	YHubPort <b>target</b> <b>muteValueCallbacks</b>

Vous pouvez utiliser cette fonction pour économiser la bande passante et le CPU sur les machines de faible puissance, ou pour éviter le déclenchement de callbacks HTTP. N'oubliez pas d'appeler la méthode `saveToFlash()` du module si le réglage doit être préservé.

**Retourne :**

YAPI . SUCCESS si l'opération se déroule sans erreur.

En cas d'erreur, déclenche une exception ou retourne un code d'erreur négatif.

**hubport**→**nextHubPort()****YHubPort**

Continue l'énumération des ports de YoctoHub commencée à l'aide de `yFirstHubPort()`. Attention, vous ne pouvez faire aucune supposition sur l'ordre dans lequel les ports de YoctoHub sont retournés.

js	function <b>nextHubPort</b> ( )
cpp	<b>YHubPort * nextHubPort</b> ( )
m	-(nullable <b>YHubPort*</b> ) <b>nextHubPort</b>
pas	<b>TYHubPort nextHubPort</b> ( ): <b>TYHubPort</b>
vb	function <b>nextHubPort</b> ( ) As <b>YHubPort</b>
cs	<b>YHubPort nextHubPort</b> ( )
java	<b>YHubPort nextHubPort</b> ( )
uwp	<b>YHubPort nextHubPort</b> ( )
py	<b>nextHubPort</b> ( )
php	function <b>nextHubPort</b> ( )
ts	<b>nextHubPort</b> ( ): <b>YHubPort</b>   null
es	<b>nextHubPort</b> ( )

Si vous souhaitez retrouver un port de YoctoHub spécifique, utilisez `HubPort.findHubPort()` avec un `hardwareID` ou un nom logique.

**Retourne :**

un pointeur sur un objet `YHubPort` accessible en ligne, ou `null` lorsque l'énumération est terminée.

## hubport→registerValueCallback()

YHubPort

Enregistre la fonction de callback qui est appelée à chaque changement de la valeur publiée.

js	function <b>registerValueCallback</b> ( <b>callback</b> )
cpp	int <b>registerValueCallback</b> ( YHubPortValueCallback <b>callback</b> )
m	-(int) <b>registerValueCallback</b> : (YHubPortValueCallback _Nullable) <b>callback</b>
pas	LongInt <b>registerValueCallback</b> ( <b>callback</b> : TYHubPortValueCallback): LongInt
vb	function <b>registerValueCallback</b> ( ByVal <b>callback</b> As YHubPortValueCallback) As Integer
cs	int <b>registerValueCallback</b> ( ValueCallback <b>callback</b> )
java	int <b>registerValueCallback</b> ( UpdateCallback <b>callback</b> )
uwp	async Task<int> <b>registerValueCallback</b> ( ValueCallback <b>callback</b> )
py	<b>registerValueCallback</b> ( <b>callback</b> )
php	function <b>registerValueCallback</b> ( <b>\$callback</b> )
ts	async <b>registerValueCallback</b> ( <b>callback</b> : YHubPortValueCallback   null): Promise<number>
es	async <b>registerValueCallback</b> ( <b>callback</b> )

Ce callback n'est appelé que durant l'exécution de `ySleep` ou `yHandleEvents`. Cela permet à l'appelant de contrôler quand les callback peuvent se produire. Il est important d'appeler l'une de ces deux fonctions périodiquement pour garantir que les callback ne soient pas appelés trop tard. Pour désactiver un callback, il suffit d'appeler cette méthode en lui passant un pointeur nul.

**Paramètres :**

**callback** la fonction de callback à rappeler, ou un pointeur nul. La fonction de callback doit accepter deux arguments: l'object fonction dont la valeur a changé, et la chaîne de caractère décrivant la nouvelle valeur publiée.

**hubport**→**set\_enabled()****YHubPort****hubport**→**setEnabled()**

Modifie le mode d'activation du port du YoctoHub.

js	function <b>set_enabled</b> ( <b>newval</b> )
cpp	int <b>set_enabled</b> ( Y_ENABLED_enum <b>newval</b> )
m	-(int) <b>setEnabled</b> : (Y_ENABLED_enum) <b>newval</b>
pas	integer <b>set_enabled</b> ( <b>newval</b> : Integer): integer
vb	function <b>set_enabled</b> ( ByVal <b>newval</b> As Integer) As Integer
cs	int <b>set_enabled</b> ( int <b>newval</b> )
java	int <b>set_enabled</b> ( int <b>newval</b> )
uwp	async Task<int> <b>set_enabled</b> ( int <b>newval</b> )
py	<b>set_enabled</b> ( <b>newval</b> )
php	function <b>set_enabled</b> ( <b>\$newval</b> )
ts	async <b>set_enabled</b> ( <b>newval</b> : YHubPort_Enabled): Promise<number>
es	async <b>set_enabled</b> ( <b>newval</b> )
dnp	int <b>set_enabled</b> ( int <b>newval</b> )
cp	int <b>set_enabled</b> ( int <b>newval</b> )
cmd	YHubPort <b>target set_enabled newval</b>

Si le port est actif, il sera alimenté. Sinon, l'alimentation du module est coupée.

**Paramètres :**

**newval** soit `YHubPort.ENABLED_FALSE`, soit `YHubPort.ENABLED_TRUE`, selon le mode d'activation du port du YoctoHub

**Retourne :**

`YAPI.SUCCESS` si l'opération se déroule sans erreur.

En cas d'erreur, déclenche une exception ou retourne un code d'erreur négatif.

**hubport**→**set\_logicalName()****YHubPort****hubport**→**setLogicalName()**

Modifie le nom logique du port de YoctoHub.

js	function <b>set_logicalName</b> ( <b>newval</b> )
cpp	int <b>set_logicalName</b> ( string <b>newval</b> )
m	-(int) setLogicalName : (NSString*) <b>newval</b>
pas	integer <b>set_logicalName</b> ( <b>newval</b> : string): integer
vb	function <b>set_logicalName</b> ( ByVal <b>newval</b> As String) As Integer
cs	int <b>set_logicalName</b> ( string <b>newval</b> )
java	int <b>set_logicalName</b> ( String <b>newval</b> )
uwp	async Task<int> <b>set_logicalName</b> ( string <b>newval</b> )
py	<b>set_logicalName</b> ( <b>newval</b> )
php	function <b>set_logicalName</b> ( <b>\$newval</b> )
ts	async <b>set_logicalName</b> ( <b>newval</b> : string): Promise<number>
es	async <b>set_logicalName</b> ( <b>newval</b> )
dnp	int <b>set_logicalName</b> ( string <b>newval</b> )
cp	int <b>set_logicalName</b> ( string <b>newval</b> )
cmd	YHubPort <b>target</b> <b>set_logicalName</b> <b>newval</b>

Vous pouvez utiliser `yCheckLogicalName()` pour vérifier si votre paramètre est valide. N'oubliez pas d'appeler la méthode `saveToFlash()` du module si le réglage doit être préservé.

**Paramètres :**

**newval** une chaîne de caractères représentant le nom logique du port de YoctoHub.

**Retourne :**

YAPI . SUCCESS si l'appel se déroule sans erreur.

En cas d'erreur, déclenche une exception ou retourne un code d'erreur négatif.

**hubport**→**set\_userData()****YHubPort****hubport**→**setUserData()**

Enregistre un contexte libre dans l'attribut `userData` de la fonction, afin de le retrouver plus tard à l'aide de la méthode `get_userData`.

js	function <b>set_userData</b> ( <b>data</b> )
cpp	void <b>set_userData</b> ( void * <b>data</b> )
m	-(void) setUserData : (id) <b>data</b>
pas	<b>set_userData</b> ( <b>data</b> : Tobject)
vb	procedure <b>set_userData</b> ( ByVal <b>data</b> As Object)
cs	void <b>set_userData</b> ( object <b>data</b> )
java	void <b>set_userData</b> ( Object <b>data</b> )
py	<b>set_userData</b> ( <b>data</b> )
php	function <b>set_userData</b> ( <b>\$data</b> )
ts	async <b>set_userData</b> ( <b>data</b> : object null): Promise<void>
es	async <b>set_userData</b> ( <b>data</b> )

Cet attribut n'es pas utilisé directement par l'API. Il est à la disposition de l'appelant pour stocker un contexte.

**Paramètres :**

**data** objet quelconque à mémoriser

## hubport→unmuteValueCallbacks()

YHubPort

Réactive l'envoi de chaque changement de la valeur publiée au hub parent.

js	function <b>unmuteValueCallbacks</b> ( )
cpp	int <b>unmuteValueCallbacks</b> ( )
m	-(int) <b>unmuteValueCallbacks</b>
pas	LongInt <b>unmuteValueCallbacks</b> ( ): LongInt
vb	function <b>unmuteValueCallbacks</b> ( ) As Integer
cs	int <b>unmuteValueCallbacks</b> ( )
java	int <b>unmuteValueCallbacks</b> ( )
uwp	async Task<int> <b>unmuteValueCallbacks</b> ( )
py	<b>unmuteValueCallbacks</b> ( )
php	function <b>unmuteValueCallbacks</b> ( )
ts	async <b>unmuteValueCallbacks</b> ( ): Promise<number>
es	async <b>unmuteValueCallbacks</b> ( )
dnp	int <b>unmuteValueCallbacks</b> ( )
cp	int <b>unmuteValueCallbacks</b> ( )
cmd	YHubPort <b>target</b> <b>unmuteValueCallbacks</b>

Cette fonction annule un précédent appel à `muteValueCallbacks()`. N'oubliez pas d'appeler la méthode `saveToFlash()` du module si le réglage doit être préservé.

**Retourne :**

YAPI . SUCCESS si l'opération se déroule sans erreur.

En cas d'erreur, déclenche une exception ou retourne un code d'erreur négatif.

**hubport**→**wait\_async()****YHubPort**

Attend que toutes les commandes asynchrones en cours d'exécution sur le module soient terminées, et appelle le callback passé en paramètre.

```
js fonction wait_async( callback, context)
```

```
ts wait_async( callback: Function, context: object)
```

```
es wait_async( callback, context)
```

La fonction callback peut donc librement utiliser des fonctions synchrones ou asynchrones, sans risquer de bloquer la machine virtuelle Javascript.

**Paramètres :**

**callback** fonction de callback qui sera appelée dès que toutes les commandes en cours d'exécution sur le module seront terminées. La fonction callback reçoit deux arguments: le contexte fourni par l'appelant et l'objet fonction concerné.

**context** contexte fourni par l'appelant, et qui sera passé tel-quel à la fonction de callback

**Retourne :**

rien du tout.

## 8.2. La classe YCellular

Interface pour interagir avec les interfaces réseau cellulaire, disponibles par exemple dans le YoctoHub-GSM-2G, le YoctoHub-GSM-3G-EU, le YoctoHub-GSM-3G-NA et le YoctoHub-GSM-4G

La classe YCellular permet de configurer et de contrôler la configuration du réseau cellulaire sur les modules Yoctopuce qui en sont dotés. Notez que les paramètres TCP/IP sont configurés séparément, à l'aide de la classe YNetwork.

Pour utiliser les fonctions décrites ici, vous devez inclure:

js	<code>&lt;script type='text/javascript' src='yocto_cellular.js'&gt;&lt;/script&gt;</code>
cpp	<code>#include "yocto_cellular.h"</code>
m	<code>#import "yocto_cellular.h"</code>
pas	<code>uses yocto_cellular;</code>
vb	<code>yocto_cellular.vb</code>
cs	<code>yocto_cellular.cs</code>
java	<code>import com.yoctopuce.YoctoAPI.YCellular;</code>
uwp	<code>import com.yoctopuce.YoctoAPI.YCellular;</code>
py	<code>from yocto_cellular import *</code>
php	<code>require_once('yocto_cellular.php');</code>
ts	<code>in HTML: import { YCellular } from './../dist/esm/yocto_cellular.js'; in Node.js: import { YCellular } from 'yoctolib-cjs/yocto_cellular.js';</code>
es	<code>in HTML: &lt;script src='./../lib/yocto_cellular.js'&gt;&lt;/script&gt; in node.js: require('yoctolib-es2017/yocto_cellular.js');</code>
dnf	<code>import YoctoProxyAPI.YCellularProxy</code>
cp	<code>#include "yocto_cellular_proxy.h"</code>
vi	<code>YCellular.vi</code>
ml	<code>import YoctoProxyAPI.YCellularProxy</code>

### Fonction globales

#### YCellular.FindCellular(func)

Permet de retrouver une interface cellulaire d'après un identifiant donné.

#### YCellular.FindCellularInContext(yctx, func)

Permet de retrouver une interface cellulaire d'après un identifiant donné dans un Context YAPI.

#### YCellular.FirstCellular()

Commence l'énumération des interfaces réseau cellulaire accessibles par la librairie.

#### YCellular.FirstCellularInContext(yctx)

Commence l'énumération des interfaces réseau cellulaire accessibles par la librairie.

#### YCellular.GetSimilarFunctions()

Enumère toutes les fonctions de type Cellular disponibles sur les modules actuellement joignables par la librairie, et retourne leurs identifiants matériels uniques (hardwareId).

### Propriétés des objets YCellularProxy

#### cellular→AdvertisedValue [lecture seule]

Courte chaîne de caractères représentant l'état courant de la fonction.

#### cellular→Apn [modifiable]

Nom du point d'accès (APN) à utiliser, si nécessaire.

#### cellular→CellOperator [lecture seule]

Nom de l'opérateur de réseau cellulaire actuellement utilisé.

#### cellular→EnableData [modifiable]

Condition dans laquelle le service de données IP (GRPS) doit être activé.

**cellular**→**FriendlyName** [*lecture seule*]

Identifiant global de la fonction au format NOM\_MODULE . NOM\_FONCTION.

**cellular**→**FunctionId** [*lecture seule*]

Identifiant matériel de l'interface cellulaire, sans référence au module.

**cellular**→**HardwareId** [*lecture seule*]

Identifiant matériel unique de la fonction au format SERIAL . FUNCTIONID.

**cellular**→**IsOnline** [*lecture seule*]

Vérifie si le module hébergeant la fonction est joignable, sans déclencher d'erreur.

**cellular**→**LinkQuality** [*lecture seule*]

Qualité de la connection, exprimée en pourcents.

**cellular**→**LockedOperator** [*modifiable*]

Nom de l'opérateur de réseau cellulaire à utiliser exclusivement, si le choix automatique est désactivé, ou une chaîne vide si la carte SIM sélectionne automatiquement l'opérateur selon ceux disponibles.

**cellular**→**LogicalName** [*modifiable*]

Nom logique de la fonction.

**cellular**→**Pin** [*modifiable*]

String opaque si un code PIN a été configuré dans le module pour accéder à la carte SIM, ou une chaîne vide il n'a pas été configuré ou si la SIM a rejeté le code indiqué.

**cellular**→**PingInterval** [*modifiable*]

Intervalle entre les tests de connectivité spontanés, en secondes.

**cellular**→**RadioConfig** [*modifiable*]

Type de protocole utilisé sur la communication série, sous forme d'une chaîne de caractères.

**cellular**→**SerialNumber** [*lecture seule*]

Numéro de série du module, préprogrammé en usine.

### Méthodes des objets YCellular

**cellular**→**\_AT**(cmd)

Envoie une commande AT au module GSM, et retourne le résultat.

**cellular**→**clearCache**()

Invalide le cache.

**cellular**→**clearDataCounters**()

Réinitialise les compteurs de données transmises et reçues.

**cellular**→**decodePLMN**(mccmnc)

Retourne le nom usuel de l'opérateur cellulaire pour une paire MCC/MNC donnée.

**cellular**→**describe**()

Retourne un court texte décrivant de manière non-ambigüe l'instance de l'interface cellulaire au format TYPE ( NAME ) = SERIAL . FUNCTIONID.

**cellular**→**get\_advertisedValue**()

Retourne la valeur courante de l'interface cellulaire (pas plus de 6 caractères).

**cellular**→**get\_airplaneMode**()

Retourne vrai si le mode avion est activé (radio désactivée).

**cellular**→**get\_apn**()

Retourne le nom du point d'accès (APN) à utiliser, si nécessaire.

**cellular**→**get\_apnSecret**()

Retourne une string opaque si des paramètres d'identification sur l'APN ont été configurés dans le module, ou une chaîne vide autrement.

**cellular**→**get\_availableOperators()**

Retourne la liste des opérateurs GSM disponibles à proximité.

**cellular**→**get\_cellIdentifier()**

Retourne l'identifiant unique de la station de base utilisée: MCC, MNC, LAC et Cell ID.

**cellular**→**get\_cellOperator()**

Retourne le nom de l'opérateur de réseau cellulaire actuellement utilisé.

**cellular**→**get\_cellType()**

Type de connexion cellulaire active.

**cellular**→**get\_dataReceived()**

Retourne le nombre d'octets reçus jusqu'à présent.

**cellular**→**get\_dataSent()**

Retourne le nombre d'octets envoyés jusqu'à présent.

**cellular**→**get\_enableData()**

Retourne la condition dans laquelle le service de données IP (GRPS) doit être activé.

**cellular**→**get\_errorMessage()**

Retourne le message correspondant à la dernière erreur survenue lors de l'utilisation de l'interface cellulaire.

**cellular**→**get\_errorType()**

Retourne le code d'erreur correspondant à la dernière erreur survenue lors de l'utilisation de l'interface cellulaire.

**cellular**→**get\_friendlyName()**

Retourne un identifiant global de l'interface cellulaire au format NOM\_MODULE . NOM\_FONCTION.

**cellular**→**get\_functionDescriptor()**

Retourne un identifiant unique de type YFUN\_DESCR correspondant à la fonction.

**cellular**→**get\_functionId()**

Retourne l'identifiant matériel de l'interface cellulaire, sans référence au module.

**cellular**→**get\_hardwareId()**

Retourne l'identifiant matériel unique de l'interface cellulaire au format SERIAL . FUNCTIONID.

**cellular**→**get\_imsi()**

Retourne le "International Mobile Subscriber Identity" (MSI) qui identifie de manière unique la carte SIM.

**cellular**→**get\_linkQuality()**

Retourne la qualité de la connexion, exprimée en pourcents.

**cellular**→**get\_lockedOperator()**

Retourne le nom de l'opérateur de réseau cellulaire à utiliser exclusivement, si le choix automatique est désactivé, ou une chaîne vide si la carte SIM sélectionne automatiquement l'opérateur selon ceux disponibles.

**cellular**→**get\_logicalName()**

Retourne le nom logique de l'interface cellulaire.

**cellular**→**get\_message()**

Retourne le dernier message de diagnostic de l'interface au réseau sans fil.

**cellular**→**get\_module()**

Retourne l'objet YModule correspondant au module Yoctopuce qui héberge la fonction.

**cellular**→**get\_module\_async(callback, context)**

Retourne l'objet YModule correspondant au module Yoctopuce qui héberge la fonction.

**cellular**→**get\_pin()**

Retourne une string opaque si un code PIN a été configuré dans le module pour accéder à la carte SIM, ou une chaîne vide il n'a pas été configuré ou si la SIM a rejeté le code indiqué.

**cellular**→**get\_pingInterval()**

Retourne l'intervalle entre les tests de connectivité spontanés, en secondes.

**cellular**→**get\_radioConfig()**

Retourne le type de protocole utilisé sur la communication série, sous forme d'une chaîne de caractères.

**cellular**→**get\_serialNumber()**

Retourne le numéro de série du module, préprogrammé en usine.

**cellular**→**get\_userData()**

Retourne le contenu de l'attribut userData, précédemment stocké à l'aide de la méthode `set_userData`.

**cellular**→**isOnline()**

Vérifie si le module hébergeant l'interface cellulaire est joignable, sans déclencher d'erreur.

**cellular**→**isOnline\_async(callback, context)**

Vérifie si le module hébergeant l'interface cellulaire est joignable, sans déclencher d'erreur.

**cellular**→**isReadOnly()**

Test si la fonction est en lecture seule.

**cellular**→**load(msValidity)**

Met en cache les valeurs courantes de l'interface cellulaire, avec une durée de validité spécifiée.

**cellular**→**loadAttribute(attrName)**

Retourne la valeur actuelle d'un attribut spécifique de la fonction, sous forme de texte, le plus rapidement possible mais sans passer par le cache.

**cellular**→**load\_async(msValidity, callback, context)**

Met en cache les valeurs courantes de l'interface cellulaire, avec une durée de validité spécifiée.

**cellular**→**muteValueCallbacks()**

Désactive l'envoi de chaque changement de la valeur publiée au hub parent.

**cellular**→**nextCellular()**

Continue l'énumération des interfaces réseau cellulaire commencée à l'aide de `yFirstCellular()`. Attention, vous ne pouvez faire aucune supposition sur l'ordre dans lequel les interfaces réseau cellulaire sont retournés.

**cellular**→**quickCellSurvey()**

Retourne la liste d'identifiants pour les antennes GSM à proximité, telle que requise pour géolocaliser rapidement le module.

**cellular**→**registerValueCallback(callback)**

Enregistre la fonction de callback qui est appelée à chaque changement de la valeur publiée.

**cellular**→**sendPUK(puk, newPin)**

Envoie le code PUK à la carte SIM pour la débloquer après trois échecs consécutifs de code PIN, et établit un nouveau code PIN dans la SIM.

**cellular**→**set\_airplaneMode(newval)**

Modifie l'état du mode avion (radio désactivée).

**cellular**→**set\_apn(newval)**

Retourne le nom du point d'accès (APN) à utiliser, si nécessaire.

**cellular**→**set\_apnAuth(username, password)**

Configure les paramètres d'identification pour se connecter à l'APN.

**cellular**→**set\_dataReceived(newval)**

Modifie la valeur du compteur d'octets reçus.

**cellular**→**set\_dataSent(newval)**

Modifie la valeur du compteur d'octets envoyés.

**cellular**→**set\_enableData(newval)**

Modifie la condition dans laquelle le service de données IP (GRPS) doit être activé.

**cellular**→**set\_lockedOperator(newval)**

Modifie le nom de l'opérateur de réseau cellulaire à utiliser.

**cellular**→**set\_logicalName(newval)**

Modifie le nom logique de l'interface cellulaire.

**cellular**→**set\_pin(newval)**

Modifie le code PIN utilisé par le module pour accéder à la carte SIM.

**cellular**→**set\_pingInterval(newval)**

Modifie l'intervalle entre les tests de connectivité spontanés, en secondes.

**cellular**→**set\_radioConfig(newval)**

Modifie le type de protocole utilisé sur la communication série.

**cellular**→**set\_userData(data)**

Enregistre un contexte libre dans l'attribut userData de la fonction, afin de le retrouver plus tard à l'aide de la méthode `get_userData`.

**cellular**→**unmuteValueCallbacks()**

Réactive l'envoi de chaque changement de la valeur publiée au hub parent.

**cellular**→**wait\_async(callback, context)**

Attend que toutes les commandes asynchrones en cours d'exécution sur le module soient terminées, et appelle le callback passé en paramètre.

## YCellular.FindCellular() YCellular.FindCellular()

YCellular

Permet de retrouver une interface cellulaire d'après un identifiant donné.

js	function <b>yFindCellular</b> ( <b>func</b> )
cpp	YCellular* <b>FindCellular</b> ( string <b>func</b> )
m	+(YCellular*) <b>FindCellular</b> : (NSString*) <b>func</b>
pas	TYCellular <b>yFindCellular</b> ( <b>func</b> : string): TYCellular
vb	function <b>FindCellular</b> ( ByVal <b>func</b> As String) As YCellular
cs	static YCellular <b>FindCellular</b> ( string <b>func</b> )
java	static YCellular <b>FindCellular</b> ( String <b>func</b> )
uwp	static YCellular <b>FindCellular</b> ( string <b>func</b> )
py	<b>FindCellular</b> ( <b>func</b> )
php	function <b>FindCellular</b> ( <b>\$func</b> )
ts	static <b>FindCellular</b> ( <b>func</b> : string): YCellular
es	static <b>FindCellular</b> ( <b>func</b> )
dnp	static YCellularProxy <b>FindCellular</b> ( string <b>func</b> )
cp	static YCellularProxy * <b>FindCellular</b> ( string <b>func</b> )

L'identifiant peut être spécifié sous plusieurs formes:

- NomLogiqueFonction
- NoSerieModule.IdentifiantFonction
- NoSerieModule.NomLogiqueFonction
- NomLogiqueModule.IdentifiantMatériel
- NomLogiqueModule.NomLogiqueFonction

Cette fonction n'exige pas que l'interface cellulaire soit en ligne au moment où elle est appelée, l'objet retourné sera néanmoins valide. Utiliser la méthode `YCellular.isOnline()` pour tester si l'interface cellulaire est utilisable à un moment donné. En cas d'ambiguïté lorsqu'on fait une recherche par nom logique, aucune erreur ne sera notifiée: la première instance trouvée sera renvoyée. La recherche se fait d'abord par nom matériel, puis par nom logique.

Si un appel à la méthode `is_online()` de cet objet renvoie FAUX alors que vous êtes sûr que le module correspondant est bien branché, vérifiez que vous n'avez pas oublié d'appeler `registerHub()` à l'initialisation de de l'application.

### Paramètres :

**func** une chaîne de caractères qui référence l'interface cellulaire sans ambiguïté, par exemple `YHUBGSM1.cellular`.

### Retourne :

un objet de classe `YCellular` qui permet ensuite de contrôler l'interface cellulaire.

## YCellular.FindCellularInContext() YCellular.FindCellularInContext()

YCellular

Permet de retrouver une interface cellulaire d'après un identifiant donné dans un Context YAPI.

```
java static YCellular FindCellularInContext( YAPIContext yctx, String func)
uwp static YCellular FindCellularInContext( YAPIContext yctx, string func)
ts static FindCellularInContext( yctx: YAPIContext, func: string): YCellular
es static FindCellularInContext( yctx, func)
```

L'identifiant peut être spécifié sous plusieurs formes:

- NomLogiqueFonction
- NoSerieModule.IdentifiantFonction
- NoSerieModule.NomLogiqueFonction
- NomLogiqueModule.IdentifiantMatériel
- NomLogiqueModule.NomLogiqueFonction

Cette fonction n'exige pas que l'interface cellulaire soit en ligne au moment où elle est appelée, l'objet retourné sera néanmoins valide. Utiliser la méthode `YCellular.isOnline()` pour tester si l'interface cellulaire est utilisable à un moment donné. En cas d'ambiguïté lorsqu'on fait une recherche par nom logique, aucune erreur ne sera notifiée: la première instance trouvée sera renvoyée. La recherche se fait d'abord par nom matériel, puis par nom logique.

### Paramètres :

**yctx** un contexte YAPI

**func** une chaîne de caractères qui référence l'interface cellulaire sans ambiguïté, par exemple `YHUBGSM1.cellular`.

### Retourne :

un objet de classe `YCellular` qui permet ensuite de contrôler l'interface cellulaire.

## YCellular.FirstCellular() YCellular.FirstCellular()

YCellular

Commence l'énumération des interfaces réseau cellulaire accessibles par la librairie.

js	function <b>yFirstCellular</b> ( )
cpp	YCellular * <b>FirstCellular</b> ( )
m	+(YCellular*) <b>FirstCellular</b>
pas	TYCellular <b>yFirstCellular</b> ( ): TYCellular
vb	function <b>FirstCellular</b> ( ) As YCellular
cs	static YCellular <b>FirstCellular</b> ( )
java	static YCellular <b>FirstCellular</b> ( )
uwp	static YCellular <b>FirstCellular</b> ( )
py	<b>FirstCellular</b> ( )
php	function <b>FirstCellular</b> ( )
ts	static <b>FirstCellular</b> ( ): YCellular   null
es	static <b>FirstCellular</b> ( )

Utiliser la fonction `YCellular.nextCellular()` pour itérer sur les autres interfaces réseau cellulaire.

**Retourne :**

un pointeur sur un objet `YCellular`, correspondant à la première interface cellulaire accessible en ligne, ou `null` si il n'y a pas de interfaces réseau cellulaire disponibles.

## YCellular.FirstCellularInContext() YCellular.FirstCellularInContext()

YCellular

Commence l'énumération des interfaces réseau cellulaire accessibles par la librairie.

```
java static YCellular FirstCellularInContext( YAPIContext yctx)
uwp static YCellular FirstCellularInContext( YAPIContext yctx)
ts static FirstCellularInContext( yctx: YAPIContext): YCellular | null
es static FirstCellularInContext( yctx)
```

Utiliser la fonction `YCellular.nextCellular()` pour itérer sur les autres interfaces réseau cellulaire.

### Paramètres :

**yctx** un contexte YAPI.

### Retourne :

un pointeur sur un objet `YCellular`, correspondant à la première interface cellulaire accessible en ligne, ou `null` si il n'y a pas de interfaces réseau cellulaire disponibles.

## YCellular.GetSimilarFunctions() YCellular.GetSimilarFunctions()

YCellular

Enumère toutes les fonctions de type Cellular disponibles sur les modules actuellement joignables par la librairie, et retourne leurs identifiants matériels uniques (hardwareId).

```
dnf static new string[] GetSimilarFunctions( )
```

```
cp static vector<string> GetSimilarFunctions( )
```

Chaque chaîne retournée peut être passée en argument à la méthode `YCellular.FindCellular` pour obtenir un objet permettant d'interagir avec le module correspondant.

**Retourne :**

un tableau de chaînes de caractères, contenant les identifiants matériels de chaque fonction disponible trouvée.

## cellular → AdvertisedValue

YCellular

---

Courte chaîne de caractères représentant l'état courant de la fonction.

`dnsp` string **AdvertisedValue**

---

**cellular→Apn****YCellular**

---

Nom du point d'accès (APN) à utiliser, si nécessaire.

`dnf` `string Apn`

Lorsque l'APN est vide, celui proposé par l'opérateur cellulaire est utilisée.

**Modifiable.** Retourne le nom du point d'accès (APN) à utiliser, si nécessaire. Lorsque l'APN est vide, celui proposé par l'opérateur cellulaire est utilisée. N'oubliez pas d'appeler la méthode `saveToFlash()` du module si le réglage doit être préservé.

**cellular** → **CellOperator**

**YCellular**

---

Nom de l'opérateur de réseau cellulaire actuellement utilisé.

`dnsp` string **CellOperator**

**cellular** → **EnableData****YCellular**

Condition dans laquelle le service de données IP (GRPS) doit être activé.

`dnsp` `int EnableData`

Lorsque le service de donnée n'est pas actif, seules les communications par SMS sont possibles.

**Modifiable.** Le service peut être soit complètement désactivé, soit limité au réseau de de l'émetteur de la carte SIM, soit être activé pour tous les réseaux en partenariat avec la carte SIM (roaming). Attention, l'utilisation de données en roaming peut conduire à des coûts de télécommunication exorbitants !

Lorsque le service de donnée n'est pas actif, seules les communications par SMS sont possibles. N'oubliez pas d'appeler la méthode `saveToFlash()` du module si le réglage doit être préservé.

## cellular→FriendlyName

YCellular

---

Identifiant global de la fonction au format NOM\_MODULE . NOM\_FONCTION.

`dnsp` string **FriendlyName**

Le chaîne retournée utilise soit les noms logiques du module et de la fonction si ils sont définis, soit respectivement le numéro de série du module et l'identifiant matériel de la fonction (par exemple: `MyCustomName.relay1`)

---

**cellular**→**FunctionId****YCellular**

---

Identifiant matériel de l'interface cellulaire, sans référence au module.

`dnsp` string **FunctionId**

Par exemple `relay1`.

**cellular** → **HardwareId**

**YCellular**

---

Identifiant matériel unique de la fonction au format `SERIAL.FUNCTIONID`.

`dnsp` `string` **HardwareId**

L'identifiant unique est composé du numéro de série du module et de l'identifiant matériel de la fonction (par exemple `RELAYLO1-123456.relay1`).

**cellular→IsOnline****YCellular**

Vérifie si le module hébergeant la fonction est joignable, sans déclencher d'erreur.

`dnsp` `bool IsOnline`

Si les valeurs des attributs en cache de la fonction sont valides au moment de l'appel, le module est considéré joignable. Cette fonction ne cause en aucun cas d'exception, quelle que soit l'erreur qui pourrait se produire lors de la vérification de joignabilité.

## cellular → LinkQuality

YCellular

---

Qualité de la connexion, exprimée en pourcents.

`int` LinkQuality

**cellular** → **LockedOperator****YCellular**

Nom de l'opérateur de réseau cellulaire à utiliser exclusivement, si le choix automatique est désactivé, ou une chaîne vide si la carte SIM sélectionne automatiquement l'opérateur selon ceux disponibles.

`dnsp` string **LockedOperator**

**Modifiable.** Modifie le nom de l'opérateur de réseau cellulaire à utiliser. Si le nom est une chaîne vide, le choix sera fait automatiquement selon la carte SIM. Sinon, seul l'opérateur choisi sera utilisé. N'oubliez pas d'appeler la méthode `saveToFlash()` du module si le réglage doit être préservé.

## cellular→LogicalName

YCellular

---

Nom logique de la fonction.

```
dnf string LogicalName
```

**Modifiable.** Vous pouvez utiliser `yCheckLogicalName()` pour vérifier si votre paramètre est valide. N'oubliez pas d'appeler la méthode `saveToFlash()` du module si le réglage doit être préservé.

**cellular→Pin****YCellular**

String opaque si un code PIN a été configuré dans le module pour accéder à la carte SIM, ou une chaîne vide il n'a pas été configuré ou si la SIM a rejeté le code indiqué.

`dnv` **string Pin**

**Modifiable.** Modifie le code PIN utilisé par le module pour accéder à la carte SIM. Cette fonction ne change pas le code sur la carte SIM elle-même, mais uniquement le paramètre utilisé par le module pour essayer d'en obtenir l'accès. Si le code SIM ne fonctionne pas dès le premier essai, il sera automatiquement oublié et un message "Enter SIM PIN" apparaîtra dans l'attribut 'message'. Il faudra alors appeler à nouveau cette méthode avec le bon code PIN. Après trois essais infructueux consécutifs le message devient "Enter SIM PUK" et il faut alors entrer le code PUK de la carte SIM avec la méthode `sendPUK`.

N'oubliez pas d'appeler la méthode `saveToFlash()` du module pour que le paramètre soit sauvegardé dans la flash.

## cellular → PingInterval

YCellular

---

Intervalle entre les tests de connectivité spontanés, en secondes.

`int` `PingInterval`

**Modifiable.** N'oubliez pas d'appeler la méthode `saveToFlash()` du module si le réglage doit être préservé.

**cellular** → **RadioConfig****YCellular**

Type de protocole utilisé sur la communication série, sous forme d'une chaîne de caractères.

`dnv` `string` **RadioConfig**

Les valeurs possibles sont "Line" pour des messages ASCII séparés par des retours de ligne, "Frame:[timeout]ms" pour des messages binaires séparés par une temporisation, "Char" pour un flux ASCII continu ou "Byte" pour un flux binaire continu.

**Modifiable.** Modifie le type de protocole utilisé sur la communication série. Les valeurs possibles sont "Line" pour des messages ASCII séparés par des retours de ligne, "Frame:[timeout]ms" pour des messages binaires séparés par une temporisation, "Char" pour un flux ASCII continu ou "Byte" pour un flux binaire continu. Le suffixe "[wait]ms" peut être ajouté pour réduire la cadence d'émission de sorte à ce qu'il y ait au minimum le nombre spécifié de millisecondes d'intervalle entre l'envoi de chaque byte. N'oubliez pas d'appeler la méthode `saveToFlash()` du module si le réglage doit être préservé.

**cellular** → **SerialNumber**

**YCellular**

---

Numéro de série du module, préprogrammé en usine.

`dnsp` `string` **SerialNumber**

**cellular→\_AT()****YCellular**

Envoie une commande AT au module GSM, et retourne le résultat.

cpp	string _AT( string <b>cmd</b> )
m	-(NSString*) _AT : (NSString*) <b>cmd</b>
pas	string _AT( <b>cmd</b> : string): string
vb	function _AT( ByVal <b>cmd</b> As String) As String
cs	string _AT( string <b>cmd</b> )
java	String _AT( String <b>cmd</b> )
uwp	async Task<string> _AT( string <b>cmd</b> )
py	_AT( <b>cmd</b> )
php	function _AT( \$ <b>cmd</b> )
ts	async _AT( <b>cmd</b> : string): Promise<string>
es	async _AT( <b>cmd</b> )
dnp	string _AT( string <b>cmd</b> )
cp	string _AT( string <b>cmd</b> )
cmd	YCellular <b>target</b> _AT <b>cmd</b>

La commande ne s'exécute que lorsque le module GSM dans un état standard, et doit le laisser exactement dans le même état. N'utilisez cette fonction qu'avec la plus grande prudence !

**Paramètres :**

**cmd** la commande AT à exécuter, comme par exemple: "+CCLK?"

**Retourne :**

une chaîne de caractères contenant le résultat de la commande. Les lignes vides sont automatiquement filtrées du résultat.

**cellular** → **clearCache()**

YCellular

Invalide le cache.

js	function <b>clearCache</b> ( )
cpp	void <b>clearCache</b> ( )
m	-(void) <b>clearCache</b>
pas	<b>clearCache</b> ( )
vb	procedure <b>clearCache</b> ( )
cs	void <b>clearCache</b> ( )
java	void <b>clearCache</b> ( )
py	<b>clearCache</b> ( )
php	function <b>clearCache</b> ( )
ts	async <b>clearCache</b> ( ): Promise<void>
es	async <b>clearCache</b> ( )

Invalide le cache des valeurs courantes de l'interface cellulaire. Force le prochain appel à une méthode `get_xxx()` ou `loadxxx()` pour charger les les données depuis le module.

**cellular**→**clearDataCounters()****YCellular**

Réinitialise les compteurs de données transmises et reçues.

js	function <b>clearDataCounters</b> ( )
cpp	int <b>clearDataCounters</b> ( )
m	-(int) <b>clearDataCounters</b>
pas	LongInt <b>clearDataCounters</b> ( ): LongInt
vb	function <b>clearDataCounters</b> ( ) As Integer
cs	int <b>clearDataCounters</b> ( )
java	int <b>clearDataCounters</b> ( )
uwp	async Task<int> <b>clearDataCounters</b> ( )
py	<b>clearDataCounters</b> ( )
php	function <b>clearDataCounters</b> ( )
ts	async <b>clearDataCounters</b> ( ): Promise<number>
es	async <b>clearDataCounters</b> ( )
dnp	int <b>clearDataCounters</b> ( )
cp	int <b>clearDataCounters</b> ( )
cmd	YCellular <b>target clearDataCounters</b>

**Retourne :**

YAPI . SUCCESS si l'opération se déroule sans erreur.

En cas d'erreur, déclenche une exception ou retourne un code d'erreur négatif.

## cellular → decodePLMN()

YCellular

Retourne le nom usuel de l'opérateur cellulaire pour une paire MCC/MNC donnée.

js	function <b>decodePLMN</b> ( <b>mccmnc</b> )
cpp	string <b>decodePLMN</b> ( string <b>mccmnc</b> )
m	-(NSString*) <b>decodePLMN</b> : (NSString*) <b>mccmnc</b>
pas	string <b>decodePLMN</b> ( <b>mccmnc</b> : string): string
vb	function <b>decodePLMN</b> ( ByVal <b>mccmnc</b> As String) As String
cs	string <b>decodePLMN</b> ( string <b>mccmnc</b> )
java	String <b>decodePLMN</b> ( String <b>mccmnc</b> )
uwp	async Task<string> <b>decodePLMN</b> ( string <b>mccmnc</b> )
py	<b>decodePLMN</b> ( <b>mccmnc</b> )
php	function <b>decodePLMN</b> ( <b>\$mccmnc</b> )
ts	async <b>decodePLMN</b> ( <b>mccmnc</b> : string): Promise<string>
es	async <b>decodePLMN</b> ( <b>mccmnc</b> )
dnp	string <b>decodePLMN</b> ( string <b>mccmnc</b> )
cp	string <b>decodePLMN</b> ( string <b>mccmnc</b> )
cmd	YCellular <b>target decodePLMN</b> <b>mccmnc</b>

**Paramètres :**

**mccmnc** une chaîne de caractères commençant par un code MCC, suivi par un code MNC.

**Retourne :**

une chaîne contenant le nom usuel de l'opérateur correspondant.

**cellular→describe()****YCellular**

Retourne un court texte décrivant de manière non-ambigüe l'instance de l'interface cellulaire au format `TYPE (NAME) =SERIAL . FUNCTIONID`.

js	function <b>describe</b> ( )
cpp	string <b>describe</b> ( )
m	-(NSString*) <b>describe</b>
pas	string <b>describe</b> ( ): string
vb	function <b>describe</b> ( ) As String
cs	string <b>describe</b> ( )
java	String <b>describe</b> ( )
py	<b>describe</b> ( )
php	function <b>describe</b> ( )
ts	async <b>describe</b> ( ): Promise<string>
es	async <b>describe</b> ( )

Plus précisément, `TYPE` correspond au type de fonction, `NAME` correspond au nom utilisé lors du premier accès à la fonction, `SERIAL` correspond au numéro de série du module si le module est connecté, ou "unresolved" sinon, et `FUNCTIONID` correspond à l'identifiant matériel de la fonction si le module est connecté. Par exemple, La méthode va retourner `Relay(MyCustomName.relay1)=RELAYLO1-123456.relay1` si le module est déjà connecté ou `Relay(BadCustomeName.relay1)=unresolved` si le module n'est pas déjà connecté. Cette méthode ne déclenche aucune transaction USB ou TCP et peut donc être utilisé dans un debugueur.

**Retourne :**

une chaîne de caractères décrivant l'interface cellulaire (ex:  
`Relay(MyCustomName.relay1)=RELAYLO1-123456.relay1`)

**cellular**→**get\_advertisedValue()****YCellular****cellular**→**advertisedValue()**

Retourne la valeur courante de l'interface cellulaire (pas plus de 6 caractères).

js	function <b>get_advertisedValue</b> ( )
cpp	string <b>get_advertisedValue</b> ( )
m	-(NSString*) advertisedValue
pas	string <b>get_advertisedValue</b> ( ): string
vb	function <b>get_advertisedValue</b> ( ) As String
cs	string <b>get_advertisedValue</b> ( )
java	String <b>get_advertisedValue</b> ( )
uwp	async Task<string> <b>get_advertisedValue</b> ( )
py	<b>get_advertisedValue</b> ( )
php	function <b>get_advertisedValue</b> ( )
ts	async <b>get_advertisedValue</b> ( ): Promise<string>
es	async <b>get_advertisedValue</b> ( )
dnp	string <b>get_advertisedValue</b> ( )
cp	string <b>get_advertisedValue</b> ( )
cmd	YCellular <b>target</b> <b>get_advertisedValue</b>

**Retourne :**

une chaîne de caractères représentant la valeur courante de l'interface cellulaire (pas plus de 6 caractères).

En cas d'erreur, déclenche une exception ou retourne `YCellular.ADVERTISEDVALUE_INVALID`.

**cellular**→**get\_airplaneMode()****YCellular****cellular**→**airplaneMode()**

Retourne vrai si le mode avion est activé (radio désactivée).

js	function <b>get_airplaneMode</b> ( )
cpp	Y_AIRPLANEMODE_enum <b>get_airplaneMode</b> ( )
m	-(Y_AIRPLANEMODE_enum) airplaneMode
pas	Integer <b>get_airplaneMode</b> ( ): Integer
vb	function <b>get_airplaneMode</b> ( ) As Integer
cs	int <b>get_airplaneMode</b> ( )
java	int <b>get_airplaneMode</b> ( )
uwp	async Task<int> <b>get_airplaneMode</b> ( )
py	<b>get_airplaneMode</b> ( )
php	function <b>get_airplaneMode</b> ( )
ts	async <b>get_airplaneMode</b> ( ): Promise<YCellular_AirplaneMode>
es	async <b>get_airplaneMode</b> ( )
dnp	int <b>get_airplaneMode</b> ( )
cp	int <b>get_airplaneMode</b> ( )
cmd	YCellular <b>target get_airplaneMode</b>

**Retourne :**

soit `YCellular.AIRPLANEMODE_OFF`, soit `YCellular.AIRPLANEMODE_ON`, selon vrai si le mode avion est activé (radio désactivée)

En cas d'erreur, déclenche une exception ou retourne `YCellular.AIRPLANEMODE_INVALID`.

**cellular**→**get\_apn()****cellular**→**apn()**

Retourne le nom du point d'accès (APN) à utiliser, si nécessaire.

js	function <b>get_apn</b> ( )
cpp	string <b>get_apn</b> ( )
m	-(NSString*) apn
pas	string <b>get_apn</b> ( ): string
vb	function <b>get_apn</b> ( ) As String
cs	string <b>get_apn</b> ( )
java	String <b>get_apn</b> ( )
uwp	async Task<string> <b>get_apn</b> ( )
py	<b>get_apn</b> ( )
php	function <b>get_apn</b> ( )
ts	async <b>get_apn</b> ( ): Promise<string>
es	async <b>get_apn</b> ( )
dnp	string <b>get_apn</b> ( )
cp	string <b>get_apn</b> ( )
cmd	YCellular <b>target get_apn</b>

Lorsque l'APN est vide, celui proposé par l'opérateur cellulaire est utilisée.

**Retourne :**

une chaîne de caractères représentant le nom du point d'accès (APN) à utiliser, si nécessaire

En cas d'erreur, déclenche une exception ou retourne `YCellular.APN_INVALID`.

**cellular**→**get\_apnSecret()****YCellular****cellular**→**apnSecret()**

Retourne une string opaque si des paramètres d'identification sur l'APN ont été configurés dans le module, ou une chaîne vide autrement.

js	function <b>get_apnSecret</b> ( )
cpp	string <b>get_apnSecret</b> ( )
m	-(NSString*) apnSecret
pas	string <b>get_apnSecret</b> ( ): string
vb	function <b>get_apnSecret</b> ( ) As String
cs	string <b>get_apnSecret</b> ( )
java	String <b>get_apnSecret</b> ( )
uwp	async Task<string> <b>get_apnSecret</b> ( )
py	<b>get_apnSecret</b> ( )
php	function <b>get_apnSecret</b> ( )
ts	async <b>get_apnSecret</b> ( ): Promise<string>
es	async <b>get_apnSecret</b> ( )
dnp	string <b>get_apnSecret</b> ( )
cp	string <b>get_apnSecret</b> ( )
cmd	YCellular <b>target</b> <b>get_apnSecret</b>

Pour configurer ces paramètres, utilisez la méthode `set_apnAuth()`.

**Retourne :**

une chaîne de caractères représentant une string opaque si des paramètres d'identification sur l'APN ont été configurés dans le module, ou une chaîne vide autrement

En cas d'erreur, déclenche une exception ou retourne `YCellular.APNSECRET_INVALID`.

**cellular**→**get\_availableOperators()****cellular**→**availableOperators()**

Retourne la liste des opérateurs GSM disponibles à proximité.

js	function <b>get_availableOperators</b> ( )
cpp	vector<string> <b>get_availableOperators</b> ( )
m	-(NSMutableArray*) availableOperators
pas	TStringArray <b>get_availableOperators</b> ( ): TStringArray
vb	function <b>get_availableOperators</b> ( ) As List
cs	List<string> <b>get_availableOperators</b> ( )
java	ArrayList<String> <b>get_availableOperators</b> ( )
uwp	async Task<List<string>> <b>get_availableOperators</b> ( )
py	<b>get_availableOperators</b> ( )
php	function <b>get_availableOperators</b> ( )
ts	async <b>get_availableOperators</b> ( ): Promise<string[]
es	async <b>get_availableOperators</b> ( )
dnp	string[] <b>get_availableOperators</b> ( )
cp	vector<string> <b>get_availableOperators</b> ( )
cmd	YCellular <b>target</b> <b>get_availableOperators</b>

Cette fonction peut typiquement prendre 30 secondes à une minute pour rendre la main. Notez qu'en général une SIM ne permet de se connecter qu'à certains opérateur, et donc pas forcément à tous ceux listés par cette fonction.

**Retourne :**

une liste de noms d'opérateur.

**cellular**→**get\_cellIdentifier()****YCellular****cellular**→**cellIdentifier()**

Retourne l'identifiant unique de la station de base utilisée: MCC, MNC, LAC et Cell ID.

js	function <b>get_cellIdentifier</b> ( )
cpp	string <b>get_cellIdentifier</b> ( )
m	-(NSString*) cellIdentifier
pas	string <b>get_cellIdentifier</b> ( ): string
vb	function <b>get_cellIdentifier</b> ( ) As String
cs	string <b>get_cellIdentifier</b> ( )
java	String <b>get_cellIdentifier</b> ( )
uwp	async Task<string> <b>get_cellIdentifier</b> ( )
py	<b>get_cellIdentifier</b> ( )
php	function <b>get_cellIdentifier</b> ( )
ts	async <b>get_cellIdentifier</b> ( ): Promise<string>
es	async <b>get_cellIdentifier</b> ( )
dnp	string <b>get_cellIdentifier</b> ( )
cp	string <b>get_cellIdentifier</b> ( )
cmd	YCellular <b>target</b> <b>get_cellIdentifier</b>

**Retourne :**

une chaîne de caractères représentant l'identifiant unique de la station de base utilisée: MCC, MNC, LAC et Cell ID

En cas d'erreur, déclenche une exception ou retourne `YCellular.CELLIDENTIFIER_INVALID`.

**cellular**→**get\_cellOperator()****YCellular****cellular**→**cellOperator()**

Retourne le nom de l'opérateur de réseau cellulaire actuellement utilisé.

js	function <b>get_cellOperator</b> ( )
cpp	string <b>get_cellOperator</b> ( )
m	-(NSString*) <b>cellOperator</b>
pas	string <b>get_cellOperator</b> ( ): string
vb	function <b>get_cellOperator</b> ( ) As String
cs	string <b>get_cellOperator</b> ( )
java	String <b>get_cellOperator</b> ( )
uwp	async Task<string> <b>get_cellOperator</b> ( )
py	<b>get_cellOperator</b> ( )
php	function <b>get_cellOperator</b> ( )
ts	async <b>get_cellOperator</b> ( ): Promise<string>
es	async <b>get_cellOperator</b> ( )
dnp	string <b>get_cellOperator</b> ( )
cp	string <b>get_cellOperator</b> ( )
cmd	YCellular <b>target</b> <b>get_cellOperator</b>

**Retourne :**

une chaîne de caractères représentant le nom de l'opérateur de réseau cellulaire actuellement utilisé

En cas d'erreur, déclenche une exception ou retourne `YCellular.CELLOPERATOR_INVALID`.

**cellular**→**get\_cellType()****YCellular****cellular**→**cellType()**

Type de connection cellulaire active.

js	function <b>get_cellType</b> ( )
cpp	Y_CELLTYPE_enum <b>get_cellType</b> ( )
m	-(Y_CELLTYPE_enum) cellType
pas	Integer <b>get_cellType</b> ( ): Integer
vb	function <b>get_cellType</b> ( ) As Integer
cs	int <b>get_cellType</b> ( )
java	int <b>get_cellType</b> ( )
uwp	async Task<int> <b>get_cellType</b> ( )
py	<b>get_cellType</b> ( )
php	function <b>get_cellType</b> ( )
ts	async <b>get_cellType</b> ( ): Promise<YCellular_CellType>
es	async <b>get_cellType</b> ( )
dnp	int <b>get_cellType</b> ( )
cp	int <b>get_cellType</b> ( )
cmd	YCellular <b>target get_cellType</b>

**Retourne :**

une valeur parmi YCellular.CELLTYPE\_GPRS, YCellular.CELLTYPE\_EGPRS,  
YCellular.CELLTYPE\_WCDMA, YCellular.CELLTYPE\_HSDPA,  
YCellular.CELLTYPE\_NONE, YCellular.CELLTYPE\_CDMA,  
YCellular.CELLTYPE\_LTE\_M, YCellular.CELLTYPE\_NB\_IOT et  
YCellular.CELLTYPE\_EC\_GSM\_IOT

En cas d'erreur, déclenche une exception ou retourne YCellular.CELLTYPE\_INVALID.

**cellular**→**get\_dataReceived()**

YCellular

**cellular**→**dataReceived()**

Retourne le nombre d'octets reçus jusqu'à présent.

js	function <b>get_dataReceived</b> ( )
cpp	int <b>get_dataReceived</b> ( )
m	-(int) dataReceived
pas	LongInt <b>get_dataReceived</b> ( ): LongInt
vb	function <b>get_dataReceived</b> ( ) As Integer
cs	int <b>get_dataReceived</b> ( )
java	int <b>get_dataReceived</b> ( )
uwp	async Task<int> <b>get_dataReceived</b> ( )
py	<b>get_dataReceived</b> ( )
php	function <b>get_dataReceived</b> ( )
ts	async <b>get_dataReceived</b> ( ): Promise<number>
es	async <b>get_dataReceived</b> ( )
dnp	int <b>get_dataReceived</b> ( )
cp	int <b>get_dataReceived</b> ( )
cmd	YCellular <b>target</b> <b>get_dataReceived</b>

**Retourne :**

un entier représentant le nombre d'octets reçus jusqu'à présent

En cas d'erreur, déclenche une exception ou retourne `YCellular.DATARECEIVED_INVALID`.

**cellular**→**get\_dataSent()****YCellular****cellular**→**dataSent()**

Retourne le nombre d'octets envoyés jusqu'à présent.

js	function <b>get_dataSent</b> ( )
cpp	int <b>get_dataSent</b> ( )
m	-(int) dataSent
pas	LongInt <b>get_dataSent</b> ( ): LongInt
vb	function <b>get_dataSent</b> ( ) As Integer
cs	int <b>get_dataSent</b> ( )
java	int <b>get_dataSent</b> ( )
uwp	async Task<int> <b>get_dataSent</b> ( )
py	<b>get_dataSent</b> ( )
php	function <b>get_dataSent</b> ( )
ts	async <b>get_dataSent</b> ( ): Promise<number>
es	async <b>get_dataSent</b> ( )
dnp	int <b>get_dataSent</b> ( )
cp	int <b>get_dataSent</b> ( )
cmd	YCellular <b>target get_dataSent</b>

**Retourne :**

un entier représentant le nombre d'octets envoyés jusqu'à présent

En cas d'erreur, déclenche une exception ou retourne `YCellular.DATASENT_INVALID`.

**cellular**→**get\_enableData()****YCellular****cellular**→**enableData()**

Retourne la condition dans laquelle le service de données IP (GRPS) doit être activé.

js	function <b>get_enableData</b> ( )
cpp	Y_ENABLEDATA_enum <b>get_enableData</b> ( )
m	-(Y_ENABLEDATA_enum) enableData
pas	Integer <b>get_enableData</b> ( ): Integer
vb	function <b>get_enableData</b> ( ) As Integer
cs	int <b>get_enableData</b> ( )
java	int <b>get_enableData</b> ( )
uwp	async Task<int> <b>get_enableData</b> ( )
py	<b>get_enableData</b> ( )
php	function <b>get_enableData</b> ( )
ts	async <b>get_enableData</b> ( ): Promise<YCellular_EnableData>
es	async <b>get_enableData</b> ( )
dnp	int <b>get_enableData</b> ( )
cp	int <b>get_enableData</b> ( )
cmd	YCellular <b>target get_enableData</b>

Lorsque le service de donnée n'est pas actif, seules les communications par SMS sont possibles.

**Retourne :**

une valeur parmi `YCellular.ENABLEDATA_HOMENETWORK`, `YCellular.ENABLEDATA_ROAMING`, `YCellular.ENABLEDATA_NEVER` et `YCellular.ENABLEDATA_NEUTRALITY` représentant la condition dans laquelle le service de données IP (GRPS) doit être activé

En cas d'erreur, déclenche une exception ou retourne `YCellular.ENABLEDATA_INVALID`.

**cellular**→**get\_errorMessage()****YCellular****cellular**→**errorMessage()**

Retourne le message correspondant à la dernière erreur survenue lors de l'utilisation de l'interface cellulaire.

js	function <b>get_errorMessage</b> ( )
cpp	string <b>get_errorMessage</b> ( )
m	-(NSString*) <b>errorMessage</b>
pas	string <b>get_errorMessage</b> ( ): string
vb	function <b>get_errorMessage</b> ( ) As String
cs	string <b>get_errorMessage</b> ( )
java	String <b>get_errorMessage</b> ( )
py	<b>get_errorMessage</b> ( )
php	function <b>get_errorMessage</b> ( )
ts	<b>get_errorMessage</b> ( ): string
es	<b>get_errorMessage</b> ( )

Cette méthode est principalement utile lorsque la librairie Yoctopuce est utilisée en désactivant la gestion des exceptions.

**Retourne :**

une chaîne de caractères correspondant au message de la dernière erreur qui s'est produit lors de l'utilisation de l'interface cellulaire.

**cellular**→**get\_errorType()****YCellular****cellular**→**errorType()**

Retourne le code d'erreur correspondant à la dernière erreur survenue lors de l'utilisation de l'interface cellulaire.

js	function <b>get_errorType</b> ( )
cpp	YRETCODE <b>get_errorType</b> ( )
m	-(YRETCODE) errorType
pas	YRETCODE <b>get_errorType</b> ( ): YRETCODE
vb	function <b>get_errorType</b> ( ) As YRETCODE
cs	YRETCODE <b>get_errorType</b> ( )
java	int <b>get_errorType</b> ( )
py	<b>get_errorType</b> ( )
php	function <b>get_errorType</b> ( )
ts	<b>get_errorType</b> ( ): number
es	<b>get_errorType</b> ( )

Cette méthode est principalement utile lorsque la librairie Yoctopuce est utilisée en désactivant la gestion des exceptions.

**Retourne :**

un nombre correspondant au code de la dernière erreur qui s'est produit lors de l'utilisation de l'interface cellulaire.

**cellular**→**get\_friendlyName()****YCellular****cellular**→**friendlyName()**

Retourne un identifiant global de l'interface cellulaire au format `NOM_MODULE . NOM_FONCTION`.

js	function <b>get_friendlyName</b> ( )
cpp	string <b>get_friendlyName</b> ( )
m	-(NSString*) friendlyName
cs	string <b>get_friendlyName</b> ( )
java	String <b>get_friendlyName</b> ( )
py	<b>get_friendlyName</b> ( )
php	function <b>get_friendlyName</b> ( )
ts	async <b>get_friendlyName</b> ( ): Promise<string>
es	async <b>get_friendlyName</b> ( )
dnp	string <b>get_friendlyName</b> ( )
cp	string <b>get_friendlyName</b> ( )

Le chaîne retournée utilise soit les noms logiques du module et de l'interface cellulaire si ils sont définis, soit respectivement le numéro de série du module et l'identifiant matériel de l'interface cellulaire (par exemple: `MyCustomName.relay1`)

**Retourne :**

une chaîne de caractères identifiant l'interface cellulaire en utilisant les noms logiques (ex: `MyCustomName.relay1`)

En cas d'erreur, déclenche une exception ou retourne `YCellular.FRIENDLYNAME_INVALID`.

**cellular**→**get\_functionDescriptor()**

YCellular

**cellular**→**functionDescriptor()**

Retourne un identifiant unique de type YFUN\_DESCR correspondant à la fonction.

js	function <b>get_functionDescriptor</b> ( )
cpp	YFUN_DESCR <b>get_functionDescriptor</b> ( )
m	-(YFUN_DESCR) functionDescriptor
pas	YFUN_DESCR <b>get_functionDescriptor</b> ( ): YFUN_DESCR
vb	function <b>get_functionDescriptor</b> ( ) As YFUN_DESCR
cs	YFUN_DESCR <b>get_functionDescriptor</b> ( )
java	String <b>get_functionDescriptor</b> ( )
py	<b>get_functionDescriptor</b> ( )
php	function <b>get_functionDescriptor</b> ( )
ts	async <b>get_functionDescriptor</b> ( ): Promise<string>
es	async <b>get_functionDescriptor</b> ( )

Cet identifiant peut être utilisé pour tester si deux instance de YFunction référencent physiquement la même fonction sur le même module.

**Retourne :**

un identifiant de type YFUN\_DESCR.

Si la fonction n'a jamais été contactée, la valeur retournée sera Y\$CLASSNAME\$.FUNCTIONDESCRIPTOR\_INVALID

**cellular**→**get\_functionId()****YCellular****cellular**→**functionId()**

Retourne l'identifiant matériel de l'interface cellulaire, sans référence au module.

js	function <b>get_functionId</b> ( )
cpp	string <b>get_functionId</b> ( )
m	-(NSString*) <b>functionId</b>
vb	function <b>get_functionId</b> ( ) As String
cs	string <b>get_functionId</b> ( )
java	String <b>get_functionId</b> ( )
py	<b>get_functionId</b> ( )
php	function <b>get_functionId</b> ( )
ts	async <b>get_functionId</b> ( ): Promise<string>
es	async <b>get_functionId</b> ( )
dnp	string <b>get_functionId</b> ( )
cp	string <b>get_functionId</b> ( )

Par exemple `relay1`.

**Retourne :**

une chaîne de caractères identifiant l'interface cellulaire (ex: `relay1`)

En cas d'erreur, déclenche une exception ou retourne `YCellular.FUNCTIONID_INVALID`.

**cellular**→**get\_hardwareId()****cellular**→**hardwareId()**

Retourne l'identifiant matériel unique de l'interface cellulaire au format `SERIAL.FUNCTIONID`.

js	function <b>get_hardwareId</b> ( )
cpp	string <b>get_hardwareId</b> ( )
m	-(NSString*) hardwareId
vb	function <b>get_hardwareId</b> ( ) As String
cs	string <b>get_hardwareId</b> ( )
java	String <b>get_hardwareId</b> ( )
py	<b>get_hardwareId</b> ( )
php	function <b>get_hardwareId</b> ( )
ts	async <b>get_hardwareId</b> ( ): Promise<string>
es	async <b>get_hardwareId</b> ( )
dnp	string <b>get_hardwareId</b> ( )
cp	string <b>get_hardwareId</b> ( )

L'identifiant unique est composé du numéro de série du module et de l'identifiant matériel de l'interface cellulaire (par exemple `RELAYLO1-123456.relay1`).

**Retourne :**

une chaîne de caractères identifiant l'interface cellulaire (ex: `RELAYLO1-123456.relay1`)

En cas d'erreur, déclenche une exception ou retourne `YCellular.HARDWAREID_INVALID`.

**cellular**→**get\_imsi()****YCellular****cellular**→**imsi()**

Retourne le "International Mobile Subscriber Identity" (MSI) qui identifie de manière unique la carte SIM.

js	function <b>get_imsi</b> ( )
cpp	string <b>get_imsi</b> ( )
m	-(NSString*) <b>imsi</b>
pas	string <b>get_imsi</b> ( ): string
vb	function <b>get_imsi</b> ( ) As String
cs	string <b>get_imsi</b> ( )
java	String <b>get_imsi</b> ( )
uwp	async Task<string> <b>get_imsi</b> ( )
py	<b>get_imsi</b> ( )
php	function <b>get_imsi</b> ( )
ts	async <b>get_imsi</b> ( ): Promise<string>
es	async <b>get_imsi</b> ( )
dnp	string <b>get_imsi</b> ( )
cp	string <b>get_imsi</b> ( )
cmd	YCellular <b>target</b> <b>get_imsi</b>

Les trois premiers chiffres représentent le code du pays (MCC), suivi par le code du réseau émetteur de la SIM, sur 2 ou 3 chiffres selon la zone géographique.

**Retourne :**

une chaîne de caractères représentant le "International Mobile Subscriber Identity" (MSI) qui identifie de manière unique la carte SIM

En cas d'erreur, déclenche une exception ou retourne `YCellular.IMSI_INVALID`.

**cellular**→**get\_linkQuality()****YCellular****cellular**→**linkQuality()**

Retourne la qualité de la connexion, exprimée en pourcents.

js	function <b>get_linkQuality</b> ( )
cpp	int <b>get_linkQuality</b> ( )
m	-(int) linkQuality
pas	LongInt <b>get_linkQuality</b> ( ): LongInt
vb	function <b>get_linkQuality</b> ( ) As Integer
cs	int <b>get_linkQuality</b> ( )
java	int <b>get_linkQuality</b> ( )
uwp	async Task<int> <b>get_linkQuality</b> ( )
py	<b>get_linkQuality</b> ( )
php	function <b>get_linkQuality</b> ( )
ts	async <b>get_linkQuality</b> ( ): Promise<number>
es	async <b>get_linkQuality</b> ( )
dnp	int <b>get_linkQuality</b> ( )
cp	int <b>get_linkQuality</b> ( )
cmd	YCellular <b>target</b> <b>get_linkQuality</b>

**Retourne :**

un entier représentant la qualité de la connexion, exprimée en pourcents

En cas d'erreur, déclenche une exception ou retourne `YCellular.LINKQUALITY_INVALID`.

**cellular**→**get\_lockedOperator()****YCellular****cellular**→**lockedOperator()**

Retourne le nom de l'opérateur de réseau cellulaire à utiliser exclusivement, si le choix automatique est désactivé, ou une chaîne vide si la carte SIM sélectionne automatiquement l'opérateur selon ceux disponibles.

js	function <b>get_lockedOperator</b> ( )
cpp	string <b>get_lockedOperator</b> ( )
m	-(NSString*) <b>lockedOperator</b>
pas	string <b>get_lockedOperator</b> ( ): string
vb	function <b>get_lockedOperator</b> ( ) As String
cs	string <b>get_lockedOperator</b> ( )
java	String <b>get_lockedOperator</b> ( )
uwp	async Task<string> <b>get_lockedOperator</b> ( )
py	<b>get_lockedOperator</b> ( )
php	function <b>get_lockedOperator</b> ( )
ts	async <b>get_lockedOperator</b> ( ): Promise<string>
es	async <b>get_lockedOperator</b> ( )
dnp	string <b>get_lockedOperator</b> ( )
cp	string <b>get_lockedOperator</b> ( )
cmd	YCellular <b>target get_lockedOperator</b>

**Retourne :**

une chaîne de caractères représentant le nom de l'opérateur de réseau cellulaire à utiliser exclusivement, si le choix automatique est désactivé, ou une chaîne vide si la carte SIM sélectionne automatiquement l'opérateur selon ceux disponibles

En cas d'erreur, déclenche une exception ou retourne `YCellular.LOCKEDOPERATOR_INVALID`.

**cellular**→**get\_logicalName()****YCellular****cellular**→**logicalName()**

Retourne le nom logique de l'interface cellulaire.

js	function <b>get_logicalName</b> ( )
cpp	string <b>get_logicalName</b> ( )
m	-(NSString*) logicalName
pas	string <b>get_logicalName</b> ( ): string
vb	function <b>get_logicalName</b> ( ) As String
cs	string <b>get_logicalName</b> ( )
java	String <b>get_logicalName</b> ( )
uwp	async Task<string> <b>get_logicalName</b> ( )
py	<b>get_logicalName</b> ( )
php	function <b>get_logicalName</b> ( )
ts	async <b>get_logicalName</b> ( ): Promise<string>
es	async <b>get_logicalName</b> ( )
dnp	string <b>get_logicalName</b> ( )
cp	string <b>get_logicalName</b> ( )
cmd	YCellular <b>target</b> <b>get_logicalName</b>

**Retourne :**

une chaîne de caractères représentant le nom logique de l'interface cellulaire.

En cas d'erreur, déclenche une exception ou retourne `YCellular.LOGICALNAME_INVALID`.

**cellular**→**get\_message()****YCellular****cellular**→**message()**

Retourne le dernier message de diagnostic de l'interface au réseau sans fil.

js	function <b>get_message</b> ( )
cpp	string <b>get_message</b> ( )
m	-(NSString*) message
pas	string <b>get_message</b> ( ): string
vb	function <b>get_message</b> ( ) As String
cs	string <b>get_message</b> ( )
java	String <b>get_message</b> ( )
uwp	async Task<string> <b>get_message</b> ( )
py	<b>get_message</b> ( )
php	function <b>get_message</b> ( )
ts	async <b>get_message</b> ( ): Promise<string>
es	async <b>get_message</b> ( )
dnp	string <b>get_message</b> ( )
cp	string <b>get_message</b> ( )
cmd	YCellular <b>target get_message</b>

**Retourne :**

une chaîne de caractères représentant le dernier message de diagnostic de l'interface au réseau sans fil

En cas d'erreur, déclenche une exception ou retourne `YCellular.MESSAGE_INVALID`.

**cellular**→**get\_module()****cellular**→**module()**

Retourne l'objet `YModule` correspondant au module Yoctopuce qui héberge la fonction.

js	function <b>get_module</b> ( )
cpp	<code>YModule *</code> <b>get_module</b> ( )
m	-( <code>YModule*</code> ) module
pas	<code>TYModule</code> <b>get_module</b> ( ): <code>TYModule</code>
vb	function <b>get_module</b> ( ) As <code>YModule</code>
cs	<code>YModule</code> <b>get_module</b> ( )
java	<code>YModule</code> <b>get_module</b> ( )
py	<b>get_module</b> ( )
php	function <b>get_module</b> ( )
ts	async <b>get_module</b> ( ): <code>Promise&lt;YModule&gt;</code>
es	async <b>get_module</b> ( )
dnp	<code>YModuleProxy</code> <b>get_module</b> ( )
cp	<code>YModuleProxy *</code> <b>get_module</b> ( )

Si la fonction ne peut être trouvée sur aucun module, l'instance de `YModule` retournée ne sera pas joignable.

**Retourne :**

une instance de `YModule`

**cellular**→**get\_module\_async()****YCellular****cellular**→**module\_async()**

Retourne l'objet `YModule` correspondant au module Yoctopuce qui héberge la fonction.

```
js function get_module_async( callback, context)
```

Si la fonction ne peut être trouvée sur aucun module, l'instance de `YModule` retournée ne sera pas joignable.

Cette version asynchrone n'existe qu'en Javascript. Elle utilise une fonction de callback plutôt qu'une simple valeur de retour, pour éviter de bloquer la VM Javascript de Firefox, qui n'implémente pas le passage de contrôle entre threads durant les appels d'entrée/sortie bloquants.

**Paramètres :**

**callback** fonction de callback qui sera appelée dès que le résultat sera connu. La fonction callback reçoit trois arguments: le contexte fourni par l'appelant, l'objet fonction concerné et l'instance demandée de `YModule`

**context** contexte fourni par l'appelant, et qui sera passé tel-quel à la fonction de callback

**Retourne :**

rien du tout : le résultat sera passé en paramètre à la fonction de callback.

**cellular**→**get\_pin()****cellular**→**pin()**

Retourne une string opaque si un code PIN a été configuré dans le module pour accéder à la carte SIM, ou une chaîne vide il n'a pas été configuré ou si la SIM a rejeté le code indiqué.

js	function <b>get_pin</b> ( )
cpp	string <b>get_pin</b> ( )
m	-(NSString*) pin
pas	string <b>get_pin</b> ( ): string
vb	function <b>get_pin</b> ( ) As String
cs	string <b>get_pin</b> ( )
java	String <b>get_pin</b> ( )
uwp	async Task<string> <b>get_pin</b> ( )
py	<b>get_pin</b> ( )
php	function <b>get_pin</b> ( )
ts	async <b>get_pin</b> ( ): Promise<string>
es	async <b>get_pin</b> ( )
dnp	string <b>get_pin</b> ( )
cp	string <b>get_pin</b> ( )
cmd	YCellular <b>target get_pin</b>

**Retourne :**

une chaîne de caractères représentant une string opaque si un code PIN a été configuré dans le module pour accéder à la carte SIM, ou une chaîne vide il n'a pas été configuré ou si la SIM a rejeté le code indiqué

En cas d'erreur, déclenche une exception ou retourne `YCellular.PIN_INVALID`.

**cellular**→**get\_pingInterval()****YCellular****cellular**→**pingInterval()**

Retourne l'intervalle entre les tests de connectivité spontanés, en secondes.

js	function <b>get_pingInterval</b> ( )
cpp	int <b>get_pingInterval</b> ( )
m	-(int) pingInterval
pas	LongInt <b>get_pingInterval</b> ( ): LongInt
vb	function <b>get_pingInterval</b> ( ) As Integer
cs	int <b>get_pingInterval</b> ( )
java	int <b>get_pingInterval</b> ( )
uwp	async Task<int> <b>get_pingInterval</b> ( )
py	<b>get_pingInterval</b> ( )
php	function <b>get_pingInterval</b> ( )
ts	async <b>get_pingInterval</b> ( ): Promise<number>
es	async <b>get_pingInterval</b> ( )
dnp	int <b>get_pingInterval</b> ( )
cp	int <b>get_pingInterval</b> ( )
cmd	YCellular <b>target</b> <b>get_pingInterval</b>

**Retourne :**

un entier représentant l'intervalle entre les tests de connectivité spontanés, en secondes

En cas d'erreur, déclenche une exception ou retourne `YCellular.PINGINTERVAL_INVALID`.

**cellular**→**get\_radioConfig()****YCellular****cellular**→**radioConfig()**

Retourne le type de protocole utilisé sur la communication série, sous forme d'une chaîne de caractères.

js	function <b>get_radioConfig</b> ( )
cpp	string <b>get_radioConfig</b> ( )
m	-(NSString*) radioConfig
pas	string <b>get_radioConfig</b> ( ): string
vb	function <b>get_radioConfig</b> ( ) As String
cs	string <b>get_radioConfig</b> ( )
java	String <b>get_radioConfig</b> ( )
uwp	async Task<string> <b>get_radioConfig</b> ( )
py	<b>get_radioConfig</b> ( )
php	function <b>get_radioConfig</b> ( )
ts	async <b>get_radioConfig</b> ( ): Promise<string>
es	async <b>get_radioConfig</b> ( )
dnp	string <b>get_radioConfig</b> ( )
cp	string <b>get_radioConfig</b> ( )
cmd	YCellular <b>target get_radioConfig</b>

Les valeurs possibles sont "Line" pour des messages ASCII séparés par des retours de ligne, "Frame:[timeout]ms" pour des messages binaires séparés par une temporisation, "Char" pour un flux ASCII continu ou "Byte" pour un flux binaire continue.

**Retourne :**

une chaîne de caractères représentant le type de protocole utilisé sur la communication série, sous forme d'une chaîne de caractères

En cas d'erreur, déclenche une exception ou retourne `YCellular.RADIOCONFIG_INVALID`.

**cellular**→**get\_serialNumber()****YCellular****cellular**→**serialNumber()**

Retourne le numéro de série du module, préprogrammé en usine.

js	function <b>get_serialNumber</b> ( )
cpp	string <b>get_serialNumber</b> ( )
m	-(NSString*) <b>serialNumber</b>
pas	string <b>get_serialNumber</b> ( ): string
vb	function <b>get_serialNumber</b> ( ) As String
cs	string <b>get_serialNumber</b> ( )
java	String <b>get_serialNumber</b> ( )
uwp	async Task<string> <b>get_serialNumber</b> ( )
py	<b>get_serialNumber</b> ( )
php	function <b>get_serialNumber</b> ( )
ts	async <b>get_serialNumber</b> ( ): Promise<string>
es	async <b>get_serialNumber</b> ( )
dnp	string <b>get_serialNumber</b> ( )
cp	string <b>get_serialNumber</b> ( )
cmd	YCellular <b>target</b> <b>get_serialNumber</b>

**Retourne :**

: une chaîne de caractères représentant le numéro de série du module, préprogrammé en usine.

En cas d'erreur, déclenche une exception ou retourne YFunction.SERIALNUMBER\_INVALID.

**cellular**→**get\_userData()****cellular**→**userData()**

Retourne le contenu de l'attribut `userData`, précédemment stocké à l'aide de la méthode `set_userData`.

js	function <b>get_userData</b> ( )
cpp	void * <b>get_userData</b> ( )
m	-(id) userData
pas	Tobject <b>get_userData</b> ( ): Tobject
vb	function <b>get_userData</b> ( ) As Object
cs	object <b>get_userData</b> ( )
java	Object <b>get_userData</b> ( )
py	<b>get_userData</b> ( )
php	function <b>get_userData</b> ( )
ts	async <b>get_userData</b> ( ): Promise<object null>
es	async <b>get_userData</b> ( )

Cet attribut n'est pas utilisé directement par l'API. Il est à la disposition de l'appelant pour stocker un contexte.

**Retourne :**

l'objet stocké précédemment par l'appelant.

**cellular**→**isOnline()****YCellular**

Vérifie si le module hébergeant l'interface cellulaire est joignable, sans déclencher d'erreur.

js	function <b>isOnline</b> ( )
cpp	bool <b>isOnline</b> ( )
m	-(BOOL) <b>isOnline</b>
pas	boolean <b>isOnline</b> ( ): boolean
vb	function <b>isOnline</b> ( ) As Boolean
cs	bool <b>isOnline</b> ( )
java	boolean <b>isOnline</b> ( )
py	<b>isOnline</b> ( )
php	function <b>isOnline</b> ( )
ts	async <b>isOnline</b> ( ): Promise<boolean>
es	async <b>isOnline</b> ( )
dnp	bool <b>isOnline</b> ( )
cp	bool <b>isOnline</b> ( )

Si les valeurs des attributs en cache de l'interface cellulaire sont valides au moment de l'appel, le module est considéré joignable. Cette fonction ne cause en aucun cas d'exception, quelle que soit l'erreur qui pourrait se produire lors de la vérification de joignabilité.

**Retourne :**

`true` si l'interface cellulaire est joignable, `false` sinon

**cellular**→**isOnline\_async()****YCellular**

Vérifie si le module hébergeant l'interface cellulaire est joignable, sans déclencher d'erreur.

```
js fonction isOnline_async( callback, context)
```

Si les valeurs des attributs en cache de l'interface cellulaire sont valides au moment de l'appel, le module est considéré joignable. Cette fonction ne cause en aucun cas d'exception, quelle que soit l'erreur qui pourrait se produire lors de la vérification de joignabilité.

Cette version asynchrone n'existe qu'en Javascript. Elle utilise une fonction de callback plutôt qu'une simple valeur de retour, pour éviter de bloquer la machine virtuelle Javascript avec une attente active.

**Paramètres :**

**callback** fonction de callback qui sera appelée dès que le résultat sera connu. La fonction callback reçoit trois arguments: le contexte fourni par l'appelant, l'objet fonction concerné et le résultat booléen

**context** contexte fourni par l'appelant, et qui sera passé tel-quel à la fonction de callback

**Retourne :**

rien du tout : le résultat sera passé en paramètre à la fonction de callback.

**cellular→isReadOnly()****YCellular**

Test si la fonction est en lecture seule.

cpp	bool <b>isReadOnly</b> ( )
m	-(bool) <b>isReadOnly</b>
pas	boolean <b>isReadOnly</b> ( ): boolean
vb	function <b>isReadOnly</b> ( ) As Boolean
cs	bool <b>isReadOnly</b> ( )
java	boolean <b>isReadOnly</b> ( )
uwp	async Task<bool> <b>isReadOnly</b> ( )
py	<b>isReadOnly</b> ( )
php	function <b>isReadOnly</b> ( )
ts	async <b>isReadOnly</b> ( ): Promise<boolean>
es	async <b>isReadOnly</b> ( )
dnp	bool <b>isReadOnly</b> ( )
cp	bool <b>isReadOnly</b> ( )
cmd	YCellular <b>target isReadOnly</b>

Retourne vrais si la fonction est protégé en ecriture ou que la fontion n'est pas disponible.

**Retourne :**

`true` si la fonction est protégé en ecriture ou que la fontion n'est pas disponible

## cellular→load()

YCellular

Met en cache les valeurs courantes de l'interface cellulaire, avec une durée de validité spécifiée.

js	function <b>load</b> ( <b>msValidity</b> )
cpp	YRETCODE <b>load</b> ( int <b>msValidity</b> )
m	-(YRETCODE) <b>load</b> : (u64) <b>msValidity</b>
pas	YRETCODE <b>load</b> ( <b>msValidity</b> : u64): YRETCODE
vb	function <b>load</b> ( ByVal <b>msValidity</b> As Long) As YRETCODE
cs	YRETCODE <b>load</b> ( ulong <b>msValidity</b> )
java	int <b>load</b> ( long <b>msValidity</b> )
py	<b>load</b> ( <b>msValidity</b> )
php	function <b>load</b> ( <b>\$msValidity</b> )
ts	async <b>load</b> ( <b>msValidity</b> : number): Promise<number>
es	async <b>load</b> ( <b>msValidity</b> )

Par défaut, lorsqu'on accède à un module, tous les attributs des fonctions du module sont automatiquement mises en cache pour la durée standard (5 ms). Cette méthode peut être utilisée pour marquer occasionnellement les données cachées comme valides pour une plus longue période, par exemple dans le but de réduire le trafic réseau.

**Paramètres :**

**msValidity** un entier correspondant à la durée de validité attribuée aux les paramètres chargés, en millisecondes

**Retourne :**

YAPI . SUCCESS si l'opération se déroule sans erreur.

En cas d'erreur, déclenche une exception ou retourne un code d'erreur négatif.

**cellular**→**loadAttribute()****YCellular**

Retourne la valeur actuelle d'un attribut spécifique de la fonction, sous forme de texte, le plus rapidement possible mais sans passer par le cache.

js	function <b>loadAttribute</b> ( <b>attrName</b> )
cpp	string <b>loadAttribute</b> ( string <b>attrName</b> )
m	-(NSString*) <b>loadAttribute</b> : (NSString*) <b>attrName</b>
pas	string <b>loadAttribute</b> ( <b>attrName</b> : string): string
vb	function <b>loadAttribute</b> ( ByVal <b>attrName</b> As String) As String
cs	string <b>loadAttribute</b> ( string <b>attrName</b> )
java	String <b>loadAttribute</b> ( String <b>attrName</b> )
uwp	async Task<string> <b>loadAttribute</b> ( string <b>attrName</b> )
py	<b>loadAttribute</b> ( <b>attrName</b> )
php	function <b>loadAttribute</b> ( <b>\$attrName</b> )
ts	async <b>loadAttribute</b> ( <b>attrName</b> : string): Promise<string>
es	async <b>loadAttribute</b> ( <b>attrName</b> )
dnb	string <b>loadAttribute</b> ( string <b>attrName</b> )
cp	string <b>loadAttribute</b> ( string <b>attrName</b> )

**Paramètres :**

**attrName** le nom de l'attribut désiré

**Retourne :**

une chaîne de caractères représentant la valeur actuelle de l'attribut.

En cas d'erreur, déclenche une exception ou retourne un chaîne vide.

**cellular**→**load\_async()****YCellular**

Met en cache les valeurs courantes de l'interface cellulaire, avec une durée de validité spécifiée.

```
js fonction load_async( msValidity, callback, context)
```

Par défaut, lorsqu'on accède à un module, tous les attributs des fonctions du module sont automatiquement mises en cache pour la durée standard (5 ms). Cette méthode peut être utilisée pour marquer occasionnellement les données cachées comme valides pour une plus longue période, par exemple dans le but de réduire le trafic réseau.

Cette version asynchrone n'existe qu'en Javascript. Elle utilise une fonction de callback plutôt qu'une simple valeur de retour, pour éviter de bloquer la machine virtuelle Javascript avec une attente active.

**Paramètres :**

- msValidity** un entier correspondant à la durée de validité attribuée aux les paramètres chargés, en millisecondes
- callback** fonction de callback qui sera appelée dès que le résultat sera connu. La fonction callback reçoit trois arguments: le contexte fourni par l'appelant, l'objet fonction concerné et le code d'erreur (ou YAPI . SUCCESS)
- context** contexte fourni par l'appelant, et qui sera passé tel-quel à la fonction de callback

**Retourne :**

rien du tout : le résultat sera passé en paramètre à la fonction de callback.

**cellular**→**muteValueCallbacks()****YCellular**

Désactive l'envoi de chaque changement de la valeur publiée au hub parent.

js	function <b>muteValueCallbacks</b> ( )
cpp	int <b>muteValueCallbacks</b> ( )
m	-(int) <b>muteValueCallbacks</b>
pas	LongInt <b>muteValueCallbacks</b> ( ): LongInt
vb	function <b>muteValueCallbacks</b> ( ) As Integer
cs	int <b>muteValueCallbacks</b> ( )
java	int <b>muteValueCallbacks</b> ( )
uwp	async Task<int> <b>muteValueCallbacks</b> ( )
py	<b>muteValueCallbacks</b> ( )
php	function <b>muteValueCallbacks</b> ( )
ts	async <b>muteValueCallbacks</b> ( ): Promise<number>
es	async <b>muteValueCallbacks</b> ( )
dnp	int <b>muteValueCallbacks</b> ( )
cp	int <b>muteValueCallbacks</b> ( )
cmd	YCellular <b>target</b> <b>muteValueCallbacks</b>

Vous pouvez utiliser cette fonction pour économiser la bande passante et le CPU sur les machines de faible puissance, ou pour éviter le déclenchement de callbacks HTTP. N'oubliez pas d'appeler la méthode `saveToFlash()` du module si le réglage doit être préservé.

**Retourne :**

YAPI . SUCCESS si l'opération se déroule sans erreur.

En cas d'erreur, déclenche une exception ou retourne un code d'erreur négatif.

**cellular** → **nextCellular()****YCellular**

Continue l'énumération des interfaces réseau cellulaire commencée à l'aide de `yFirstCellular()`. Attention, vous ne pouvez faire aucune supposition sur l'ordre dans lequel les interfaces réseau cellulaire sont retournés.

js	function <b>nextCellular</b> ( )
cpp	YCellular * <b>nextCellular</b> ( )
m	-(nullable YCellular*) <b>nextCellular</b>
pas	TYCellular <b>nextCellular</b> ( ): TYCellular
vb	function <b>nextCellular</b> ( ) As YCellular
cs	YCellular <b>nextCellular</b> ( )
java	YCellular <b>nextCellular</b> ( )
uwp	YCellular <b>nextCellular</b> ( )
py	<b>nextCellular</b> ( )
php	function <b>nextCellular</b> ( )
ts	<b>nextCellular</b> ( ): YCellular   null
es	<b>nextCellular</b> ( )

Si vous souhaitez retrouver une interface cellulaire spécifique, utilisez `Cellular.findCellular()` avec un `hardwareID` ou un nom logique.

**Retourne :**

un pointeur sur un objet `YCellular` accessible en ligne, ou `null` lorsque l'énumération est terminée.

**cellular**→**quickCellSurvey()****YCellular**

Retourne la liste d'identifiants pour les antennes GSM à proximité, telle que requise pour géolocaliser rapidement le module.

js	function <b>quickCellSurvey</b> ( )
cpp	vector<YCellRecord> <b>quickCellSurvey</b> ( )
m	-(NSMutableArray*) <b>quickCellSurvey</b>
pas	TYCellRecordArray <b>quickCellSurvey</b> ( ): TYCellRecordArray
vb	function <b>quickCellSurvey</b> ( ) As List
cs	List<YCellRecord> <b>quickCellSurvey</b> ( )
java	ArrayList<YCellRecord> <b>quickCellSurvey</b> ( )
uwp	async Task<List<YCellRecord>> <b>quickCellSurvey</b> ( )
py	<b>quickCellSurvey</b> ( )
php	function <b>quickCellSurvey</b> ( )
ts	async <b>quickCellSurvey</b> ( ): Promise<YCellRecord[]
es	async <b>quickCellSurvey</b> ( )
dnx	YCellRecordProxy[] <b>quickCellSurvey</b> ( )
cp	vector<YCellRecordProxy> <b>quickCellSurvey</b> ( )
cmd	YCellular <b>target quickCellSurvey</b>

La première antenne listée est la cellule active, et les suivantes sont les cellules voisines listée par la cellule active.

**Retourne :**

une liste de YCellRecord.

**cellular** → **registerValueCallback()****YCellular**

Enregistre la fonction de callback qui est appelée à chaque changement de la valeur publiée.

js	function <b>registerValueCallback</b> ( <b>callback</b> )
cpp	int <b>registerValueCallback</b> ( YCellularValueCallback <b>callback</b> )
m	-(int) <b>registerValueCallback</b> : (YCellularValueCallback _Nullable) <b>callback</b>
pas	LongInt <b>registerValueCallback</b> ( <b>callback</b> : TYCellularValueCallback): LongInt
vb	function <b>registerValueCallback</b> ( ByVal <b>callback</b> As YCellularValueCallback) As Integer
cs	int <b>registerValueCallback</b> ( ValueCallback <b>callback</b> )
java	int <b>registerValueCallback</b> ( UpdateCallback <b>callback</b> )
uwp	async Task<int> <b>registerValueCallback</b> ( ValueCallback <b>callback</b> )
py	<b>registerValueCallback</b> ( <b>callback</b> )
php	function <b>registerValueCallback</b> ( <b>\$callback</b> )
ts	async <b>registerValueCallback</b> ( <b>callback</b> : YCellularValueCallback   null): Promise<number>
es	async <b>registerValueCallback</b> ( <b>callback</b> )

Ce callback n'est appelé que durant l'exécution de `ySleep` ou `yHandleEvents`. Cela permet à l'appelant de contrôler quand les callback peuvent se produire. Il est important d'appeler l'une de ces deux fonctions périodiquement pour garantir que les callback ne soient pas appelés trop tard. Pour désactiver un callback, il suffit d'appeler cette méthode en lui passant un pointeur nul.

**Paramètres :**

**callback** la fonction de callback à rappeler, ou un pointeur nul. La fonction de callback doit accepter deux arguments: l'object fonction dont la valeur a changé, et la chaîne de caractère décrivant la nouvelle valeur publiée.

**cellular**→**sendPUK()****YCellular**

Envoie le code PUK à la carte SIM pour la débloquent après trois échecs consécutifs de code PIN, et établit un nouveau code PIN dans la SIM.

js	function <b>sendPUK</b> ( <b>puk</b> , <b>newPin</b> )
cpp	int <b>sendPUK</b> ( string <b>puk</b> , string <b>newPin</b> )
m	-(int) <b>sendPUK</b> : (NSString*) <b>puk</b> : (NSString*) <b>newPin</b>
pas	LongInt <b>sendPUK</b> ( <b>puk</b> : string, <b>newPin</b> : string): LongInt
vb	function <b>sendPUK</b> ( ByVal <b>puk</b> As String, ByVal <b>newPin</b> As String) As Integer
cs	int <b>sendPUK</b> ( string <b>puk</b> , string <b>newPin</b> )
java	int <b>sendPUK</b> ( String <b>puk</b> , String <b>newPin</b> )
uwp	async Task<int> <b>sendPUK</b> ( string <b>puk</b> , string <b>newPin</b> )
py	<b>sendPUK</b> ( <b>puk</b> , <b>newPin</b> )
php	function <b>sendPUK</b> ( \$ <b>puk</b> , \$ <b>newPin</b> )
ts	async <b>sendPUK</b> ( <b>puk</b> : string, <b>newPin</b> : string): Promise<number>
es	async <b>sendPUK</b> ( <b>puk</b> , <b>newPin</b> )
dnp	int <b>sendPUK</b> ( string <b>puk</b> , string <b>newPin</b> )
cp	int <b>sendPUK</b> ( string <b>puk</b> , string <b>newPin</b> )
cmd	YCellular <b>target sendPUK puk newPin</b>

Seules dix tentatives consécutives de déblocage sont possibles: après dix tentatives infructueuses, la carte SIM sera définitivement inutilisable. Après avoir appelé cette fonction, vous devrez aussi appeler la méthode `set_pin()` pour indiquer au YoctoHub le nouveau PIN à utiliser dans le futur.

**Paramètres :**

- puk** code PUK de la carte SIM
- newPin** nouveau code PIN à configurer dans la carte SIM

**Retourne :**

YAPI . SUCCESS si l'opération se déroule sans erreur.

En cas d'erreur, déclenche une exception ou retourne un code d'erreur négatif.

**cellular**→**set\_airplaneMode()****cellular**→**setAirplaneMode()**

Modifie l'état du mode avion (radio désactivée).

js	function <b>set_airplaneMode</b> ( <b>newval</b> )
cpp	int <b>set_airplaneMode</b> ( Y_AIRPLANEMODE_enum <b>newval</b> )
m	-(int) setAirplaneMode : (Y_AIRPLANEMODE_enum) <b>newval</b>
pas	integer <b>set_airplaneMode</b> ( <b>newval</b> : Integer): integer
vb	function <b>set_airplaneMode</b> ( ByVal <b>newval</b> As Integer) As Integer
cs	int <b>set_airplaneMode</b> ( int <b>newval</b> )
java	int <b>set_airplaneMode</b> ( int <b>newval</b> )
uwp	async Task<int> <b>set_airplaneMode</b> ( int <b>newval</b> )
py	<b>set_airplaneMode</b> ( <b>newval</b> )
php	function <b>set_airplaneMode</b> ( \$ <b>newval</b> )
ts	async <b>set_airplaneMode</b> ( <b>newval</b> : YCellular_AirplaneMode): Promise<number>
es	async <b>set_airplaneMode</b> ( <b>newval</b> )
dnp	int <b>set_airplaneMode</b> ( int <b>newval</b> )
cp	int <b>set_airplaneMode</b> ( int <b>newval</b> )
cmd	YCellular <b>target set_airplaneMode newval</b>

**Paramètres :**

**newval** soit YCellular.AIRPLANEMODE\_OFF, soit YCellular.AIRPLANEMODE\_ON, selon l'état du mode avion (radio désactivée)

**Retourne :**

YAPI . SUCCESS si l'opération se déroule sans erreur.

En cas d'erreur, déclenche une exception ou retourne un code d'erreur négatif.

**cellular**→**set\_apn()****YCellular****cellular**→**setApn()**

Retourne le nom du point d'accès (APN) à utiliser, si nécessaire.

js	function <b>set_apn</b> ( <b>newval</b> )
cpp	int <b>set_apn</b> ( string <b>newval</b> )
m	-(int) setApn : (NSString*) <b>newval</b>
pas	integer <b>set_apn</b> ( <b>newval</b> : string): integer
vb	function <b>set_apn</b> ( ByVal <b>newval</b> As String) As Integer
cs	int <b>set_apn</b> ( string <b>newval</b> )
java	int <b>set_apn</b> ( String <b>newval</b> )
uwp	async Task<int> <b>set_apn</b> ( string <b>newval</b> )
py	<b>set_apn</b> ( <b>newval</b> )
php	function <b>set_apn</b> ( \$ <b>newval</b> )
ts	async <b>set_apn</b> ( <b>newval</b> : string): Promise<number>
es	async <b>set_apn</b> ( <b>newval</b> )
dnp	int <b>set_apn</b> ( string <b>newval</b> )
cp	int <b>set_apn</b> ( string <b>newval</b> )
cmd	YCellular <b>target set_apn newval</b>

Lorsque l'APN est vide, celui proposé par l'opérateur cellulaire est utilisée. N'oubliez pas d'appeler la méthode `saveToFlash()` du module si le réglage doit être préservé.

**Paramètres :**

**newval** une chaîne de caractères

**Retourne :**

YAPI . SUCCESS si l'opération se déroule sans erreur.

En cas d'erreur, déclenche une exception ou retourne un code d'erreur négatif.

**cellular**→**set\_apnAuth()****cellular**→**setApnAuth()**

Configure les paramètres d'identification pour se connecter à l'APN.

js	function <b>set_apnAuth</b> ( <b>username</b> , <b>password</b> )
c++	int <b>set_apnAuth</b> ( string <b>username</b> , string <b>password</b> )
m	-(int) setApnAuth : (NSString*) <b>username</b> : (NSString*) <b>password</b>
pas	LongInt <b>set_apnAuth</b> ( <b>username</b> : string, <b>password</b> : string): LongInt
vb	function <b>set_apnAuth</b> ( ByVal <b>username</b> As String, ByVal <b>password</b> As String) As Integer
cs	int <b>set_apnAuth</b> ( string <b>username</b> , string <b>password</b> )
java	int <b>set_apnAuth</b> ( String <b>username</b> , String <b>password</b> )
uwp	async Task<int> <b>set_apnAuth</b> ( string <b>username</b> , string <b>password</b> )
py	<b>set_apnAuth</b> ( <b>username</b> , <b>password</b> )
php	function <b>set_apnAuth</b> ( \$ <b>username</b> , \$ <b>password</b> )
ts	async <b>set_apnAuth</b> ( <b>username</b> : string, <b>password</b> : string): Promise<number>
es	async <b>set_apnAuth</b> ( <b>username</b> , <b>password</b> )
dnp	int <b>set_apnAuth</b> ( string <b>username</b> , string <b>password</b> )
cp	int <b>set_apnAuth</b> ( string <b>username</b> , string <b>password</b> )
cmd	YCellular <b>target set_apnAuth</b> <b>username password</b>

Les protocoles PAP et CHAP sont tous deux supportés.

**Paramètres :**

**username** nom d'utilisateur  
**password** mot de passe

**Retourne :**

YAPI . SUCCESS si l'opération se déroule sans erreur.

En cas d'erreur, déclenche une exception ou retourne un code d'erreur négatif.

**cellular**→**set\_dataReceived()****YCellular****cellular**→**setDataReceived()**

Modifie la valeur du compteur d'octets reçus.

js	function <b>set_dataReceived</b> ( <b>newval</b> )
cpp	int <b>set_dataReceived</b> ( int <b>newval</b> )
m	-(int) setDataReceived : (int) <b>newval</b>
pas	integer <b>set_dataReceived</b> ( <b>newval</b> : LongInt): integer
vb	function <b>set_dataReceived</b> ( ByVal <b>newval</b> As Integer) As Integer
cs	int <b>set_dataReceived</b> ( int <b>newval</b> )
java	int <b>set_dataReceived</b> ( int <b>newval</b> )
uwp	async Task<int> <b>set_dataReceived</b> ( int <b>newval</b> )
py	<b>set_dataReceived</b> ( <b>newval</b> )
php	function <b>set_dataReceived</b> ( <b>\$newval</b> )
ts	async <b>set_dataReceived</b> ( <b>newval</b> : number): Promise<number>
es	async <b>set_dataReceived</b> ( <b>newval</b> )
dnp	int <b>set_dataReceived</b> ( int <b>newval</b> )
cp	int <b>set_dataReceived</b> ( int <b>newval</b> )
cmd	YCellular <b>target</b> <b>set_dataReceived</b> <b>newval</b>

**Paramètres :**

**newval** un entier représentant la valeur du compteur d'octets reçus

**Retourne :**

YAPI . SUCCESS si l'opération se déroule sans erreur.

En cas d'erreur, déclenche une exception ou retourne un code d'erreur négatif.

**cellular** → **set\_dataSent()****cellular** → **setDataSent()**

Modifie la valeur du compteur d'octets envoyés.

js	function <b>set_dataSent</b> ( <b>newval</b> )
cpp	int <b>set_dataSent</b> ( int <b>newval</b> )
m	-(int) setDataSent : (int) <b>newval</b>
pas	integer <b>set_dataSent</b> ( <b>newval</b> : LongInt): integer
vb	function <b>set_dataSent</b> ( ByVal <b>newval</b> As Integer) As Integer
cs	int <b>set_dataSent</b> ( int <b>newval</b> )
java	int <b>set_dataSent</b> ( int <b>newval</b> )
uwp	async Task<int> <b>set_dataSent</b> ( int <b>newval</b> )
py	<b>set_dataSent</b> ( <b>newval</b> )
php	function <b>set_dataSent</b> ( <b>\$newval</b> )
ts	async <b>set_dataSent</b> ( <b>newval</b> : number): Promise<number>
es	async <b>set_dataSent</b> ( <b>newval</b> )
dnp	int <b>set_dataSent</b> ( int <b>newval</b> )
cp	int <b>set_dataSent</b> ( int <b>newval</b> )
cmd	YCellular <b>target set_dataSent newval</b>

**Paramètres :**

**newval** un entier représentant la valeur du compteur d'octets envoyés

**Retourne :**

YAPI . SUCCESS si l'opération se déroule sans erreur.

En cas d'erreur, déclenche une exception ou retourne un code d'erreur négatif.

**cellular**→**set\_enableData()****YCellular****cellular**→**setEnabledData()**

Modifie la condition dans laquelle le service de données IP (GRPS) doit être activé.

js	function <b>set_enableData</b> ( <b>newval</b> )
cpp	int <b>set_enableData</b> ( Y_ENABLEDATA_enum <b>newval</b> )
m	-(int) <b>setEnabledData</b> : (Y_ENABLEDATA_enum) <b>newval</b>
pas	integer <b>set_enableData</b> ( <b>newval</b> : Integer): integer
vb	function <b>set_enableData</b> ( ByVal <b>newval</b> As Integer) As Integer
cs	int <b>set_enableData</b> ( int <b>newval</b> )
java	int <b>set_enableData</b> ( int <b>newval</b> )
uwp	async Task<int> <b>set_enableData</b> ( int <b>newval</b> )
py	<b>set_enableData</b> ( <b>newval</b> )
php	function <b>set_enableData</b> ( \$ <b>newval</b> )
ts	async <b>set_enableData</b> ( <b>newval</b> : YCellular_Enabledata): Promise<number>
es	async <b>set_enableData</b> ( <b>newval</b> )
dnp	int <b>set_enableData</b> ( int <b>newval</b> )
cp	int <b>set_enableData</b> ( int <b>newval</b> )
cmd	YCellular <b>target</b> <b>set_enableData</b> <b>newval</b>

Le service peut être soit complètement désactivé, soit limité au réseau de de l'émetteur de la carte SIM, soit être activé pour tous les réseaux en partenariat avec la carte SIM (roaming). Attention, l'utilisation de données en roaming peut conduire à des coûts de télécommunication exorbitants !

Lorsque le service de donnée n'est pas actif, seules les communications par SMS sont possibles. N'oubliez pas d'appeler la méthode `saveToFlash()` du module si le réglage doit être préservé.

**Paramètres :**

**newval** une valeur parmi `YCellular.ENABLEDATA_HOMENETWORK`, `YCellular.ENABLEDATA_ROAMING`, `YCellular.ENABLEDATA_NEVER` et `YCellular.ENABLEDATA_NEUTRALITY` représentant la condition dans laquelle le service de données IP (GRPS) doit être activé

**Retourne :**

`YAPI.SUCCESS` si l'opération se déroule sans erreur.

En cas d'erreur, déclenche une exception ou retourne un code d'erreur négatif.

**cellular**→**set\_lockedOperator()**

YCellular

**cellular**→**setLockedOperator()**

Modifie le nom de l'opérateur de réseau cellulaire à utiliser.

js	function <b>set_lockedOperator</b> ( <b>newval</b> )
cpp	int <b>set_lockedOperator</b> ( string <b>newval</b> )
m	-(int) setLockedOperator : (NSString*) <b>newval</b>
pas	integer <b>set_lockedOperator</b> ( <b>newval</b> : string): integer
vb	function <b>set_lockedOperator</b> ( ByVal <b>newval</b> As String) As Integer
cs	int <b>set_lockedOperator</b> ( string <b>newval</b> )
java	int <b>set_lockedOperator</b> ( String <b>newval</b> )
uwp	async Task<int> <b>set_lockedOperator</b> ( string <b>newval</b> )
py	<b>set_lockedOperator</b> ( <b>newval</b> )
php	function <b>set_lockedOperator</b> ( \$ <b>newval</b> )
ts	async <b>set_lockedOperator</b> ( <b>newval</b> : string): Promise<number>
es	async <b>set_lockedOperator</b> ( <b>newval</b> )
dnp	int <b>set_lockedOperator</b> ( string <b>newval</b> )
cp	int <b>set_lockedOperator</b> ( string <b>newval</b> )
cmd	YCellular <b>target set_lockedOperator newval</b>

Si le nom est une chaîne vide, le choix sera fait automatiquement selon la carte SIM. Sinon, seul l'opérateur choisi sera utilisé. N'oubliez pas d'appeler la méthode `saveToFlash()` du module si le réglage doit être préservé.

**Paramètres :**

**newval** une chaîne de caractères représentant le nom de l'opérateur de réseau cellulaire à utiliser

**Retourne :**

YAPI . SUCCESS si l'opération se déroule sans erreur.

En cas d'erreur, déclenche une exception ou retourne un code d'erreur négatif.

**cellular**→**set\_logicalName()****YCellular****cellular**→**setLogicalName()**

Modifie le nom logique de l'interface cellulaire.

js	function <b>set_logicalName</b> ( <b>newval</b> )
cpp	int <b>set_logicalName</b> ( string <b>newval</b> )
m	-(int) setLogicalName : (NSString*) <b>newval</b>
pas	integer <b>set_logicalName</b> ( <b>newval</b> : string): integer
vb	function <b>set_logicalName</b> ( ByVal <b>newval</b> As String) As Integer
cs	int <b>set_logicalName</b> ( string <b>newval</b> )
java	int <b>set_logicalName</b> ( String <b>newval</b> )
uwp	async Task<int> <b>set_logicalName</b> ( string <b>newval</b> )
py	<b>set_logicalName</b> ( <b>newval</b> )
php	function <b>set_logicalName</b> ( \$ <b>newval</b> )
ts	async <b>set_logicalName</b> ( <b>newval</b> : string): Promise<number>
es	async <b>set_logicalName</b> ( <b>newval</b> )
dnp	int <b>set_logicalName</b> ( string <b>newval</b> )
cp	int <b>set_logicalName</b> ( string <b>newval</b> )
cmd	YCellular <b>target</b> <b>set_logicalName</b> <b>newval</b>

Vous pouvez utiliser `yCheckLogicalName()` pour vérifier si votre paramètre est valide. N'oubliez pas d'appeler la méthode `saveToFlash()` du module si le réglage doit être préservé.

**Paramètres :**

**newval** une chaîne de caractères représentant le nom logique de l'interface cellulaire.

**Retourne :**

YAPI . SUCCESS si l'appel se déroule sans erreur.

En cas d'erreur, déclenche une exception ou retourne un code d'erreur négatif.

**cellular**→**set\_pin()****cellular**→**setPin()**

Modifie le code PIN utilisé par le module pour accéder à la carte SIM.

js	function <b>set_pin</b> ( <b>newval</b> )
cpp	int <b>set_pin</b> ( string <b>newval</b> )
m	-(int) setPin : (NSString*) <b>newval</b>
pas	integer <b>set_pin</b> ( <b>newval</b> : string): integer
vb	function <b>set_pin</b> ( ByVal <b>newval</b> As String) As Integer
cs	int <b>set_pin</b> ( string <b>newval</b> )
java	int <b>set_pin</b> ( String <b>newval</b> )
uwp	async Task<int> <b>set_pin</b> ( string <b>newval</b> )
py	<b>set_pin</b> ( <b>newval</b> )
php	function <b>set_pin</b> ( <b>\$newval</b> )
ts	async <b>set_pin</b> ( <b>newval</b> : string): Promise<number>
es	async <b>set_pin</b> ( <b>newval</b> )
dnp	int <b>set_pin</b> ( string <b>newval</b> )
cp	int <b>set_pin</b> ( string <b>newval</b> )
cmd	YCellular <b>target set_pin newval</b>

Cette fonction ne change pas le code sur la carte SIM elle-même, mais uniquement le paramètre utilisé par le module pour essayer d'en obtenir l'accès. Si le code SIM ne fonctionne pas dès le premier essai, il sera automatiquement oublié et un message "Enter SIM PIN" apparaîtra dans l'attribut 'message'. Il faudra alors appeler à nouveau cette méthode avec le bon code PIN. Après trois essais infructueux consécutifs le message devient "Enter SIM PUK" et il faut alors entrer le code PUK de la carte SIM avec la méthode `sendPUK`.

N'oubliez pas d'appeler la méthode `saveToFlash()` du module pour que le paramètre soit sauvegardé dans la flash.

**Paramètres :**

**newval** une chaîne de caractères représentant le code PIN utilisé par le module pour accéder à la carte SIM

**Retourne :**

YAPI . SUCCESS si l'opération se déroule sans erreur.

En cas d'erreur, déclenche une exception ou retourne un code d'erreur négatif.

**cellular**→**set\_pingInterval()****YCellular****cellular**→**setPingInterval()**

Modifie l'intervalle entre les tests de connectivité spontanés, en secondes.

js	function <b>set_pingInterval</b> ( <b>newval</b> )
cpp	int <b>set_pingInterval</b> ( int <b>newval</b> )
m	-(int) <b>setPingInterval</b> : (int) <b>newval</b>
pas	integer <b>set_pingInterval</b> ( <b>newval</b> : LongInt): integer
vb	function <b>set_pingInterval</b> ( ByVal <b>newval</b> As Integer) As Integer
cs	int <b>set_pingInterval</b> ( int <b>newval</b> )
java	int <b>set_pingInterval</b> ( int <b>newval</b> )
uwp	async Task<int> <b>set_pingInterval</b> ( int <b>newval</b> )
py	<b>set_pingInterval</b> ( <b>newval</b> )
php	function <b>set_pingInterval</b> ( <b>\$newval</b> )
ts	async <b>set_pingInterval</b> ( <b>newval</b> : number): Promise<number>
es	async <b>set_pingInterval</b> ( <b>newval</b> )
dnp	int <b>set_pingInterval</b> ( int <b>newval</b> )
cp	int <b>set_pingInterval</b> ( int <b>newval</b> )
cmd	YCellular <b>target set_pingInterval newval</b>

N'oubliez pas d'appeler la méthode `saveToFlash()` du module si le réglage doit être préservé.

**Paramètres :**

**newval** un entier représentant l'intervalle entre les tests de connectivité spontanés, en secondes

**Retourne :**

YAPI . SUCCESS si l'opération se déroule sans erreur.

En cas d'erreur, déclenche une exception ou retourne un code d'erreur négatif.

**cellular**→**set\_radioConfig()****cellular**→**setRadioConfig()**

Modifie le type de protocole utilisé sur la communication série.

js	function <b>set_radioConfig</b> ( <b>newval</b> )
cpp	int <b>set_radioConfig</b> ( string <b>newval</b> )
m	-(int) setRadioConfig : (NSString*) <b>newval</b>
pas	integer <b>set_radioConfig</b> ( <b>newval</b> : string): integer
vb	function <b>set_radioConfig</b> ( ByVal <b>newval</b> As String) As Integer
cs	int <b>set_radioConfig</b> ( string <b>newval</b> )
java	int <b>set_radioConfig</b> ( String <b>newval</b> )
uwp	async Task<int> <b>set_radioConfig</b> ( string <b>newval</b> )
py	<b>set_radioConfig</b> ( <b>newval</b> )
php	function <b>set_radioConfig</b> ( \$ <b>newval</b> )
ts	async <b>set_radioConfig</b> ( <b>newval</b> : string): Promise<number>
es	async <b>set_radioConfig</b> ( <b>newval</b> )
dnp	int <b>set_radioConfig</b> ( string <b>newval</b> )
cp	int <b>set_radioConfig</b> ( string <b>newval</b> )
cmd	YCellular <b>target set_radioConfig newval</b>

Les valeurs possibles sont "Line" pour des messages ASCII séparés par des retours de ligne, "Frame:[timeout]ms" pour des messages binaires séparés par une temporisation, "Char" pour un flux ASCII continu ou "Byte" pour un flux binaire continue. Le suffixe "[wait]ms" peut être ajouté pour réduire la cadence d'émission de sorte à ce qu'il y ait au minimum le nombre spécifié de millisecondes d'intervalle entre l'envoi de chaque byte. N'oubliez pas d'appeler la méthode `saveToFlash()` du module si le réglage doit être préservé.

**Paramètres :**

**newval** une chaîne de caractères représentant le type de protocole utilisé sur la communication série

**Retourne :**

YAPI . SUCCESS si l'opération se déroule sans erreur.

En cas d'erreur, déclenche une exception ou retourne un code d'erreur négatif.

**cellular**→**set\_userData()****YCellular****cellular**→**setUserData()**

Enregistre un contexte libre dans l'attribut `userData` de la fonction, afin de le retrouver plus tard à l'aide de la méthode `get_userData`.

js	function <b>set_userData</b> ( <b>data</b> )
cpp	void <b>set_userData</b> ( void * <b>data</b> )
m	-(void) setUserData : (id) <b>data</b>
pas	<b>set_userData</b> ( <b>data</b> : Tobject)
vb	procedure <b>set_userData</b> ( ByVal <b>data</b> As Object)
cs	void <b>set_userData</b> ( object <b>data</b> )
java	void <b>set_userData</b> ( Object <b>data</b> )
py	<b>set_userData</b> ( <b>data</b> )
php	function <b>set_userData</b> ( <b>\$data</b> )
ts	async <b>set_userData</b> ( <b>data</b> : object null): Promise<void>
es	async <b>set_userData</b> ( <b>data</b> )

Cet attribut n'es pas utilisé directement par l'API. Il est à la disposition de l'appelant pour stocker un contexte.

**Paramètres :**

**data** objet quelconque à mémoriser

cellular → **unmuteValueCallbacks()**

YCellular

Réactive l'envoi de chaque changement de la valeur publiée au hub parent.

js	function <b>unmuteValueCallbacks</b> ( )
cpp	int <b>unmuteValueCallbacks</b> ( )
m	-(int) <b>unmuteValueCallbacks</b>
pas	LongInt <b>unmuteValueCallbacks</b> ( ): LongInt
vb	function <b>unmuteValueCallbacks</b> ( ) As Integer
cs	int <b>unmuteValueCallbacks</b> ( )
java	int <b>unmuteValueCallbacks</b> ( )
uwp	async Task<int> <b>unmuteValueCallbacks</b> ( )
py	<b>unmuteValueCallbacks</b> ( )
php	function <b>unmuteValueCallbacks</b> ( )
ts	async <b>unmuteValueCallbacks</b> ( ): Promise<number>
es	async <b>unmuteValueCallbacks</b> ( )
dnp	int <b>unmuteValueCallbacks</b> ( )
cp	int <b>unmuteValueCallbacks</b> ( )
cmd	YCellular <b>target unmuteValueCallbacks</b>

Cette fonction annule un précédent appel à `muteValueCallbacks()`. N'oubliez pas d'appeler la méthode `saveToFlash()` du module si le réglage doit être préservé.

**Retourne :**

YAPI . SUCCESS si l'opération se déroule sans erreur.

En cas d'erreur, déclenche une exception ou retourne un code d'erreur négatif.

**cellular**→**wait\_async()****YCellular**

Attend que toutes les commandes asynchrones en cours d'exécution sur le module soient terminées, et appelle le callback passé en paramètre.

```
js fonction wait_async( callback, context)
```

```
ts wait_async( callback: Function, context: object)
```

```
es wait_async( callback, context)
```

La fonction callback peut donc librement utiliser des fonctions synchrones ou asynchrones, sans risquer de bloquer la machine virtuelle Javascript.

**Paramètres :**

**callback** fonction de callback qui sera appelée dès que toutes les commandes en cours d'exécution sur le module seront terminées. La fonction callback reçoit deux arguments: le contexte fourni par l'appelant et l'objet fonction concerné.

**context** contexte fourni par l'appelant, et qui sera passé tel-qu'il est à la fonction de callback

**Retourne :**

rien du tout.

## 8.3. La classe YNetwork

Interface pour interagir avec les interfaces réseau, disponibles par exemple dans le YoctoHub-Ethernet, le YoctoHub-GSM-3G-NA, le YoctoHub-GSM-4G et le YoctoHub-Wireless-n

La classe `YNetwork` permet de contrôler les paramètres TCP/IP des modules Yoctopuce dotés d'une interface réseau.

Pour utiliser les fonctions décrites ici, vous devez inclure:

es	in HTML: <code>&lt;script src="../../lib/yocto_network.js"&gt;&lt;/script&gt;</code> in node.js: <code>require('yoctolib-es2017/yocto_network.js');</code>
js	<code>&lt;script type='text/javascript' src='yocto_network.js'&gt;&lt;/script&gt;</code>
cpp	<code>#include "yocto_network.h"</code>
m	<code>#import "yocto_network.h"</code>
pas	<code>uses yocto_network;</code>
vb	<code>yocto_network.vb</code>
cs	<code>yocto_network.cs</code>
java	<code>import com.yoctopuce.YoctoAPI.YNetwork;</code>
uwp	<code>import com.yoctopuce.YoctoAPI.YNetwork;</code>
py	<code>from yocto_network import *</code>
php	<code>require_once('yocto_network.php');</code>
ts	in HTML: <code>import { YNetwork } from '../../dist/esm/yocto_network.js';</code> in Node.js: <code>import { YNetwork } from 'yoctolib-cjs/yocto_network.js';</code>
dnp	<code>import YoctoProxyAPI.YNetworkProxy</code>
cp	<code>#include "yocto_network_proxy.h"</code>
vi	<code>YNetwork.vi</code>
ml	<code>import YoctoProxyAPI.YNetworkProxy</code>

### Fonction globales

#### **YNetwork.FindNetwork(func)**

Permet de retrouver une interface réseau d'après un identifiant donné.

#### **YNetwork.FindNetworkInContext(yctx, func)**

Permet de retrouver une interface réseau d'après un identifiant donné dans un Context YAPI.

#### **YNetwork.FirstNetwork()**

Commence l'énumération des interfaces réseau accessibles par la librairie.

#### **YNetwork.FirstNetworkInContext(yctx)**

Commence l'énumération des interfaces réseau accessibles par la librairie.

#### **YNetwork.GetSimilarFunctions()**

Enumère toutes les fonctions de type Network disponibles sur les modules actuellement joignables par la librairie, et retourne leurs identifiants matériels uniques (hardwareId).

### Propriétés des objets YNetworkProxy

#### **network→AdminPassword [modifiable]**

Chaîne de hash si un mot de passe a été configuré pour l'utilisateur "admin", ou sinon une chaîne vide.

#### **network→AdvertisedValue [lecture seule]**

Courte chaîne de caractères représentant l'état courant de la fonction.

#### **network→CallbackCredentials [modifiable]**

Version hashée du laisser-passer pour le callback de notification s'il a été configuré, ou sinon une chaîne vide.

#### **network→CallbackEncoding [modifiable]**

Encodage à utiliser pour représenter les valeurs notifiées par callback.
<b>network→CallbackInitialDelay</b> <i>[modifiable]</i> Attente initiale avant la première notification par callback, en secondes.
<b>network→CallbackMaxDelay</b> <i>[modifiable]</i> Attente entre deux callback HTTP lorsque rien n'est à signaler, en secondes.
<b>network→CallbackMethod</b> <i>[modifiable]</i> Méthode HTTP à utiliser pour signaler les changements d'état par callback.
<b>network→CallbackMinDelay</b> <i>[modifiable]</i> Attente minimale entre deux callbacks HTTP, en secondes.
<b>network→CallbackSchedule</b> <i>[modifiable]</i> Planification des callbacks HTTP, sous forme de chaîne de caractères.
<b>network→CallbackUrl</b> <i>[modifiable]</i> Adresse (URL) de callback à notifier lors de changement d'état significatifs.
<b>network→DefaultPage</b> <i>[modifiable]</i> Page HTML à envoyer pour l'URL "/" <b>Modifiable</b> .
<b>network→Discoverable</b> <i>[modifiable]</i> état d'activation du protocole d'annonce sur le réseau permettant de retrouver facilement le module (protocoles uPnP/Bonjour).
<b>network→FriendlyName</b> <i>[lecture seule]</i> Identifiant global de la fonction au format NOM_MODULE . NOM_FONCTION.
<b>network→FunctionId</b> <i>[lecture seule]</i> Identifiant matériel de l'interface réseau, sans référence au module.
<b>network→HardwareId</b> <i>[lecture seule]</i> Identifiant matériel unique de la fonction au format SERIAL . FUNCTIONID.
<b>network→HttpPort</b> <i>[modifiable]</i> Port TCP utilisé pour l'interface Web du hub.
<b>network→IpAddress</b> <i>[lecture seule]</i> Adresse IP utilisée par le module Yoctopuce.
<b>network→IsOnline</b> <i>[lecture seule]</i> Vérifie si le module hébergeant la fonction est joignable, sans déclencher d'erreur.
<b>network→LogicalName</b> <i>[modifiable]</i> Nom logique de la fonction.
<b>network→MacAddress</b> <i>[lecture seule]</i> Adresse MAC de l'interface réseau, unique pour chaque module.
<b>network→NtpServer</b> <i>[modifiable]</i> Adresse IP du serveur de NTP à utiliser pour maintenir le module à l'heure.
<b>network→PrimaryDNS</b> <i>[modifiable]</i> Adresse IP du serveur de noms primaire que le module doit utiliser.
<b>network→Readiness</b> <i>[lecture seule]</i> état de fonctionnement atteint par l'interface réseau.
<b>network→SecondaryDNS</b> <i>[modifiable]</i> Adresse IP du serveur de noms secondaire que le module doit utiliser.
<b>network→SerialNumber</b> <i>[lecture seule]</i> Numéro de série du module, préprogrammé en usine.
<b>network→UserPassword</b> <i>[modifiable]</i>

Chaîne de hash si un mot de passe a été configuré pour l'utilisateur "user", ou sinon une chaîne vide.

**network**→**WwwWatchdogDelay** [*modifiable*]

Durée de perte de connexion WWW tolérée (en secondes) avant de déclencher un redémarrage automatique pour tenter de récupérer la connectivité Internet.

### Méthodes des objets YNetwork

**network**→**callbackLogin**(**username**, **password**)

Contacte le callback de notification et sauvegarde un laisser-passer pour s'y connecter.

**network**→**clearCache**()

Invalide le cache.

**network**→**describe**()

Retourne un court texte décrivant de manière non-ambigüe l'instance de l'interface réseau au format `TYPE (NAME) =SERIAL . FUNCTIONID`.

**network**→**get\_adminPassword**()

Retourne une chaîne de hash si un mot de passe a été configuré pour l'utilisateur "admin", ou sinon une chaîne vide.

**network**→**get\_advertisedValue**()

Retourne la valeur courante de l'interface réseau (pas plus de 6 caractères).

**network**→**get\_callbackCredentials**()

Retourne une version hashée du laisser-passer pour le callback de notification s'il a été configuré, ou sinon une chaîne vide.

**network**→**get\_callbackEncoding**()

Retourne l'encodage à utiliser pour représenter les valeurs notifiées par callback.

**network**→**get\_callbackInitialDelay**()

Retourne l'attente initiale avant la première notification par callback, en secondes.

**network**→**get\_callbackMaxDelay**()

Retourne l'attente entre deux callback HTTP lorsque rien n'est à signaler, en secondes.

**network**→**get\_callbackMethod**()

Retourne la méthode HTTP à utiliser pour signaler les changements d'état par callback.

**network**→**get\_callbackMinDelay**()

Retourne l'attente minimale entre deux callbacks HTTP, en secondes.

**network**→**get\_callbackSchedule**()

Retourne la planification des callbacks HTTP, sous forme de chaîne de caractères.

**network**→**get\_callbackUrl**()

Retourne l'adresse (URL) de callback à notifier lors de changement d'état significatifs.

**network**→**get\_defaultPage**()

Retourne la page HTML à envoyer pour l'URL "/"

**network**→**get\_discoverable**()

Retourne l'état d'activation du protocole d'annonce sur le réseau permettant de retrouver facilement le module (protocoles uPnP/Bonjour).

**network**→**get\_errorMessage**()

Retourne le message correspondant à la dernière erreur survenue lors de l'utilisation de l'interface réseau.

**network**→**get\_errorType**()

Retourne le code d'erreur correspondant à la dernière erreur survenue lors de l'utilisation de l'interface réseau.

**network**→**get\_friendlyName**()

Retourne un identifiant global de l'interface réseau au format `NOM_MODULE . NOM_FONCTION`.

**network**→**get\_functionDescriptor**()

Retourne un identifiant unique de type YFUN\_DESCR correspondant à la fonction.

#### **network→get\_functionId()**

Retourne l'identifiant matériel de l'interface réseau, sans référence au module.

#### **network→get\_hardwareId()**

Retourne l'identifiant matériel unique de l'interface réseau au format SERIAL . FUNCTIONID.

#### **network→get\_httpPort()**

Retourne le port TCP utilisé pour l'interface Web du hub.

#### **network→get\_ipAddress()**

Retourne l'adresse IP utilisée par le module Yoctopuce.

#### **network→get\_ipConfig()**

Retourne la configuration IP de l'interface réseau.

#### **network→get\_logicalName()**

Retourne le nom logique de l'interface réseau.

#### **network→get\_macAddress()**

Retourne l'adresse MAC de l'interface réseau, unique pour chaque module.

#### **network→get\_module()**

Retourne l'objet YModule correspondant au module Yoctopuce qui héberge la fonction.

#### **network→get\_module\_async(callback, context)**

Retourne l'objet YModule correspondant au module Yoctopuce qui héberge la fonction.

#### **network→get\_ntpServer()**

Retourne l'adresse IP du serveur de NTP à utiliser pour maintenir le module à l'heure.

#### **network→get\_poeCurrent()**

Retourne le courant consommé par le module depuis Power-over-Ethernet (PoE), en milliampères.

#### **network→get\_primaryDNS()**

Retourne l'adresse IP du serveur de noms primaire que le module doit utiliser.

#### **network→get\_readiness()**

Retourne l'état de fonctionnement atteint par l'interface réseau.

#### **network→get\_router()**

Retourne l'adresse IP du routeur (passerelle) utilisé par le module (*default gateway*).

#### **network→get\_secondaryDNS()**

Retourne l'adresse IP du serveur de noms secondaire que le module doit utiliser.

#### **network→get\_serialNumber()**

Retourne le numéro de série du module, préprogrammé en usine.

#### **network→get\_subnetMask()**

Retourne le masque de sous-réseau utilisé par le module.

#### **network→get\_userData()**

Retourne le contenu de l'attribut userData, précédemment stocké à l'aide de la méthode set\_userdata.

#### **network→get\_userPassword()**

Retourne une chaîne de hash si un mot de passe a été configuré pour l'utilisateur "user", ou sinon une chaîne vide.

#### **network→get\_wwwWatchdogDelay()**

Retourne la durée de perte de connexion WWW tolérée (en secondes) avant de déclencher un redémarrage automatique pour tenter de récupérer la connectivité Internet.

#### **network→isOnline()**

Vérifie si le module hébergeant l'interface réseau est joignable, sans déclencher d'erreur.

**network**→**isOnline\_async(callback, context)**

Vérifie si le module hébergeant l'interface réseau est joignable, sans déclencher d'erreur.

**network**→**isReadOnly()**

Test si la fonction est en lecture seule.

**network**→**load(msValidity)**

Met en cache les valeurs courantes de l'interface réseau, avec une durée de validité spécifiée.

**network**→**loadAttribute(attrName)**

Retourne la valeur actuelle d'un attribut spécifique de la fonction, sous forme de texte, le plus rapidement possible mais sans passer par le cache.

**network**→**load\_async(msValidity, callback, context)**

Met en cache les valeurs courantes de l'interface réseau, avec une durée de validité spécifiée.

**network**→**muteValueCallbacks()**

Désactive l'envoi de chaque changement de la valeur publiée au hub parent.

**network**→**nextNetwork()**

Continue l'énumération des interfaces réseau commencée à l'aide de `yFirstNetwork()` Attention, vous ne pouvez faire aucune supposition sur l'ordre dans lequel les interfaces réseau sont retournés.

**network**→**ping(host)**

Ping l'adresse choisie pour vérifier la connexion réseau.

**network**→**registerValueCallback(callback)**

Enregistre la fonction de callback qui est appelée à chaque changement de la valeur publiée.

**network**→**set\_adminPassword(newval)**

Modifie le mot de passe pour l'utilisateur "admin", qui devient alors instantanément nécessaire pour toute altération de l'état du module.

**network**→**set\_callbackCredentials(newval)**

Modifie le laisser-passer pour se connecter à l'adresse de callback.

**network**→**set\_callbackEncoding(newval)**

Modifie l'encodage à utiliser pour représenter les valeurs notifiées par callback.

**network**→**set\_callbackInitialDelay(newval)**

Modifie l'attente initiale avant la première notification par callback, en secondes.

**network**→**set\_callbackMaxDelay(newval)**

Modifie l'attente entre deux callback HTTP lorsque rien n'est à signaler.

**network**→**set\_callbackMethod(newval)**

Modifie la méthode HTTP à utiliser pour signaler les changements d'état par callback.

**network**→**set\_callbackMinDelay(newval)**

Modifie l'attente minimale entre deux callbacks HTTP, en secondes.

**network**→**set\_callbackSchedule(newval)**

Modifie la planification des callbacks HTTP, sous forme de chaîne de caractères.

**network**→**set\_callbackUrl(newval)**

Modifie l'adresse (URL) de callback à notifier lors de changement d'état significatifs.

**network**→**set\_defaultPage(newval)**

Modifie la page HTML par défaut du hub.

**network**→**set\_discoverable(newval)**

Modifie l'état d'activation du protocole d'annonce sur le réseau permettant de retrouver facilement le module (protocoles uPnP/Bonjour).

**network**→**set\_httpPort(newval)**

Modifie le port TCP utilisé pour l'interface Web du hub.

**network**→**set\_logicalName(newval)**

Modifie le nom logique de l'interface réseau.

**network**→**set\_ntpServer(newval)**

Modifie l'adresse IP du serveur NTP que le module doit utiliser.

**network**→**set\_periodicCallbackSchedule(interval, offset)**

Configure la planification de callbacks HTTP périodiques (fonction simplifiée).

**network**→**set\_primaryDNS(newval)**

Modifie l'adresse IP du serveur de noms primaire que le module doit utiliser.

**network**→**set\_secondaryDNS(newval)**

Modifie l'adresse IP du serveur de nom secondaire que le module doit utiliser.

**network**→**set\_userdata(data)**

Enregistre un contexte libre dans l'attribut `userData` de la fonction, afin de le retrouver plus tard à l'aide de la méthode `get_userdata`.

**network**→**set\_userPassword(newval)**

Modifie le mode de passe pour l'utilisateur "user", qui devient alors instantanément nécessaire pour tout accès au module.

**network**→**set\_wwwWatchdogDelay(newval)**

Modifie la durée de perte de connexion WWW tolérée (en secondes) avant de déclencher un redémarrage automatique pour tenter de récupérer la connectivité Internet.

**network**→**triggerCallback()**

Déclenche un callback HTTP rapidement.

**network**→**unmuteValueCallbacks()**

Réactive l'envoi de chaque changement de la valeur publiée au hub parent.

**network**→**useDHCP(fallbackIpAddr, fallbackSubnetMaskLen, fallbackRouter)**

Modifie la configuration de l'interface réseau pour utiliser une adresse assignée automatiquement par le serveur DHCP.

**network**→**useDHCPauto()**

Modifie la configuration de l'interface réseau pour utiliser une adresse assignée automatiquement par le serveur DHCP.

**network**→**useStaticIP(ipAddress, subnetMaskLen, router)**

Modifie la configuration de l'interface réseau pour utiliser une adresse IP assignée manuellement (adresse IP statique).

**network**→**wait\_async(callback, context)**

Attend que toutes les commandes asynchrones en cours d'exécution sur le module soient terminées, et appelle le callback passé en paramètre.

## YNetwork.FindNetwork() YNetwork.FindNetwork()

YNetwork

Permet de retrouver une interface réseau d'après un identifiant donné.

js	function <b>yFindNetwork</b> ( <b>func</b> )
c++	YNetwork* <b>FindNetwork</b> ( string <b>func</b> )
m	+(YNetwork*) <b>FindNetwork</b> : (NSString*) <b>func</b>
pas	TYNetwork <b>yFindNetwork</b> ( <b>func</b> : string): TYNetwork
vb	function <b>FindNetwork</b> ( ByVal <b>func</b> As String) As YNetwork
cs	static YNetwork <b>FindNetwork</b> ( string <b>func</b> )
java	static YNetwork <b>FindNetwork</b> ( String <b>func</b> )
uwp	static YNetwork <b>FindNetwork</b> ( string <b>func</b> )
py	<b>FindNetwork</b> ( <b>func</b> )
php	function <b>FindNetwork</b> ( <b>\$func</b> )
ts	static <b>FindNetwork</b> ( <b>func</b> : string): YNetwork
es	static <b>FindNetwork</b> ( <b>func</b> )
dnp	static YNetworkProxy <b>FindNetwork</b> ( string <b>func</b> )
cp	static YNetworkProxy * <b>FindNetwork</b> ( string <b>func</b> )

L'identifiant peut être spécifié sous plusieurs formes:

- NomLogiqueFonction
- NoSerieModule.IdentifiantFonction
- NoSerieModule.NomLogiqueFonction
- NomLogiqueModule.IdentifiantMatériel
- NomLogiqueModule.NomLogiqueFonction

Cette fonction n'exige pas que l'interface réseau soit en ligne au moment où elle est appelée, l'objet retourné sera néanmoins valide. Utiliser la méthode `YNetwork.isOnline()` pour tester si l'interface réseau est utilisable à un moment donné. En cas d'ambiguïté lorsqu'on fait une recherche par nom logique, aucune erreur ne sera notifiée: la première instance trouvée sera renvoyée. La recherche se fait d'abord par nom matériel, puis par nom logique.

Si un appel à la méthode `is_online()` de cet objet renvoie FAUX alors que vous êtes sûr que le module correspondant est bien branché, vérifiez que vous n'avez pas oublié d'appeler `registerHub()` à l'initialisation de de l'application.

### Paramètres :

**func** une chaîne de caractères qui référence l'interface réseau sans ambiguïté, par exemple `YHUBETH1.network`.

### Retourne :

un objet de classe `YNetwork` qui permet ensuite de contrôler l'interface réseau.

## YNetwork.FindNetworkInContext() YNetwork.FindNetworkInContext()

YNetwork

Permet de retrouver une interface réseau d'après un identifiant donné dans un Contexte YAPI.

```
java static YNetwork FindNetworkInContext( YAPIContext yctx, String func)
uwp static YNetwork FindNetworkInContext( YAPIContext yctx, string func)
ts static FindNetworkInContext( yctx: YAPIContext, func: string): YNetwork
es static FindNetworkInContext( yctx, func)
```

L'identifiant peut être spécifié sous plusieurs formes:

- NomLogiqueFonction
- NoSerieModule.IdentifiantFonction
- NoSerieModule.NomLogiqueFonction
- NomLogiqueModule.IdentifiantMatériel
- NomLogiqueModule.NomLogiqueFonction

Cette fonction n'exige pas que l'interface réseau soit en ligne au moment où elle est appelée, l'objet retourné sera néanmoins valide. Utiliser la méthode `YNetwork.isOnline()` pour tester si l'interface réseau est utilisable à un moment donné. En cas d'ambiguïté lorsqu'on fait une recherche par nom logique, aucune erreur ne sera notifiée: la première instance trouvée sera renvoyée. La recherche se fait d'abord par nom matériel, puis par nom logique.

### Paramètres :

**yctx** un contexte YAPI

**func** une chaîne de caractères qui référence l'interface réseau sans ambiguïté, par exemple `YHUBETH1.network`.

### Retourne :

un objet de classe `YNetwork` qui permet ensuite de contrôler l'interface réseau.

## YNetwork.FirstNetwork() YNetwork.FirstNetwork()

YNetwork

Commence l'énumération des interfaces réseau accessibles par la librairie.

js	function <b>yFirstNetwork</b> ( )
c++	YNetwork * <b>FirstNetwork</b> ( )
m	+(YNetwork*) <b>FirstNetwork</b>
pas	TYNetwork <b>yFirstNetwork</b> ( ): TYNetwork
vb	function <b>FirstNetwork</b> ( ) As YNetwork
cs	static YNetwork <b>FirstNetwork</b> ( )
java	static YNetwork <b>FirstNetwork</b> ( )
uwp	static YNetwork <b>FirstNetwork</b> ( )
py	<b>FirstNetwork</b> ( )
php	function <b>FirstNetwork</b> ( )
ts	static <b>FirstNetwork</b> ( ): YNetwork   null
es	static <b>FirstNetwork</b> ( )

Utiliser la fonction `YNetwork.nextNetwork()` pour itérer sur les autres interfaces réseau.

**Retourne :**

un pointeur sur un objet `YNetwork`, correspondant à la première interface réseau accessible en ligne, ou `null` si il n'y a pas de interfaces réseau disponibles.

## YNetwork.FirstNetworkInContext() YNetwork.FirstNetworkInContext()

YNetwork

Commence l'énumération des interfaces réseau accessibles par la librairie.

```
java static YNetwork FirstNetworkInContext( YAPIContext yctx)
uwp static YNetwork FirstNetworkInContext( YAPIContext yctx)
ts static FirstNetworkInContext( yctx: YAPIContext): YNetwork | null
es static FirstNetworkInContext( yctx)
```

Utiliser la fonction `YNetwork.nextNetwork()` pour itérer sur les autres interfaces réseau.

### Paramètres :

`yctx` un contexte YAPI.

### Retourne :

un pointeur sur un objet `YNetwork`, correspondant à la première interface réseau accessible en ligne, ou `null` si il n'y a pas de interfaces réseau disponibles.

## YNetwork.GetSimilarFunctions() YNetwork.GetSimilarFunctions()

YNetwork

Enumère toutes les fonctions de type Network disponibles sur les modules actuellement joignables par la librairie, et retourne leurs identifiants matériels uniques (hardwareId).

```
dnf static new string[] GetSimilarFunctions( )
```

```
cp static vector<string> GetSimilarFunctions( )
```

Chaque chaîne retournée peut être passée en argument à la méthode `YNetwork.FindNetwork` pour obtenir un objet permettant d'interagir avec le module correspondant.

**Retourne :**

un tableau de chaînes de caractères, contenant les identifiants matériels de chaque fonction disponible trouvée.

---

**network**→**AdminPassword****YNetwork**

---

Chaîne de hash si un mot de passe a été configuré pour l'utilisateur "admin", ou sinon une chaîne vide.

`dnP` `string AdminPassword`

**Modifiable.** Modifie le mot de passe pour l'utilisateur "admin", qui devient alors instantanément nécessaire pour toute altération de l'état du module. Si la valeur fournie est une chaîne vide, plus aucun mot de passe n'est nécessaire. N'oubliez pas d'appeler la méthode `saveToFlash()` du module si le réglage doit être préservé.

**network**→**AdvertisedValue**

**YNetwork**

---

Courte chaîne de caractères représentant l'état courant de la fonction.

`dnsp` string **AdvertisedValue**

---

**network**→**CallbackCredentials****YNetwork**

---

Version hashée du laisser-passer pour le callback de notification s'il a été configuré, ou sinon une chaîne vide.

```
dnpy string CallbackCredentials
```

**Modifiable.** Modifie le laisser-passer pour se connecter à l'adresse de callback. Le laisser-passer doit être fourni tel que retourné par la fonction `get_callbackCredentials`, sous la forme `username:hash`. La valeur du hash dépend de la méthode d'autorisation implémentée par le callback. Pour une autorisation de type Basic, le hash est le MD5 de la chaîne `username:password`. Pour une autorisation de type Digest, le hash est le MD5 de la chaîne `username:realm:password`. Pour une utilisation simplifiée, utilisez la fonction `callbackLogin`. N'oubliez pas d'appeler la méthode `saveToFlash()` du module si le réglage doit être préservé.

Encodage à utiliser pour représenter les valeurs notifiées par callback.

`dnsp` `int` **CallbackEncoding**

**Modifiable.** N'oubliez pas d'appeler la méthode `saveToFlash()` du module si le réglage doit être préservé.

---

**network**→**CallbackInitialDelay****YNetwork**

---

Attente initiale avant la première notification par callback, en secondes.

`dnsp` `int` **CallbackInitialDelay**

**Modifiable.** N'oubliez pas d'appeler la méthode `saveToFlash()` du module si le réglage doit être préservé.

## network→CallbackMaxDelay

YNetwork

---

Attente entre deux callback HTTP lorsque rien n'est à signaler, en secondes.

`dnf` `int` [CallbackMaxDelay](#)

**Modifiable.** Modifie l'attente entre deux callback HTTP lorsque rien n'est à signaler. N'oubliez pas d'appeler la méthode `saveToFlash()` du module si le réglage doit être préservé.

---

**network**→**CallbackMethod****YNetwork**

---

Méthode HTTP à utiliser pour signaler les changements d'état par callback.

`dnsp` `int` **CallbackMethod**

**Modifiable.** N'oubliez pas d'appeler la méthode `saveToFlash()` du module si le réglage doit être préservé.

## network→CallbackMinDelay

YNetwork

---

Attente minimale entre deux callbacks HTTP, en secondes.

`dnsp` `int` [CallbackMinDelay](#)

**Modifiable.** N'oubliez pas d'appeler la méthode `saveToFlash()` du module si le réglage doit être préservé.

---

**network**→**CallbackSchedule****YNetwork**

---

Planification des callbacks HTTP, sous forme de chaîne de caractères.

`dnsp` `string` **CallbackSchedule**

**Modifiable.** N'oubliez pas d'appeler la méthode `saveToFlash()` du module si le réglage doit être préservé.

## network→CallbackUrl

YNetwork

---

Adresse (URL) de callback à notifier lors de changement d'état significatifs.

`dnsp` string **CallbackUrl**

**Modifiable.** N'oubliez pas d'appeler la méthode `saveToFlash()` du module si le réglage doit être préservé.

---

**network**→**DefaultPage****YNetwork**

---

Page HTML à envoyer pour l'URL "/" **Modifiable**.

`dnp` `string` **DefaultPage**

Modifie la page HTML par défaut du hub. Si aucune valeur n'est attribuée le hub retourne index.html qui est l'interface web du hub. Il est possible de changer cet page pour un fichier qui a été uploadé sur le hub. Attention, la taille maximale permise pour le nom de fichier est de 15 caractères. Si vous changez ce paramètre, n'oubliez pas d'appeler la méthode `saveToFlash()` du module si le réglage doit être préservé.

## network→Discoverable

YNetwork

état d'activation du protocole d'annonce sur le réseau permettant de retrouver facilement le module (protocoles uPnP/Bonjour).

`dnsp` `int Discoverable`

**Modifiable.** N'oubliez pas d'appeler la méthode `saveToFlash()` du module si le réglage doit être préservé.

**network**→**FriendlyName****YNetwork**

Identifiant global de la fonction au format NOM\_MODULE . NOM\_FONCTION.

`dnf` string **FriendlyName**

Le chaîne retournée utilise soit les noms logiques du module et de la fonction si ils sont définis, soit respectivement le numéro de série du module et l'identifiant matériel de la fonction (par exemple: `MyCustomName.relay1`)

**network**→**FunctionId**

**YNetwork**

---

Identifiant matériel de l'interface réseau, sans référence au module.

`dnsp` string **FunctionId**

Par exemple `relay1`.

---

**network**→**HardwareId****YNetwork**

---

Identifiant matériel unique de la fonction au format SERIAL.FUNCTIONID.

`dnsp` string **HardwareId**

L'identifiant unique est composé du numéro de série du module et de l'identifiant matériel de la fonction (par exemple RELAYLO1-123456.relay1).

## network→HttpPort

YNetwork

---

Port TCP utilisé pour l'interface Web du hub.

`dnsp` `int HttpPort`

**Modifiable.** La valeur par défaut est le port 80, utilisé habituellement par tous les serveurs web. Indépendamment de la valeur de ce paramètre, le hub répond toujours au port 4444, qui est utilisé par défaut par la librairie de programmation Yoctopuce. Si vous changez ce paramètre, n'oubliez pas d'appeler la méthode `saveToFlash()` du module si le réglage doit être préservé.

---

**network**→**IpAddress****YNetwork**

---

Adresse IP utilisée par le module Yoctopuce.

`dnsp` `string` **IpAddress**

Il peut s'agir d'une adresse configurée statiquement, ou d'une adresse reçue par un serveur DHCP.

Vérifie si le module hébergeant la fonction est joignable, sans déclencher d'erreur.

`dnf` **bool IsOnline**

Si les valeurs des attributs en cache de la fonction sont valides au moment de l'appel, le module est considéré joignable. Cette fonction ne cause en aucun cas d'exception, quelle que soit l'erreur qui pourrait se produire lors de la vérification de joignabilité.

---

**network**→**LogicalName****YNetwork**

---

Nom logique de la fonction.

`dnf` `string LogicalName`

**Modifiable.** Vous pouvez utiliser `yCheckLogicalName()` pour vérifier si votre paramètre est valide. N'oubliez pas d'appeler la méthode `saveToFlash()` du module si le réglage doit être préservé.

## network→MacAddress

YNetwork

---

Adresse MAC de l'interface réseau, unique pour chaque module.

`dnsp` string **MacAddress**

L'adresse MAC est aussi présente sur un autocollant sur le module, représentée en chiffres et en code-barres.

---

**network**→**NtpServer****YNetwork**

---

Adresse IP du serveur de NTP à utiliser pour maintenir le module à l'heure.

`dnsp` `string NtpServer`

**Modifiable.** Modifie l'adresse IP du serveur NTP que le module doit utiliser. Utilisez une chaîne vide pour restaurer l'adresse pré-configurée en usine. N'oubliez pas d'appeler la méthode `saveToFlash()` et de redémarrer le module pour que le paramètre soit appliqué.

## network→PrimaryDNS

YNetwork

---

Adresse IP du serveur de noms primaire que le module doit utiliser.

`dnsp` string PrimaryDNS

**Modifiable.** En mode DHCP, si une valeur est spécifiée, elle remplacera celle reçue du serveur DHCP. N'oubliez pas d'appeler la méthode `saveToFlash()` et de redémarrer le module pour que le paramètre soit appliqué.

**network→Readiness****YNetwork**

état de fonctionnement atteint par l'interface réseau.

`dnsp` [int Readiness](#)

Le niveau zéro (DOWN\_0) signifie qu'aucun support réseau matériel n'a été détecté. Soit il n'y a pas de signal sur le câble réseau, soit le point d'accès sans fil choisi n'est pas détecté. Le niveau 1 (LIVE\_1) est atteint lorsque le réseau est détecté, mais n'est pas encore connecté. Pour un réseau sans fil, cela confirme l'existence du SSID configuré. Le niveau 2 (LINK\_2) est atteint lorsque le support matériel du réseau est fonctionnel. Pour une connection réseau filaire, le niveau 2 signifie que le câble est connecté aux deux bouts. Pour une connection à un point d'accès réseau sans fil, il démontre que les paramètres de sécurité configurés sont corrects. Pour une connection sans fil en mode ad-hoc, cela signifie qu'il y a au moins un partenaire sur le réseau ad-hoc. Le niveau 3 (DHCP\_3) est atteint lorsque qu'une adresse IP a été obtenue par DHCP. Le niveau 4 (DNS\_4) est atteint lorsqu'un serveur DNS est joignable par le réseau. Le niveau 5 (WWW\_5) est atteint lorsque la connectivité globale à internet est avérée par l'obtention de l'heure courante sur un serveur NTP.

## network→SecondaryDNS

YNetwork

---

Adresse IP du serveur de noms secondaire que le module doit utiliser.

`dnsp` string **SecondaryDNS**

**Modifiable.** Modifie l'adresse IP du serveur de nom secondaire que le module doit utiliser. En mode DHCP, si une valeur est spécifiée, elle remplacera celle reçue du serveur DHCP. N'oubliez pas d'appeler la méthode `saveToFlash()` et de redémarrer le module pour que le paramètre soit appliqué.

---

**network**→**SerialNumber****YNetwork**

---

Numéro de série du module, préprogrammé en usine.

`dnsp` string **SerialNumber**

## network→UserPassword

YNetwork

Chaîne de hash si un mot de passe a été configuré pour l'utilisateur "user", ou sinon une chaîne vide.

`dnsp` string **UserPassword**

**Modifiable.** Modifie le mode de passe pour l'utilisateur "user", qui devient alors instantanément nécessaire pour tout accès au module. Si la valeur fournie est une chaîne vide, plus aucun mot de passe n'est nécessaire. N'oubliez pas d'appeler la méthode `saveToFlash()` du module si le réglage doit être préservé.

---

**network**→**WwwWatchdogDelay****YNetwork**

---

Durée de perte de connection WWW tolérée (en secondes) avant de déclencher un redémarrage automatique pour tenter de récupérer la connectivité Internet.

`dnsp` `int` **WwwWatchdogDelay**

Une valeur nulle désactive le redémarrage automatique en cas de perte de connectivité WWW.

**Modifiable.** Une valeur nulle désactive le redémarrage automatique en cas de perte de connectivité WWW. La plus petite durée non-nulle utilisable est 90 secondes. N'oubliez pas d'appeler la méthode `saveToFlash()` du module si le réglage doit être préservé.

## network→callbackLogin()

YNetwork

Contacte le callback de notification et sauvegarde un laisser-passer pour s'y connecter.

js	function <b>callbackLogin</b> ( <b>username</b> , <b>password</b> )
cpp	int <b>callbackLogin</b> ( string <b>username</b> , string <b>password</b> )
m	-(int) <b>callbackLogin</b> : (NSString*) <b>username</b> : (NSString*) <b>password</b>
pas	integer <b>callbackLogin</b> ( <b>username</b> : string, <b>password</b> : string): integer
vb	function <b>callbackLogin</b> ( ByVal <b>username</b> As String, ByVal <b>password</b> As String) As Integer
cs	int <b>callbackLogin</b> ( string <b>username</b> , string <b>password</b> )
java	int <b>callbackLogin</b> ( String <b>username</b> , String <b>password</b> )
py	<b>callbackLogin</b> ( <b>username</b> , <b>password</b> )
php	function <b>callbackLogin</b> ( \$ <b>username</b> , \$ <b>password</b> )
ts	async <b>callbackLogin</b> ( <b>username</b> : string, <b>password</b> : string): Promise<number>
es	async <b>callbackLogin</b> ( <b>username</b> , <b>password</b> )
dnp	int <b>callbackLogin</b> ( string <b>username</b> , string <b>password</b> )
cp	int <b>callbackLogin</b> ( string <b>username</b> , string <b>password</b> )
cmd	YNetwork <b>target</b> <b>callbackLogin</b> <b>username</b> <b>password</b>

Le mot de passe ne sera pas stocké dans le module, mais seulement une version hashée non réversible. N'oubliez pas d'appeler la méthode `saveToFlash()` du module si le réglage doit être préservé.

**Paramètres :**

- username** nom d'utilisateur pour s'identifier au callback
- password** mot de passe pour s'identifier au callback

**Retourne :**

YAPI . SUCCESS si l'opération se déroule sans erreur.

En cas d'erreur, déclenche une exception ou retourne un code d'erreur négatif.

**network**→**clearCache()****YNetwork**

Invalide le cache.

js	function <b>clearCache</b> ( )
cpp	void <b>clearCache</b> ( )
m	-(void) <b>clearCache</b>
pas	<b>clearCache</b> ( )
vb	procedure <b>clearCache</b> ( )
cs	void <b>clearCache</b> ( )
java	void <b>clearCache</b> ( )
py	<b>clearCache</b> ( )
php	function <b>clearCache</b> ( )
ts	async <b>clearCache</b> ( ): Promise<void>
es	async <b>clearCache</b> ( )

Invalide le cache des valeurs courantes de l'interface réseau. Force le prochain appel à une méthode `get_xxx()` ou `loadxxx()` pour charger les les données depuis le module.

**network**→**describe()**

Retourne un court texte décrivant de manière non-ambigüe l'instance de l'interface réseau au format `TYPE (NAME) =SERIAL . FUNCTIONID`.

js	function <b>describe</b> ( )
cpp	string <b>describe</b> ( )
m	-(NSString*) <b>describe</b>
pas	string <b>describe</b> ( ): string
vb	function <b>describe</b> ( ) As String
cs	string <b>describe</b> ( )
java	String <b>describe</b> ( )
py	<b>describe</b> ( )
php	function <b>describe</b> ( )
ts	async <b>describe</b> ( ): Promise<string>
es	async <b>describe</b> ( )

Plus précisément, `TYPE` correspond au type de fonction, `NAME` correspond au nom utilisé lors du premier accès a la fonction, `SERIAL` correspond au numéro de série du module si le module est connecté, ou "unresolved" sinon, et `FUNCTIONID` correspond à l'identifiant matériel de la fonction si le module est connecté. Par exemple, La methode va retourner `Relay(MyCustomName.relay1)=RELAYLO1-123456.relay1` si le module est déjà connecté ou `Relay(BadCustomeName.relay1)=unresolved` si le module n'est pas déjà connecté. Cette methode ne declenche aucune transaction USB ou TCP et peut donc être utilisé dans un debugueur.

**Retourne :**

une chaîne de caractères décrivant l'interface réseau (ex:  
`Relay(MyCustomName.relay1)=RELAYLO1-123456.relay1`)

**network**→**get\_adminPassword()****YNetwork****network**→**adminPassword()**

Retourne une chaîne de hash si un mot de passe a été configuré pour l'utilisateur "admin", ou sinon une chaîne vide.

js	function <b>get_adminPassword</b> ( )
cpp	string <b>get_adminPassword</b> ( )
m	-(NSString*) <b>adminPassword</b>
pas	string <b>get_adminPassword</b> ( ): string
vb	function <b>get_adminPassword</b> ( ) As String
cs	string <b>get_adminPassword</b> ( )
java	String <b>get_adminPassword</b> ( )
uwp	async Task<string> <b>get_adminPassword</b> ( )
py	<b>get_adminPassword</b> ( )
php	function <b>get_adminPassword</b> ( )
ts	async <b>get_adminPassword</b> ( ): Promise<string>
es	async <b>get_adminPassword</b> ( )
dnp	string <b>get_adminPassword</b> ( )
cp	string <b>get_adminPassword</b> ( )
cmd	YNetwork <b>target</b> <b>get_adminPassword</b>

**Retourne :**

une chaîne de caractères représentant une chaîne de hash si un mot de passe a été configuré pour l'utilisateur "admin", ou sinon une chaîne vide

En cas d'erreur, déclenche une exception ou retourne `YNetwork.ADMINPASSWORD_INVALID`.

**network**→**get\_advertisedValue()****YNetwork****network**→**advertisedValue()**

Retourne la valeur courante de l'interface réseau (pas plus de 6 caractères).

js	function <b>get_advertisedValue</b> ( )
cpp	string <b>get_advertisedValue</b> ( )
m	-(NSString*) advertisedValue
pas	string <b>get_advertisedValue</b> ( ): string
vb	function <b>get_advertisedValue</b> ( ) As String
cs	string <b>get_advertisedValue</b> ( )
java	String <b>get_advertisedValue</b> ( )
uwp	async Task<string> <b>get_advertisedValue</b> ( )
py	<b>get_advertisedValue</b> ( )
php	function <b>get_advertisedValue</b> ( )
ts	async <b>get_advertisedValue</b> ( ): Promise<string>
es	async <b>get_advertisedValue</b> ( )
dnp	string <b>get_advertisedValue</b> ( )
cp	string <b>get_advertisedValue</b> ( )
cmd	YNetwork <b>target</b> <b>get_advertisedValue</b>

**Retourne :**

une chaîne de caractères représentant la valeur courante de l'interface réseau (pas plus de 6 caractères).

En cas d'erreur, déclenche une exception ou retourne `YNetwork.ADVERTISEDVALUE_INVALID`.

**network**→**get\_callbackCredentials()****YNetwork****network**→**callbackCredentials()**

Retourne une version hashée du laisser-passer pour le callback de notification s'il a été configuré, ou sinon une chaîne vide.

js	function <b>get_callbackCredentials</b> ( )
cpp	string <b>get_callbackCredentials</b> ( )
m	-(NSString*) <b>callbackCredentials</b>
pas	string <b>get_callbackCredentials</b> ( ): string
vb	function <b>get_callbackCredentials</b> ( ) As String
cs	string <b>get_callbackCredentials</b> ( )
java	String <b>get_callbackCredentials</b> ( )
uwp	async Task<string> <b>get_callbackCredentials</b> ( )
py	<b>get_callbackCredentials</b> ( )
php	function <b>get_callbackCredentials</b> ( )
ts	async <b>get_callbackCredentials</b> ( ): Promise<string>
es	async <b>get_callbackCredentials</b> ( )
dnp	string <b>get_callbackCredentials</b> ( )
cp	string <b>get_callbackCredentials</b> ( )
cmd	YNetwork <b>target</b> <b>get_callbackCredentials</b>

**Retourne :**

une chaîne de caractères représentant une version hashée du laisser-passer pour le callback de notification s'il a été configuré, ou sinon une chaîne vide

En cas d'erreur, déclenche une exception ou retourne `YNetwork.CALLBACKCREDENTIALS_INVALID`.

**network**→**get\_callbackEncoding()****network**→**callbackEncoding()**

Retourne l'encodage à utiliser pour représenter les valeurs notifiées par callback.

js	function <b>get_callbackEncoding</b> ( )
cpp	Y_CALLBACKENCODING_enum <b>get_callbackEncoding</b> ( )
m	-(Y_CALLBACKENCODING_enum) callbackEncoding
pas	Integer <b>get_callbackEncoding</b> ( ): Integer
vb	function <b>get_callbackEncoding</b> ( ) As Integer
cs	int <b>get_callbackEncoding</b> ( )
java	int <b>get_callbackEncoding</b> ( )
uwp	async Task<int> <b>get_callbackEncoding</b> ( )
py	<b>get_callbackEncoding</b> ( )
php	function <b>get_callbackEncoding</b> ( )
ts	async <b>get_callbackEncoding</b> ( ): Promise<YNetwork_CallbackEncoding>
es	async <b>get_callbackEncoding</b> ( )
dnp	int <b>get_callbackEncoding</b> ( )
cp	int <b>get_callbackEncoding</b> ( )
cmd	YNetwork <b>target</b> <b>get_callbackEncoding</b>

**Retourne :**

une valeur parmi YNetwork.CALLBACKENCODING\_FORM, YNetwork.CALLBACKENCODING\_JSON, YNetwork.CALLBACKENCODING\_JSON\_ARRAY, YNetwork.CALLBACKENCODING\_CSV, YNetwork.CALLBACKENCODING\_YOCTO\_API, YNetwork.CALLBACKENCODING\_JSON\_NUM, YNetwork.CALLBACKENCODING\_EMONCMS, YNetwork.CALLBACKENCODING\_AZURE, YNetwork.CALLBACKENCODING\_INFLUXDB, YNetwork.CALLBACKENCODING\_MQTT, YNetwork.CALLBACKENCODING\_YOCTO\_API\_JSON, YNetwork.CALLBACKENCODING\_PRTG et YNetwork.CALLBACKENCODING\_INFLUXDB\_V2 représentant l'encodage à utiliser pour représenter les valeurs notifiées par callback

En cas d'erreur, déclenche une exception ou retourne YNetwork.CALLBACKENCODING\_INVALID.

**network**→**get\_callbackInitialDelay()****YNetwork****network**→**callbackInitialDelay()**

Retourne l'attente initiale avant la première notification par callback, en secondes.

js	function <b>get_callbackInitialDelay</b> ( )
cpp	int <b>get_callbackInitialDelay</b> ( )
m	-(int) <b>callbackInitialDelay</b>
pas	LongInt <b>get_callbackInitialDelay</b> ( ): LongInt
vb	function <b>get_callbackInitialDelay</b> ( ) As Integer
cs	int <b>get_callbackInitialDelay</b> ( )
java	int <b>get_callbackInitialDelay</b> ( )
uwp	async Task<int> <b>get_callbackInitialDelay</b> ( )
py	<b>get_callbackInitialDelay</b> ( )
php	function <b>get_callbackInitialDelay</b> ( )
ts	async <b>get_callbackInitialDelay</b> ( ): Promise<number>
es	async <b>get_callbackInitialDelay</b> ( )
dnp	int <b>get_callbackInitialDelay</b> ( )
cp	int <b>get_callbackInitialDelay</b> ( )
cmd	YNetwork <b>target</b> <b>get_callbackInitialDelay</b>

**Retourne :**

un entier représentant l'attente initiale avant la première notification par callback, en secondes

En cas d'erreur, déclenche une exception ou retourne `YNetwork.CALLBACKINITIALDELAY_INVALID`.

**network**→**get\_callbackMaxDelay()****YNetwork****network**→**callbackMaxDelay()**

Retourne l'attente entre deux callback HTTP lorsque rien n'est à signaler, en secondes.

js	function <b>get_callbackMaxDelay</b> ( )
cpp	int <b>get_callbackMaxDelay</b> ( )
m	-(int) callbackMaxDelay
pas	LongInt <b>get_callbackMaxDelay</b> ( ): LongInt
vb	function <b>get_callbackMaxDelay</b> ( ) As Integer
cs	int <b>get_callbackMaxDelay</b> ( )
java	int <b>get_callbackMaxDelay</b> ( )
uwp	async Task<int> <b>get_callbackMaxDelay</b> ( )
py	<b>get_callbackMaxDelay</b> ( )
php	function <b>get_callbackMaxDelay</b> ( )
ts	async <b>get_callbackMaxDelay</b> ( ): Promise<number>
es	async <b>get_callbackMaxDelay</b> ( )
dnp	int <b>get_callbackMaxDelay</b> ( )
cp	int <b>get_callbackMaxDelay</b> ( )
cmd	YNetwork <b>target</b> <b>get_callbackMaxDelay</b>

**Retourne :**

un entier représentant l'attente entre deux callback HTTP lorsque rien n'est à signaler, en secondes

En cas d'erreur, déclenche une exception ou retourne `YNetwork.CALLBACKMAXDELAY_INVALID`.

**network**→**get\_callbackMethod()****YNetwork****network**→**callbackMethod()**

Retourne la méthode HTTP à utiliser pour signaler les changements d'état par callback.

js	function <b>get_callbackMethod</b> ( )
cpp	Y_CALLBACKMETHOD_enum <b>get_callbackMethod</b> ( )
m	-(Y_CALLBACKMETHOD_enum) callbackMethod
pas	Integer <b>get_callbackMethod</b> ( ): Integer
vb	function <b>get_callbackMethod</b> ( ) As Integer
cs	int <b>get_callbackMethod</b> ( )
java	int <b>get_callbackMethod</b> ( )
uwp	async Task<int> <b>get_callbackMethod</b> ( )
py	<b>get_callbackMethod</b> ( )
php	function <b>get_callbackMethod</b> ( )
ts	async <b>get_callbackMethod</b> ( ): Promise<YNetwork_CallbackMethod>
es	async <b>get_callbackMethod</b> ( )
dnf	int <b>get_callbackMethod</b> ( )
cp	int <b>get_callbackMethod</b> ( )
cmd	YNetwork <b>target</b> <b>get_callbackMethod</b>

**Retourne :**

une valeur parmi `YNetwork.CALLBACKMETHOD_POST`, `YNetwork.CALLBACKMETHOD_GET` et `YNetwork.CALLBACKMETHOD_PUT` représentant la méthode HTTP à utiliser pour signaler les changements d'état par callback

En cas d'erreur, déclenche une exception ou retourne `YNetwork.CALLBACKMETHOD_INVALID`.

**network**→**get\_callbackMinDelay()****YNetwork****network**→**callbackMinDelay()**

Retourne l'attente minimale entre deux callbacks HTTP, en secondes.

js	function <b>get_callbackMinDelay</b> ( )
cpp	int <b>get_callbackMinDelay</b> ( )
m	-(int) callbackMinDelay
pas	LongInt <b>get_callbackMinDelay</b> ( ): LongInt
vb	function <b>get_callbackMinDelay</b> ( ) As Integer
cs	int <b>get_callbackMinDelay</b> ( )
java	int <b>get_callbackMinDelay</b> ( )
uwp	async Task<int> <b>get_callbackMinDelay</b> ( )
py	<b>get_callbackMinDelay</b> ( )
php	function <b>get_callbackMinDelay</b> ( )
ts	async <b>get_callbackMinDelay</b> ( ): Promise<number>
es	async <b>get_callbackMinDelay</b> ( )
dnp	int <b>get_callbackMinDelay</b> ( )
cp	int <b>get_callbackMinDelay</b> ( )
cmd	YNetwork <b>target</b> <b>get_callbackMinDelay</b>

**Retourne :**

un entier représentant l'attente minimale entre deux callbacks HTTP, en secondes

En cas d'erreur, déclenche une exception ou retourne `YNetwork.CALLBACKMINDELAY_INVALID`.

**network**→**get\_callbackSchedule()****YNetwork****network**→**callbackSchedule()**

Retourne la planification des callbacks HTTP, sous forme de chaîne de caractères.

js	function <b>get_callbackSchedule</b> ( )
cpp	string <b>get_callbackSchedule</b> ( )
m	-(NSString*) <b>callbackSchedule</b>
pas	string <b>get_callbackSchedule</b> ( ): string
vb	function <b>get_callbackSchedule</b> ( ) As String
cs	string <b>get_callbackSchedule</b> ( )
java	String <b>get_callbackSchedule</b> ( )
uwp	async Task<string> <b>get_callbackSchedule</b> ( )
py	<b>get_callbackSchedule</b> ( )
php	function <b>get_callbackSchedule</b> ( )
ts	async <b>get_callbackSchedule</b> ( ): Promise<string>
es	async <b>get_callbackSchedule</b> ( )
dnf	string <b>get_callbackSchedule</b> ( )
cp	string <b>get_callbackSchedule</b> ( )
cmd	YNetwork <b>target</b> <b>get_callbackSchedule</b>

**Retourne :**

une chaîne de caractères représentant la planification des callbacks HTTP, sous forme de chaîne de caractères

En cas d'erreur, déclenche une exception ou retourne `YNetwork.CALLBACKSCHEDULE_INVALID`.

**network**→**get\_callbackUrl()****network**→**callbackUrl()**

Retourne l'adresse (URL) de callback à notifier lors de changement d'état significatifs.

js	function <b>get_callbackUrl</b> ( )
cpp	string <b>get_callbackUrl</b> ( )
m	-(NSString*) <b>callbackUrl</b>
pas	string <b>get_callbackUrl</b> ( ): string
vb	function <b>get_callbackUrl</b> ( ) As String
cs	string <b>get_callbackUrl</b> ( )
java	String <b>get_callbackUrl</b> ( )
uwp	async Task<string> <b>get_callbackUrl</b> ( )
py	<b>get_callbackUrl</b> ( )
php	function <b>get_callbackUrl</b> ( )
ts	async <b>get_callbackUrl</b> ( ): Promise<string>
es	async <b>get_callbackUrl</b> ( )
dnp	string <b>get_callbackUrl</b> ( )
cp	string <b>get_callbackUrl</b> ( )
cmd	YNetwork <b>target</b> <b>get_callbackUrl</b>

**Retourne :**

une chaîne de caractères représentant l'adresse (URL) de callback à notifier lors de changement d'état significatifs

En cas d'erreur, déclenche une exception ou retourne `YNetwork.CALLBACKURL_INVALID`.

**network**→**get\_defaultPage()****YNetwork****network**→**defaultPage()**

Retourne la page HTML à envoyer pour l'URL "/"

js	function <b>get_defaultPage</b> ( )
cpp	string <b>get_defaultPage</b> ( )
m	-(NSString*) defaultPage
pas	string <b>get_defaultPage</b> ( ): string
vb	function <b>get_defaultPage</b> ( ) As String
cs	string <b>get_defaultPage</b> ( )
java	String <b>get_defaultPage</b> ( )
uwp	async Task<string> <b>get_defaultPage</b> ( )
py	<b>get_defaultPage</b> ( )
php	function <b>get_defaultPage</b> ( )
ts	async <b>get_defaultPage</b> ( ): Promise<string>
es	async <b>get_defaultPage</b> ( )
dnp	string <b>get_defaultPage</b> ( )
cp	string <b>get_defaultPage</b> ( )
cmd	YNetwork <b>target</b> <b>get_defaultPage</b>

**Retourne :**

une chaîne de caractères représentant la page HTML à envoyer pour l'URL "/"

En cas d'erreur, déclenche une exception ou retourne `YNetwork.DEFAULTPAGE_INVALID`.

**network**→**get\_discoverable()****YNetwork****network**→**discoverable()**

Retourne l'état d'activation du protocole d'annonce sur le réseau permettant de retrouver facilement le module (protocoles uPnP/Bonjour).

js	function <b>get_discoverable</b> ( )
cpp	Y_DISCOVERABLE_enum <b>get_discoverable</b> ( )
m	-(Y_DISCOVERABLE_enum) discoverable
pas	Integer <b>get_discoverable</b> ( ): Integer
vb	function <b>get_discoverable</b> ( ) As Integer
cs	int <b>get_discoverable</b> ( )
java	int <b>get_discoverable</b> ( )
uwp	async Task<int> <b>get_discoverable</b> ( )
py	<b>get_discoverable</b> ( )
php	function <b>get_discoverable</b> ( )
ts	async <b>get_discoverable</b> ( ): Promise<YNetwork_Discoverable>
es	async <b>get_discoverable</b> ( )
dnp	int <b>get_discoverable</b> ( )
cp	int <b>get_discoverable</b> ( )
cmd	YNetwork <b>target get_discoverable</b>

**Retourne :**

soit `YNetwork.DISCOVERABLE_FALSE`, soit `YNetwork.DISCOVERABLE_TRUE`, selon l'état d'activation du protocole d'annonce sur le réseau permettant de retrouver facilement le module (protocoles uPnP/Bonjour)

En cas d'erreur, déclenche une exception ou retourne `YNetwork.DISCOVERABLE_INVALID`.

**network**→**get\_errorMessage()****YNetwork****network**→**errorMessage()**

Retourne le message correspondant à la dernière erreur survenue lors de l'utilisation de l'interface réseau.

js	function <b>get_errorMessage</b> ( )
cpp	string <b>get_errorMessage</b> ( )
m	-(NSString*) <b>errorMessage</b>
pas	string <b>get_errorMessage</b> ( ): string
vb	function <b>get_errorMessage</b> ( ) As String
cs	string <b>get_errorMessage</b> ( )
java	String <b>get_errorMessage</b> ( )
py	<b>get_errorMessage</b> ( )
php	function <b>get_errorMessage</b> ( )
ts	<b>get_errorMessage</b> ( ): string
es	<b>get_errorMessage</b> ( )

Cette méthode est principalement utile lorsque la librairie Yoctopuce est utilisée en désactivant la gestion des exceptions.

**Retourne :**

une chaîne de caractères correspondant au message de la dernière erreur qui s'est produit lors de l'utilisation de l'interface réseau.

**network**→**get\_errorType()****network**→**errorType()**

Retourne le code d'erreur correspondant à la dernière erreur survenue lors de l'utilisation de l'interface réseau.

js	function <b>get_errorType</b> ( )
cpp	YRETCODE <b>get_errorType</b> ( )
m	-(YRETCODE) errorType
pas	YRETCODE <b>get_errorType</b> ( ): YRETCODE
vb	function <b>get_errorType</b> ( ) As YRETCODE
cs	YRETCODE <b>get_errorType</b> ( )
java	int <b>get_errorType</b> ( )
py	<b>get_errorType</b> ( )
php	function <b>get_errorType</b> ( )
ts	<b>get_errorType</b> ( ): number
es	<b>get_errorType</b> ( )

Cette méthode est principalement utile lorsque la librairie Yoctopuce est utilisée en désactivant la gestion des exceptions.

**Retourne :**

un nombre correspondant au code de la dernière erreur qui s'est produit lors de l'utilisation de l'interface réseau.

**network**→**get\_friendlyName()****YNetwork****network**→**friendlyName()**

Retourne un identifiant global de l'interface réseau au format `NOM_MODULE . NOM_FONCTION`.

js	function <b>get_friendlyName</b> ( )
cpp	string <b>get_friendlyName</b> ( )
m	-(NSString*) friendlyName
cs	string <b>get_friendlyName</b> ( )
java	String <b>get_friendlyName</b> ( )
py	<b>get_friendlyName</b> ( )
php	function <b>get_friendlyName</b> ( )
ts	async <b>get_friendlyName</b> ( ): Promise<string>
es	async <b>get_friendlyName</b> ( )
dnf	string <b>get_friendlyName</b> ( )
cp	string <b>get_friendlyName</b> ( )

Le chaîne retournée utilise soit les noms logiques du module et de l'interface réseau si ils sont définis, soit respectivement le numéro de série du module et l'identifiant matériel de l'interface réseau (par exemple: `MyCustomName.relay1`)

**Retourne :**

une chaîne de caractères identifiant l'interface réseau en utilisant les noms logiques (ex: `MyCustomName.relay1`)

En cas d'erreur, déclenche une exception ou retourne `YNetwork.FRIENDLYNAME_INVALID`.

**network**→**get\_functionDescriptor()****network**→**functionDescriptor()**

Retourne un identifiant unique de type YFUN\_DESCR correspondant à la fonction.

js	function <b>get_functionDescriptor</b> ( )
cpp	YFUN_DESCR <b>get_functionDescriptor</b> ( )
m	-(YFUN_DESCR) functionDescriptor
pas	YFUN_DESCR <b>get_functionDescriptor</b> ( ): YFUN_DESCR
vb	function <b>get_functionDescriptor</b> ( ) As YFUN_DESCR
cs	YFUN_DESCR <b>get_functionDescriptor</b> ( )
java	String <b>get_functionDescriptor</b> ( )
py	<b>get_functionDescriptor</b> ( )
php	function <b>get_functionDescriptor</b> ( )
ts	async <b>get_functionDescriptor</b> ( ): Promise<string>
es	async <b>get_functionDescriptor</b> ( )

Cet identifiant peut être utilisé pour tester si deux instance de YFunction référencent physiquement la même fonction sur le même module.

**Retourne :**

un identifiant de type YFUN\_DESCR.

Si la fonction n'a jamais été contactée, la valeur retournée sera Y\$CLASSNAME\$.FUNCTIONDESCRIPTOR\_INVALID

**network**→**get\_functionId()****YNetwork****network**→**functionId()**

Retourne l'identifiant matériel de l'interface réseau, sans référence au module.

js	function <b>get_functionId</b> ( )
cpp	string <b>get_functionId</b> ( )
m	-(NSString*) <b>functionId</b>
vb	function <b>get_functionId</b> ( ) As String
cs	string <b>get_functionId</b> ( )
java	String <b>get_functionId</b> ( )
py	<b>get_functionId</b> ( )
php	function <b>get_functionId</b> ( )
ts	async <b>get_functionId</b> ( ): Promise<string>
es	async <b>get_functionId</b> ( )
dnp	string <b>get_functionId</b> ( )
cp	string <b>get_functionId</b> ( )

Par exemple `relay1`.

**Retourne :**

une chaîne de caractères identifiant l'interface réseau (ex: `relay1`)

En cas d'erreur, déclenche une exception ou retourne `YNetwork.FUNCTIONID_INVALID`.

**network**→**get\_hardwareId()****network**→**hardwareId()**

Retourne l'identifiant matériel unique de l'interface réseau au format `SERIAL.FUNCTIONID`.

js	function <b>get_hardwareId</b> ( )
cpp	string <b>get_hardwareId</b> ( )
m	-(NSString*) hardwareId
vb	function <b>get_hardwareId</b> ( ) As String
cs	string <b>get_hardwareId</b> ( )
java	String <b>get_hardwareId</b> ( )
py	<b>get_hardwareId</b> ( )
php	function <b>get_hardwareId</b> ( )
ts	async <b>get_hardwareId</b> ( ): Promise<string>
es	async <b>get_hardwareId</b> ( )
dnp	string <b>get_hardwareId</b> ( )
cp	string <b>get_hardwareId</b> ( )

L'identifiant unique est composé du numéro de série du module et de l'identifiant matériel de l'interface réseau (par exemple `RELAYLO1-123456.relay1`).

**Retourne :**

une chaîne de caractères identifiant l'interface réseau (ex: `RELAYLO1-123456.relay1`)

En cas d'erreur, déclenche une exception ou retourne `YNetwork.HARDWAREID_INVALID`.

**network**→**get\_httpPort()****YNetwork****network**→**httpPort()**

Retourne le port TCP utilisé pour l'interface Web du hub.

js	function <b>get_httpPort</b> ( )
cpp	int <b>get_httpPort</b> ( )
m	-(int) httpPort
pas	LongInt <b>get_httpPort</b> ( ): LongInt
vb	function <b>get_httpPort</b> ( ) As Integer
cs	int <b>get_httpPort</b> ( )
java	int <b>get_httpPort</b> ( )
uwp	async Task<int> <b>get_httpPort</b> ( )
py	<b>get_httpPort</b> ( )
php	function <b>get_httpPort</b> ( )
ts	async <b>get_httpPort</b> ( ): Promise<number>
es	async <b>get_httpPort</b> ( )
dnp	int <b>get_httpPort</b> ( )
cp	int <b>get_httpPort</b> ( )
cmd	YNetwork <b>target get_httpPort</b>

**Retourne :**

un entier représentant le port TCP utilisé pour l'interface Web du hub

En cas d'erreur, déclenche une exception ou retourne `YNetwork.HTTPPORT_INVALID`.

**network**→**get\_ipAddress()****network**→**ipAddress()**

Retourne l'adresse IP utilisée par le module Yoctopuce.

js	function <b>get_ipAddress</b> ( )
cpp	string <b>get_ipAddress</b> ( )
m	-(NSString*) ipAddress
pas	string <b>get_ipAddress</b> ( ): string
vb	function <b>get_ipAddress</b> ( ) As String
cs	string <b>get_ipAddress</b> ( )
java	String <b>get_ipAddress</b> ( )
uwp	async Task<string> <b>get_ipAddress</b> ( )
py	<b>get_ipAddress</b> ( )
php	function <b>get_ipAddress</b> ( )
ts	async <b>get_ipAddress</b> ( ): Promise<string>
es	async <b>get_ipAddress</b> ( )
dnp	string <b>get_ipAddress</b> ( )
cp	string <b>get_ipAddress</b> ( )
cmd	YNetwork <b>target</b> <b>get_ipAddress</b>

Il peut s'agir d'une adresse configurée statiquement, ou d'une adresse reçue par un serveur DHCP.

**Retourne :**

une chaîne de caractères représentant l'adresse IP utilisée par le module Yoctopuce

En cas d'erreur, déclenche une exception ou retourne `YNetwork.IPADDRESS_INVALID`.

**network**→**get\_ipConfig()****YNetwork****network**→**ipConfig()**

Retourne la configuration IP de l'interface réseau.

js	function <b>get_ipConfig</b> ( )
cpp	string <b>get_ipConfig</b> ( )
m	-(NSString*) ipConfig
pas	string <b>get_ipConfig</b> ( ): string
vb	function <b>get_ipConfig</b> ( ) As String
cs	string <b>get_ipConfig</b> ( )
java	String <b>get_ipConfig</b> ( )
uwp	async Task<string> <b>get_ipConfig</b> ( )
py	<b>get_ipConfig</b> ( )
php	function <b>get_ipConfig</b> ( )
ts	async <b>get_ipConfig</b> ( ): Promise<string>
es	async <b>get_ipConfig</b> ( )
dnp	string <b>get_ipConfig</b> ( )
cp	string <b>get_ipConfig</b> ( )
cmd	YNetwork <b>target get_ipConfig</b>

Si l'interface réseau est configurée pour utiliser une adresse IP assignée manuellement (adresse IP statique) la chaîne commence par "STATIC:" et est suivie pas l'adresse IP, la longueur du masque de sous-réseau et l'adresse IP de la passerelle. Ces trois paramètres sont séparés par le caractère "/". Par exemple: "STATIC:192.168.1.14/16/192.168.1.1"

Si l'interface réseau est configurée pour utiliser une adresse assignée automatiquement par DHCP la chaîne commence par "DHCP:" et est suivie d'une adresse IP, d'une longueur du masque de sous-réseau et d'une adresse IP de passerelle. Ces trois paramètres sont séparés par le caractère "/" et sont utilisés si aucun serveur DHCP ne répond.

**Retourne :**

une chaîne de caractères représentant la configuration IP de l'interface réseau

En cas d'erreur, déclenche une exception ou retourne `YNetwork.IPCONFIG_INVALID`.

**network**→**get\_logicalName()****network**→**logicalName()**

Retourne le nom logique de l'interface réseau.

js	function <b>get_logicalName</b> ( )
cpp	string <b>get_logicalName</b> ( )
m	-(NSString*) logicalName
pas	string <b>get_logicalName</b> ( ): string
vb	function <b>get_logicalName</b> ( ) As String
cs	string <b>get_logicalName</b> ( )
java	String <b>get_logicalName</b> ( )
uwp	async Task<string> <b>get_logicalName</b> ( )
py	<b>get_logicalName</b> ( )
php	function <b>get_logicalName</b> ( )
ts	async <b>get_logicalName</b> ( ): Promise<string>
es	async <b>get_logicalName</b> ( )
dnp	string <b>get_logicalName</b> ( )
cp	string <b>get_logicalName</b> ( )
cmd	YNetwork <b>target</b> <b>get_logicalName</b>

**Retourne :**

une chaîne de caractères représentant le nom logique de l'interface réseau.

En cas d'erreur, déclenche une exception ou retourne `YNetwork.LOGICALNAME_INVALID`.

**network**→**get\_macAddress()****YNetwork****network**→**macAddress()**

Retourne l'adresse MAC de l'interface réseau, unique pour chaque module.

js	function <b>get_macAddress</b> ( )
cpp	string <b>get_macAddress</b> ( )
m	-(NSString*) <b>macAddress</b>
pas	string <b>get_macAddress</b> ( ): string
vb	function <b>get_macAddress</b> ( ) As String
cs	string <b>get_macAddress</b> ( )
java	String <b>get_macAddress</b> ( )
uwp	async Task<string> <b>get_macAddress</b> ( )
py	<b>get_macAddress</b> ( )
php	function <b>get_macAddress</b> ( )
ts	async <b>get_macAddress</b> ( ): Promise<string>
es	async <b>get_macAddress</b> ( )
dnp	string <b>get_macAddress</b> ( )
cp	string <b>get_macAddress</b> ( )
cmd	YNetwork <b>target</b> <b>get_macAddress</b>

L'adresse MAC est aussi présente sur un autocollant sur le module, représentée en chiffres et en code-barres.

**Retourne :**

une chaîne de caractères représentant l'adresse MAC de l'interface réseau, unique pour chaque module

En cas d'erreur, déclenche une exception ou retourne `YNetwork.MACADDRESS_INVALID`.

**network**→**get\_module()****network**→**module()**

Retourne l'objet `YModule` correspondant au module Yoctopuce qui héberge la fonction.

js	function <b>get_module</b> ( )
cpp	<code>YModule *</code> <b>get_module</b> ( )
m	-( <code>YModule*</code> ) <b>module</b>
pas	<code>TYModule</code> <b>get_module</b> ( ): <code>TYModule</code>
vb	function <b>get_module</b> ( ) As <code>YModule</code>
cs	<code>YModule</code> <b>get_module</b> ( )
java	<code>YModule</code> <b>get_module</b> ( )
py	<b>get_module</b> ( )
php	function <b>get_module</b> ( )
ts	async <b>get_module</b> ( ): <code>Promise&lt;YModule&gt;</code>
es	async <b>get_module</b> ( )
dnp	<code>YModuleProxy</code> <b>get_module</b> ( )
cp	<code>YModuleProxy *</code> <b>get_module</b> ( )

Si la fonction ne peut être trouvée sur aucun module, l'instance de `YModule` retournée ne sera pas joignable.

**Retourne :**

une instance de `YModule`

**network**→**get\_module\_async()****YNetwork****network**→**module\_async()**

Retourne l'objet `YModule` correspondant au module Yoctopuce qui héberge la fonction.

```
js function get_module_async( callback, context)
```

Si la fonction ne peut être trouvée sur aucun module, l'instance de `YModule` retournée ne sera pas joignable.

Cette version asynchrone n'existe qu'en Javascript. Elle utilise une fonction de callback plutôt qu'une simple valeur de retour, pour éviter de bloquer la VM Javascript de Firefox, qui n'implémente pas le passage de contrôle entre threads durant les appels d'entrée/sortie bloquants.

**Paramètres :**

**callback** fonction de callback qui sera appelée dès que le résultat sera connu. La fonction callback reçoit trois arguments: le contexte fourni par l'appelant, l'objet fonction concerné et l'instance demandée de `YModule`

**context** contexte fourni par l'appelant, et qui sera passé tel-quel à la fonction de callback

**Retourne :**

rien du tout : le résultat sera passé en paramètre à la fonction de callback.

**network**→**get\_ntpServer()****network**→**ntpServer()**

Retourne l'adresse IP du serveur de NTP à utiliser pour maintenir le module à l'heure.

js	function <b>get_ntpServer</b> ( )
cpp	string <b>get_ntpServer</b> ( )
m	-(NSString*) ntpServer
pas	string <b>get_ntpServer</b> ( ): string
vb	function <b>get_ntpServer</b> ( ) As String
cs	string <b>get_ntpServer</b> ( )
java	String <b>get_ntpServer</b> ( )
uwp	async Task<string> <b>get_ntpServer</b> ( )
py	<b>get_ntpServer</b> ( )
php	function <b>get_ntpServer</b> ( )
ts	async <b>get_ntpServer</b> ( ): Promise<string>
es	async <b>get_ntpServer</b> ( )
dnp	string <b>get_ntpServer</b> ( )
cp	string <b>get_ntpServer</b> ( )
cmd	YNetwork <b>target</b> <b>get_ntpServer</b>

**Retourne :**

une chaîne de caractères représentant l'adresse IP du serveur de NTP à utiliser pour maintenir le module à l'heure

En cas d'erreur, déclenche une exception ou retourne `YNetwork.NTPSERVER_INVALID`.

**network**→**get\_poeCurrent()****YNetwork****network**→**poeCurrent()**

Retourne le courant consommé par le module depuis Power-over-Ethernet (PoE), en milliampères.

js	function <b>get_poeCurrent</b> ( )
cpp	int <b>get_poeCurrent</b> ( )
m	-(int) poeCurrent
pas	LongInt <b>get_poeCurrent</b> ( ): LongInt
vb	function <b>get_poeCurrent</b> ( ) As Integer
cs	int <b>get_poeCurrent</b> ( )
java	int <b>get_poeCurrent</b> ( )
uwp	async Task<int> <b>get_poeCurrent</b> ( )
py	<b>get_poeCurrent</b> ( )
php	function <b>get_poeCurrent</b> ( )
ts	async <b>get_poeCurrent</b> ( ): Promise<number>
es	async <b>get_poeCurrent</b> ( )
dnp	int <b>get_poeCurrent</b> ( )
cp	int <b>get_poeCurrent</b> ( )
cmd	YNetwork <b>target</b> <b>get_poeCurrent</b>

La consommation est mesurée après conversion en 5 Volt, et ne doit jamais dépasser 1800 mA.

**Retourne :**

un entier représentant le courant consommé par le module depuis Power-over-Ethernet (PoE), en milliampères

En cas d'erreur, déclenche une exception ou retourne `YNetwork.POECURRENT_INVALID`.

**network**→**get\_primaryDNS()****YNetwork****network**→**primaryDNS()**

Retourne l'adresse IP du serveur de noms primaire que le module doit utiliser.

js	function <b>get_primaryDNS</b> ( )
cpp	string <b>get_primaryDNS</b> ( )
m	-(NSString*) <b>primaryDNS</b>
pas	string <b>get_primaryDNS</b> ( ): string
vb	function <b>get_primaryDNS</b> ( ) As String
cs	string <b>get_primaryDNS</b> ( )
java	String <b>get_primaryDNS</b> ( )
uwp	async Task<string> <b>get_primaryDNS</b> ( )
py	<b>get_primaryDNS</b> ( )
php	function <b>get_primaryDNS</b> ( )
ts	async <b>get_primaryDNS</b> ( ): Promise<string>
es	async <b>get_primaryDNS</b> ( )
dnp	string <b>get_primaryDNS</b> ( )
cp	string <b>get_primaryDNS</b> ( )
cmd	YNetwork <b>target</b> <b>get_primaryDNS</b>

**Retourne :**

une chaîne de caractères représentant l'adresse IP du serveur de noms primaire que le module doit utiliser

En cas d'erreur, déclenche une exception ou retourne `YNetwork.PRIMARYDNS_INVALID`.

**network→get\_readiness()****YNetwork****network→readiness()**

Retourne l'état de fonctionnement atteint par l'interface réseau.

js	function <b>get_readiness</b> ( )
cpp	Y_READINESS_enum <b>get_readiness</b> ( )
m	-(Y_READINESS_enum) readiness
pas	Integer <b>get_readiness</b> ( ): Integer
vb	function <b>get_readiness</b> ( ) As Integer
cs	int <b>get_readiness</b> ( )
java	int <b>get_readiness</b> ( )
uwp	async Task<int> <b>get_readiness</b> ( )
py	<b>get_readiness</b> ( )
php	function <b>get_readiness</b> ( )
ts	async <b>get_readiness</b> ( ): Promise<YNetwork_Readiness>
es	async <b>get_readiness</b> ( )
dnf	int <b>get_readiness</b> ( )
cp	int <b>get_readiness</b> ( )
cmd	YNetwork <b>target get_readiness</b>

Le niveau zéro (DOWN\_0) signifie qu'aucun support réseau matériel n'a été détecté. Soit il n'y a pas de signal sur le câble réseau, soit le point d'accès sans fil choisi n'est pas détecté. Le niveau 1 (LIVE\_1) est atteint lorsque le réseau est détecté, mais n'est pas encore connecté. Pour un réseau sans fil, cela confirme l'existence du SSID configuré. Le niveau 2 (LINK\_2) est atteint lorsque le support matériel du réseau est fonctionnel. Pour une connexion réseau filaire, le niveau 2 signifie que le câble est connecté aux deux bouts. Pour une connexion à un point d'accès réseau sans fil, il démontre que les paramètres de sécurité configurés sont corrects. Pour une connexion sans fil en mode ad-hoc, cela signifie qu'il y a au moins un partenaire sur le réseau ad-hoc. Le niveau 3 (DHCP\_3) est atteint lorsque qu'une adresse IP a été obtenue par DHCP. Le niveau 4 (DNS\_4) est atteint lorsqu'un serveur DNS est joignable par le réseau. Le niveau 5 (WWW\_5) est atteint lorsque la connectivité globale à internet est avérée par l'obtention de l'heure courante sur un serveur NTP.

**Retourne :**

une valeur parmi YNetwork.READINESS\_DOWN, YNetwork.READINESS\_EXISTS, YNetwork.READINESS\_LINKED, YNetwork.READINESS\_LAN\_OK et YNetwork.READINESS\_WWW\_OK représentant l'état de fonctionnement atteint par l'interface réseau

En cas d'erreur, déclenche une exception ou retourne YNetwork.READINESS\_INVALID.

**network**→**get\_router()****network**→**router()**

Retourne l'adresse IP du routeur (passerelle) utilisé par le module (*default gateway*).

js	function <b>get_router</b> ( )
cpp	string <b>get_router</b> ( )
m	-(NSString*) router
pas	string <b>get_router</b> ( ): string
vb	function <b>get_router</b> ( ) As String
cs	string <b>get_router</b> ( )
java	String <b>get_router</b> ( )
uwp	async Task<string> <b>get_router</b> ( )
py	<b>get_router</b> ( )
php	function <b>get_router</b> ( )
ts	async <b>get_router</b> ( ): Promise<string>
es	async <b>get_router</b> ( )
dnp	string <b>get_router</b> ( )
cp	string <b>get_router</b> ( )
cmd	YNetwork <b>target get_router</b>

**Retourne :**

une chaîne de caractères représentant l'adresse IP du routeur (passerelle) utilisé par le module (*default gateway*)

En cas d'erreur, déclenche une exception ou retourne `YNetwork.ROUTER_INVALID`.

**network**→**get\_secondaryDNS()****YNetwork****network**→**secondaryDNS()**

Retourne l'adresse IP du serveur de noms secondaire que le module doit utiliser.

js	function <b>get_secondaryDNS</b> ( )
cpp	string <b>get_secondaryDNS</b> ( )
m	-(NSString*) <b>secondaryDNS</b>
pas	string <b>get_secondaryDNS</b> ( ): string
vb	function <b>get_secondaryDNS</b> ( ) As String
cs	string <b>get_secondaryDNS</b> ( )
java	String <b>get_secondaryDNS</b> ( )
uwp	async Task<string> <b>get_secondaryDNS</b> ( )
py	<b>get_secondaryDNS</b> ( )
php	function <b>get_secondaryDNS</b> ( )
ts	async <b>get_secondaryDNS</b> ( ): Promise<string>
es	async <b>get_secondaryDNS</b> ( )
dnf	string <b>get_secondaryDNS</b> ( )
cp	string <b>get_secondaryDNS</b> ( )
cmd	YNetwork <b>target</b> <b>get_secondaryDNS</b>

**Retourne :**

une chaîne de caractères représentant l'adresse IP du serveur de noms secondaire que le module doit utiliser

En cas d'erreur, déclenche une exception ou retourne `YNetwork.SECONDARYDNS_INVALID`.

**network**→**get\_serialNumber()****network**→**serialNumber()**

Retourne le numéro de série du module, préprogrammé en usine.

js	function <b>get_serialNumber</b> ( )
cpp	string <b>get_serialNumber</b> ( )
m	-(NSString*) <b>serialNumber</b>
pas	string <b>get_serialNumber</b> ( ): string
vb	function <b>get_serialNumber</b> ( ) As String
cs	string <b>get_serialNumber</b> ( )
java	String <b>get_serialNumber</b> ( )
uwp	async Task<string> <b>get_serialNumber</b> ( )
py	<b>get_serialNumber</b> ( )
php	function <b>get_serialNumber</b> ( )
ts	async <b>get_serialNumber</b> ( ): Promise<string>
es	async <b>get_serialNumber</b> ( )
dnp	string <b>get_serialNumber</b> ( )
cp	string <b>get_serialNumber</b> ( )
cmd	YNetwork <b>target</b> <b>get_serialNumber</b>

**Retourne :**

: une chaîne de caractères représentant le numéro de série du module, préprogrammé en usine.

En cas d'erreur, déclenche une exception ou retourne YFunction.SERIALNUMBER\_INVALID.

**network**→**get\_subnetMask()****YNetwork****network**→**subnetMask()**

Retourne le masque de sous-réseau utilisé par le module.

js	function <b>get_subnetMask</b> ( )
cpp	string <b>get_subnetMask</b> ( )
m	-(NSString*) subnetMask
pas	string <b>get_subnetMask</b> ( ): string
vb	function <b>get_subnetMask</b> ( ) As String
cs	string <b>get_subnetMask</b> ( )
java	String <b>get_subnetMask</b> ( )
uwp	async Task<string> <b>get_subnetMask</b> ( )
py	<b>get_subnetMask</b> ( )
php	function <b>get_subnetMask</b> ( )
ts	async <b>get_subnetMask</b> ( ): Promise<string>
es	async <b>get_subnetMask</b> ( )
dnp	string <b>get_subnetMask</b> ( )
cp	string <b>get_subnetMask</b> ( )
cmd	YNetwork <b>target</b> <b>get_subnetMask</b>

**Retourne :**

une chaîne de caractères représentant le masque de sous-réseau utilisé par le module

En cas d'erreur, déclenche une exception ou retourne `YNetwork.SUBNETMASK_INVALID`.

**network**→**get\_userData()****network**→**userData()**

Retourne le contenu de l'attribut `userData`, précédemment stocké à l'aide de la méthode `set_userData`.

js	function <b>get_userData</b> ( )
cpp	void * <b>get_userData</b> ( )
m	-(id) <code>userData</code>
pas	Tobject <b>get_userData</b> ( ): Tobject
vb	function <b>get_userData</b> ( ) As Object
cs	object <b>get_userData</b> ( )
java	Object <b>get_userData</b> ( )
py	<b>get_userData</b> ( )
php	function <b>get_userData</b> ( )
ts	async <b>get_userData</b> ( ): Promise<object null>
es	async <b>get_userData</b> ( )

Cet attribut n'est pas utilisé directement par l'API. Il est à la disposition de l'appelant pour stocker un contexte.

**Retourne :**

l'objet stocké précédemment par l'appelant.

**network**→**get\_userPassword()****YNetwork****network**→**userPassword()**

Retourne une chaîne de hash si un mot de passe a été configuré pour l'utilisateur "user", ou sinon une chaîne vide.

js	function <b>get_userPassword</b> ( )
cpp	string <b>get_userPassword</b> ( )
m	-(NSString*) <b>userPassword</b>
pas	string <b>get_userPassword</b> ( ): string
vb	function <b>get_userPassword</b> ( ) As String
cs	string <b>get_userPassword</b> ( )
java	String <b>get_userPassword</b> ( )
uwp	async Task<string> <b>get_userPassword</b> ( )
py	<b>get_userPassword</b> ( )
php	function <b>get_userPassword</b> ( )
ts	async <b>get_userPassword</b> ( ): Promise<string>
es	async <b>get_userPassword</b> ( )
dnp	string <b>get_userPassword</b> ( )
cp	string <b>get_userPassword</b> ( )
cmd	YNetwork <b>target</b> <b>get_userPassword</b>

**Retourne :**

une chaîne de caractères représentant une chaîne de hash si un mot de passe a été configuré pour l'utilisateur "user", ou sinon une chaîne vide

En cas d'erreur, déclenche une exception ou retourne `YNetwork.USERPASSWORD_INVALID`.

**network**→**get\_wwwWatchdogDelay()****YNetwork****network**→**wwwWatchdogDelay()**

Retourne la durée de perte de connexion WWW tolérée (en secondes) avant de déclencher un redémarrage automatique pour tenter de récupérer la connectivité Internet.

js	function <b>get_wwwWatchdogDelay</b> ( )
cpp	int <b>get_wwwWatchdogDelay</b> ( )
m	-(int) wwwWatchdogDelay
pas	LongInt <b>get_wwwWatchdogDelay</b> ( ): LongInt
vb	function <b>get_wwwWatchdogDelay</b> ( ) As Integer
cs	int <b>get_wwwWatchdogDelay</b> ( )
java	int <b>get_wwwWatchdogDelay</b> ( )
uwp	async Task<int> <b>get_wwwWatchdogDelay</b> ( )
py	<b>get_wwwWatchdogDelay</b> ( )
php	function <b>get_wwwWatchdogDelay</b> ( )
ts	async <b>get_wwwWatchdogDelay</b> ( ): Promise<number>
es	async <b>get_wwwWatchdogDelay</b> ( )
dnp	int <b>get_wwwWatchdogDelay</b> ( )
cp	int <b>get_wwwWatchdogDelay</b> ( )
cmd	YNetwork <b>target</b> <b>get_wwwWatchdogDelay</b>

Une valeur nulle désactive le redémarrage automatique en cas de perte de connectivité WWW.

**Retourne :**

un entier représentant la durée de perte de connexion WWW tolérée (en secondes) avant de déclencher un redémarrage automatique pour tenter de récupérer la connectivité Internet

En cas d'erreur, déclenche une exception ou retourne `YNetwork.WWWWATCHDOGDELAY_INVALID`.

**network**→**isOnline()****YNetwork**

Vérifie si le module hébergeant l'interface réseau est joignable, sans déclencher d'erreur.

js	function <b>isOnline</b> ( )
cpp	bool <b>isOnline</b> ( )
m	-(BOOL) <b>isOnline</b>
pas	boolean <b>isOnline</b> ( ): boolean
vb	function <b>isOnline</b> ( ) As Boolean
cs	bool <b>isOnline</b> ( )
java	boolean <b>isOnline</b> ( )
py	<b>isOnline</b> ( )
php	function <b>isOnline</b> ( )
ts	async <b>isOnline</b> ( ): Promise<boolean>
es	async <b>isOnline</b> ( )
dnp	bool <b>isOnline</b> ( )
cp	bool <b>isOnline</b> ( )

Si les valeurs des attributs en cache de l'interface réseau sont valides au moment de l'appel, le module est considéré joignable. Cette fonction ne cause en aucun cas d'exception, quelle que soit l'erreur qui pourrait se produire lors de la vérification de joignabilité.

**Retourne :**

`true` si l'interface réseau est joignable, `false` sinon

**network**→**isOnline\_async()****YNetwork**

Vérifie si le module hébergeant l'interface réseau est joignable, sans déclencher d'erreur.

```
js fonction isOnline_async( callback, context)
```

Si les valeurs des attributs en cache de l'interface réseau sont valides au moment de l'appel, le module est considéré joignable. Cette fonction ne cause en aucun cas d'exception, quelle que soit l'erreur qui pourrait se produire lors de la vérification de joignabilité.

Cette version asynchrone n'existe qu'en Javascript. Elle utilise une fonction de callback plutôt qu'une simple valeur de retour, pour éviter de bloquer la machine virtuelle Javascript avec une attente active.

**Paramètres :**

**callback** fonction de callback qui sera appelée dès que le résultat sera connu. La fonction callback reçoit trois arguments: le contexte fourni par l'appelant, l'objet fonction concerné et le résultat booléen

**context** contexte fourni par l'appelant, et qui sera passé tel-quel à la fonction de callback

**Retourne :**

rien du tout : le résultat sera passé en paramètre à la fonction de callback.

**network**→**isReadOnly()****YNetwork**

Test si la fonction est en lecture seule.

cpp	bool <b>isReadOnly</b> ( )
m	-(bool) <b>isReadOnly</b>
pas	boolean <b>isReadOnly</b> ( ): boolean
vb	function <b>isReadOnly</b> ( ) As Boolean
cs	bool <b>isReadOnly</b> ( )
java	boolean <b>isReadOnly</b> ( )
uwp	async Task<bool> <b>isReadOnly</b> ( )
py	<b>isReadOnly</b> ( )
php	function <b>isReadOnly</b> ( )
ts	async <b>isReadOnly</b> ( ): Promise<boolean>
es	async <b>isReadOnly</b> ( )
dnp	bool <b>isReadOnly</b> ( )
cp	bool <b>isReadOnly</b> ( )
cmd	YNetwork <b>target isReadOnly</b>

Retourne vrais si la fonction est protégé en ecriture ou que la fontion n'est pas disponible.

**Retourne :**

`true` si la fonction est protégé en ecriture ou que la fontion n'est pas disponible

## network→load()

YNetwork

Met en cache les valeurs courantes de l'interface réseau, avec une durée de validité spécifiée.

js	function <b>load</b> ( <b>msValidity</b> )
cpp	YRETCODE <b>load</b> ( int <b>msValidity</b> )
m	-(YRETCODE) <b>load</b> : (u64) <b>msValidity</b>
pas	YRETCODE <b>load</b> ( <b>msValidity</b> : u64): YRETCODE
vb	function <b>load</b> ( ByVal <b>msValidity</b> As Long) As YRETCODE
cs	YRETCODE <b>load</b> ( ulong <b>msValidity</b> )
java	int <b>load</b> ( long <b>msValidity</b> )
py	<b>load</b> ( <b>msValidity</b> )
php	function <b>load</b> ( <b>\$msValidity</b> )
ts	async <b>load</b> ( <b>msValidity</b> : number): Promise<number>
es	async <b>load</b> ( <b>msValidity</b> )

Par défaut, lorsqu'on accède à un module, tous les attributs des fonctions du module sont automatiquement mises en cache pour la durée standard (5 ms). Cette méthode peut être utilisée pour marquer occasionnellement les données cachées comme valides pour une plus longue période, par exemple dans le but de réduire le trafic réseau.

**Paramètres :**

**msValidity** un entier correspondant à la durée de validité attribuée aux les paramètres chargés, en millisecondes

**Retourne :**

YAPI . SUCCESS si l'opération se déroule sans erreur.

En cas d'erreur, déclenche une exception ou retourne un code d'erreur négatif.

**network→loadAttribute()****YNetwork**

Retourne la valeur actuelle d'un attribut spécifique de la fonction, sous forme de texte, le plus rapidement possible mais sans passer par le cache.

js	function <b>loadAttribute</b> ( <b>attrName</b> )
cpp	string <b>loadAttribute</b> ( string <b>attrName</b> )
m	-(NSString*) <b>loadAttribute</b> : (NSString*) <b>attrName</b>
pas	string <b>loadAttribute</b> ( <b>attrName</b> : string): string
vb	function <b>loadAttribute</b> ( ByVal <b>attrName</b> As String) As String
cs	string <b>loadAttribute</b> ( string <b>attrName</b> )
java	String <b>loadAttribute</b> ( String <b>attrName</b> )
uwp	async Task<string> <b>loadAttribute</b> ( string <b>attrName</b> )
py	<b>loadAttribute</b> ( <b>attrName</b> )
php	function <b>loadAttribute</b> ( \$ <b>attrName</b> )
ts	async <b>loadAttribute</b> ( <b>attrName</b> : string): Promise<string>
es	async <b>loadAttribute</b> ( <b>attrName</b> )
dnb	string <b>loadAttribute</b> ( string <b>attrName</b> )
cp	string <b>loadAttribute</b> ( string <b>attrName</b> )

**Paramètres :**

**attrName** le nom de l'attribut désiré

**Retourne :**

une chaîne de caractères représentant la valeur actuelle de l'attribut.

En cas d'erreur, déclenche une exception ou retourne un chaîne vide.

**network**→**load\_async()****YNetwork**

Met en cache les valeurs courantes de l'interface réseau, avec une durée de validité spécifiée.

```
js fonction load_async( msValidity, callback, context)
```

Par défaut, lorsqu'on accède à un module, tous les attributs des fonctions du module sont automatiquement mises en cache pour la durée standard (5 ms). Cette méthode peut être utilisée pour marquer occasionnellement les données cachées comme valides pour une plus longue période, par exemple dans le but de réduire le trafic réseau.

Cette version asynchrone n'existe qu'en Javascript. Elle utilise une fonction de callback plutôt qu'une simple valeur de retour, pour éviter de bloquer la machine virtuelle Javascript avec une attente active.

**Paramètres :**

- msValidity** un entier correspondant à la durée de validité attribuée aux les paramètres chargés, en millisecondes
- callback** fonction de callback qui sera appelée dès que le résultat sera connu. La fonction callback reçoit trois arguments: le contexte fourni par l'appelant, l'objet fonction concerné et le code d'erreur (ou `YAPI . SUCCESS`)
- context** contexte fourni par l'appelant, et qui sera passé tel-quel à la fonction de callback

**Retourne :**

rien du tout : le résultat sera passé en paramètre à la fonction de callback.

**network**→**muteValueCallbacks()****YNetwork**

Désactive l'envoi de chaque changement de la valeur publiée au hub parent.

js	function <b>muteValueCallbacks</b> ( )
cpp	int <b>muteValueCallbacks</b> ( )
m	-(int) <b>muteValueCallbacks</b>
pas	LongInt <b>muteValueCallbacks</b> ( ): LongInt
vb	function <b>muteValueCallbacks</b> ( ) As Integer
cs	int <b>muteValueCallbacks</b> ( )
java	int <b>muteValueCallbacks</b> ( )
uwp	async Task<int> <b>muteValueCallbacks</b> ( )
py	<b>muteValueCallbacks</b> ( )
php	function <b>muteValueCallbacks</b> ( )
ts	async <b>muteValueCallbacks</b> ( ): Promise<number>
es	async <b>muteValueCallbacks</b> ( )
dnp	int <b>muteValueCallbacks</b> ( )
cp	int <b>muteValueCallbacks</b> ( )
cmd	YNetwork <b>target</b> <b>muteValueCallbacks</b>

Vous pouvez utiliser cette fonction pour économiser la bande passante et le CPU sur les machines de faible puissance, ou pour éviter le déclenchement de callbacks HTTP. N'oubliez pas d'appeler la méthode `saveToFlash()` du module si le réglage doit être préservé.

**Retourne :**

YAPI . SUCCESS si l'opération se déroule sans erreur.

En cas d'erreur, déclenche une exception ou retourne un code d'erreur négatif.

**network**→**nextNetwork()****YNetwork**

Continue l'énumération des interfaces réseau commencée à l'aide de `yFirstNetwork()`. Attention, vous ne pouvez faire aucune supposition sur l'ordre dans lequel les interfaces réseau sont retournés.

js	function <b>nextNetwork</b> ( )
cpp	YNetwork * <b>nextNetwork</b> ( )
m	-(nullable YNetwork*) <b>nextNetwork</b>
pas	TYNetwork <b>nextNetwork</b> ( ): TYNetwork
vb	function <b>nextNetwork</b> ( ) As YNetwork
cs	YNetwork <b>nextNetwork</b> ( )
java	YNetwork <b>nextNetwork</b> ( )
uwp	YNetwork <b>nextNetwork</b> ( )
py	<b>nextNetwork</b> ( )
php	function <b>nextNetwork</b> ( )
ts	<b>nextNetwork</b> ( ): YNetwork   null
es	<b>nextNetwork</b> ( )

Si vous souhaitez retrouver une interface réseau spécifique, utilisez `Network.findNetwork()` avec un `hardwareID` ou un nom logique.

**Retourne :**

un pointeur sur un objet `YNetwork` accessible en ligne, ou `null` lorsque l'énumération est terminée.

**network→ping()****YNetwork**

Ping l'adresse choisie pour vérifier la connexion réseau.

js	function <b>ping</b> ( <b>host</b> )
cpp	string <b>ping</b> ( string <b>host</b> )
m	-(NSString*) <b>ping</b> : (NSString*) <b>host</b>
pas	string <b>ping</b> ( <b>host</b> : string): string
vb	function <b>ping</b> ( ByVal <b>host</b> As String) As String
cs	string <b>ping</b> ( string <b>host</b> )
java	String <b>ping</b> ( String <b>host</b> )
uwp	async Task<string> <b>ping</b> ( string <b>host</b> )
py	<b>ping</b> ( <b>host</b> )
php	function <b>ping</b> ( <b>\$host</b> )
ts	async <b>ping</b> ( <b>host</b> : string): Promise<string>
es	async <b>ping</b> ( <b>host</b> )
dnp	string <b>ping</b> ( string <b>host</b> )
cp	string <b>ping</b> ( string <b>host</b> )
cmd	YNetwork <b>target ping host</b>

Envoie quatre requêtes ICMP ECHO\_REQUEST à la cible host depuis le module. Cette méthode retourne une chaîne de caractères avec le résultat des 4 requêtes ICMP ECHO\_RESPONSE.

**Paramètres :**

**host** le nom d'hôte ou l'adresse IP de la cible

**Retourne :**

une chaîne de caractères contenant le résultat du ping.

## network→registerValueCallback()

YNetwork

Enregistre la fonction de callback qui est appelée à chaque changement de la valeur publiée.

js	function <b>registerValueCallback</b> ( <b>callback</b> )
cpp	int <b>registerValueCallback</b> ( YNetworkValueCallback <b>callback</b> )
m	-(int) <b>registerValueCallback</b> : (YNetworkValueCallback _Nullable) <b>callback</b>
pas	LongInt <b>registerValueCallback</b> ( <b>callback</b> : TYNetworkValueCallback): LongInt
vb	function <b>registerValueCallback</b> ( ByVal <b>callback</b> As YNetworkValueCallback) As Integer
cs	int <b>registerValueCallback</b> ( ValueCallback <b>callback</b> )
java	int <b>registerValueCallback</b> ( UpdateCallback <b>callback</b> )
uwp	async Task<int> <b>registerValueCallback</b> ( ValueCallback <b>callback</b> )
py	<b>registerValueCallback</b> ( <b>callback</b> )
php	function <b>registerValueCallback</b> ( <b>\$callback</b> )
ts	async <b>registerValueCallback</b> ( <b>callback</b> : YNetworkValueCallback   null): Promise<number>
es	async <b>registerValueCallback</b> ( <b>callback</b> )

Ce callback n'est appelé que durant l'exécution de `ySleep` ou `yHandleEvents`. Cela permet à l'appelant de contrôler quand les callback peuvent se produire. Il est important d'appeler l'une de ces deux fonctions périodiquement pour garantir que les callback ne soient pas appelés trop tard. Pour désactiver un callback, il suffit d'appeler cette méthode en lui passant un pointeur nul.

**Paramètres :**

**callback** la fonction de callback à rappeler, ou un pointeur nul. La fonction de callback doit accepter deux arguments: l'object fonction dont la valeur a changé, et la chaîne de caractère décrivant la nouvelle valeur publiée.

**network**→**set\_adminPassword()****YNetwork****network**→**setAdminPassword()**

Modifie le mot de passe pour l'utilisateur "admin", qui devient alors instantanément nécessaire pour toute altération de l'état du module.

js	function <b>set_adminPassword</b> ( <b>newval</b> )
cpp	int <b>set_adminPassword</b> ( string <b>newval</b> )
m	-(int) <b>setAdminPassword</b> : (NSString*) <b>newval</b>
pas	integer <b>set_adminPassword</b> ( <b>newval</b> : string): integer
vb	function <b>set_adminPassword</b> ( ByVal <b>newval</b> As String) As Integer
cs	int <b>set_adminPassword</b> ( string <b>newval</b> )
java	int <b>set_adminPassword</b> ( String <b>newval</b> )
uwp	async Task<int> <b>set_adminPassword</b> ( string <b>newval</b> )
py	<b>set_adminPassword</b> ( <b>newval</b> )
php	function <b>set_adminPassword</b> ( \$ <b>newval</b> )
ts	async <b>set_adminPassword</b> ( <b>newval</b> : string): Promise<number>
es	async <b>set_adminPassword</b> ( <b>newval</b> )
dnp	int <b>set_adminPassword</b> ( string <b>newval</b> )
cp	int <b>set_adminPassword</b> ( string <b>newval</b> )
cmd	YNetwork <b>target set_adminPassword newval</b>

Si la valeur fournie est une chaîne vide, plus aucun mot de passe n'est nécessaire. N'oubliez pas d'appeler la méthode `saveToFlash()` du module si le réglage doit être préservé.

**Paramètres :**

**newval** une chaîne de caractères représentant le mot de passe pour l'utilisateur "admin", qui devient alors instantanément nécessaire pour toute altération de l'état du module

**Retourne :**

YAPI . SUCCESS si l'opération se déroule sans erreur.

En cas d'erreur, déclenche une exception ou retourne un code d'erreur négatif.

**network**→**set\_callbackCredentials()****network**→**setCallbackCredentials()**

Modifie le laisser-passer pour se connecter à l'adresse de callback.

js	function <b>set_callbackCredentials</b> ( <b>newval</b> )
cpp	int <b>set_callbackCredentials</b> ( string <b>newval</b> )
m	-(int) <b>setCallbackCredentials</b> : (NSString*) <b>newval</b>
pas	integer <b>set_callbackCredentials</b> ( <b>newval</b> : string): integer
vb	function <b>set_callbackCredentials</b> ( ByVal <b>newval</b> As String) As Integer
cs	int <b>set_callbackCredentials</b> ( string <b>newval</b> )
java	int <b>set_callbackCredentials</b> ( String <b>newval</b> )
uwp	async Task<int> <b>set_callbackCredentials</b> ( string <b>newval</b> )
py	<b>set_callbackCredentials</b> ( <b>newval</b> )
php	function <b>set_callbackCredentials</b> ( \$ <b>newval</b> )
ts	async <b>set_callbackCredentials</b> ( <b>newval</b> : string): Promise<number>
es	async <b>set_callbackCredentials</b> ( <b>newval</b> )
dnp	int <b>set_callbackCredentials</b> ( string <b>newval</b> )
cp	int <b>set_callbackCredentials</b> ( string <b>newval</b> )
cmd	YNetwork <b>target</b> <b>set_callbackCredentials</b> <b>newval</b>

Le laisser-passer doit être fourni tel que retourné par la fonction `get_callbackCredentials`, sous la forme `username:hash`. La valeur du hash dépend de la méthode d'autorisation implémentée par le callback. Pour une autorisation de type Basic, le hash est le MD5 de la chaîne `username:password`. Pour une autorisation de type Digest, le hash est le MD5 de la chaîne `username:realm:password`. Pour une utilisation simplifiée, utilisez la fonction `callbackLogin`. N'oubliez pas d'appeler la méthode `saveToFlash()` du module si le réglage doit être préservé.

**Paramètres :**

**newval** une chaîne de caractères représentant le laisser-passer pour se connecter à l'adresse de callback

**Retourne :**

YAPI . SUCCESS si l'opération se déroule sans erreur.

En cas d'erreur, déclenche une exception ou retourne un code d'erreur négatif.

**network**→**set\_callbackEncoding()****YNetwork****network**→**setCallbackEncoding()**

Modifie l'encodage à utiliser pour représenter les valeurs notifiées par callback.

js	function <b>set_callbackEncoding</b> ( <b>newval</b> )
cpp	int <b>set_callbackEncoding</b> ( Y_CALLBACKENCODING_enum <b>newval</b> )
m	-(int) setCallbackEncoding : (Y_CALLBACKENCODING_enum) <b>newval</b>
pas	integer <b>set_callbackEncoding</b> ( <b>newval</b> : Integer): integer
vb	function <b>set_callbackEncoding</b> ( ByVal <b>newval</b> As Integer) As Integer
cs	int <b>set_callbackEncoding</b> ( int <b>newval</b> )
java	int <b>set_callbackEncoding</b> ( int <b>newval</b> )
uwp	async Task<int> <b>set_callbackEncoding</b> ( int <b>newval</b> )
py	<b>set_callbackEncoding</b> ( <b>newval</b> )
php	function <b>set_callbackEncoding</b> ( \$ <b>newval</b> )
ts	async <b>set_callbackEncoding</b> ( <b>newval</b> : YNetwork_CallbackEncoding): Promise<number>
es	async <b>set_callbackEncoding</b> ( <b>newval</b> )
dnf	int <b>set_callbackEncoding</b> ( int <b>newval</b> )
cp	int <b>set_callbackEncoding</b> ( int <b>newval</b> )
cmd	YNetwork <b>target</b> <b>set_callbackEncoding</b> <b>newval</b>

N'oubliez pas d'appeler la méthode `saveToFlash()` du module si le réglage doit être préservé.

**Paramètres :**

**newval** une valeur parmi `YNetwork.CALLBACKENCODING_FORM`,  
`YNetwork.CALLBACKENCODING_JSON`,  
`YNetwork.CALLBACKENCODING_JSON_ARRAY`,  
`YNetwork.CALLBACKENCODING_CSV`,  
`YNetwork.CALLBACKENCODING_YOCTO_API`,  
`YNetwork.CALLBACKENCODING_JSON_NUM`,  
`YNetwork.CALLBACKENCODING_EMONCMS`,  
`YNetwork.CALLBACKENCODING_AZURE`,  
`YNetwork.CALLBACKENCODING_INFLUXDB`,  
`YNetwork.CALLBACKENCODING_MQTT`,  
`YNetwork.CALLBACKENCODING_YOCTO_API_JZON`,  
`YNetwork.CALLBACKENCODING_PRTG` et  
`YNetwork.CALLBACKENCODING_INFLUXDB_V2` représentant l'encodage à utiliser pour  
représenter les valeurs notifiées par callback

**Retourne :**

`YAPI.SUCCESS` si l'opération se déroule sans erreur.

En cas d'erreur, déclenche une exception ou retourne un code d'erreur négatif.

**network**→**set\_callbackInitialDelay()****YNetwork****network**→**setCallbackInitialDelay()**

Modifie l'attente initiale avant la première notification par callback, en secondes.

js	function <b>set_callbackInitialDelay</b> ( <b>newval</b> )
cpp	int <b>set_callbackInitialDelay</b> ( int <b>newval</b> )
m	-(int) setCallbackInitialDelay : (int) <b>newval</b>
pas	integer <b>set_callbackInitialDelay</b> ( <b>newval</b> : LongInt): integer
vb	function <b>set_callbackInitialDelay</b> ( ByVal <b>newval</b> As Integer) As Integer
cs	int <b>set_callbackInitialDelay</b> ( int <b>newval</b> )
java	int <b>set_callbackInitialDelay</b> ( int <b>newval</b> )
uwp	async Task<int> <b>set_callbackInitialDelay</b> ( int <b>newval</b> )
py	<b>set_callbackInitialDelay</b> ( <b>newval</b> )
php	function <b>set_callbackInitialDelay</b> ( <b>\$newval</b> )
ts	async <b>set_callbackInitialDelay</b> ( <b>newval</b> : number): Promise<number>
es	async <b>set_callbackInitialDelay</b> ( <b>newval</b> )
dnp	int <b>set_callbackInitialDelay</b> ( int <b>newval</b> )
cp	int <b>set_callbackInitialDelay</b> ( int <b>newval</b> )
cmd	YNetwork <b>target</b> <b>set_callbackInitialDelay</b> <b>newval</b>

N'oubliez pas d'appeler la méthode `saveToFlash()` du module si le réglage doit être préservé.

**Paramètres :**

**newval** un entier représentant l'attente initiale avant la première notification par callback, en secondes

**Retourne :**

YAPI . SUCCESS si l'opération se déroule sans erreur.

En cas d'erreur, déclenche une exception ou retourne un code d'erreur négatif.

**network**→**set\_callbackMaxDelay()****YNetwork****network**→**setCallbackMaxDelay()**

Modifie l'attente entre deux callback HTTP lorsque rien n'est à signaler.

js	function <b>set_callbackMaxDelay</b> ( <b>newval</b> )
cpp	int <b>set_callbackMaxDelay</b> ( int <b>newval</b> )
m	-(int) <b>setCallbackMaxDelay</b> : (int) <b>newval</b>
pas	integer <b>set_callbackMaxDelay</b> ( <b>newval</b> : LongInt): integer
vb	function <b>set_callbackMaxDelay</b> ( ByVal <b>newval</b> As Integer) As Integer
cs	int <b>set_callbackMaxDelay</b> ( int <b>newval</b> )
java	int <b>set_callbackMaxDelay</b> ( int <b>newval</b> )
uwp	async Task<int> <b>set_callbackMaxDelay</b> ( int <b>newval</b> )
py	<b>set_callbackMaxDelay</b> ( <b>newval</b> )
php	function <b>set_callbackMaxDelay</b> ( <b>\$newval</b> )
ts	async <b>set_callbackMaxDelay</b> ( <b>newval</b> : number): Promise<number>
es	async <b>set_callbackMaxDelay</b> ( <b>newval</b> )
dnp	int <b>set_callbackMaxDelay</b> ( int <b>newval</b> )
cp	int <b>set_callbackMaxDelay</b> ( int <b>newval</b> )
cmd	YNetwork <b>target</b> <b>set_callbackMaxDelay</b> <b>newval</b>

N'oubliez pas d'appeler la méthode `saveToFlash()` du module si le réglage doit être préservé.

**Paramètres :**

**newval** un entier représentant l'attente entre deux callback HTTP lorsque rien n'est à signaler

**Retourne :**

YAPI . SUCCESS si l'opération se déroule sans erreur.

En cas d'erreur, déclenche une exception ou retourne un code d'erreur négatif.

**network**→**set\_callbackMethod()****YNetwork****network**→**setCallbackMethod()**

Modifie la méthode HTTP à utiliser pour signaler les changements d'état par callback.

js	function <b>set_callbackMethod</b> ( <b>newval</b> )
cpp	int <b>set_callbackMethod</b> ( Y_CALLBACKMETHOD_enum <b>newval</b> )
m	-(int) setCallbackMethod : (Y_CALLBACKMETHOD_enum) <b>newval</b>
pas	integer <b>set_callbackMethod</b> ( <b>newval</b> : Integer): integer
vb	function <b>set_callbackMethod</b> ( ByVal <b>newval</b> As Integer) As Integer
cs	int <b>set_callbackMethod</b> ( int <b>newval</b> )
java	int <b>set_callbackMethod</b> ( int <b>newval</b> )
uwp	async Task<int> <b>set_callbackMethod</b> ( int <b>newval</b> )
py	<b>set_callbackMethod</b> ( <b>newval</b> )
php	function <b>set_callbackMethod</b> ( \$ <b>newval</b> )
ts	async <b>set_callbackMethod</b> ( <b>newval</b> : YNetwork_CallbackMethod): Promise<number>
es	async <b>set_callbackMethod</b> ( <b>newval</b> )
dnp	int <b>set_callbackMethod</b> ( int <b>newval</b> )
cp	int <b>set_callbackMethod</b> ( int <b>newval</b> )
cmd	YNetwork <b>target</b> <b>set_callbackMethod</b> <b>newval</b>

N'oubliez pas d'appeler la méthode `saveToFlash()` du module si le réglage doit être préservé.

**Paramètres :**

**newval** une valeur parmi `YNetwork.CALLBACKMETHOD_POST`, `YNetwork.CALLBACKMETHOD_GET` et `YNetwork.CALLBACKMETHOD_PUT` représentant la méthode HTTP à utiliser pour signaler les changements d'état par callback

**Retourne :**

`YAPI.SUCCESS` si l'opération se déroule sans erreur.

En cas d'erreur, déclenche une exception ou retourne un code d'erreur négatif.

**network**→**set\_callbackMinDelay()****YNetwork****network**→**setCallbackMinDelay()**

Modifie l'attente minimale entre deux callbacks HTTP, en secondes.

js	function <b>set_callbackMinDelay</b> ( <b>newval</b> )
cpp	int <b>set_callbackMinDelay</b> ( int <b>newval</b> )
m	-(int) <b>setCallbackMinDelay</b> : (int) <b>newval</b>
pas	integer <b>set_callbackMinDelay</b> ( <b>newval</b> : LongInt): integer
vb	function <b>set_callbackMinDelay</b> ( ByVal <b>newval</b> As Integer) As Integer
cs	int <b>set_callbackMinDelay</b> ( int <b>newval</b> )
java	int <b>set_callbackMinDelay</b> ( int <b>newval</b> )
uwp	async Task<int> <b>set_callbackMinDelay</b> ( int <b>newval</b> )
py	<b>set_callbackMinDelay</b> ( <b>newval</b> )
php	function <b>set_callbackMinDelay</b> ( \$ <b>newval</b> )
ts	async <b>set_callbackMinDelay</b> ( <b>newval</b> : number): Promise<number>
es	async <b>set_callbackMinDelay</b> ( <b>newval</b> )
dnp	int <b>set_callbackMinDelay</b> ( int <b>newval</b> )
cp	int <b>set_callbackMinDelay</b> ( int <b>newval</b> )
cmd	YNetwork <b>target</b> <b>set_callbackMinDelay</b> <b>newval</b>

N'oubliez pas d'appeler la méthode `saveToFlash()` du module si le réglage doit être préservé.

**Paramètres :**

**newval** un entier représentant l'attente minimale entre deux callbacks HTTP, en secondes

**Retourne :**

YAPI . SUCCESS si l'opération se déroule sans erreur.

En cas d'erreur, déclenche une exception ou retourne un code d'erreur négatif.

**network**→**set\_callbackSchedule()**

YNetwork

**network**→**setCallbackSchedule()**

Modifie la planification des callbacks HTTP, sous forme de chaîne de caractères.

js	function <b>set_callbackSchedule</b> ( <b>newval</b> )
cpp	int <b>set_callbackSchedule</b> ( string <b>newval</b> )
m	-(int) <b>setCallbackSchedule</b> : (NSString*) <b>newval</b>
pas	integer <b>set_callbackSchedule</b> ( <b>newval</b> : string): integer
vb	function <b>set_callbackSchedule</b> ( ByVal <b>newval</b> As String) As Integer
cs	int <b>set_callbackSchedule</b> ( string <b>newval</b> )
java	int <b>set_callbackSchedule</b> ( String <b>newval</b> )
uwp	async Task<int> <b>set_callbackSchedule</b> ( string <b>newval</b> )
py	<b>set_callbackSchedule</b> ( <b>newval</b> )
php	function <b>set_callbackSchedule</b> ( \$ <b>newval</b> )
ts	async <b>set_callbackSchedule</b> ( <b>newval</b> : string): Promise<number>
es	async <b>set_callbackSchedule</b> ( <b>newval</b> )
dnp	int <b>set_callbackSchedule</b> ( string <b>newval</b> )
cp	int <b>set_callbackSchedule</b> ( string <b>newval</b> )
cmd	YNetwork <b>target</b> <b>set_callbackSchedule</b> <b>newval</b>

N'oubliez pas d'appeler la méthode `saveToFlash()` du module si le réglage doit être préservé.

**Paramètres :**

**newval** une chaîne de caractères représentant la planification des callbacks HTTP, sous forme de chaîne de caractères

**Retourne :**

YAPI . SUCCESS si l'opération se déroule sans erreur.

En cas d'erreur, déclenche une exception ou retourne un code d'erreur négatif.

**network**→**set\_callbackUrl()****YNetwork****network**→**setCallbackUrl()**

Modifie l'adresse (URL) de callback à notifier lors de changement d'état significatifs.

js	function <b>set_callbackUrl</b> ( <b>newval</b> )
cpp	int <b>set_callbackUrl</b> ( string <b>newval</b> )
m	-(int) setCallbackUrl : (NSString*) <b>newval</b>
pas	integer <b>set_callbackUrl</b> ( <b>newval</b> : string): integer
vb	function <b>set_callbackUrl</b> ( ByVal <b>newval</b> As String) As Integer
cs	int <b>set_callbackUrl</b> ( string <b>newval</b> )
java	int <b>set_callbackUrl</b> ( String <b>newval</b> )
uwp	async Task<int> <b>set_callbackUrl</b> ( string <b>newval</b> )
py	<b>set_callbackUrl</b> ( <b>newval</b> )
php	function <b>set_callbackUrl</b> ( <b>\$newval</b> )
ts	async <b>set_callbackUrl</b> ( <b>newval</b> : string): Promise<number>
es	async <b>set_callbackUrl</b> ( <b>newval</b> )
dnp	int <b>set_callbackUrl</b> ( string <b>newval</b> )
cp	int <b>set_callbackUrl</b> ( string <b>newval</b> )
cmd	YNetwork <b>target</b> <b>set_callbackUrl</b> <b>newval</b>

N'oubliez pas d'appeler la méthode `saveToFlash()` du module si le réglage doit être préservé.

**Paramètres :**

**newval** une chaîne de caractères représentant l'adresse (URL) de callback à notifier lors de changement d'état significatifs

**Retourne :**

YAPI . SUCCESS si l'opération se déroule sans erreur.

En cas d'erreur, déclenche une exception ou retourne un code d'erreur négatif.

**network**→**set\_defaultPage()****network**→**setDefaultPage()**

Modifie la page HTML par défaut du hub.

js	function <b>set_defaultPage</b> ( <b>newval</b> )
cpp	int <b>set_defaultPage</b> ( string <b>newval</b> )
m	-(int) setDefaultPage : (NSString*) <b>newval</b>
pas	integer <b>set_defaultPage</b> ( <b>newval</b> : string): integer
vb	function <b>set_defaultPage</b> ( ByVal <b>newval</b> As String) As Integer
cs	int <b>set_defaultPage</b> ( string <b>newval</b> )
java	int <b>set_defaultPage</b> ( String <b>newval</b> )
uwp	async Task<int> <b>set_defaultPage</b> ( string <b>newval</b> )
py	<b>set_defaultPage</b> ( <b>newval</b> )
php	function <b>set_defaultPage</b> ( \$ <b>newval</b> )
ts	async <b>set_defaultPage</b> ( <b>newval</b> : string): Promise<number>
es	async <b>set_defaultPage</b> ( <b>newval</b> )
dnp	int <b>set_defaultPage</b> ( string <b>newval</b> )
cp	int <b>set_defaultPage</b> ( string <b>newval</b> )
cmd	YNetwork <b>target</b> <b>set_defaultPage</b> <b>newval</b>

Si aucune valeur n'est attribuée le hub retourne index.html qui est l'interface web du hub. Il est possible de changer cet page pour un fichier qui a été uploadé sur le hub. Attention, la taille maximale permise pour le nom de fichier est de 15 caractères. Si vous changez ce paramètre, n'oubliez pas d'appeler la méthode `saveToFlash()` du module si le réglage doit être préservé.

**Paramètres :**

**newval** une chaîne de caractères représentant la page HTML par défaut du hub

**Retourne :**

YAPI . SUCCESS si l'opération se déroule sans erreur.

En cas d'erreur, déclenche une exception ou retourne un code d'erreur négatif.

**network→set\_discoverable()****YNetwork****network→setDiscoverable()**

Modifie l'état d'activation du protocole d'annonce sur le réseau permettant de retrouver facilement le module (protocoles uPnP/Bonjour).

js	function <b>set_discoverable</b> ( <b>newval</b> )
cpp	int <b>set_discoverable</b> ( Y_DISCOVERABLE_enum <b>newval</b> )
m	-(int) setDiscoverable : (Y_DISCOVERABLE_enum) <b>newval</b>
pas	integer <b>set_discoverable</b> ( <b>newval</b> : Integer): integer
vb	function <b>set_discoverable</b> ( ByVal <b>newval</b> As Integer) As Integer
cs	int <b>set_discoverable</b> ( int <b>newval</b> )
java	int <b>set_discoverable</b> ( int <b>newval</b> )
uwp	async Task<int> <b>set_discoverable</b> ( int <b>newval</b> )
py	<b>set_discoverable</b> ( <b>newval</b> )
php	function <b>set_discoverable</b> ( \$ <b>newval</b> )
ts	async <b>set_discoverable</b> ( <b>newval</b> : YNetwork_Discoverable): Promise<number>
es	async <b>set_discoverable</b> ( <b>newval</b> )
dnp	int <b>set_discoverable</b> ( int <b>newval</b> )
cp	int <b>set_discoverable</b> ( int <b>newval</b> )
cmd	YNetwork <b>target set_discoverable newval</b>

N'oubliez pas d'appeler la méthode `saveToFlash()` du module si le réglage doit être préservé.

**Paramètres :**

**newval** soit `YNetwork.DISCOVERABLE_FALSE`, soit `YNetwork.DISCOVERABLE_TRUE`, selon l'état d'activation du protocole d'annonce sur le réseau permettant de retrouver facilement le module (protocoles uPnP/Bonjour)

**Retourne :**

`YAPI.SUCCESS` si l'opération se déroule sans erreur.

En cas d'erreur, déclenche une exception ou retourne un code d'erreur négatif.

**network**→**set\_httpPort()****network**→**setHttpPort()**

Modifie le port TCP utilisé pour l'interface Web du hub.

js	function <b>set_httpPort</b> ( <b>newval</b> )
cpp	int <b>set_httpPort</b> ( int <b>newval</b> )
m	-(int) setHttpPort : (int) <b>newval</b>
pas	integer <b>set_httpPort</b> ( <b>newval</b> : LongInt): integer
vb	function <b>set_httpPort</b> ( ByVal <b>newval</b> As Integer) As Integer
cs	int <b>set_httpPort</b> ( int <b>newval</b> )
java	int <b>set_httpPort</b> ( int <b>newval</b> )
uwp	async Task<int> <b>set_httpPort</b> ( int <b>newval</b> )
py	<b>set_httpPort</b> ( <b>newval</b> )
php	function <b>set_httpPort</b> ( <b>\$newval</b> )
ts	async <b>set_httpPort</b> ( <b>newval</b> : number): Promise<number>
es	async <b>set_httpPort</b> ( <b>newval</b> )
dnp	int <b>set_httpPort</b> ( int <b>newval</b> )
cp	int <b>set_httpPort</b> ( int <b>newval</b> )
cmd	YNetwork <b>target</b> <b>set_httpPort</b> <b>newval</b>

La valeur par défaut est le port 80, utilisé habituellement par tous les serveurs web. Indépendamment de la valeur de ce paramètre, le hub répond toujours au port 4444, qui est utilisé par défaut par la bibliothèque de programmation Yoctopuce. Si vous changez ce paramètre, n'oubliez pas d'appeler la méthode `saveToFlash()` du module si le réglage doit être préservé.

**Paramètres :**

**newval** un entier représentant le port TCP utilisé pour l'interface Web du hub

**Retourne :**

YAPI . SUCCESS si l'opération se déroule sans erreur.

En cas d'erreur, déclenche une exception ou retourne un code d'erreur négatif.

**network**→**set\_logicalName()****YNetwork****network**→**setLogicalName()**

Modifie le nom logique de l'interface réseau.

js	function <b>set_logicalName</b> ( <b>newval</b> )
cpp	int <b>set_logicalName</b> ( string <b>newval</b> )
m	-(int) setLogicalName : (NSString*) <b>newval</b>
pas	integer <b>set_logicalName</b> ( <b>newval</b> : string): integer
vb	function <b>set_logicalName</b> ( ByVal <b>newval</b> As String) As Integer
cs	int <b>set_logicalName</b> ( string <b>newval</b> )
java	int <b>set_logicalName</b> ( String <b>newval</b> )
uwp	async Task<int> <b>set_logicalName</b> ( string <b>newval</b> )
py	<b>set_logicalName</b> ( <b>newval</b> )
php	function <b>set_logicalName</b> ( \$ <b>newval</b> )
ts	async <b>set_logicalName</b> ( <b>newval</b> : string): Promise<number>
es	async <b>set_logicalName</b> ( <b>newval</b> )
dnp	int <b>set_logicalName</b> ( string <b>newval</b> )
cp	int <b>set_logicalName</b> ( string <b>newval</b> )
cmd	YNetwork <b>target set_logicalName newval</b>

Vous pouvez utiliser `yCheckLogicalName()` pour vérifier si votre paramètre est valide. N'oubliez pas d'appeler la méthode `saveToFlash()` du module si le réglage doit être préservé.

**Paramètres :**

**newval** une chaîne de caractères représentant le nom logique de l'interface réseau.

**Retourne :**

YAPI . SUCCESS si l'appel se déroule sans erreur.

En cas d'erreur, déclenche une exception ou retourne un code d'erreur négatif.

**network**→**set\_ntpServer()****network**→**setNtpServer()**

Modifie l'adresse IP du serveur NTP que le module doit utiliser.

js	function <b>set_ntpServer</b> ( <b>newval</b> )
c++	int <b>set_ntpServer</b> ( string <b>newval</b> )
m	-(int) setNtpServer : (NSString*) <b>newval</b>
pas	integer <b>set_ntpServer</b> ( <b>newval</b> : string): integer
vb	function <b>set_ntpServer</b> ( ByVal <b>newval</b> As String) As Integer
cs	int <b>set_ntpServer</b> ( string <b>newval</b> )
java	int <b>set_ntpServer</b> ( String <b>newval</b> )
uwp	async Task<int> <b>set_ntpServer</b> ( string <b>newval</b> )
py	<b>set_ntpServer</b> ( <b>newval</b> )
php	function <b>set_ntpServer</b> ( \$ <b>newval</b> )
ts	async <b>set_ntpServer</b> ( <b>newval</b> : string): Promise<number>
es	async <b>set_ntpServer</b> ( <b>newval</b> )
dnp	int <b>set_ntpServer</b> ( string <b>newval</b> )
cp	int <b>set_ntpServer</b> ( string <b>newval</b> )
cmd	YNetwork <b>target</b> <b>set_ntpServer</b> <b>newval</b>

Utilisez une chaîne vide pour restaurer l'adresse pré-configurée en usine. N'oubliez pas d'appeler la méthode `saveToFlash()` et de redémarrer le module pour que le paramètre soit appliqué.

**Paramètres :**

**newval** une chaîne de caractères représentant l'adresse IP du serveur NTP que le module doit utiliser

**Retourne :**

YAPI . SUCCESS si l'opération se déroule sans erreur.

En cas d'erreur, déclenche une exception ou retourne un code d'erreur négatif.

**network→set\_periodicCallbackSchedule()****YNetwork****network→setPeriodicCallbackSchedule()**

Configure la planification de callbacks HTTP périodiques (fonction simplifiée).

js	<code>function set_periodicCallbackSchedule( interval, offset)</code>
cpp	<code>int set_periodicCallbackSchedule( string interval, int offset)</code>
m	<code>-(int) setPeriodicCallbackSchedule : (NSString*) interval : (int) offset</code>
pas	<code>LongInt set_periodicCallbackSchedule( interval: string, offset: LongInt): LongInt</code>
vb	<code>function set_periodicCallbackSchedule( ByVal interval As String, ByVal offset As Integer) As Integer</code>
cs	<code>int set_periodicCallbackSchedule( string interval, int offset)</code>
java	<code>int set_periodicCallbackSchedule( String interval, int offset)</code>
uwp	<code>async Task&lt;int&gt; set_periodicCallbackSchedule( string interval, int offset)</code>
py	<code>set_periodicCallbackSchedule( interval, offset)</code>
php	<code>function set_periodicCallbackSchedule( \$interval, \$offset)</code>
ts	<code>async set_periodicCallbackSchedule( interval: string, offset: number): Promise&lt;number&gt;</code>
es	<code>async set_periodicCallbackSchedule( interval, offset)</code>
dnp	<code>int set_periodicCallbackSchedule( string interval, int offset)</code>
cp	<code>int set_periodicCallbackSchedule( string interval, int offset)</code>
cmd	<code>YNetwork target set_periodicCallbackSchedule interval offset</code>

**Paramètres :**

**interval** la périodicité du callback sous forme textuelle, exprimée en secondes, minutes ou en heures. Par exemple: "60s", "5m", "1h", "48h".

**offset** un entier décrivant le décalage du callback par rapport au début de la période. Par exemple, si la périodicité est 24h, un offset de 7 déclanchera le callback chaque jour à 7h du matin.

**Retourne :**

YAPI . SUCCESS when the call succeeds.

On failure, throws an exception or returns a negative error code.

**network**→**set\_primaryDNS()**

YNetwork

**network**→**setPrimaryDNS()**

Modifie l'adresse IP du serveur de noms primaire que le module doit utiliser.

js	function <b>set_primaryDNS</b> ( <b>newval</b> )
cpp	int <b>set_primaryDNS</b> ( string <b>newval</b> )
m	-(int) setPrimaryDNS : (NSString*) <b>newval</b>
pas	integer <b>set_primaryDNS</b> ( <b>newval</b> : string): integer
vb	function <b>set_primaryDNS</b> ( ByVal <b>newval</b> As String) As Integer
cs	int <b>set_primaryDNS</b> ( string <b>newval</b> )
java	int <b>set_primaryDNS</b> ( String <b>newval</b> )
uwp	async Task<int> <b>set_primaryDNS</b> ( string <b>newval</b> )
py	<b>set_primaryDNS</b> ( <b>newval</b> )
php	function <b>set_primaryDNS</b> ( \$ <b>newval</b> )
ts	async <b>set_primaryDNS</b> ( <b>newval</b> : string): Promise<number>
es	async <b>set_primaryDNS</b> ( <b>newval</b> )
dnp	int <b>set_primaryDNS</b> ( string <b>newval</b> )
cp	int <b>set_primaryDNS</b> ( string <b>newval</b> )
cmd	YNetwork <b>target</b> <b>set_primaryDNS</b> <b>newval</b>

En mode DHCP, si une valeur est spécifiée, elle remplacera celle reçue du serveur DHCP. N'oubliez pas d'appeler la méthode `saveToFlash()` et de redémarrer le module pour que le paramètre soit appliqué.

**Paramètres :**

**newval** une chaîne de caractères représentant l'adresse IP du serveur de noms primaire que le module doit utiliser

**Retourne :**

YAPI . SUCCESS si l'opération se déroule sans erreur.

En cas d'erreur, déclenche une exception ou retourne un code d'erreur négatif.

**network**→**set\_secondaryDNS()****YNetwork****network**→**setSecondaryDNS()**

Modifie l'adresse IP du serveur de nom secondaire que le module doit utiliser.

js	function <b>set_secondaryDNS</b> ( <b>newval</b> )
cpp	int <b>set_secondaryDNS</b> ( string <b>newval</b> )
m	-(int) <b>setSecondaryDNS</b> : (NSString*) <b>newval</b>
pas	integer <b>set_secondaryDNS</b> ( <b>newval</b> : string): integer
vb	function <b>set_secondaryDNS</b> ( ByVal <b>newval</b> As String) As Integer
cs	int <b>set_secondaryDNS</b> ( string <b>newval</b> )
java	int <b>set_secondaryDNS</b> ( String <b>newval</b> )
uwp	async Task<int> <b>set_secondaryDNS</b> ( string <b>newval</b> )
py	<b>set_secondaryDNS</b> ( <b>newval</b> )
php	function <b>set_secondaryDNS</b> ( \$ <b>newval</b> )
ts	async <b>set_secondaryDNS</b> ( <b>newval</b> : string): Promise<number>
es	async <b>set_secondaryDNS</b> ( <b>newval</b> )
dnp	int <b>set_secondaryDNS</b> ( string <b>newval</b> )
cp	int <b>set_secondaryDNS</b> ( string <b>newval</b> )
cmd	YNetwork <b>target</b> <b>set_secondaryDNS</b> <b>newval</b>

En mode DHCP, si une valeur est spécifiée, elle remplacera celle reçue du serveur DHCP. N'oubliez pas d'appeler la méthode `saveToFlash()` et de redémarrer le module pour que le paramètre soit appliqué.

**Paramètres :**

**newval** une chaîne de caractères représentant l'adresse IP du serveur de nom secondaire que le module doit utiliser

**Retourne :**

YAPI . SUCCESS si l'opération se déroule sans erreur.

En cas d'erreur, déclenche une exception ou retourne un code d'erreur négatif.

**network**→**set\_userdata()****network**→**setUserData()**

Enregistre un contexte libre dans l'attribut `userData` de la fonction, afin de le retrouver plus tard à l'aide de la méthode `get_userdata`.

js	function <b>set_userdata</b> ( <b>data</b> )
cpp	void <b>set_userdata</b> ( void * <b>data</b> )
m	-(void) setUserData : (id) <b>data</b>
pas	<b>set_userdata</b> ( <b>data</b> : Tobject)
vb	procedure <b>set_userdata</b> ( ByVal <b>data</b> As Object)
cs	void <b>set_userdata</b> ( object <b>data</b> )
java	void <b>set_userdata</b> ( Object <b>data</b> )
py	<b>set_userdata</b> ( <b>data</b> )
php	function <b>set_userdata</b> ( <b>\$data</b> )
ts	async <b>set_userdata</b> ( <b>data</b> : object null): Promise<void>
es	async <b>set_userdata</b> ( <b>data</b> )

Cet attribut n'est pas utilisé directement par l'API. Il est à la disposition de l'appelant pour stocker un contexte.

**Paramètres :**

**data** objet quelconque à mémoriser

**network**→**set\_userPassword()****YNetwork****network**→**setUserPassword()**

Modifie le mode de passe pour l'utilisateur "user", qui devient alors instantanément nécessaire pour tout accès au module.

js	function <b>set_userPassword</b> ( <b>newval</b> )
cpp	int <b>set_userPassword</b> ( string <b>newval</b> )
m	-(int) setUserPassword : (NSString*) <b>newval</b>
pas	integer <b>set_userPassword</b> ( <b>newval</b> : string): integer
vb	function <b>set_userPassword</b> ( ByVal <b>newval</b> As String) As Integer
cs	int <b>set_userPassword</b> ( string <b>newval</b> )
java	int <b>set_userPassword</b> ( String <b>newval</b> )
uwp	async Task<int> <b>set_userPassword</b> ( string <b>newval</b> )
py	<b>set_userPassword</b> ( <b>newval</b> )
php	function <b>set_userPassword</b> ( \$ <b>newval</b> )
ts	async <b>set_userPassword</b> ( <b>newval</b> : string): Promise<number>
es	async <b>set_userPassword</b> ( <b>newval</b> )
dnp	int <b>set_userPassword</b> ( string <b>newval</b> )
cp	int <b>set_userPassword</b> ( string <b>newval</b> )
cmd	YNetwork <b>target set_userPassword newval</b>

Si la valeur fournie est une chaîne vide, plus aucun mot de passe n'est nécessaire. N'oubliez pas d'appeler la méthode `saveToFlash()` du module si le réglage doit être préservé.

**Paramètres :**

**newval** une chaîne de caractères représentant le mode de passe pour l'utilisateur "user", qui devient alors instantanément nécessaire pour tout accès au module

**Retourne :**

YAPI . SUCCESS si l'opération se déroule sans erreur.

En cas d'erreur, déclenche une exception ou retourne un code d'erreur négatif.

**network**→**set\_wwwWatchdogDelay()****YNetwork****network**→**setWwwWatchdogDelay()**

Modifie la durée de perte de connection WWW tolérée (en secondes) avant de déclencher un redémarrage automatique pour tenter de récupérer la connectivité Internet.

js	function <b>set_wwwWatchdogDelay</b> ( <b>newval</b> )
cpp	int <b>set_wwwWatchdogDelay</b> ( int <b>newval</b> )
m	-(int) <b>setWwwWatchdogDelay</b> : (int) <b>newval</b>
pas	integer <b>set_wwwWatchdogDelay</b> ( <b>newval</b> : LongInt): integer
vb	function <b>set_wwwWatchdogDelay</b> ( ByVal <b>newval</b> As Integer) As Integer
cs	int <b>set_wwwWatchdogDelay</b> ( int <b>newval</b> )
java	int <b>set_wwwWatchdogDelay</b> ( int <b>newval</b> )
uwp	async Task<int> <b>set_wwwWatchdogDelay</b> ( int <b>newval</b> )
py	<b>set_wwwWatchdogDelay</b> ( <b>newval</b> )
php	function <b>set_wwwWatchdogDelay</b> ( <b>\$newval</b> )
ts	async <b>set_wwwWatchdogDelay</b> ( <b>newval</b> : number): Promise<number>
es	async <b>set_wwwWatchdogDelay</b> ( <b>newval</b> )
dnp	int <b>set_wwwWatchdogDelay</b> ( int <b>newval</b> )
cp	int <b>set_wwwWatchdogDelay</b> ( int <b>newval</b> )
cmd	YNetwork <b>target set_wwwWatchdogDelay newval</b>

Une valeur nulle désactive le redémarrage automatique en cas de perte de connectivité WWW. La plus petite durée non-nulle utilisable est 90 secondes. N'oubliez pas d'appeler la méthode `saveToFlash()` du module si le réglage doit être préservé.

**Paramètres :**

**newval** un entier représentant la durée de perte de connection WWW tolérée (en secondes) avant de déclencher un redémarrage automatique pour tenter de récupérer la connectivité Internet

**Retourne :**

YAPI . SUCCESS si l'opération se déroule sans erreur.

En cas d'erreur, déclenche une exception ou retourne un code d'erreur négatif.

**network**→**triggerCallback()****YNetwork**

Déclenche un callback HTTP rapidement.

js	function <b>triggerCallback</b> ( )
cpp	int <b>triggerCallback</b> ( )
m	-(int) <b>triggerCallback</b>
pas	LongInt <b>triggerCallback</b> ( ): LongInt
vb	function <b>triggerCallback</b> ( ) As Integer
cs	int <b>triggerCallback</b> ( )
java	int <b>triggerCallback</b> ( )
uwp	async Task<int> <b>triggerCallback</b> ( )
py	<b>triggerCallback</b> ( )
php	function <b>triggerCallback</b> ( )
ts	async <b>triggerCallback</b> ( ): Promise<number>
es	async <b>triggerCallback</b> ( )
dnp	int <b>triggerCallback</b> ( )
cp	int <b>triggerCallback</b> ( )
cmd	YNetwork <b>target triggerCallback</b>

Cette fonction peut même être appelée à l'intérieur d'un callback HTTP, dans quel cas le callback HTTP suivant sera appelé 5 secondes après la fin du callback courant, indépendamment de l'intervalle minimal configuré dans le module.

**Retourne :**

une chaîne de caractères contenant le résultat du ping.

**network**→**unmuteValueCallbacks()****YNetwork**

Réactive l'envoi de chaque changement de la valeur publiée au hub parent.

js	function <b>unmuteValueCallbacks</b> ( )
cpp	int <b>unmuteValueCallbacks</b> ( )
m	-(int) <b>unmuteValueCallbacks</b>
pas	LongInt <b>unmuteValueCallbacks</b> ( ): LongInt
vb	function <b>unmuteValueCallbacks</b> ( ) As Integer
cs	int <b>unmuteValueCallbacks</b> ( )
java	int <b>unmuteValueCallbacks</b> ( )
uwp	async Task<int> <b>unmuteValueCallbacks</b> ( )
py	<b>unmuteValueCallbacks</b> ( )
php	function <b>unmuteValueCallbacks</b> ( )
ts	async <b>unmuteValueCallbacks</b> ( ): Promise<number>
es	async <b>unmuteValueCallbacks</b> ( )
dnp	int <b>unmuteValueCallbacks</b> ( )
cp	int <b>unmuteValueCallbacks</b> ( )
cmd	YNetwork <b>target</b> <b>unmuteValueCallbacks</b>

Cette fonction annule un précédent appel à `muteValueCallbacks()`. N'oubliez pas d'appeler la méthode `saveToFlash()` du module si le réglage doit être préservé.

**Retourne :**

YAPI . SUCCESS si l'opération se déroule sans erreur.

En cas d'erreur, déclenche une exception ou retourne un code d'erreur négatif.

**network→useDHCP()****YNetwork**

Modifie la configuration de l'interface réseau pour utiliser une adresse assignée automatiquement par le serveur DHCP.

js	<code>function useDHCP( fallbackIpAddr, fallbackSubnetMaskLen, fallbackRouter)</code>
cpp	<code>int useDHCP( string fallbackIpAddr, int fallbackSubnetMaskLen, string fallbackRouter)</code>
m	<code>-(int) useDHCP : (NSString*) fallbackIpAddr : (int) fallbackSubnetMaskLen : (NSString*) fallbackRouter</code>
pas	<code>LongInt useDHCP( fallbackIpAddr: string, fallbackSubnetMaskLen: LongInt, fallbackRouter: string): LongInt</code>
vb	<code>function useDHCP( ByVal fallbackIpAddr As String, ByVal fallbackSubnetMaskLen As Integer, ByVal fallbackRouter As String) As Integer</code>
cs	<code>int useDHCP( string fallbackIpAddr, int fallbackSubnetMaskLen, string fallbackRouter)</code>
java	<code>int useDHCP( String fallbackIpAddr, int fallbackSubnetMaskLen, String fallbackRouter)</code>
uwp	<code>async Task&lt;int&gt; useDHCP( string fallbackIpAddr, int fallbackSubnetMaskLen, string fallbackRouter)</code>
py	<code>useDHCP( fallbackIpAddr, fallbackSubnetMaskLen, fallbackRouter)</code>
php	<code>function useDHCP( \$fallbackIpAddr, \$fallbackSubnetMaskLen, \$fallbackRouter)</code>
ts	<code>async useDHCP( fallbackIpAddr: string, fallbackSubnetMaskLen: number, fallbackRouter: string): Promise&lt;number&gt;</code>
es	<code>async useDHCP( fallbackIpAddr, fallbackSubnetMaskLen, fallbackRouter)</code>
dnp	<code>int useDHCP( string fallbackIpAddr, int fallbackSubnetMaskLen, string fallbackRouter)</code>
cp	<code>int useDHCP( string fallbackIpAddr, int fallbackSubnetMaskLen, string fallbackRouter)</code>
cmd	<code>YNetwork target useDHCP fallbackIpAddr fallbackSubnetMaskLen fallbackRouter</code>

En attendant qu'une adresse soit reçue (et indéfiniment si aucun serveur DHCP ne répond), le module utilisera les paramètres IP spécifiés à cette fonction. N'oubliez pas d'appeler la méthode `saveToFlash()` et de redémarrer le module pour que le paramètre soit appliqué.

**Paramètres :**

<b>fallbackIpAddr</b>	adresse IP à utiliser si aucun serveur DHCP ne répond
<b>fallbackSubnetMaskLen</b>	longueur du masque de sous-réseau à utiliser si aucun serveur DHCP ne répond. Par exemple, la valeur 24 représente 255.255.255.0.
<b>fallbackRouter</b>	adresse de la passerelle à utiliser si aucun serveur DHCP ne répond

**Retourne :**

YAPI . SUCCESS si l'opération se déroule sans erreur.

En cas d'erreur, déclenche une exception ou retourne un code d'erreur négatif.

**network→useDHCPauto()****YNetwork**

Modifie la configuration de l'interface réseau pour utiliser une adresse assignée automatiquement par le serveur DHCP.

js	function <b>useDHCPauto</b> ( )
cpp	int <b>useDHCPauto</b> ( )
m	-(int) <b>useDHCPauto</b>
pas	LongInt <b>useDHCPauto</b> ( ): LongInt
vb	function <b>useDHCPauto</b> ( ) As Integer
cs	int <b>useDHCPauto</b> ( )
java	int <b>useDHCPauto</b> ( )
uwp	async Task<int> <b>useDHCPauto</b> ( )
py	<b>useDHCPauto</b> ( )
php	function <b>useDHCPauto</b> ( )
ts	async <b>useDHCPauto</b> ( ): Promise<number>
es	async <b>useDHCPauto</b> ( )
dnp	int <b>useDHCPauto</b> ( )
cp	int <b>useDHCPauto</b> ( )
cmd	YNetwork <b>target useDHCPauto</b>

En attendant qu'une adresse soit reçue (et indéfiniment si aucun serveur DHCP ne répond), le module utilise une adresse IP du réseau 169.254.0.0/16 (APIPA). N'oubliez pas d'appeler la méthode `saveToFlash()` et de redémarrer le module pour que le paramètre soit appliqué.

**Retourne :**

YAPI . SUCCESS si l'opération se déroule sans erreur.

En cas d'erreur, déclenche une exception ou retourne un code d'erreur négatif.

## network→useStaticIP()

YNetwork

Modifie la configuration de l'interface réseau pour utiliser une adresse IP assignée manuellement (adresse IP statique).

js	function <b>useStaticIP</b> ( <b>ipAddress</b> , <b>subnetMaskLen</b> , <b>router</b> )
cpp	int <b>useStaticIP</b> ( string <b>ipAddress</b> , int <b>subnetMaskLen</b> , string <b>router</b> )
m	-(int) <b>useStaticIP</b> : (NSString*) <b>ipAddress</b> : (int) <b>subnetMaskLen</b> : (NSString*) <b>router</b>
pas	LongInt <b>useStaticIP</b> ( <b>ipAddress</b> : string, <b>subnetMaskLen</b> : LongInt, <b>router</b> : string): LongInt
vb	function <b>useStaticIP</b> ( ByVal <b>ipAddress</b> As String, ByVal <b>subnetMaskLen</b> As Integer, ByVal <b>router</b> As String) As Integer
cs	int <b>useStaticIP</b> ( string <b>ipAddress</b> , int <b>subnetMaskLen</b> , string <b>router</b> )
java	int <b>useStaticIP</b> ( String <b>ipAddress</b> , int <b>subnetMaskLen</b> , String <b>router</b> )
uwp	async Task<int> <b>useStaticIP</b> ( string <b>ipAddress</b> , int <b>subnetMaskLen</b> , string <b>router</b> )
py	<b>useStaticIP</b> ( <b>ipAddress</b> , <b>subnetMaskLen</b> , <b>router</b> )
php	function <b>useStaticIP</b> ( \$ <b>ipAddress</b> , \$ <b>subnetMaskLen</b> , \$ <b>router</b> )
ts	async <b>useStaticIP</b> ( <b>ipAddress</b> : string, <b>subnetMaskLen</b> : number, <b>router</b> : string): Promise<number>
es	async <b>useStaticIP</b> ( <b>ipAddress</b> , <b>subnetMaskLen</b> , <b>router</b> )
dnp	int <b>useStaticIP</b> ( string <b>ipAddress</b> , int <b>subnetMaskLen</b> , string <b>router</b> )
cp	int <b>useStaticIP</b> ( string <b>ipAddress</b> , int <b>subnetMaskLen</b> , string <b>router</b> )
cmd	YNetwork <b>target useStaticIP</b> <b>ipAddress</b> <b>subnetMaskLen</b> <b>router</b>

N'oubliez pas d'appeler la méthode `saveToFlash()` et de redémarrer le module pour que le paramètre soit appliqué.

**Paramètres :**

- ipAddress** adresse IP à utiliser par le module
- subnetMaskLen** longueur du masque de sous-réseau à utiliser. Par exemple, la valeur 24 représente 255.255.255.0.
- router** adresse IP de la passerelle à utiliser ("default gateway")

**Retourne :**

YAPI . SUCCESS si l'opération se déroule sans erreur.

En cas d'erreur, déclenche une exception ou retourne un code d'erreur négatif.

**network**→**wait\_async()****YNetwork**

Attend que toutes les commandes asynchrones en cours d'exécution sur le module soient terminées, et appelle le callback passé en paramètre.

```
js fonction wait_async( callback, context)
```

```
ts wait_async( callback: Function, context: object)
```

```
es wait_async( callback, context)
```

La fonction callback peut donc librement utiliser des fonctions synchrones ou asynchrones, sans risquer de bloquer la machine virtuelle Javascript.

**Paramètres :**

**callback** fonction de callback qui sera appelée dès que toutes les commandes en cours d'exécution sur le module seront terminées. La fonction callback reçoit deux arguments: le contexte fourni par l'appelant et l'objet fonction concerné.

**context** contexte fourni par l'appelant, et qui sera passé tel-qu'el à la fonction de callback

**Retourne :**

rien du tout.

## 8.4. La classe YFiles

Interface pour interagir avec les systèmes de fichier, disponibles par exemple dans le Yocto-Color-V2, le Yocto-Serial, le YoctoHub-Ethernet et le YoctoHub-Wireless-n

La class YFiles permet d'accéder au système de fichier embarqué sur certains modules Yoctopuce. Le stockage de fichiers permet par exemple de personnaliser un service web (dans le cas d'un module connecté au réseau) ou pour d'ajouter un police de caractères (dans le cas d'un module d'affichage).

Pour utiliser les fonctions décrites ici, vous devez inclure:

js	<code>&lt;script type='text/javascript' src='yocto_files.js'&gt;&lt;/script&gt;</code>
cpp	<code>#include "yocto_files.h"</code>
m	<code>#import "yocto_files.h"</code>
pas	<code>uses yocto_files;</code>
vb	<code>yocto_files.vb</code>
cs	<code>yocto_files.cs</code>
java	<code>import com.yoctopuce.YoctoAPI.YFiles;</code>
uwp	<code>import com.yoctopuce.YoctoAPI.YFiles;</code>
py	<code>from yocto_files import *</code>
php	<code>require_once('yocto_files.php');</code>
ts	<code>in HTML: import { YFiles } from '../dist/esm/yocto_files.js';</code> <code>in Node.js: import { YFiles } from 'yoctolib-cjs/yocto_files.js';</code>
es	<code>in HTML: &lt;script src='../lib/yocto_files.js'&gt;&lt;/script&gt;</code> <code>in node.js: require('yoctolib-es2017/yocto_files.js');</code>
dnf	<code>import YoctoProxyAPI.YFilesProxy</code>
cp	<code>#include "yocto_files_proxy.h"</code>
vi	<code>YFiles.vi</code>
ml	<code>import YoctoProxyAPI.YFilesProxy</code>

### Fonction globales

#### **YFiles.FindFiles(func)**

Permet de retrouver un système de fichier d'après un identifiant donné.

#### **YFiles.FindFilesInContext(yctx, func)**

Permet de retrouver un système de fichier d'après un identifiant donné dans un Context YAPI.

#### **YFiles.FirstFiles()**

Commence l'énumération des systèmes de fichier accessibles par la librairie.

#### **YFiles.FirstFilesInContext(yctx)**

Commence l'énumération des systèmes de fichier accessibles par la librairie.

#### **YFiles.GetSimilarFunctions()**

Enumère toutes les fonctions de type Files disponibles sur les modules actuellement joignables par la librairie, et retourne leurs identifiants matériels uniques (hardwareId).

### Propriétés des objets YFilesProxy

#### **files**→**AdvertisedValue** *[lecture seule]*

Courte chaîne de caractères représentant l'état courant de la fonction.

#### **files**→**FilesCount** *[lecture seule]*

Nombre de fichiers présents dans le système de fichier.

#### **files**→**FriendlyName** *[lecture seule]*

Identifiant global de la fonction au format NOM\_MODULE . NOM\_FONCTION.

**files**→**FunctionId** [*lecture seule*]

Identifiant matériel du système de fichier, sans référence au module.

**files**→**HardwareId** [*lecture seule*]

Identifiant matériel unique de la fonction au format SERIAL . FUNCTIONID.

**files**→**IsOnline** [*lecture seule*]

Vérifie si le module hébergeant la fonction est joignable, sans déclencher d'erreur.

**files**→**LogicalName** [*modifiable*]

Nom logique de la fonction.

**files**→**SerialNumber** [*lecture seule*]

Numéro de série du module, préprogrammé en usine.

### Méthodes des objets YFiles

**files**→**clearCache()**

Invalide le cache.

**files**→**describe()**

Retourne un court texte décrivant de manière non-ambigüe l'instance du système de fichier au format TYPE ( NAME ) = SERIAL . FUNCTIONID.

**files**→**download(pathname)**

Télécharge le fichier choisi du filesystem et retourne son contenu.

**files**→**download\_async(pathname, callback, context)**

Procède au chargement du bloc suivant de mesures depuis l'enregistreur de données du module, de manière asynchrone.

**files**→**fileExist(filename)**

Test si un fichier existe dans le système de fichier du module.

**files**→**format\_fs()**

Rétabli le système de fichier dans son état original, défragmenté.

**files**→**get\_advertisedValue()**

Retourne la valeur courante du système de fichier (pas plus de 6 caractères).

**files**→**get\_errorMessage()**

Retourne le message correspondant à la dernière erreur survenue lors de l'utilisation du système de fichier.

**files**→**get\_errorType()**

Retourne le code d'erreur correspondant à la dernière erreur survenue lors de l'utilisation du système de fichier.

**files**→**get\_filesCount()**

Retourne le nombre de fichiers présents dans le système de fichier.

**files**→**get\_freeSpace()**

Retourne l'espace disponible dans le système de fichier pour charger des nouveaux fichiers, en octets.

**files**→**get\_friendlyName()**

Retourne un identifiant global du système de fichier au format NOM\_MODULE . NOM\_FONCTION.

**files**→**get\_functionDescriptor()**

Retourne un identifiant unique de type YFUN\_DESCR correspondant à la fonction.

**files**→**get\_functionId()**

Retourne l'identifiant matériel du système de fichier, sans référence au module.

**files**→**get\_hardwareId()**

Retourne l'identifiant matériel unique du système de fichier au format SERIAL . FUNCTIONID.

**files**→**get\_list(pattern)**

Retourne une liste d'objets objet YFileRecord qui décrivent les fichiers présents dans le système de fichier.

**files**→get\_logicalName()

Retourne le nom logique du système de fichier.

**files**→get\_module()

Retourne l'objet YModule correspondant au module Yoctopuce qui héberge la fonction.

**files**→get\_module\_async(callback, context)

Retourne l'objet YModule correspondant au module Yoctopuce qui héberge la fonction.

**files**→get\_serialNumber()

Retourne le numéro de série du module, préprogrammé en usine.

**files**→get\_userData()

Retourne le contenu de l'attribut userData, précédemment stocké à l'aide de la méthode set\_userData.

**files**→isOnline()

Vérifie si le module hébergeant le système de fichier est joignable, sans déclencher d'erreur.

**files**→isOnline\_async(callback, context)

Vérifie si le module hébergeant le système de fichier est joignable, sans déclencher d'erreur.

**files**→isReadOnly()

Test si la fonction est en lecture seule.

**files**→load(msValidity)

Met en cache les valeurs courantes du système de fichier, avec une durée de validité spécifiée.

**files**→loadAttribute(attrName)

Retourne la valeur actuelle d'un attribut spécifique de la fonction, sous forme de texte, le plus rapidement possible mais sans passer par le cache.

**files**→load\_async(msValidity, callback, context)

Met en cache les valeurs courantes du système de fichier, avec une durée de validité spécifiée.

**files**→muteValueCallbacks()

Désactive l'envoi de chaque changement de la valeur publiée au hub parent.

**files**→nextFiles()

Continue l'énumération des systèmes de fichier commencée à l'aide de yFirstFiles() Attention, vous ne pouvez faire aucune supposition sur l'ordre dans lequel les systèmes de fichier sont retournés.

**files**→registerValueCallback(callback)

Enregistre la fonction de callback qui est appelée à chaque changement de la valeur publiée.

**files**→remove(pathname)

Efface un fichier, spécifié par son path complet, du système de fichier.

**files**→set\_logicalName(newval)

Modifie le nom logique du système de fichier.

**files**→set\_userData(data)

Enregistre un contexte libre dans l'attribut userData de la fonction, afin de le retrouver plus tard à l'aide de la méthode get\_userData.

**files**→unmuteValueCallbacks()

Réactive l'envoi de chaque changement de la valeur publiée au hub parent.

**files**→upload(pathname, content)

Télécharge un contenu vers le système de fichier, au chemin d'accès spécifié.

**files**→wait\_async(callback, context)

Attend que toutes les commandes asynchrones en cours d'exécution sur le module soient terminées, et appelle le callback passé en paramètre.

## YFiles.FindFiles() YFiles.FindFiles()

YFiles

Permet de retrouver un système de fichier d'après un identifiant donné.

js	function <b>yFindFiles( func)</b>
cpp	YFiles* <b>FindFiles( string func)</b>
m	+(YFiles*) <b>FindFiles : (NSString*) func</b>
pas	TYFiles <b>yFindFiles( func: string): TYFiles</b>
vb	function <b>FindFiles( ByVal func As String) As YFiles</b>
cs	static YFiles <b>FindFiles( string func)</b>
java	static YFiles <b>FindFiles( String func)</b>
uwp	static YFiles <b>FindFiles( string func)</b>
py	<b>FindFiles( func)</b>
php	function <b>FindFiles( \$func)</b>
ts	static <b>FindFiles( func: string): YFiles</b>
es	static <b>FindFiles( func)</b>
dnp	static YFilesProxy <b>FindFiles( string func)</b>
cp	static YFilesProxy * <b>FindFiles( string func)</b>

L'identifiant peut être spécifié sous plusieurs formes:

- NomLogiqueFonction
- NoSerieModule.IdentifiantFonction
- NoSerieModule.NomLogiqueFonction
- NomLogiqueModule.IdentifiantMatériel
- NomLogiqueModule.NomLogiqueFonction

Cette fonction n'exige pas que le système de fichier soit en ligne au moment où elle est appelée, l'objet retourné sera néanmoins valide. Utiliser la méthode `YFiles.isOnline()` pour tester si le système de fichier est utilisable à un moment donné. En cas d'ambiguïté lorsqu'on fait une recherche par nom logique, aucune erreur ne sera notifiée: la première instance trouvée sera renvoyée. La recherche se fait d'abord par nom matériel, puis par nom logique.

Si un appel à la méthode `is_online()` de cet objet renvoie FAUX alors que vous êtes sûr que le module correspondant est bien branché, vérifiez que vous n'avez pas oublié d'appeler `registerHub()` à l'initialisation de de l'application.

### Paramètres :

**func** une chaîne de caractères qui référence le système de fichier sans ambiguïté, par exemple `YRGBLED2.files`.

### Retourne :

un objet de classe `YFiles` qui permet ensuite de contrôler le système de fichier.

## YFiles.FindFilesInContext() YFiles.FindFilesInContext()

YFiles

Permet de retrouver un système de fichier d'après un identifiant donné dans un Context YAPI.

```
java static YFiles FindFilesInContext( YAPIContext yctx, String func)
uwp static YFiles FindFilesInContext( YAPIContext yctx, string func)
ts static FindFilesInContext( yctx: YAPIContext, func: string): YFiles
es static FindFilesInContext( yctx, func)
```

L'identifiant peut être spécifié sous plusieurs formes:

- NomLogiqueFonction
- NoSerieModule.IdentifiantFonction
- NoSerieModule.NomLogiqueFonction
- NomLogiqueModule.IdentifiantMatériel
- NomLogiqueModule.NomLogiqueFonction

Cette fonction n'exige pas que le système de fichier soit en ligne au moment où elle est appelée, l'objet retourné sera néanmoins valide. Utiliser la méthode `YFiles.isOnline()` pour tester si le système de fichier est utilisable à un moment donné. En cas d'ambiguïté lorsqu'on fait une recherche par nom logique, aucune erreur ne sera notifiée: la première instance trouvée sera renvoyée. La recherche se fait d'abord par nom matériel, puis par nom logique.

### Paramètres :

**yctx** un contexte YAPI

**func** une chaîne de caractères qui référence le système de fichier sans ambiguïté, par exemple `YRGBLED2.files`.

### Retourne :

un objet de classe `YFiles` qui permet ensuite de contrôler le système de fichier.

## YFiles.FirstFiles() YFiles.FirstFiles()

YFiles

Commence l'énumération des systèmes de fichier accessibles par la librairie.

js	function <b>yFirstFiles</b> ( )
cpp	YFiles * <b>FirstFiles</b> ( )
m	+(YFiles*) <b>FirstFiles</b>
pas	TYFiles <b>yFirstFiles</b> ( ): TYFiles
vb	function <b>FirstFiles</b> ( ) As YFiles
cs	static YFiles <b>FirstFiles</b> ( )
java	static YFiles <b>FirstFiles</b> ( )
uwp	static YFiles <b>FirstFiles</b> ( )
py	<b>FirstFiles</b> ( )
php	function <b>FirstFiles</b> ( )
ts	static <b>FirstFiles</b> ( ): YFiles   null
es	static <b>FirstFiles</b> ( )

Utiliser la fonction `YFiles.nextFiles()` pour itérer sur les autres systèmes de fichier.

**Retourne :**

un pointeur sur un objet `YFiles`, correspondant au premier système de fichier accessible en ligne, ou `null` si il n'y a pas de systèmes de fichier disponibles.

## YFiles.FirstFilesInContext() YFiles.FirstFilesInContext()

YFiles

Commence l'énumération des systèmes de fichier accessibles par la librairie.

```
java static YFiles FirstFilesInContext( YAPIContext yctx)
uwp static YFiles FirstFilesInContext( YAPIContext yctx)
ts static FirstFilesInContext( yctx: YAPIContext): YFiles | null
es static FirstFilesInContext( yctx)
```

Utiliser la fonction `YFiles.nextFiles()` pour itérer sur les autres systèmes de fichier.

**Paramètres :**

`yctx` un contexte YAPI.

**Retourne :**

un pointeur sur un objet `YFiles`, correspondant au premier système de fichier accessible en ligne, ou `null` si il n'y a pas de systèmes de fichier disponibles.

## YFiles.GetSimilarFunctions() YFiles.GetSimilarFunctions()

YFiles

Enumère toutes les fonctions de type Files disponibles sur les modules actuellement joignables par la librairie, et retourne leurs identifiants matériels uniques (hardwareId).

```
dnf static new string[] GetSimilarFunctions( )
```

```
cp static vector<string> GetSimilarFunctions( )
```

Chaque chaîne retournée peut être passée en argument à la méthode `YFiles.FindFiles` pour obtenir un objet permettant d'interagir avec le module correspondant.

**Retourne :**

un tableau de chaînes de caractères, contenant les identifiants matériels de chaque fonction disponible trouvée.

## files→AdvertisedValue

YFiles

---

Courte chaîne de caractères représentant l'état courant de la fonction.

`dnf` string **AdvertisedValue**

---

**files**→**FilesCount****YFiles**

---

Nombre de fichiers présents dans le système de fichier.

`int` **FilesCount**

## files→FriendlyName

YFiles

---

Identifiant global de la fonction au format NOM\_MODULE . NOM\_FONCTION.

`dnf` string **FriendlyName**

Le chaîne retournée utilise soit les noms logiques du module et de la fonction si ils sont définis, soit respectivement le numéro de série du module et l'identifiant matériel de la fonction (par exemple: `MyCustomName.relay1`)

**files**→**FunctionId****YFiles**

Identifiant matériel du système de fichier, sans référence au module.

`dnf` string **FunctionId**

Par exemple `relay1`.

---

Identifiant matériel unique de la fonction au format `SERIAL.FUNCTIONID`.

`dnsp` string **HardwareId**

L'identifiant unique est composé du numéro de série du module et de l'identifiant matériel de la fonction (par exemple `RELAYLO1-123456.relay1`).

**files**→**IsOnline****YFiles**

Vérifie si le module hébergeant la fonction est joignable, sans déclencher d'erreur.

`dnf` `bool IsOnline`

Si les valeurs des attributs en cache de la fonction sont valides au moment de l'appel, le module est considéré joignable. Cette fonction ne cause en aucun cas d'exception, quelle que soit l'erreur qui pourrait se produire lors de la vérification de joignabilité.

## files→LogicalName

YFiles

Nom logique de la fonction.

```
dnf string LogicalName
```

**Modifiable.** Vous pouvez utiliser `yCheckLogicalName( )` pour vérifier si votre paramètre est valide. N'oubliez pas d'appeler la méthode `saveToFlash( )` du module si le réglage doit être préservé.

---

**files**→**SerialNumber****YFiles**

---

Numéro de série du module, préprogrammé en usine.

`dnsp` string **SerialNumber**

## files → clearCache()

YFiles

Invalide le cache.

js	function <b>clearCache</b> ( )
cpp	void <b>clearCache</b> ( )
m	-(void) <b>clearCache</b>
pas	<b>clearCache</b> ( )
vb	procedure <b>clearCache</b> ( )
cs	void <b>clearCache</b> ( )
java	void <b>clearCache</b> ( )
py	<b>clearCache</b> ( )
php	function <b>clearCache</b> ( )
ts	async <b>clearCache</b> ( ): Promise<void>
es	async <b>clearCache</b> ( )

Invalide le cache des valeurs courantes du système de fichier. Force le prochain appel à une méthode `get_xxx()` ou `loadxxx()` pour charger les les données depuis le module.

**files**→**describe()****YFiles**

Retourne un court texte décrivant de manière non-ambigüe l'instance du système de fichier au format `TYPE (NAME) =SERIAL . FUNCTIONID`.

js	function <b>describe</b> ( )
cpp	string <b>describe</b> ( )
m	-(NSString*) <b>describe</b>
pas	string <b>describe</b> ( ): string
vb	function <b>describe</b> ( ) As String
cs	string <b>describe</b> ( )
java	String <b>describe</b> ( )
py	<b>describe</b> ( )
php	function <b>describe</b> ( )
ts	async <b>describe</b> ( ): Promise<string>
es	async <b>describe</b> ( )

Plus précisément, `TYPE` correspond au type de fonction, `NAME` correspond au nom utilisé lors du premier accès à la fonction, `SERIAL` correspond au numéro de série du module si le module est connecté, ou "unresolved" sinon, et `FUNCTIONID` correspond à l'identifiant matériel de la fonction si le module est connecté. Par exemple, La méthode va retourner `Relay(MyCustomName.relay1)=RELAYLO1-123456.relay1` si le module est déjà connecté ou `Relay(BadCustomName.relay1)=unresolved` si le module n'est pas déjà connecté. Cette méthode ne déclenche aucune transaction USB ou TCP et peut donc être utilisé dans un debugeur.

**Retourne :**

une chaîne de caractères décrivant le système de fichier (ex:  
`Relay(MyCustomName.relay1)=RELAYLO1-123456.relay1`)

## files→download()

Télécharge le fichier choisi du filesystem et retourne son contenu.

js	function <b>download</b> ( <b>pathname</b> )
cpp	string <b>download</b> ( string <b>pathname</b> )
m	-(NSMutableData*) <b>download</b> : (NSString*) <b>pathname</b>
pas	TByteArray <b>download</b> ( <b>pathname</b> : string): TByteArray
vb	function <b>download</b> ( ByVal <b>pathname</b> As String) As Byte
cs	byte[] <b>download</b> ( string <b>pathname</b> )
java	byte[] <b>download</b> ( String <b>pathname</b> )
uwp	async Task<byte[]> <b>download</b> ( string <b>pathname</b> )
py	<b>download</b> ( <b>pathname</b> )
php	function <b>download</b> ( <b>\$pathname</b> )
ts	async <b>download</b> ( <b>pathname</b> : string): Promise<Uint8Array>
es	async <b>download</b> ( <b>pathname</b> )
dnp	byte[] <b>download</b> ( string <b>pathname</b> )
cp	string <b>download</b> ( string <b>pathname</b> )
cmd	YFiles <b>target download</b> <b>pathname</b>

**Paramètres :**

**pathname** nom complet du fichier à charger, y compris le chemin d'accès.

**Retourne :**

le contenu du fichier chargé sous forme d'objet binaire

En cas d'erreur, déclenche une exception ou retourne un contenu vide.

**files**→**download\_async()****YFiles**

Procède au chargement du bloc suivant de mesures depuis l'enregistreur de données du module, de manière asynchrone.

```
js function download_async( pathname, callback, context)
```

**Paramètres :**

**pathname** nom complet du fichier à charger, y compris le chemin d'accès.

**callback** fonction fournie par l'utilisateur, qui sera appelée lorsque la suite du chargement aura été effectué. La fonction callback doit prendre trois arguments: - la variable de contexte à disposition de l'utilisateur - l'objet YFiles dont la méthode `download_async` a été appelée - le contenu du fichier chargé sous forme d'objet binaire

**context** variable de contexte à disposition de l'utilisateur

**Retourne :**

rien.

## files → fileExist()

YFiles

Test si un fichier existe dans le système de fichier du module.

js	function <b>fileExist</b> ( <b>filename</b> )
c++	bool <b>fileExist</b> ( string <b>filename</b> )
m	-(bool) <b>fileExist</b> : (NSString*) <b>filename</b>
pas	boolean <b>fileExist</b> ( <b>filename</b> : string): boolean
vb	function <b>fileExist</b> ( ByVal <b>filename</b> As String) As Boolean
cs	bool <b>fileExist</b> ( string <b>filename</b> )
java	boolean <b>fileExist</b> ( String <b>filename</b> )
uwp	async Task<bool> <b>fileExist</b> ( string <b>filename</b> )
py	<b>fileExist</b> ( <b>filename</b> )
php	function <b>fileExist</b> ( <b>\$filename</b> )
ts	async <b>fileExist</b> ( <b>filename</b> : string): Promise<boolean>
es	async <b>fileExist</b> ( <b>filename</b> )
dnp	bool <b>fileExist</b> ( string <b>filename</b> )
cp	bool <b>fileExist</b> ( string <b>filename</b> )
cmd	YFiles <b>target fileExist filename</b>

**Paramètres :**

**filename** le nom de fichier.

**Retourne :**

vrai si le fichier existe, et faux si le fichier n'existe pas.

En cas d'erreur, déclenche une exception.

**files**→**format\_fs()****YFiles**

Rétabli le système de fichier dans on état original, défragmenté.

js	function <b>format_fs</b> ( )
cpp	int <b>format_fs</b> ( )
m	-(int) <b>format_fs</b>
pas	LongInt <b>format_fs</b> ( ): LongInt
vb	function <b>format_fs</b> ( ) As Integer
cs	int <b>format_fs</b> ( )
java	int <b>format_fs</b> ( )
uwp	async Task<int> <b>format_fs</b> ( )
py	<b>format_fs</b> ( )
php	function <b>format_fs</b> ( )
ts	async <b>format_fs</b> ( ): Promise<number>
es	async <b>format_fs</b> ( )
dnp	int <b>format_fs</b> ( )
cp	int <b>format_fs</b> ( )
cmd	YFiles <b>target format_fs</b>

entièrement vide. Tous les fichiers précédemment chargés sont irrémédiablement effacés.

**Retourne :**

YAPI . SUCCESS si l'opération se déroule sans erreur.

En cas d'erreur, déclenche une exception ou retourne un code d'erreur négatif.

**files**→**get\_advertisedValue()****files**→**advertisedValue()**

Retourne la valeur courante du système de fichier (pas plus de 6 caractères).

js	function <b>get_advertisedValue</b> ( )
cpp	string <b>get_advertisedValue</b> ( )
m	-(NSString*) advertisedValue
pas	string <b>get_advertisedValue</b> ( ): string
vb	function <b>get_advertisedValue</b> ( ) As String
cs	string <b>get_advertisedValue</b> ( )
java	String <b>get_advertisedValue</b> ( )
uwp	async Task<string> <b>get_advertisedValue</b> ( )
py	<b>get_advertisedValue</b> ( )
php	function <b>get_advertisedValue</b> ( )
ts	async <b>get_advertisedValue</b> ( ): Promise<string>
es	async <b>get_advertisedValue</b> ( )
dnp	string <b>get_advertisedValue</b> ( )
cp	string <b>get_advertisedValue</b> ( )
cmd	YFiles <b>target</b> <b>get_advertisedValue</b>

**Retourne :**

une chaîne de caractères représentant la valeur courante du système de fichier (pas plus de 6 caractères).

En cas d'erreur, déclenche une exception ou retourne `YFiles.ADVERTISEDVALUE_INVALID`.

**files**→**get\_errorMessage()****YFiles****files**→**errorMessage()**

Retourne le message correspondant à la dernière erreur survenue lors de l'utilisation du système de fichier.

js	function <b>get_errorMessage</b> ( )
cpp	string <b>get_errorMessage</b> ( )
m	-(NSString*) <b>errorMessage</b>
pas	string <b>get_errorMessage</b> ( ): string
vb	function <b>get_errorMessage</b> ( ) As String
cs	string <b>get_errorMessage</b> ( )
java	String <b>get_errorMessage</b> ( )
py	<b>get_errorMessage</b> ( )
php	function <b>get_errorMessage</b> ( )
ts	<b>get_errorMessage</b> ( ): string
es	<b>get_errorMessage</b> ( )

Cette méthode est principalement utile lorsque la librairie Yoctopuce est utilisée en désactivant la gestion des exceptions.

**Retourne :**

une chaîne de caractères correspondant au message de la dernière erreur qui s'est produit lors de l'utilisation du système de fichier.

**files**→**get\_errorType()****files**→**errorType()**

Retourne le code d'erreur correspondant à la dernière erreur survenue lors de l'utilisation du système de fichier.

js	function <b>get_errorType</b> ( )
cpp	YRETCODE <b>get_errorType</b> ( )
m	-(YRETCODE) errorType
pas	YRETCODE <b>get_errorType</b> ( ): YRETCODE
vb	function <b>get_errorType</b> ( ) As YRETCODE
cs	YRETCODE <b>get_errorType</b> ( )
java	int <b>get_errorType</b> ( )
py	<b>get_errorType</b> ( )
php	function <b>get_errorType</b> ( )
ts	<b>get_errorType</b> ( ): number
es	<b>get_errorType</b> ( )

Cette méthode est principalement utile lorsque la librairie Yoctopuce est utilisée en désactivant la gestion des exceptions.

**Retourne :**

un nombre correspondant au code de la dernière erreur qui s'est produit lors de l'utilisation du système de fichier.

**files**→**get\_filesCount()****YFiles****files**→**filesCount()**

Retourne le nombre de fichiers présents dans le système de fichier.

js	function <b>get_filesCount</b> ( )
cpp	int <b>get_filesCount</b> ( )
m	-(int) filesCount
pas	LongInt <b>get_filesCount</b> ( ): LongInt
vb	function <b>get_filesCount</b> ( ) As Integer
cs	int <b>get_filesCount</b> ( )
java	int <b>get_filesCount</b> ( )
uwp	async Task<int> <b>get_filesCount</b> ( )
py	<b>get_filesCount</b> ( )
php	function <b>get_filesCount</b> ( )
ts	async <b>get_filesCount</b> ( ): Promise<number>
es	async <b>get_filesCount</b> ( )
dnp	int <b>get_filesCount</b> ( )
cp	int <b>get_filesCount</b> ( )
cmd	YFiles <b>target</b> <b>get_filesCount</b>

**Retourne :**

un entier représentant le nombre de fichiers présents dans le système de fichier

En cas d'erreur, déclenche une exception ou retourne `YFiles.FILES_COUNT_INVALID`.

## files→get\_freeSpace()

## files→freeSpace()

Retourne l'espace disponible dans le système de fichier pour charger des nouveaux fichiers, en octets.

js	function <b>get_freeSpace</b> ( )
cpp	int <b>get_freeSpace</b> ( )
m	-(int) freeSpace
pas	LongInt <b>get_freeSpace</b> ( ): LongInt
vb	function <b>get_freeSpace</b> ( ) As Integer
cs	int <b>get_freeSpace</b> ( )
java	int <b>get_freeSpace</b> ( )
uwp	async Task<int> <b>get_freeSpace</b> ( )
py	<b>get_freeSpace</b> ( )
php	function <b>get_freeSpace</b> ( )
ts	async <b>get_freeSpace</b> ( ): Promise<number>
es	async <b>get_freeSpace</b> ( )
dnp	int <b>get_freeSpace</b> ( )
cp	int <b>get_freeSpace</b> ( )
cmd	YFiles <b>target</b> <b>get_freeSpace</b>

**Retourne :**

un entier représentant l'espace disponible dans le système de fichier pour charger des nouveaux fichiers, en octets

En cas d'erreur, déclenche une exception ou retourne `YFiles.FREESPACE_INVALID`.

**files**→**get\_friendlyName()****YFiles****files**→**friendlyName()**

Retourne un identifiant global du système de fichier au format `NOM_MODULE . NOM_FONCTION`.

js	function <b>get_friendlyName</b> ( )
cpp	string <b>get_friendlyName</b> ( )
m	-(NSString*) <b>friendlyName</b>
cs	string <b>get_friendlyName</b> ( )
java	String <b>get_friendlyName</b> ( )
py	<b>get_friendlyName</b> ( )
php	function <b>get_friendlyName</b> ( )
ts	async <b>get_friendlyName</b> ( ): Promise<string>
es	async <b>get_friendlyName</b> ( )
dnp	string <b>get_friendlyName</b> ( )
cp	string <b>get_friendlyName</b> ( )

Le chaîne retournée utilise soit les noms logiques du module et du système de fichier si ils sont définis, soit respectivement le numéro de série du module et l'identifiant matériel du système de fichier (par exemple: `MyCustomName.relay1`)

**Retourne :**

une chaîne de caractères identifiant le système de fichier en utilisant les noms logiques (ex: `MyCustomName.relay1`)

En cas d'erreur, déclenche une exception ou retourne `YFiles.FRIENDLYNAME_INVALID`.

**files**→**get\_functionDescriptor()****files**→**functionDescriptor()**

Retourne un identifiant unique de type YFUN\_DESCR correspondant à la fonction.

js	function <b>get_functionDescriptor</b> ( )
cpp	YFUN_DESCR <b>get_functionDescriptor</b> ( )
m	-(YFUN_DESCR) functionDescriptor
pas	YFUN_DESCR <b>get_functionDescriptor</b> ( ): YFUN_DESCR
vb	function <b>get_functionDescriptor</b> ( ) As YFUN_DESCR
cs	YFUN_DESCR <b>get_functionDescriptor</b> ( )
java	String <b>get_functionDescriptor</b> ( )
py	<b>get_functionDescriptor</b> ( )
php	function <b>get_functionDescriptor</b> ( )
ts	async <b>get_functionDescriptor</b> ( ): Promise<string>
es	async <b>get_functionDescriptor</b> ( )

Cet identifiant peut être utilisé pour tester si deux instance de YFunction référencent physiquement la même fonction sur le même module.

**Retourne :**

un identifiant de type YFUN\_DESCR.

Si la fonction n'a jamais été contactée, la valeur retournée sera Y\$CLASSNAME\$.FUNCTIONDESCRIPTOR\_INVALID

**files**→**get\_functionId()****YFiles****files**→**functionId()**

Retourne l'identifiant matériel du système de fichier, sans référence au module.

js	function <b>get_functionId</b> ( )
cpp	string <b>get_functionId</b> ( )
m	-(NSString*) <b>functionId</b>
vb	function <b>get_functionId</b> ( ) As String
cs	string <b>get_functionId</b> ( )
java	String <b>get_functionId</b> ( )
py	<b>get_functionId</b> ( )
php	function <b>get_functionId</b> ( )
ts	async <b>get_functionId</b> ( ): Promise<string>
es	async <b>get_functionId</b> ( )
dnp	string <b>get_functionId</b> ( )
cp	string <b>get_functionId</b> ( )

Par exemple `relay1`.

**Retourne :**

une chaîne de caractères identifiant le système de fichier (ex: `relay1`)

En cas d'erreur, déclenche une exception ou retourne `YFiles.FUNCTIONID_INVALID`.

**files**→**get\_hardwareId()****files**→**hardwareId()**

Retourne l'identifiant matériel unique du système de fichier au format SERIAL . FUNCTIONID.

js	function <b>get_hardwareId</b> ( )
cpp	string <b>get_hardwareId</b> ( )
m	-(NSString*) hardwareId
vb	function <b>get_hardwareId</b> ( ) As String
cs	string <b>get_hardwareId</b> ( )
java	String <b>get_hardwareId</b> ( )
py	<b>get_hardwareId</b> ( )
php	function <b>get_hardwareId</b> ( )
ts	async <b>get_hardwareId</b> ( ): Promise<string>
es	async <b>get_hardwareId</b> ( )
dnp	string <b>get_hardwareId</b> ( )
cp	string <b>get_hardwareId</b> ( )

L'identifiant unique est composé du numéro de série du module et de l'identifiant matériel du système de fichier (par exemple RELAYLO1-123456 . relay1).

**Retourne :**

une chaîne de caractères identifiant le système de fichier (ex: RELAYLO1-123456 . relay1)

En cas d'erreur, déclenche une exception ou retourne YFiles . HARDWAREID\_INVALID.

**files**→**get\_list()****YFiles****files**→**list()**

Retourne une liste d'objets objet YFileRecord qui décrivent les fichiers présents dans le système de fichier.

js	function <b>get_list</b> ( <b>pattern</b> )
cpp	vector<YFileRecord> <b>get_list</b> ( string <b>pattern</b> )
m	-(NSMutableArray*) list : (NSString*) <b>pattern</b>
pas	TYFileRecordArray <b>get_list</b> ( <b>pattern</b> : string): TYFileRecordArray
vb	function <b>get_list</b> ( ByVal <b>pattern</b> As String) As List
cs	List<YFileRecord> <b>get_list</b> ( string <b>pattern</b> )
java	ArrayList<YFileRecord> <b>get_list</b> ( String <b>pattern</b> )
uwp	async Task<List<YFileRecord>> <b>get_list</b> ( string <b>pattern</b> )
py	<b>get_list</b> ( <b>pattern</b> )
php	function <b>get_list</b> ( \$ <b>pattern</b> )
ts	async <b>get_list</b> ( <b>pattern</b> : string): Promise<YFileRecord[]>
es	async <b>get_list</b> ( <b>pattern</b> )
dnp	YFileRecordProxy[] <b>get_list</b> ( string <b>pattern</b> )
cp	vector<YFileRecordProxy> <b>get_list</b> ( string <b>pattern</b> )
cmd	YFiles <b>target get_list pattern</b>

**Paramètres :**

**pattern** un filtre optionnel sur les noms de fichiers retournés, pouvant contenir des astérisques et des points d'interrogations comme jokers. Si le pattern fourni est vide, tous les fichiers sont retournés.

**Retourne :**

une liste d'objets YFileRecord, contenant le nom complet (y compris le chemin d'accès), la taille en octets et le CRC 32-bit du contenu du fichier.

En cas d'erreur, déclenche une exception ou retourne une liste vide.

**files**→**get\_logicalName()****files**→**logicalName()**

Retourne le nom logique du système de fichier.

js	function <b>get_logicalName</b> ( )
cpp	string <b>get_logicalName</b> ( )
m	-(NSString*) logicalName
pas	string <b>get_logicalName</b> ( ): string
vb	function <b>get_logicalName</b> ( ) As String
cs	string <b>get_logicalName</b> ( )
java	String <b>get_logicalName</b> ( )
uwp	async Task<string> <b>get_logicalName</b> ( )
py	<b>get_logicalName</b> ( )
php	function <b>get_logicalName</b> ( )
ts	async <b>get_logicalName</b> ( ): Promise<string>
es	async <b>get_logicalName</b> ( )
dnp	string <b>get_logicalName</b> ( )
cp	string <b>get_logicalName</b> ( )
cmd	YFiles <b>target</b> <b>get_logicalName</b>

**Retourne :**

une chaîne de caractères représentant le nom logique du système de fichier.

En cas d'erreur, déclenche une exception ou retourne `YFiles.LOGICALNAME_INVALID`.

**files**→**get\_module()**

YFiles

**files**→**module()**

Retourne l'objet `YModule` correspondant au module Yoctopuce qui héberge la fonction.

js	function <b>get_module</b> ( )
cpp	<code>YModule *</code> <b>get_module</b> ( )
m	-( <code>YModule*</code> ) module
pas	<code>TYModule</code> <b>get_module</b> ( ): <code>TYModule</code>
vb	function <b>get_module</b> ( ) As <code>YModule</code>
cs	<code>YModule</code> <b>get_module</b> ( )
java	<code>YModule</code> <b>get_module</b> ( )
py	<b>get_module</b> ( )
php	function <b>get_module</b> ( )
ts	async <b>get_module</b> ( ): <code>Promise&lt;YModule&gt;</code>
es	async <b>get_module</b> ( )
dnp	<code>YModuleProxy</code> <b>get_module</b> ( )
cp	<code>YModuleProxy *</code> <b>get_module</b> ( )

Si la fonction ne peut être trouvée sur aucun module, l'instance de `YModule` retournée ne sera pas joignable.

**Retourne :**

une instance de `YModule`

**files**→**get\_module\_async()**

YFiles

**files**→**module\_async()**

Retourne l'objet `YModule` correspondant au module Yoctopuce qui héberge la fonction.

```
js fonction get_module_async( callback, context)
```

Si la fonction ne peut être trouvée sur aucun module, l'instance de `YModule` retournée ne sera pas joignable.

Cette version asynchrone n'existe qu'en Javascript. Elle utilise une fonction de callback plutôt qu'une simple valeur de retour, pour éviter de bloquer la VM Javascript de Firefox, qui n'implémente pas le passage de contrôle entre threads durant les appels d'entrée/sortie bloquants.

**Paramètres :**

**callback** fonction de callback qui sera appelée dès que le résultat sera connu. La fonction callback reçoit trois arguments: le contexte fourni par l'appelant, l'objet fonction concerné et l'instance demandée de `YModule`

**context** contexte fourni par l'appelant, et qui sera passé tel-quel à la fonction de callback

**Retourne :**

rien du tout : le résultat sera passé en paramètre à la fonction de callback.

**files**→**get\_serialNumber()****YFiles****files**→**serialNumber()**

Retourne le numéro de série du module, préprogrammé en usine.

js	function <b>get_serialNumber</b> ( )
cpp	string <b>get_serialNumber</b> ( )
m	-(NSString*) <b>serialNumber</b>
pas	string <b>get_serialNumber</b> ( ): string
vb	function <b>get_serialNumber</b> ( ) As String
cs	string <b>get_serialNumber</b> ( )
java	String <b>get_serialNumber</b> ( )
uwp	async Task<string> <b>get_serialNumber</b> ( )
py	<b>get_serialNumber</b> ( )
php	function <b>get_serialNumber</b> ( )
ts	async <b>get_serialNumber</b> ( ): Promise<string>
es	async <b>get_serialNumber</b> ( )
dnp	string <b>get_serialNumber</b> ( )
cp	string <b>get_serialNumber</b> ( )
cmd	YFiles <b>target get_serialNumber</b>

**Retourne :**

: une chaîne de caractères représentant le numéro de série du module, préprogrammé en usine.

En cas d'erreur, déclenche une exception ou retourne YFunction.SERIALNUMBER\_INVALID.

**files**→**get\_userdata()****files**→**userData()**

Retourne le contenu de l'attribut `userData`, précédemment stocké à l'aide de la méthode `set_userdata`.

js	function <b>get_userdata</b> ( )
cpp	void * <b>get_userdata</b> ( )
m	-(id) <code>userData</code>
pas	Tobject <b>get_userdata</b> ( ): Tobject
vb	function <b>get_userdata</b> ( ) As Object
cs	object <b>get_userdata</b> ( )
java	Object <b>get_userdata</b> ( )
py	<b>get_userdata</b> ( )
php	function <b>get_userdata</b> ( )
ts	async <b>get_userdata</b> ( ): Promise<object null>
es	async <b>get_userdata</b> ( )

Cet attribut n'est pas utilisé directement par l'API. Il est à la disposition de l'appelant pour stocker un contexte.

**Retourne :**

l'objet stocké précédemment par l'appelant.

**files**→**isOnline()****YFiles**

Vérifie si le module hébergeant le système de fichier est joignable, sans déclencher d'erreur.

js	function <b>isOnline</b> ( )
cpp	bool <b>isOnline</b> ( )
m	-(BOOL) <b>isOnline</b>
pas	boolean <b>isOnline</b> ( ): boolean
vb	function <b>isOnline</b> ( ) As Boolean
cs	bool <b>isOnline</b> ( )
java	boolean <b>isOnline</b> ( )
py	<b>isOnline</b> ( )
php	function <b>isOnline</b> ( )
ts	async <b>isOnline</b> ( ): Promise<boolean>
es	async <b>isOnline</b> ( )
dnp	bool <b>isOnline</b> ( )
cp	bool <b>isOnline</b> ( )

Si les valeurs des attributs en cache du système de fichier sont valides au moment de l'appel, le module est considéré joignable. Cette fonction ne cause en aucun cas d'exception, quelle que soit l'erreur qui pourrait se produire lors de la vérification de joignabilité.

**Retourne :**

`true` si le système de fichier est joignable, `false` sinon

**files**→**isOnline\_async()****YFiles**

Vérifie si le module hébergeant le système de fichier est joignable, sans déclencher d'erreur.

```
js fonction isOnline_async( callback, context)
```

Si les valeurs des attributs en cache du système de fichier sont valides au moment de l'appel, le module est considéré joignable. Cette fonction ne cause en aucun cas d'exception, quelle que soit l'erreur qui pourrait se produire lors de la vérification de joignabilité.

Cette version asynchrone n'existe qu'en Javascript. Elle utilise une fonction de callback plutôt qu'une simple valeur de retour, pour éviter de bloquer la machine virtuelle Javascript avec une attente active.

**Paramètres :**

**callback** fonction de callback qui sera appelée dès que le résultat sera connu. La fonction callback reçoit trois arguments: le contexte fourni par l'appelant, l'objet fonction concerné et le résultat booléen

**context** contexte fourni par l'appelant, et qui sera passé tel-quel à la fonction de callback

**Retourne :**

rien du tout : le résultat sera passé en paramètre à la fonction de callback.

**files**→**isReadOnly()****YFiles**

Test si la fonction est en lecture seule.

cpp	bool <b>isReadOnly</b> ( )
m	-(bool) <b>isReadOnly</b>
pas	boolean <b>isReadOnly</b> ( ): boolean
vb	function <b>isReadOnly</b> ( ) As Boolean
cs	bool <b>isReadOnly</b> ( )
java	boolean <b>isReadOnly</b> ( )
uwp	async Task<bool> <b>isReadOnly</b> ( )
py	<b>isReadOnly</b> ( )
php	function <b>isReadOnly</b> ( )
ts	async <b>isReadOnly</b> ( ): Promise<boolean>
es	async <b>isReadOnly</b> ( )
dnp	bool <b>isReadOnly</b> ( )
cp	bool <b>isReadOnly</b> ( )
cmd	YFiles <b>target isReadOnly</b>

Retourne vrais si la fonction est protégé en ecriture ou que la fontion n'est pas disponible.

**Retourne :**

`true` si la fonction est protégé en ecriture ou que la fontion n'est pas disponible

## files→load()

Met en cache les valeurs courantes du système de fichier, avec une durée de validité spécifiée.

js	function <b>load</b> ( <b>msValidity</b> )
cpp	YRETCODE <b>load</b> ( int <b>msValidity</b> )
m	-(YRETCODE) <b>load</b> : (u64) <b>msValidity</b>
pas	YRETCODE <b>load</b> ( <b>msValidity</b> : u64): YRETCODE
vb	function <b>load</b> ( ByVal <b>msValidity</b> As Long) As YRETCODE
cs	YRETCODE <b>load</b> ( ulong <b>msValidity</b> )
java	int <b>load</b> ( long <b>msValidity</b> )
py	<b>load</b> ( <b>msValidity</b> )
php	function <b>load</b> ( <b>\$msValidity</b> )
ts	async <b>load</b> ( <b>msValidity</b> : number): Promise<number>
es	async <b>load</b> ( <b>msValidity</b> )

Par défaut, lorsqu'on accède à un module, tous les attributs des fonctions du module sont automatiquement mises en cache pour la durée standard (5 ms). Cette méthode peut être utilisée pour marquer occasionnellement les données cachées comme valides pour une plus longue période, par exemple dans le but de réduire le trafic réseau.

**Paramètres :**

**msValidity** un entier correspondant à la durée de validité attribuée aux les paramètres chargés, en millisecondes

**Retourne :**

YAPI . SUCCESS si l'opération se déroule sans erreur.

En cas d'erreur, déclenche une exception ou retourne un code d'erreur négatif.

**files**→**loadAttribute()****YFiles**

Retourne la valeur actuelle d'un attribut spécifique de la fonction, sous forme de texte, le plus rapidement possible mais sans passer par le cache.

js	function <b>loadAttribute</b> ( <b>attrName</b> )
cpp	string <b>loadAttribute</b> ( string <b>attrName</b> )
m	-(NSString*) <b>loadAttribute</b> : (NSString*) <b>attrName</b>
pas	string <b>loadAttribute</b> ( <b>attrName</b> : string): string
vb	function <b>loadAttribute</b> ( ByVal <b>attrName</b> As String) As String
cs	string <b>loadAttribute</b> ( string <b>attrName</b> )
java	String <b>loadAttribute</b> ( String <b>attrName</b> )
uwp	async Task<string> <b>loadAttribute</b> ( string <b>attrName</b> )
py	<b>loadAttribute</b> ( <b>attrName</b> )
php	function <b>loadAttribute</b> ( <b>\$attrName</b> )
ts	async <b>loadAttribute</b> ( <b>attrName</b> : string): Promise<string>
es	async <b>loadAttribute</b> ( <b>attrName</b> )
dnb	string <b>loadAttribute</b> ( string <b>attrName</b> )
cp	string <b>loadAttribute</b> ( string <b>attrName</b> )

**Paramètres :**

**attrName** le nom de l'attribut désiré

**Retourne :**

une chaîne de caractères représentant la valeur actuelle de l'attribut.

En cas d'erreur, déclenche une exception ou retourne un chaîne vide.

## files→load\_async()

YFiles

Met en cache les valeurs courantes du système de fichier, avec une durée de validité spécifiée.

```
js fonction load_async( msValidity, callback, context)
```

Par défaut, lorsqu'on accède à un module, tous les attributs des fonctions du module sont automatiquement mises en cache pour la durée standard (5 ms). Cette méthode peut être utilisée pour marquer occasionnellement les données cachées comme valides pour une plus longue période, par exemple dans le but de réduire le trafic réseau.

Cette version asynchrone n'existe qu'en Javascript. Elle utilise une fonction de callback plutôt qu'une simple valeur de retour, pour éviter de bloquer la machine virtuelle Javascript avec une attente active.

**Paramètres :**

- msValidity** un entier correspondant à la durée de validité attribuée aux les paramètres chargés, en millisecondes
- callback** fonction de callback qui sera appelée dès que le résultat sera connu. La fonction callback reçoit trois arguments: le contexte fourni par l'appelant, l'objet fonction concerné et le code d'erreur (ou YAPI . SUCCESS)
- context** contexte fourni par l'appelant, et qui sera passé tel-quel à la fonction de callback

**Retourne :**

rien du tout : le résultat sera passé en paramètre à la fonction de callback.

**files**→**muteValueCallbacks()****YFiles**

Désactive l'envoi de chaque changement de la valeur publiée au hub parent.

js	function <b>muteValueCallbacks</b> ( )
cpp	int <b>muteValueCallbacks</b> ( )
m	-(int) <b>muteValueCallbacks</b>
pas	LongInt <b>muteValueCallbacks</b> ( ): LongInt
vb	function <b>muteValueCallbacks</b> ( ) As Integer
cs	int <b>muteValueCallbacks</b> ( )
java	int <b>muteValueCallbacks</b> ( )
uwp	async Task<int> <b>muteValueCallbacks</b> ( )
py	<b>muteValueCallbacks</b> ( )
php	function <b>muteValueCallbacks</b> ( )
ts	async <b>muteValueCallbacks</b> ( ): Promise<number>
es	async <b>muteValueCallbacks</b> ( )
dnp	int <b>muteValueCallbacks</b> ( )
cp	int <b>muteValueCallbacks</b> ( )
cmd	YFiles <b>target</b> <b>muteValueCallbacks</b>

Vous pouvez utiliser cette fonction pour économiser la bande passante et le CPU sur les machines de faible puissance, ou pour éviter le déclenchement de callbacks HTTP. N'oubliez pas d'appeler la méthode `saveToFlash()` du module si le réglage doit être préservé.

**Retourne :**

YAPI . SUCCESS si l'opération se déroule sans erreur.

En cas d'erreur, déclenche une exception ou retourne un code d'erreur négatif.

**files** → **nextFiles()****YFiles**

Continue l'énumération des systèmes de fichier commencée à l'aide de `yFirstFiles()`. Attention, vous ne pouvez faire aucune supposition sur l'ordre dans lequel les systèmes de fichier sont retournés.

js	function <b>nextFiles</b> ( )
cpp	YFiles * <b>nextFiles</b> ( )
m	-(nullable YFiles*) <b>nextFiles</b>
pas	TYFiles <b>nextFiles</b> ( ): TYFiles
vb	function <b>nextFiles</b> ( ) As YFiles
cs	YFiles <b>nextFiles</b> ( )
java	YFiles <b>nextFiles</b> ( )
uwp	YFiles <b>nextFiles</b> ( )
py	<b>nextFiles</b> ( )
php	function <b>nextFiles</b> ( )
ts	<b>nextFiles</b> ( ): YFiles   null
es	<b>nextFiles</b> ( )

Si vous souhaitez retrouver un système de fichier spécifique, utilisez `Files.findFiles()` avec un `hardwareID` ou un nom logique.

**Retourne :**

un pointeur sur un objet `YFiles` accessible en ligne, ou `null` lorsque l'énumération est terminée.

## files→registerValueCallback()

## YFiles

Enregistre la fonction de callback qui est appelée à chaque changement de la valeur publiée.

js	function <b>registerValueCallback</b> ( <b>callback</b> )
cpp	int <b>registerValueCallback</b> ( YFilesValueCallback <b>callback</b> )
m	-(int) <b>registerValueCallback</b> : (YFilesValueCallback _Nullable) <b>callback</b>
pas	LongInt <b>registerValueCallback</b> ( <b>callback</b> : TYFilesValueCallback): LongInt
vb	function <b>registerValueCallback</b> ( ByVal <b>callback</b> As YFilesValueCallback) As Integer
cs	int <b>registerValueCallback</b> ( ValueCallback <b>callback</b> )
java	int <b>registerValueCallback</b> ( UpdateCallback <b>callback</b> )
uwp	async Task<int> <b>registerValueCallback</b> ( ValueCallback <b>callback</b> )
py	<b>registerValueCallback</b> ( <b>callback</b> )
php	function <b>registerValueCallback</b> ( <b>\$callback</b> )
ts	async <b>registerValueCallback</b> ( <b>callback</b> : YFilesValueCallback   null): Promise<number>
es	async <b>registerValueCallback</b> ( <b>callback</b> )

Ce callback n'est appelé que durant l'exécution de `ySleep` ou `yHandleEvents`. Cela permet à l'appelant de contrôler quand les callback peuvent se produire. Il est important d'appeler l'une de ces deux fonctions périodiquement pour garantir que les callback ne soient pas appelés trop tard. Pour désactiver un callback, il suffit d'appeler cette méthode en lui passant un pointeur nul.

**Paramètres :**

**callback** la fonction de callback à rappeler, ou un pointeur nul. La fonction de callback doit accepter deux arguments: l'object fonction dont la valeur a changé, et la chaîne de caractère décrivant la nouvelle valeur publiée.

**files→remove()**

Efface un fichier, spécifié par son path complet, du système de fichier.

js	function <b>remove</b> ( <b>pathname</b> )
cpp	int <b>remove</b> ( string <b>pathname</b> )
m	-(int) <b>remove</b> : (NSString*) <b>pathname</b>
pas	LongInt <b>remove</b> ( <b>pathname</b> : string): LongInt
vb	function <b>remove</b> ( ByVal <b>pathname</b> As String) As Integer
cs	int <b>remove</b> ( string <b>pathname</b> )
java	int <b>remove</b> ( String <b>pathname</b> )
uwp	async Task<int> <b>remove</b> ( string <b>pathname</b> )
py	<b>remove</b> ( <b>pathname</b> )
php	function <b>remove</b> ( <b>\$pathname</b> )
ts	async <b>remove</b> ( <b>pathname</b> : string): Promise<number>
es	async <b>remove</b> ( <b>pathname</b> )
dnp	int <b>remove</b> ( string <b>pathname</b> )
cp	int <b>remove</b> ( string <b>pathname</b> )
cmd	YFiles <b>target remove</b> <b>pathname</b>

A cause de la fragmentation, l'effacement d'un fichier ne libère pas toujours la totalité de l'espace qu'il occupe. Par contre, la ré-écriture d'un fichier du même nom récupérera dans tout les cas l'espace qui n'aurait éventuellement pas été libéré. Pour s'assurer de libérer la totalité de l'espace du système de fichier, utilisez la fonction `format_fs`.

**Paramètres :**

**pathname** nom complet du fichier, y compris le chemin d'accès.

**Retourne :**

YAPI . SUCCESS si l'opération se déroule sans erreur.

En cas d'erreur, déclenche une exception ou retourne un code d'erreur négatif.

**files**→**set\_logicalName()****YFiles****files**→**setLogicalName()**

Modifie le nom logique du système de fichier.

js	function <b>set_logicalName</b> ( <b>newval</b> )
cpp	int <b>set_logicalName</b> ( string <b>newval</b> )
m	-(int) setLogicalName : (NSString*) <b>newval</b>
pas	integer <b>set_logicalName</b> ( <b>newval</b> : string): integer
vb	function <b>set_logicalName</b> ( ByVal <b>newval</b> As String) As Integer
cs	int <b>set_logicalName</b> ( string <b>newval</b> )
java	int <b>set_logicalName</b> ( String <b>newval</b> )
uwp	async Task<int> <b>set_logicalName</b> ( string <b>newval</b> )
py	<b>set_logicalName</b> ( <b>newval</b> )
php	function <b>set_logicalName</b> ( \$ <b>newval</b> )
ts	async <b>set_logicalName</b> ( <b>newval</b> : string): Promise<number>
es	async <b>set_logicalName</b> ( <b>newval</b> )
dnp	int <b>set_logicalName</b> ( string <b>newval</b> )
cp	int <b>set_logicalName</b> ( string <b>newval</b> )
cmd	YFiles <b>target set_logicalName newval</b>

Vous pouvez utiliser `yCheckLogicalName()` pour vérifier si votre paramètre est valide. N'oubliez pas d'appeler la méthode `saveToFlash()` du module si le réglage doit être préservé.

**Paramètres :**

**newval** une chaîne de caractères représentant le nom logique du système de fichier.

**Retourne :**

YAPI . SUCCESS si l'appel se déroule sans erreur.

En cas d'erreur, déclenche une exception ou retourne un code d'erreur négatif.

**files**→**set\_userdata()****files**→**setUserData()**

Enregistre un contexte libre dans l'attribut `userData` de la fonction, afin de le retrouver plus tard à l'aide de la méthode `get_userdata`.

js	function <b>set_userdata</b> ( <b>data</b> )
cpp	void <b>set_userdata</b> ( void * <b>data</b> )
m	-(void) <b>setUserData</b> : (id) <b>data</b>
pas	<b>set_userdata</b> ( <b>data</b> : Tobject)
vb	procedure <b>set_userdata</b> ( ByVal <b>data</b> As Object)
cs	void <b>set_userdata</b> ( object <b>data</b> )
java	void <b>set_userdata</b> ( Object <b>data</b> )
py	<b>set_userdata</b> ( <b>data</b> )
php	function <b>set_userdata</b> ( <b>\$data</b> )
ts	async <b>set_userdata</b> ( <b>data</b> : object null): Promise<void>
es	async <b>set_userdata</b> ( <b>data</b> )

Cet attribut n'est pas utilisé directement par l'API. Il est à la disposition de l'appelant pour stocker un contexte.

**Paramètres :**

**data** objet quelconque à mémoriser

**files**→**unmuteValueCallbacks()****YFiles**

Réactive l'envoi de chaque changement de la valeur publiée au hub parent.

js	function <b>unmuteValueCallbacks</b> ( )
cpp	int <b>unmuteValueCallbacks</b> ( )
m	-(int) <b>unmuteValueCallbacks</b>
pas	LongInt <b>unmuteValueCallbacks</b> ( ): LongInt
vb	function <b>unmuteValueCallbacks</b> ( ) As Integer
cs	int <b>unmuteValueCallbacks</b> ( )
java	int <b>unmuteValueCallbacks</b> ( )
uwp	async Task<int> <b>unmuteValueCallbacks</b> ( )
py	<b>unmuteValueCallbacks</b> ( )
php	function <b>unmuteValueCallbacks</b> ( )
ts	async <b>unmuteValueCallbacks</b> ( ): Promise<number>
es	async <b>unmuteValueCallbacks</b> ( )
dnp	int <b>unmuteValueCallbacks</b> ( )
cp	int <b>unmuteValueCallbacks</b> ( )
cmd	YFiles <b>target unmuteValueCallbacks</b>

Cette fonction annule un précédent appel à `muteValueCallbacks()`. N'oubliez pas d'appeler la méthode `saveToFlash()` du module si le réglage doit être préservé.

**Retourne :**

YAPI . SUCCESS si l'opération se déroule sans erreur.

En cas d'erreur, déclenche une exception ou retourne un code d'erreur négatif.

## files→upload()

Télécharge un contenu vers le système de fichier, au chemin d'accès spécifié.

js	function <b>upload</b> ( <b>pathname</b> , <b>content</b> )
cpp	int <b>upload</b> ( string <b>pathname</b> , string <b>content</b> )
m	-(int) <b>upload</b> : (NSString*) <b>pathname</b> : (NSData*) <b>content</b>
pas	LongInt <b>upload</b> ( <b>pathname</b> : string, <b>content</b> : TByteArray): LongInt
vb	procedure <b>upload</b> ( ByVal <b>pathname</b> As String, ByVal <b>content</b> As Byte())
cs	int <b>upload</b> ( string <b>pathname</b> , byte[] <b>content</b> )
java	int <b>upload</b> ( String <b>pathname</b> , byte[] <b>content</b> )
uwp	async Task<int> <b>upload</b> ( string <b>pathname</b> , byte[] <b>content</b> )
py	<b>upload</b> ( <b>pathname</b> , <b>content</b> )
php	function <b>upload</b> ( <b>\$pathname</b> , <b>\$content</b> )
ts	async <b>upload</b> ( <b>pathname</b> : string, <b>content</b> : Uint8Array): Promise<number>
es	async <b>upload</b> ( <b>pathname</b> , <b>content</b> )
dnp	int <b>upload</b> ( string <b>pathname</b> , byte[] <b>content</b> )
cp	int <b>upload</b> ( string <b>pathname</b> , string <b>content</b> )
cmd	YFiles <b>target upload pathname content</b>

Si un fichier existe déjà pour le même chemin d'accès, son contenu est remplacé.

**Paramètres :**

**pathname** nom complet du fichier, y compris le chemin d'accès.

**content** contenu du fichier à télécharger

**Retourne :**

YAPI . SUCCESS si l'opération se déroule sans erreur.

En cas d'erreur, déclenche une exception ou retourne un code d'erreur négatif.

**files**→**wait\_async()****YFiles**

Attend que toutes les commandes asynchrones en cours d'exécution sur le module soient terminées, et appelle le callback passé en paramètre.

```
js fonction wait_async( callback, context)
```

```
ts wait_async( callback: Function, context: object)
```

```
es wait_async( callback, context)
```

La fonction callback peut donc librement utiliser des fonctions synchrones ou asynchrones, sans risquer de bloquer la machine virtuelle Javascript.

**Paramètres :**

**callback** fonction de callback qui sera appelée dès que toutes les commandes en cours d'exécution sur le module seront terminées. La fonction callback reçoit deux arguments: le contexte fourni par l'appelant et l'objet fonction concerné.

**context** contexte fourni par l'appelant, et qui sera passé tel-qu'il est à la fonction de callback

**Retourne :**

rien du tout.

## 8.5. La classe YRealTimeClock

Interface pour interagir avec les horloges à temps réel, disponibles par exemple dans le YoctoHub-GSM-3G-EU, le YoctoHub-GSM-3G-NA, le YoctoHub-GSM-4G et le YoctoHub-Wireless-n

La classe `YRealTimeClock` permet d'accéder à l'horloge embarquée sur certains modules Yoctopuce. Elle fournit la date et l'heure courante de manière persistante, même en cas de coupure de courant de plusieurs jours. Elle est le fondement des fonctions de réveil automatique implémentées par le `WakeUpScheduler`. L'heure courante peut représenter aussi bien une heure locale qu'une heure UTC, mais aucune adaptation automatique n'est fait au changement d'heure été/hiver.

Pour utiliser les fonctions décrites ici, vous devez inclure:

es	in HTML: <code>&lt;script src="../../lib/yocto_realtimedlock.js"&gt;&lt;/script&gt;</code> in node.js: <code>require('yoctolib-es2017/yocto_realtimedlock.js');</code>
js	<code>&lt;script type='text/javascript' src='yocto_realtimedlock.js'&gt;&lt;/script&gt;</code>
cpp	<code>#include "yocto_realtimedlock.h"</code>
m	<code>#import "yocto_realtimedlock.h"</code>
pas	<code>uses yocto_realtimedlock;</code>
vb	<code>yocto_realtimedlock.vb</code>
cs	<code>yocto_realtimedlock.cs</code>
java	<code>import com.yoctopuce.YoctoAPI.YRealTimeClock;</code>
uwp	<code>import com.yoctopuce.YoctoAPI.YRealTimeClock;</code>
py	<code>from yocto_realtimedlock import *</code>
php	<code>require_once('yocto_realtimedlock.php');</code>
ts	in HTML: <code>import { YRealTimeClock } from '../../dist/esm/yocto_realtimedlock.js';</code> in Node.js: <code>import { YRealTimeClock } from 'yoctolib-cjs/yocto_realtimedlock.js';</code>
dnp	<code>import YoctoProxyAPI.YRealTimeClockProxy</code>
cp	<code>#include "yocto_realtimedlock_proxy.h"</code>
vi	<code>YRealTimeClock.vi</code>
ml	<code>import YoctoProxyAPI.YRealTimeClockProxy</code>

### Fonction globales

#### **YRealTimeClock.FindRealTimeClock(func)**

Permet de retrouver une horloge à temps réel d'après un identifiant donné.

#### **YRealTimeClock.FindRealTimeClockInContext(yctx, func)**

Permet de retrouver une horloge à temps réel d'après un identifiant donné dans un Context YAPI.

#### **YRealTimeClock.FirstRealTimeClock()**

Commence l'énumération des horloges à temps réel accessibles par la librairie.

#### **YRealTimeClock.FirstRealTimeClockInContext(yctx)**

Commence l'énumération des horloges à temps réel accessibles par la librairie.

#### **YRealTimeClock.GetSimilarFunctions()**

Enumère toutes les fonctions de type `RealTimeClock` disponibles sur les modules actuellement joignables par la librairie, et retourne leurs identifiants matériels uniques (`hardwareId`).

### Propriétés des objets `YRealTimeClockProxy`

#### **realtimedlock→AdvertisedValue** [lecture seule]

Courte chaîne de caractères représentant l'état courant de la fonction.

#### **realtimedlock→FriendlyName** [lecture seule]

Identifiant global de la fonction au format `NOM_MODULE . NOM_FONCTION`.

#### **realtimedlock→FunctionId** [lecture seule]

Identifiant matériel de l'horloge à temps réel, sans référence au module.

**realtimeclock**→**HardwareId** [*lecture seule*]

Identifiant matériel unique de la fonction au format SERIAL . FUNCTIONID.

**realtimeclock**→**IsOnline** [*lecture seule*]

Vérifie si le module hébergeant la fonction est joignable, sans déclencher d'erreur.

**realtimeclock**→**LogicalName** [*modifiable*]

Nom logique de la fonction.

**realtimeclock**→**SerialNumber** [*lecture seule*]

Numéro de série du module, préprogrammé en usine.

**realtimeclock**→**UtcOffset** [*modifiable*]

Nombre de secondes de décalage entre l'heure courante et l'heure UTC (time zone).

### Méthodes des objets YRealTimeClock

**realtimeclock**→**clearCache()**

Invalide le cache.

**realtimeclock**→**describe()**

Retourne un court texte décrivant de manière non-ambigüe l'instance de l'horloge à temps réel au format TYPE ( NAME ) =SERIAL . FUNCTIONID.

**realtimeclock**→**get\_advertisedValue()**

Retourne la valeur courante de l'horloge à temps réel (pas plus de 6 caractères).

**realtimeclock**→**get\_dateTime()**

Retourne l'heure courante au format "AAAA/MM/JJ hh:mm:ss".

**realtimeclock**→**get\_errorMessage()**

Retourne le message correspondant à la dernière erreur survenue lors de l'utilisation de l'horloge à temps réel.

**realtimeclock**→**get\_errorType()**

Retourne le code d'erreur correspondant à la dernière erreur survenue lors de l'utilisation de l'horloge à temps réel.

**realtimeclock**→**get\_friendlyName()**

Retourne un identifiant global de l'horloge à temps réel au format NOM\_MODULE . NOM\_FONCTION.

**realtimeclock**→**get\_functionDescriptor()**

Retourne un identifiant unique de type YFUN\_DESCR correspondant à la fonction.

**realtimeclock**→**get\_functionId()**

Retourne l'identifiant matériel de l'horloge à temps réel, sans référence au module.

**realtimeclock**→**get\_hardwareId()**

Retourne l'identifiant matériel unique de l'horloge à temps réel au format SERIAL . FUNCTIONID.

**realtimeclock**→**get\_logicalName()**

Retourne le nom logique de l'horloge à temps réel.

**realtimeclock**→**get\_module()**

Retourne l'objet YModule correspondant au module Yoctopuce qui héberge la fonction.

**realtimeclock**→**get\_module\_async(callback, context)**

Retourne l'objet YModule correspondant au module Yoctopuce qui héberge la fonction.

**realtimeclock**→**get\_serialNumber()**

Retourne le numéro de série du module, préprogrammé en usine.

**realtimeclock**→**get\_timeSet()**

Retourne vrai si l'horloge à été mise à l'heure, sinon faux.

**realtimeclock→get\_unixTime()**

Retourne l'heure courante au format Unix (nombre de secondes secondes écoulées depuis le 1er janvier 1970).

**realtimeclock→get\_userData()**

Retourne le contenu de l'attribut userData, précédemment stocké à l'aide de la méthode set\_userData.

**realtimeclock→get\_utcOffset()**

Retourne le nombre de secondes de décalage entre l'heure courante et l'heure UTC (time zone).

**realtimeclock→isOnline()**

Vérifie si le module hébergeant l'horloge à temps réel est joignable, sans déclencher d'erreur.

**realtimeclock→isOnline\_async(callback, context)**

Vérifie si le module hébergeant l'horloge à temps réel est joignable, sans déclencher d'erreur.

**realtimeclock→isReadOnly()**

Test si la fonction est en lecture seule.

**realtimeclock→load(msValidity)**

Met en cache les valeurs courantes de l'horloge à temps réel, avec une durée de validité spécifiée.

**realtimeclock→loadAttribute(attrName)**

Retourne la valeur actuelle d'un attribut spécifique de la fonction, sous forme de texte, le plus rapidement possible mais sans passer par le cache.

**realtimeclock→load\_async(msValidity, callback, context)**

Met en cache les valeurs courantes de l'horloge à temps réel, avec une durée de validité spécifiée.

**realtimeclock→muteValueCallbacks()**

Désactive l'envoi de chaque changement de la valeur publiée au hub parent.

**realtimeclock→nextRealTimeClock()**

Continue l'énumération des horloges à temps réel commencée à l'aide de yFirstRealTimeClock( )  
Attention, vous ne pouvez faire aucune supposition sur l'ordre dans lequel les horloges à temps réel sont retournés.

**realtimeclock→registerValueCallback(callback)**

Enregistre la fonction de callback qui est appelée à chaque changement de la valeur publiée.

**realtimeclock→set\_logicalName(newval)**

Modifie le nom logique de l'horloge à temps réel.

**realtimeclock→set\_unixTime(newval)**

Modifie l'heure courante.

**realtimeclock→set\_userData(data)**

Enregistre un contexte libre dans l'attribut userData de la fonction, afin de le retrouver plus tard à l'aide de la méthode get\_userData.

**realtimeclock→set\_utcOffset(newval)**

Modifie le nombre de secondes de décalage entre l'heure courante et l'heure UTC (time zone).

**realtimeclock→unmuteValueCallbacks()**

Réactive l'envoi de chaque changement de la valeur publiée au hub parent.

**realtimeclock→wait\_async(callback, context)**

Attend que toutes les commandes asynchrones en cours d'exécution sur le module soient terminées, et appelle le callback passé en paramètre.

## YRealTimeClock.FindRealTimeClock() YRealTimeClock.FindRealTimeClock()

## YRealTimeClock

Permet de retrouver une horloge à temps réel d'après un identifiant donné.

js	function <b>yFindRealTimeClock</b> ( <b>func</b> )
cpp	YRealTimeClock* <b>FindRealTimeClock</b> ( string <b>func</b> )
m	+(YRealTimeClock*) <b>FindRealTimeClock</b> : (NSString*) <b>func</b>
pas	TYRealTimeClock <b>yFindRealTimeClock</b> ( <b>func</b> : string): TYRealTimeClock
vb	function <b>FindRealTimeClock</b> ( ByVal <b>func</b> As String) As YRealTimeClock
cs	static YRealTimeClock <b>FindRealTimeClock</b> ( string <b>func</b> )
java	static YRealTimeClock <b>FindRealTimeClock</b> ( String <b>func</b> )
uwp	static YRealTimeClock <b>FindRealTimeClock</b> ( string <b>func</b> )
py	<b>FindRealTimeClock</b> ( <b>func</b> )
php	function <b>FindRealTimeClock</b> ( <b>\$func</b> )
ts	static <b>FindRealTimeClock</b> ( <b>func</b> : string): YRealTimeClock
es	static <b>FindRealTimeClock</b> ( <b>func</b> )
dnp	static YRealTimeClockProxy <b>FindRealTimeClock</b> ( string <b>func</b> )
cp	static YRealTimeClockProxy * <b>FindRealTimeClock</b> ( string <b>func</b> )

L'identifiant peut être spécifié sous plusieurs formes:

- NomLogiqueFonction
- NoSerieModule.IdentifiantFonction
- NoSerieModule.NomLogiqueFonction
- NomLogiqueModule.IdentifiantMatériel
- NomLogiqueModule.NomLogiqueFonction

Cette fonction n'exige pas que l'horloge à temps réel soit en ligne au moment où elle est appelée, l'objet retourné sera néanmoins valide. Utiliser la méthode `YRealTimeClock.isOnline()` pour tester si l'horloge à temps réel est utilisable à un moment donné. En cas d'ambiguïté lorsqu'on fait une recherche par nom logique, aucune erreur ne sera notifiée: la première instance trouvée sera renvoyée. La recherche se fait d'abord par nom matériel, puis par nom logique.

Si un appel à la méthode `is_online()` de cet objet renvoie FAUX alors que vous êtes sûr que le module correspondant est bien branché, vérifiez que vous n'avez pas oublié d'appeler `registerHub()` à l'initialisation de de l'application.

### Paramètres :

**func** une chaîne de caractères qui référence l'horloge à temps réel sans ambiguïté, par exemple `YHUBGSM3.realTimeClock`.

### Retourne :

un objet de classe `YRealTimeClock` qui permet ensuite de contrôler l'horloge à temps réel.

## YRealTimeClock.FindRealTimeClockInContext() YRealTimeClock.FindRealTimeClockInContext()

YRealTimeClock

Permet de retrouver une horloge à temps réel d'après un identifiant donné dans un Context YAPI.

```

java static YRealTimeClock FindRealTimeClockInContext( YAPIContext yctx,
                                                       String func)
uwp static YRealTimeClock FindRealTimeClockInContext( YAPIContext yctx,
                                                       string func)
ts static FindRealTimeClockInContext( yctx: YAPIContext, func: string): YRealTimeClock
es static FindRealTimeClockInContext( yctx, func)

```

L'identifiant peut être spécifié sous plusieurs formes:

- NomLogiqueFonction
- NoSerieModule.IdentifiantFonction
- NoSerieModule.NomLogiqueFonction
- NomLogiqueModule.IdentifiantMatériel
- NomLogiqueModule.NomLogiqueFonction

Cette fonction n'exige pas que l'horloge à temps réel soit en ligne au moment où elle est appelée, l'objet retourné sera néanmoins valide. Utiliser la méthode `YRealTimeClock.isOnline()` pour tester si l'horloge à temps réel est utilisable à un moment donné. En cas d'ambiguïté lorsqu'on fait une recherche par nom logique, aucune erreur ne sera notifiée: la première instance trouvée sera renvoyée. La recherche se fait d'abord par nom matériel, puis par nom logique.

### Paramètres :

**yctx** un contexte YAPI

**func** une chaîne de caractères qui référence l'horloge à temps réel sans ambiguïté, par exemple `YHUBGSM3.realTimeClock`.

### Retourne :

un objet de classe `YRealTimeClock` qui permet ensuite de contrôler l'horloge à temps réel.

## YRealTimeClock.FirstRealTimeClock() YRealTimeClock.FirstRealTimeClock()

## YRealTimeClock

Commence l'énumération des horloges à temps réel accessibles par la librairie.

js	function <b>yFirstRealTimeClock</b> ( )
cpp	YRealTimeClock * <b>FirstRealTimeClock</b> ( )
m	+(YRealTimeClock*) <b>FirstRealTimeClock</b>
pas	TYRealTimeClock <b>yFirstRealTimeClock</b> ( ): TYRealTimeClock
vb	function <b>FirstRealTimeClock</b> ( ) As YRealTimeClock
cs	static YRealTimeClock <b>FirstRealTimeClock</b> ( )
java	static YRealTimeClock <b>FirstRealTimeClock</b> ( )
uwp	static YRealTimeClock <b>FirstRealTimeClock</b> ( )
py	<b>FirstRealTimeClock</b> ( )
php	function <b>FirstRealTimeClock</b> ( )
ts	static <b>FirstRealTimeClock</b> ( ): YRealTimeClock   null
es	static <b>FirstRealTimeClock</b> ( )

Utiliser la fonction `YRealTimeClock.nextRealTimeClock()` pour itérer sur les autres horloges à temps réel.

### Retourne :

un pointeur sur un objet `YRealTimeClock`, correspondant à la première horloge à temps réel accessible en ligne, ou `null` si il n'y a pas de horloges à temps réel disponibles.

## YRealTimeClock.FirstRealTimeClockInContext() YRealTimeClock.FirstRealTimeClockInContext()

YRealTimeClock

Commence l'énumération des horloges à temps réel accessibles par la librairie.

```
java static YRealTimeClock FirstRealTimeClockInContext( YAPIContext yctx)
uwp static YRealTimeClock FirstRealTimeClockInContext( YAPIContext yctx)
ts static FirstRealTimeClockInContext( yctx: YAPIContext): YRealTimeClock | null
es static FirstRealTimeClockInContext( yctx)
```

Utiliser la fonction `YRealTimeClock.nextRealTimeClock()` pour itérer sur les autres horloges à temps réel.

### Paramètres :

`yctx` un contexte YAPI.

### Retourne :

un pointeur sur un objet `YRealTimeClock`, correspondant à la première horloge à temps réel accessible en ligne, ou `null` si il n'y a pas de horloges à temps réel disponibles.

## YRealTimeClock.GetSimilarFunctions() YRealTimeClock.GetSimilarFunctions()

## YRealTimeClock

Enumère toutes les fonctions de type RealTimeClock disponibles sur les modules actuellement joignables par la librairie, et retourne leurs identifiants matériels uniques (hardwareId).

```
dnf static new string[] GetSimilarFunctions( )  
cp static vector<string> GetSimilarFunctions( )
```

Chaque chaîne retournée peut être passée en argument à la méthode YRealTimeClock.FindRealTimeClock pour obtenir un objet permettant d'interagir avec le module correspondant.

**Retourne :**

un tableau de chaînes de caractères, contenant les identifiants matériels de chaque fonction disponible trouvée.

**realtimeclock→AdvertisedValue**

**YRealTimeClock**

---

Courte chaîne de caractères représentant l'état courant de la fonction.

`dnv` string **AdvertisedValue**

---

**realtimeclock**→**FriendlyName****YRealTimeClock**

---

Identifiant global de la fonction au format NOM\_MODULE . NOM\_FONCTION.

`dnf` string **FriendlyName**

Le chaîne retournée utilise soit les noms logiques du module et de la fonction si ils sont définis, soit respectivement le numéro de série du module et l'identifiant matériel de la fonction (par exemple: `MyCustomName.relay1`)

**realtimeclock→FunctionId**

**YRealTimeClock**

---

Identifiant matériel de l'horloge à temps réel, sans référence au module.

`dnsp` `string` **FunctionId**

Par exemple `relay1`.

---

**realtimeclock**→**HardwareId****YRealTimeClock**

---

Identifiant matériel unique de la fonction au format `SERIAL.FUNCTIONID`.

`dnf` `string` **HardwareId**

L'identifiant unique est composé du numéro de série du module et de l'identifiant matériel de la fonction (par exemple `RELAYLO1-123456.relay1`).

Vérifie si le module hébergeant la fonction est joignable, sans déclencher d'erreur.

`bool IsOnline`

Si les valeurs des attributs en cache de la fonction sont valides au moment de l'appel, le module est considéré joignable. Cette fonction ne cause en aucun cas d'exception, quelle que soit l'erreur qui pourrait se produire lors de la vérification de joignabilité.

---

**realtimelock**→**LogicalName****YRealTimeClock**

---

Nom logique de la fonction.

`dnf` `string LogicalName`

**Modifiable.** Vous pouvez utiliser `yCheckLogicalName()` pour vérifier si votre paramètre est valide. N'oubliez pas d'appeler la méthode `saveToFlash()` du module si le réglage doit être préservé.

**realtimeclock**→**SerialNumber**

**YRealTimeClock**

---

Numéro de série du module, préprogrammé en usine.

`dnsp` `string` **SerialNumber**

---

**realtimeclock→UtcOffset****YRealTimeClock**

---

Nombre de secondes de décalage entre l'heure courante et l'heure UTC (time zone).

`dnsp` **int UtcOffset**

**Modifiable.** Modifie le nombre de secondes de décalage entre l'heure courante et l'heure UTC (time zone). Le décalage est automatiquement arrondi au quart d'heure le plus proche. N'oubliez pas d'appeler la méthode `saveToFlash()` du module si le réglage doit être préservé.

**realtimeclock**→**clearCache()****YRealTimeClock**

Invalide le cache.

js	function <b>clearCache</b> ( )
cpp	void <b>clearCache</b> ( )
m	-(void) <b>clearCache</b>
pas	<b>clearCache</b> ( )
vb	procedure <b>clearCache</b> ( )
cs	void <b>clearCache</b> ( )
java	void <b>clearCache</b> ( )
py	<b>clearCache</b> ( )
php	function <b>clearCache</b> ( )
ts	async <b>clearCache</b> ( ): Promise<void>
es	async <b>clearCache</b> ( )

Invalide le cache des valeurs courantes de l'horloge à temps réel. Force le prochain appel à une méthode `get_xxx()` ou `loadxxx()` pour charger les les données depuis le module.

**realtimeclock→describe()****YRealTimeClock**

Retourne un court texte décrivant de manière non-ambigüe l'instance de l'horloge à temps réel au format `TYPE (NAME) =SERIAL . FUNCTIONID`.

js	function <b>describe</b> ( )
cpp	string <b>describe</b> ( )
m	-(NSString*) <b>describe</b>
pas	string <b>describe</b> ( ): string
vb	function <b>describe</b> ( ) As String
cs	string <b>describe</b> ( )
java	String <b>describe</b> ( )
py	<b>describe</b> ( )
php	function <b>describe</b> ( )
ts	async <b>describe</b> ( ): Promise<string>
es	async <b>describe</b> ( )

Plus précisément, `TYPE` correspond au type de fonction, `NAME` correspond au nom utilisé lors du premier accès à la fonction, `SERIAL` correspond au numéro de série du module si le module est connecté, ou "unresolved" sinon, et `FUNCTIONID` correspond à l'identifiant matériel de la fonction si le module est connecté. Par exemple, La méthode va retourner `Relay(MyCustomName.relay1)=RELAYLO1-123456.relay1` si le module est déjà connecté ou `Relay(BadCustomName.relay1)=unresolved` si le module n'est pas déjà connecté. Cette méthode ne déclenche aucune transaction USB ou TCP et peut donc être utilisé dans un débogueur.

**Retourne :**

une chaîne de caractères décrivant l'horloge à temps réel (ex:  
`Relay(MyCustomName.relay1)=RELAYLO1-123456.relay1`)

**realtimeclock→get\_advertisedValue()****YRealTimeClock****realtimeclock→advertisedValue()**

Retourne la valeur courante de l'horloge à temps réel (pas plus de 6 caractères).

js	function <b>get_advertisedValue</b> ( )
cpp	string <b>get_advertisedValue</b> ( )
m	-(NSString*) advertisedValue
pas	string <b>get_advertisedValue</b> ( ): string
vb	function <b>get_advertisedValue</b> ( ) As String
cs	string <b>get_advertisedValue</b> ( )
java	String <b>get_advertisedValue</b> ( )
uwp	async Task<string> <b>get_advertisedValue</b> ( )
py	<b>get_advertisedValue</b> ( )
php	function <b>get_advertisedValue</b> ( )
ts	async <b>get_advertisedValue</b> ( ): Promise<string>
es	async <b>get_advertisedValue</b> ( )
dnp	string <b>get_advertisedValue</b> ( )
cp	string <b>get_advertisedValue</b> ( )
cmd	YRealTimeClock <b>target</b> <b>get_advertisedValue</b>

**Retourne :**

une chaîne de caractères représentant la valeur courante de l'horloge à temps réel (pas plus de 6 caractères).

En cas d'erreur, déclenche une exception ou retourne `YRealTimeClock.AVERTISEDVALUE_INVALID`.

**realtimeclock**→**get\_dateTime()****YRealTimeClock****realtimeclock**→**dateTime()**

Retourne l'heure courante au format "AAAA/MM/JJ hh:mm:ss".

js	function <b>get_dateTime</b> ( )
cpp	string <b>get_dateTime</b> ( )
m	-(NSString*) <b>dateTime</b>
pas	string <b>get_dateTime</b> ( ): string
vb	function <b>get_dateTime</b> ( ) As String
cs	string <b>get_dateTime</b> ( )
java	String <b>get_dateTime</b> ( )
uwp	async Task<string> <b>get_dateTime</b> ( )
py	<b>get_dateTime</b> ( )
php	function <b>get_dateTime</b> ( )
ts	async <b>get_dateTime</b> ( ): Promise<string>
es	async <b>get_dateTime</b> ( )
dnp	string <b>get_dateTime</b> ( )
cp	string <b>get_dateTime</b> ( )
cmd	YRealTimeClock <b>target</b> <b>get_dateTime</b>

**Retourne :**

une chaîne de caractères représentant l'heure courante au format "AAAA/MM/JJ hh:mm:ss"

En cas d'erreur, déclenche une exception ou retourne `YRealTimeClock.DATETIME_INVALID`.

**realtimeclock→get\_errorMessage()****YRealTimeClock****realtimeclock→errorMessage()**

Retourne le message correspondant à la dernière erreur survenue lors de l'utilisation de l'horloge à temps réel.

js	function <b>get_errorMessage</b> ( )
cpp	string <b>get_errorMessage</b> ( )
m	-(NSString*) <b>errorMessage</b>
pas	string <b>get_errorMessage</b> ( ): string
vb	function <b>get_errorMessage</b> ( ) As String
cs	string <b>get_errorMessage</b> ( )
java	String <b>get_errorMessage</b> ( )
py	<b>get_errorMessage</b> ( )
php	function <b>get_errorMessage</b> ( )
ts	<b>get_errorMessage</b> ( ): string
es	<b>get_errorMessage</b> ( )

Cette méthode est principalement utile lorsque la librairie Yoctopuce est utilisée en désactivant la gestion des exceptions.

**Retourne :**

une chaîne de caractères correspondant au message de la dernière erreur qui s'est produit lors de l'utilisation de l'horloge à temps réel.

**realtimeclock**→**get\_errorType()****YRealTimeClock****realtimeclock**→**errorType()**

Retourne le code d'erreur correspondant à la dernière erreur survenue lors de l'utilisation de l'horloge à temps réel.

js	function <b>get_errorType</b> ( )
cpp	YRETCODE <b>get_errorType</b> ( )
m	-(YRETCODE) errorType
pas	YRETCODE <b>get_errorType</b> ( ): YRETCODE
vb	function <b>get_errorType</b> ( ) As YRETCODE
cs	YRETCODE <b>get_errorType</b> ( )
java	int <b>get_errorType</b> ( )
py	<b>get_errorType</b> ( )
php	function <b>get_errorType</b> ( )
ts	<b>get_errorType</b> ( ): number
es	<b>get_errorType</b> ( )

Cette méthode est principalement utile lorsque la librairie Yoctopuce est utilisée en désactivant la gestion des exceptions.

**Retourne :**

un nombre correspondant au code de la dernière erreur qui s'est produit lors de l'utilisation de l'horloge à temps réel.

**realtimeclock**→**get\_friendlyName()****YRealTimeClock****realtimeclock**→**friendlyName()**

Retourne un identifiant global de l'horloge à temps réel au format `NOM_MODULE.NOM_FONCTION`.

js	function <b>get_friendlyName</b> ( )
cpp	string <b>get_friendlyName</b> ( )
m	-(NSString*) friendlyName
cs	string <b>get_friendlyName</b> ( )
java	String <b>get_friendlyName</b> ( )
py	<b>get_friendlyName</b> ( )
php	function <b>get_friendlyName</b> ( )
ts	async <b>get_friendlyName</b> ( ): Promise<string>
es	async <b>get_friendlyName</b> ( )
dnp	string <b>get_friendlyName</b> ( )
cp	string <b>get_friendlyName</b> ( )

Le chaîne retournée utilise soit les noms logiques du module et de l'horloge à temps réel si ils sont définis, soit respectivement le numéro de série du module et l'identifiant matériel de l'horloge à temps réel (par exemple: `MyCustomName.relay1`)

**Retourne :**

une chaîne de caractères identifiant l'horloge à temps réel en utilisant les noms logiques (ex: `MyCustomName.relay1`)

En cas d'erreur, déclenche une exception ou retourne `YRealTimeClock.FRIENDLYNAME_INVALID`.

**realtimeclock**→**get\_functionDescriptor()****YRealTimeClock****realtimeclock**→**functionDescriptor()**

Retourne un identifiant unique de type YFUN\_DESCR correspondant à la fonction.

js	function <b>get_functionDescriptor</b> ( )
cpp	YFUN_DESCR <b>get_functionDescriptor</b> ( )
m	-(YFUN_DESCR) <b>functionDescriptor</b>
pas	YFUN_DESCR <b>get_functionDescriptor</b> ( ): YFUN_DESCR
vb	function <b>get_functionDescriptor</b> ( ) As YFUN_DESCR
cs	YFUN_DESCR <b>get_functionDescriptor</b> ( )
java	String <b>get_functionDescriptor</b> ( )
py	<b>get_functionDescriptor</b> ( )
php	function <b>get_functionDescriptor</b> ( )
ts	async <b>get_functionDescriptor</b> ( ): Promise<string>
es	async <b>get_functionDescriptor</b> ( )

Cet identifiant peut être utilisé pour tester si deux instance de YFunction référencent physiquement la même fonction sur le même module.

**Retourne :**

un identifiant de type YFUN\_DESCR.

Si la fonction n'a jamais été contactée, la valeur retournée sera Y\$CLASSNAME\$.FUNCTIONDESCRIPTOR\_INVALID

**realtimeclock→get\_functionId()****YRealTimeClock****realtimeclock→functionId()**

Retourne l'identifiant matériel de l'horloge à temps réel, sans référence au module.

js	function <b>get_functionId</b> ( )
cpp	string <b>get_functionId</b> ( )
m	-(NSString*) <b>functionId</b>
vb	function <b>get_functionId</b> ( ) As String
cs	string <b>get_functionId</b> ( )
java	String <b>get_functionId</b> ( )
py	<b>get_functionId</b> ( )
php	function <b>get_functionId</b> ( )
ts	async <b>get_functionId</b> ( ): Promise<string>
es	async <b>get_functionId</b> ( )
dnp	string <b>get_functionId</b> ( )
cp	string <b>get_functionId</b> ( )

Par exemple `relay1`.

**Retourne :**

une chaîne de caractères identifiant l'horloge à temps réel (ex: `relay1`)

En cas d'erreur, déclenche une exception ou retourne `YRealTimeClock.FUNCTIONID_INVALID`.

**realtimeclock**→**get\_hardwareId()****YRealTimeClock****realtimeclock**→**hardwareId()**

Retourne l'identifiant matériel unique de l'horloge à temps réel au format `SERIAL.FUNCTIONID`.

js	function <b>get_hardwareId</b> ( )
cpp	string <b>get_hardwareId</b> ( )
m	-(NSString*) <b>hardwareId</b>
vb	function <b>get_hardwareId</b> ( ) As String
cs	string <b>get_hardwareId</b> ( )
java	String <b>get_hardwareId</b> ( )
py	<b>get_hardwareId</b> ( )
php	function <b>get_hardwareId</b> ( )
ts	async <b>get_hardwareId</b> ( ): Promise<string>
es	async <b>get_hardwareId</b> ( )
dnp	string <b>get_hardwareId</b> ( )
cp	string <b>get_hardwareId</b> ( )

L'identifiant unique est composé du numéro de série du module et de l'identifiant matériel de l'horloge à temps réel (par exemple `RELAYLO1-123456.relay1`).

**Retourne :**

une chaîne de caractères identifiant l'horloge à temps réel (ex: `RELAYLO1-123456.relay1`)

En cas d'erreur, déclenche une exception ou retourne `YRealTimeClock.HARDWAREID_INVALID`.

**realtimeclock**→**get\_logicalName()****YRealTimeClock****realtimeclock**→**logicalName()**

Retourne le nom logique de l'horloge à temps réel.

js	function <b>get_logicalName</b> ( )
cpp	string <b>get_logicalName</b> ( )
m	-(NSString*) logicalName
pas	string <b>get_logicalName</b> ( ): string
vb	function <b>get_logicalName</b> ( ) As String
cs	string <b>get_logicalName</b> ( )
java	String <b>get_logicalName</b> ( )
uwp	async Task<string> <b>get_logicalName</b> ( )
py	<b>get_logicalName</b> ( )
php	function <b>get_logicalName</b> ( )
ts	async <b>get_logicalName</b> ( ): Promise<string>
es	async <b>get_logicalName</b> ( )
dnp	string <b>get_logicalName</b> ( )
cp	string <b>get_logicalName</b> ( )
cmd	YRealTimeClock <b>target</b> <b>get_logicalName</b>

**Retourne :**

une chaîne de caractères représentant le nom logique de l'horloge à temps réel.

En cas d'erreur, déclenche une exception ou retourne `YRealTimeClock.LOGICALNAME_INVALID`.

**realtimeclock**→**get\_module()****YRealTimeClock****realtimeclock**→**module()**

Retourne l'objet `YModule` correspondant au module Yoctopuce qui héberge la fonction.

js	function <b>get_module</b> ( )
cpp	<code>YModule *</code> <b>get_module</b> ( )
m	-( <code>YModule*</code> ) module
pas	<code>TYModule</code> <b>get_module</b> ( ): <code>TYModule</code>
vb	function <b>get_module</b> ( ) As <code>YModule</code>
cs	<code>YModule</code> <b>get_module</b> ( )
java	<code>YModule</code> <b>get_module</b> ( )
py	<b>get_module</b> ( )
php	function <b>get_module</b> ( )
ts	async <b>get_module</b> ( ): <code>Promise&lt;YModule&gt;</code>
es	async <b>get_module</b> ( )
dnp	<code>YModuleProxy</code> <b>get_module</b> ( )
cp	<code>YModuleProxy *</code> <b>get_module</b> ( )

Si la fonction ne peut être trouvée sur aucun module, l'instance de `YModule` retournée ne sera pas joignable.

**Retourne :**

une instance de `YModule`

**realtimeclock**→**get\_module\_async()****YRealTimeClock****realtimeclock**→**module\_async()**

Retourne l'objet `YModule` correspondant au module Yoctopuce qui héberge la fonction.

```
js fonction get_module_async( callback, context)
```

Si la fonction ne peut être trouvée sur aucun module, l'instance de `YModule` retournée ne sera pas joignable.

Cette version asynchrone n'existe qu'en Javascript. Elle utilise une fonction de callback plutôt qu'une simple valeur de retour, pour éviter de bloquer la VM Javascript de Firefox, qui n'implémente pas le passage de contrôle entre threads durant les appels d'entrée/sortie bloquants.

**Paramètres :**

**callback** fonction de callback qui sera appelée dès que le résultat sera connu. La fonction callback reçoit trois arguments: le contexte fourni par l'appelant, l'objet fonction concerné et l'instance demandée de `YModule`

**context** contexte fourni par l'appelant, et qui sera passé tel-quel à la fonction de callback

**Retourne :**

rien du tout : le résultat sera passé en paramètre à la fonction de callback.

**realtimeclock**→**get\_serialNumber()****YRealTimeClock****realtimeclock**→**serialNumber()**

Retourne le numéro de série du module, préprogrammé en usine.

js	function <b>get_serialNumber</b> ( )
cpp	string <b>get_serialNumber</b> ( )
m	-(NSString*) serialNumber
pas	string <b>get_serialNumber</b> ( ): string
vb	function <b>get_serialNumber</b> ( ) As String
cs	string <b>get_serialNumber</b> ( )
java	String <b>get_serialNumber</b> ( )
uwp	async Task<string> <b>get_serialNumber</b> ( )
py	<b>get_serialNumber</b> ( )
php	function <b>get_serialNumber</b> ( )
ts	async <b>get_serialNumber</b> ( ): Promise<string>
es	async <b>get_serialNumber</b> ( )
dnp	string <b>get_serialNumber</b> ( )
cp	string <b>get_serialNumber</b> ( )
cmd	YRealTimeClock <b>target</b> <b>get_serialNumber</b>

**Retourne :**

: une chaîne de caractères représentant le numéro de série du module, préprogrammé en usine.

En cas d'erreur, déclenche une exception ou retourne YFunction.SERIALNUMBER\_INVALID.

**realtimeclock**→**get\_timeSet()****YRealTimeClock****realtimeclock**→**timeSet()**

Retourne vrai si l'horloge à été mise à l'heure, sinon faux.

js	function <b>get_timeSet</b> ( )
cpp	Y_TIMESET_enum <b>get_timeSet</b> ( )
m	-(Y_TIMESET_enum) timeSet
pas	Integer <b>get_timeSet</b> ( ): Integer
vb	function <b>get_timeSet</b> ( ) As Integer
cs	int <b>get_timeSet</b> ( )
java	int <b>get_timeSet</b> ( )
uwp	async Task<int> <b>get_timeSet</b> ( )
py	<b>get_timeSet</b> ( )
php	function <b>get_timeSet</b> ( )
ts	async <b>get_timeSet</b> ( ): Promise<YRealTimeClock_TimeSet>
es	async <b>get_timeSet</b> ( )
dnp	int <b>get_timeSet</b> ( )
cp	int <b>get_timeSet</b> ( )
cmd	YRealTimeClock <b>target</b> <b>get_timeSet</b>

**Retourne :**

soit `YRealTimeClock.TIMESET_FALSE`, soit `YRealTimeClock.TIMESET_TRUE`, selon vrai si l'horloge à été mise à l'heure, sinon faux

En cas d'erreur, déclenche une exception ou retourne `YRealTimeClock.TIMESET_INVALID`.

**realtimeclock**→**get\_unixTime()****YRealTimeClock****realtimeclock**→**unixTime()**

Retourne l'heure courante au format Unix (nombre de seconds secondes écoulées depuis le 1er janvier 1970).

js	function <b>get_unixTime</b> ( )
cpp	s64 <b>get_unixTime</b> ( )
m	-(s64) unixTime
pas	int64 <b>get_unixTime</b> ( ): int64
vb	function <b>get_unixTime</b> ( ) As Long
cs	long <b>get_unixTime</b> ( )
java	long <b>get_unixTime</b> ( )
uwp	async Task<long> <b>get_unixTime</b> ( )
py	<b>get_unixTime</b> ( )
php	function <b>get_unixTime</b> ( )
ts	async <b>get_unixTime</b> ( ): Promise<number>
es	async <b>get_unixTime</b> ( )
dnp	long <b>get_unixTime</b> ( )
cp	s64 <b>get_unixTime</b> ( )
cmd	YRealTimeClock <b>target</b> <b>get_unixTime</b>

**Retourne :**

un entier représentant l'heure courante au format Unix (nombre de seconds secondes écoulées depuis le 1er janvier 1970)

En cas d'erreur, déclenche une exception ou retourne `YRealTimeClock.UNIXTIME_INVALID`.

**realtimeclock→get\_userdata()****YRealTimeClock****realtimeclock→userdata()**

Retourne le contenu de l'attribut `userData`, précédemment stocké à l'aide de la méthode `set_userdata`.

js	function <b>get_userdata</b> ( )
cpp	void * <b>get_userdata</b> ( )
m	-(id) <code>userData</code>
pas	Tobject <b>get_userdata</b> ( ): Tobject
vb	function <b>get_userdata</b> ( ) As Object
cs	object <b>get_userdata</b> ( )
java	Object <b>get_userdata</b> ( )
py	<b>get_userdata</b> ( )
php	function <b>get_userdata</b> ( )
ts	async <b>get_userdata</b> ( ): Promise<object null>
es	async <b>get_userdata</b> ( )

Cet attribut n'est pas utilisé directement par l'API. Il est à la disposition de l'appelant pour stocker un contexte.

**Retourne :**

l'objet stocké précédemment par l'appelant.

**realtimeclock**→**get\_utcOffset()****YRealTimeClock****realtimeclock**→**utcOffset()**

Retourne le nombre de secondes de décalage entre l'heure courante et l'heure UTC (time zone).

js	function <b>get_utcOffset</b> ( )
cpp	int <b>get_utcOffset</b> ( )
m	-(int) utcOffset
pas	LongInt <b>get_utcOffset</b> ( ): LongInt
vb	function <b>get_utcOffset</b> ( ) As Integer
cs	int <b>get_utcOffset</b> ( )
java	int <b>get_utcOffset</b> ( )
uwp	async Task<int> <b>get_utcOffset</b> ( )
py	<b>get_utcOffset</b> ( )
php	function <b>get_utcOffset</b> ( )
ts	async <b>get_utcOffset</b> ( ): Promise<number>
es	async <b>get_utcOffset</b> ( )
dnp	int <b>get_utcOffset</b> ( )
cp	int <b>get_utcOffset</b> ( )
cmd	YRealTimeClock <b>target</b> <b>get_utcOffset</b>

**Retourne :**

un entier représentant le nombre de secondes de décalage entre l'heure courante et l'heure UTC (time zone)

En cas d'erreur, déclenche une exception ou retourne `YRealTimeClock.UTC_OFFSET_INVALID`.

**realtimeclock→isOnline()****YRealTimeClock**

Vérifie si le module hébergeant l'horloge à temps réel est joignable, sans déclencher d'erreur.

js	function <b>isOnline</b> ( )
cpp	bool <b>isOnline</b> ( )
m	-(BOOL) <b>isOnline</b>
pas	boolean <b>isOnline</b> ( ): boolean
vb	function <b>isOnline</b> ( ) As Boolean
cs	bool <b>isOnline</b> ( )
java	boolean <b>isOnline</b> ( )
py	<b>isOnline</b> ( )
php	function <b>isOnline</b> ( )
ts	async <b>isOnline</b> ( ): Promise<boolean>
es	async <b>isOnline</b> ( )
dnp	bool <b>isOnline</b> ( )
cp	bool <b>isOnline</b> ( )

Si les valeurs des attributs en cache de l'horloge à temps réel sont valides au moment de l'appel, le module est considéré joignable. Cette fonction ne cause en aucun cas d'exception, quelle que soit l'erreur qui pourrait se produire lors de la vérification de joignabilité.

**Retourne :**

`true` si l'horloge à temps réel est joignable, `false` sinon

**realtimeclock→isOnline\_async()****YRealTimeClock**

Vérifie si le module hébergeant l'horloge à temps réel est joignable, sans déclencher d'erreur.

```
js fonction isOnline_async( callback, context)
```

Si les valeurs des attributs en cache de l'horloge à temps réel sont valides au moment de l'appel, le module est considéré joignable. Cette fonction ne cause en aucun cas d'exception, quelle que soit l'erreur qui pourrait se produire lors de la vérification de joignabilité.

Cette version asynchrone n'existe qu'en Javascript. Elle utilise une fonction de callback plutôt qu'une simple valeur de retour, pour éviter de bloquer la machine virtuelle Javascript avec une attente active.

**Paramètres :**

**callback** fonction de callback qui sera appelée dès que le résultat sera connu. La fonction callback reçoit trois arguments: le contexte fourni par l'appelant, l'objet fonction concerné et le résultat booléen

**context** contexte fourni par l'appelant, et qui sera passé tel-quel à la fonction de callback

**Retourne :**

rien du tout : le résultat sera passé en paramètre à la fonction de callback.

## realtimeclock→isReadOnly()

YRealTimeClock

Test si la fonction est en lecture seule.

cpp	bool <b>isReadOnly</b> ( )
m	-(bool) <b>isReadOnly</b>
pas	boolean <b>isReadOnly</b> ( ): boolean
vb	function <b>isReadOnly</b> ( ) As Boolean
cs	bool <b>isReadOnly</b> ( )
java	boolean <b>isReadOnly</b> ( )
uwp	async Task<bool> <b>isReadOnly</b> ( )
py	<b>isReadOnly</b> ( )
php	function <b>isReadOnly</b> ( )
ts	async <b>isReadOnly</b> ( ): Promise<boolean>
es	async <b>isReadOnly</b> ( )
dnp	bool <b>isReadOnly</b> ( )
cp	bool <b>isReadOnly</b> ( )
cmd	YRealTimeClock <b>target isReadOnly</b>

Retourne vrais si la fonction est protégé en ecriture ou que la fontion n'est pas disponible.

**Retourne :**

true si la fonction est protégé en ecriture ou que la fontion n'est pas disponible

**realtimeclock→load()****YRealTimeClock**

Met en cache les valeurs courantes de l'horloge à temps réel, avec une durée de validité spécifiée.

js	function <b>load</b> ( <b>msValidity</b> )
cpp	YRETCODE <b>load</b> ( int <b>msValidity</b> )
m	-(YRETCODE) <b>load</b> : (u64) <b>msValidity</b>
pas	YRETCODE <b>load</b> ( <b>msValidity</b> : u64): YRETCODE
vb	function <b>load</b> ( ByVal <b>msValidity</b> As Long) As YRETCODE
cs	YRETCODE <b>load</b> ( ulong <b>msValidity</b> )
java	int <b>load</b> ( long <b>msValidity</b> )
py	<b>load</b> ( <b>msValidity</b> )
php	function <b>load</b> ( <b>\$msValidity</b> )
ts	async <b>load</b> ( <b>msValidity</b> : number): Promise<number>
es	async <b>load</b> ( <b>msValidity</b> )

Par défaut, lorsqu'on accède à un module, tous les attributs des fonctions du module sont automatiquement mises en cache pour la durée standard (5 ms). Cette méthode peut être utilisée pour marquer occasionnellement les données cachées comme valides pour une plus longue période, par exemple dans le but de réduire le trafic réseau.

**Paramètres :**

**msValidity** un entier correspondant à la durée de validité attribuée aux les paramètres chargés, en millisecondes

**Retourne :**

YAPI . SUCCESS si l'opération se déroule sans erreur.

En cas d'erreur, déclenche une exception ou retourne un code d'erreur négatif.

**realtimeclock→loadAttribute()****YRealTimeClock**

Retourne la valeur actuelle d'un attribut spécifique de la fonction, sous forme de texte, le plus rapidement possible mais sans passer par le cache.

js	function <b>loadAttribute</b> ( <b>attrName</b> )
cpp	string <b>loadAttribute</b> ( string <b>attrName</b> )
m	-(NSString*) <b>loadAttribute</b> : (NSString*) <b>attrName</b>
pas	string <b>loadAttribute</b> ( <b>attrName</b> : string): string
vb	function <b>loadAttribute</b> ( ByVal <b>attrName</b> As String) As String
cs	string <b>loadAttribute</b> ( string <b>attrName</b> )
java	String <b>loadAttribute</b> ( String <b>attrName</b> )
uwp	async Task<string> <b>loadAttribute</b> ( string <b>attrName</b> )
py	<b>loadAttribute</b> ( <b>attrName</b> )
php	function <b>loadAttribute</b> ( <b>\$attrName</b> )
ts	async <b>loadAttribute</b> ( <b>attrName</b> : string): Promise<string>
es	async <b>loadAttribute</b> ( <b>attrName</b> )
dnp	string <b>loadAttribute</b> ( string <b>attrName</b> )
cp	string <b>loadAttribute</b> ( string <b>attrName</b> )

**Paramètres :**

**attrName** le nom de l'attribut désiré

**Retourne :**

une chaîne de caractères représentant la valeur actuelle de l'attribut.

En cas d'erreur, déclenche une exception ou retourne un chaîne vide.

**realtimeclock→load\_async()****YRealTimeClock**

Met en cache les valeurs courantes de l'horloge à temps réel, avec une durée de validité spécifiée.

```
js function load_async( msValidity, callback, context)
```

Par défaut, lorsqu'on accède à un module, tous les attributs des fonctions du module sont automatiquement mises en cache pour la durée standard (5 ms). Cette méthode peut être utilisée pour marquer occasionnellement les données cachées comme valides pour une plus longue période, par exemple dans le but de réduire le trafic réseau.

Cette version asynchrone n'existe qu'en Javascript. Elle utilise une fonction de callback plutôt qu'une simple valeur de retour, pour éviter de bloquer la machine virtuelle Javascript avec une attente active.

**Paramètres :**

- msValidity** un entier correspondant à la durée de validité attribuée aux les paramètres chargés, en millisecondes
- callback** fonction de callback qui sera appelée dès que le résultat sera connu. La fonction callback reçoit trois arguments: le contexte fourni par l'appelant, l'objet fonction concerné et le code d'erreur (ou `YAPI . SUCCESS`)
- context** contexte fourni par l'appelant, et qui sera passé tel-quel à la fonction de callback

**Retourne :**

rien du tout : le résultat sera passé en paramètre à la fonction de callback.

**realtimeclock→muteValueCallbacks()****YRealTimeClock**

Désactive l'envoi de chaque changement de la valeur publiée au hub parent.

js	function <b>muteValueCallbacks</b> ( )
cpp	int <b>muteValueCallbacks</b> ( )
m	-(int) <b>muteValueCallbacks</b>
pas	LongInt <b>muteValueCallbacks</b> ( ): LongInt
vb	function <b>muteValueCallbacks</b> ( ) As Integer
cs	int <b>muteValueCallbacks</b> ( )
java	int <b>muteValueCallbacks</b> ( )
uwp	async Task<int> <b>muteValueCallbacks</b> ( )
py	<b>muteValueCallbacks</b> ( )
php	function <b>muteValueCallbacks</b> ( )
ts	async <b>muteValueCallbacks</b> ( ): Promise<number>
es	async <b>muteValueCallbacks</b> ( )
dnp	int <b>muteValueCallbacks</b> ( )
cp	int <b>muteValueCallbacks</b> ( )
cmd	YRealTimeClock <b>target</b> <b>muteValueCallbacks</b>

Vous pouvez utiliser cette fonction pour économiser la bande passante et le CPU sur les machines de faible puissance, ou pour éviter le déclenchement de callbacks HTTP. N'oubliez pas d'appeler la méthode `saveToFlash()` du module si le réglage doit être préservé.

**Retourne :**

YAPI . SUCCESS si l'opération se déroule sans erreur.

En cas d'erreur, déclenche une exception ou retourne un code d'erreur négatif.

**realtimeclock**→**nextRealTimeClock()****YRealTimeClock**

Continue l'énumération des horloges à temps réel commencée à l'aide de `yFirstRealTimeClock()` Attention, vous ne pouvez faire aucune supposition sur l'ordre dans lequel les horloges à temps réel sont retournés.

js	function <b>nextRealTimeClock()</b>
cpp	YRealTimeClock * <b>nextRealTimeClock()</b>
m	-(nullable YRealTimeClock*) <b>nextRealTimeClock</b>
pas	TYRealTimeClock <b>nextRealTimeClock()</b> : TYRealTimeClock
vb	function <b>nextRealTimeClock()</b> As YRealTimeClock
cs	YRealTimeClock <b>nextRealTimeClock()</b>
java	YRealTimeClock <b>nextRealTimeClock()</b>
uwp	YRealTimeClock <b>nextRealTimeClock()</b>
py	<b>nextRealTimeClock()</b>
php	function <b>nextRealTimeClock()</b>
ts	<b>nextRealTimeClock()</b> : YRealTimeClock   null
es	<b>nextRealTimeClock()</b>

Si vous souhaitez retrouver une horloge à temps réel spécifique, utilisez `RealTimeClock.findRealTimeClock()` avec un `hardwareID` ou un nom logique.

**Retourne :**

un pointeur sur un objet `YRealTimeClock` accessible en ligne, ou `null` lorsque l'énumération est terminée.

**realtimeclock→registerValueCallback()****YRealTimeClock**

Enregistre la fonction de callback qui est appelée à chaque changement de la valeur publiée.

js	function <b>registerValueCallback</b> ( <b>callback</b> )
cpp	int <b>registerValueCallback</b> ( YRealTimeClockValueCallback <b>callback</b> )
m	-(int) <b>registerValueCallback</b> : (YRealTimeClockValueCallback _Nullable) <b>callback</b>
pas	LongInt <b>registerValueCallback</b> ( <b>callback</b> : TYRealTimeClockValueCallback): LongInt
vb	function <b>registerValueCallback</b> ( ByVal <b>callback</b> As YRealTimeClockValueCallback) As Integer
cs	int <b>registerValueCallback</b> ( ValueCallback <b>callback</b> )
java	int <b>registerValueCallback</b> ( UpdateCallback <b>callback</b> )
uwp	async Task<int> <b>registerValueCallback</b> ( ValueCallback <b>callback</b> )
py	<b>registerValueCallback</b> ( <b>callback</b> )
php	function <b>registerValueCallback</b> ( <b>\$callback</b> )
ts	async <b>registerValueCallback</b> ( <b>callback</b> : YRealTimeClockValueCallback   null): Promise<number>
es	async <b>registerValueCallback</b> ( <b>callback</b> )

Ce callback n'est appelé que durant l'exécution de `ySleep` ou `yHandleEvents`. Cela permet à l'appelant de contrôler quand les callback peuvent se produire. Il est important d'appeler l'une de ces deux fonctions périodiquement pour garantir que les callback ne soient pas appelés trop tard. Pour désactiver un callback, il suffit d'appeler cette méthode en lui passant un pointeur nul.

**Paramètres :**

**callback** la fonction de callback à rappeler, ou un pointeur nul. La fonction de callback doit accepter deux arguments: l'object fonction dont la valeur a changé, et la chaîne de caractère décrivant la nouvelle valeur publiée.

**realtimeclock**→**set\_logicalName()****YRealTimeClock****realtimeclock**→**setLogicalName()**

Modifie le nom logique de l'horloge à temps réel.

js	function <b>set_logicalName</b> ( <b>newval</b> )
cpp	int <b>set_logicalName</b> ( string <b>newval</b> )
m	-(int) setLogicalName : (NSString*) <b>newval</b>
pas	integer <b>set_logicalName</b> ( <b>newval</b> : string): integer
vb	function <b>set_logicalName</b> ( ByVal <b>newval</b> As String) As Integer
cs	int <b>set_logicalName</b> ( string <b>newval</b> )
java	int <b>set_logicalName</b> ( String <b>newval</b> )
uwp	async Task<int> <b>set_logicalName</b> ( string <b>newval</b> )
py	<b>set_logicalName</b> ( <b>newval</b> )
php	function <b>set_logicalName</b> ( \$ <b>newval</b> )
ts	async <b>set_logicalName</b> ( <b>newval</b> : string): Promise<number>
es	async <b>set_logicalName</b> ( <b>newval</b> )
dnp	int <b>set_logicalName</b> ( string <b>newval</b> )
cp	int <b>set_logicalName</b> ( string <b>newval</b> )
cmd	YRealTimeClock <b>target</b> <b>set_logicalName</b> <b>newval</b>

Vous pouvez utiliser `yCheckLogicalName()` pour vérifier si votre paramètre est valide. N'oubliez pas d'appeler la méthode `saveToFlash()` du module si le réglage doit être préservé.

**Paramètres :**

**newval** une chaîne de caractères représentant le nom logique de l'horloge à temps réel.

**Retourne :**

YAPI . SUCCESS si l'appel se déroule sans erreur.

En cas d'erreur, déclenche une exception ou retourne un code d'erreur négatif.

**realtimeclock→set\_unixTime()****YRealTimeClock****realtimeclock→setUnixTime()**

Modifie l'heure courante.

js	function <b>set_unixTime</b> ( <b>newval</b> )
cpp	int <b>set_unixTime</b> ( s64 <b>newval</b> )
m	-(int) setUnixTime : (s64) <b>newval</b>
pas	integer <b>set_unixTime</b> ( <b>newval</b> : int64): integer
vb	function <b>set_unixTime</b> ( ByVal <b>newval</b> As Long) As Integer
cs	int <b>set_unixTime</b> ( long <b>newval</b> )
java	int <b>set_unixTime</b> ( long <b>newval</b> )
uwp	async Task<int> <b>set_unixTime</b> ( long <b>newval</b> )
py	<b>set_unixTime</b> ( <b>newval</b> )
php	function <b>set_unixTime</b> ( <b>\$newval</b> )
ts	async <b>set_unixTime</b> ( <b>newval</b> : number): Promise<number>
es	async <b>set_unixTime</b> ( <b>newval</b> )
dnp	int <b>set_unixTime</b> ( long <b>newval</b> )
cp	int <b>set_unixTime</b> ( s64 <b>newval</b> )
cmd	YRealTimeClock <b>target set_unixTime newval</b>

L'heure est passée au format Unix (nombre de seconds secondes écoulées depuis le 1er janvier 1970).

**Paramètres :****newval** un entier représentant l'heure courante**Retourne :**

YAPI . SUCCESS si l'opération se déroule sans erreur.

En cas d'erreur, déclenche une exception ou retourne un code d'erreur négatif.

**realtimeclock**→**set\_userdata()****YRealTimeClock****realtimeclock**→**setUserData()**

Enregistre un contexte libre dans l'attribut `userData` de la fonction, afin de le retrouver plus tard à l'aide de la méthode `get_userdata`.

js	function <b>set_userdata</b> ( <b>data</b> )
cpp	void <b>set_userdata</b> ( void * <b>data</b> )
m	-(void) setUserData : (id) <b>data</b>
pas	<b>set_userdata</b> ( <b>data</b> : Tobject)
vb	procedure <b>set_userdata</b> ( ByVal <b>data</b> As Object)
cs	void <b>set_userdata</b> ( object <b>data</b> )
java	void <b>set_userdata</b> ( Object <b>data</b> )
py	<b>set_userdata</b> ( <b>data</b> )
php	function <b>set_userdata</b> ( <b>\$data</b> )
ts	async <b>set_userdata</b> ( <b>data</b> : object null): Promise<void>
es	async <b>set_userdata</b> ( <b>data</b> )

Cet attribut n'es pas utilisé directement par l'API. Il est à la disposition de l'appelant pour stocker un contexte.

**Paramètres :**

**data** objet quelconque à mémoriser

**realtimeclock→set\_utcOffset()****YRealTimeClock****realtimeclock→setUtcOffset()**

Modifie le nombre de secondes de décalage entre l'heure courante et l'heure UTC (time zone).

js	function <b>set_utcOffset</b> ( <b>newval</b> )
cpp	int <b>set_utcOffset</b> ( int <b>newval</b> )
m	-(int) setUtcOffset : (int) <b>newval</b>
pas	integer <b>set_utcOffset</b> ( <b>newval</b> : LongInt): integer
vb	function <b>set_utcOffset</b> ( ByVal <b>newval</b> As Integer) As Integer
cs	int <b>set_utcOffset</b> ( int <b>newval</b> )
java	int <b>set_utcOffset</b> ( int <b>newval</b> )
uwp	async Task<int> <b>set_utcOffset</b> ( int <b>newval</b> )
py	<b>set_utcOffset</b> ( <b>newval</b> )
php	function <b>set_utcOffset</b> ( <b>\$newval</b> )
ts	async <b>set_utcOffset</b> ( <b>newval</b> : number): Promise<number>
es	async <b>set_utcOffset</b> ( <b>newval</b> )
dnp	int <b>set_utcOffset</b> ( int <b>newval</b> )
cp	int <b>set_utcOffset</b> ( int <b>newval</b> )
cmd	YRealTimeClock <b>target set_utcOffset newval</b>

Le décalage est automatiquement arrondi au quart d'heure le plus proche. N'oubliez pas d'appeler la méthode `saveToFlash()` du module si le réglage doit être préservé.

**Paramètres :**

**newval** un entier représentant le nombre de secondes de décalage entre l'heure courante et l'heure UTC (time zone)

**Retourne :**

YAPI . SUCCESS si l'opération se déroule sans erreur.

En cas d'erreur, déclenche une exception ou retourne un code d'erreur négatif.

**realtimeclock→unmuteValueCallbacks()****YRealTimeClock**

Réactive l'envoi de chaque changement de la valeur publiée au hub parent.

js	function <b>unmuteValueCallbacks</b> ( )
cpp	int <b>unmuteValueCallbacks</b> ( )
m	-(int) <b>unmuteValueCallbacks</b>
pas	LongInt <b>unmuteValueCallbacks</b> ( ): LongInt
vb	function <b>unmuteValueCallbacks</b> ( ) As Integer
cs	int <b>unmuteValueCallbacks</b> ( )
java	int <b>unmuteValueCallbacks</b> ( )
uwp	async Task<int> <b>unmuteValueCallbacks</b> ( )
py	<b>unmuteValueCallbacks</b> ( )
php	function <b>unmuteValueCallbacks</b> ( )
ts	async <b>unmuteValueCallbacks</b> ( ): Promise<number>
es	async <b>unmuteValueCallbacks</b> ( )
dnp	int <b>unmuteValueCallbacks</b> ( )
cp	int <b>unmuteValueCallbacks</b> ( )
cmd	YRealTimeClock <b>target unmuteValueCallbacks</b>

Cette fonction annule un précédent appel à `muteValueCallbacks()`. N'oubliez pas d'appeler la méthode `saveToFlash()` du module si le réglage doit être préservé.

**Retourne :**

YAPI . SUCCESS si l'opération se déroule sans erreur.

En cas d'erreur, déclenche une exception ou retourne un code d'erreur négatif.

**realtimeclock**→**wait\_async()****YRealTimeClock**

Attend que toutes les commandes asynchrones en cours d'exécution sur le module soient terminées, et appelle le callback passé en paramètre.

```
js  fonction wait_async( callback, context)  
ts  wait_async( callback: Function, context: object)  
es  wait_async( callback, context)
```

La fonction callback peut donc librement utiliser des fonctions synchrones ou asynchrones, sans risquer de bloquer la machine virtuelle Javascript.

**Paramètres :**

**callback** fonction de callback qui sera appelée dès que toutes les commandes en cours d'exécution sur le module seront terminées La fonction callback reçoit deux arguments: le contexte fourni par l'appelant et l'objet fonction concerné.

**context** contexte fourni par l'appelant, et qui sera passé tel-quel à la fonction de callback

**Retourne :**

rien du tout.

## 8.6. La classe YWakeUpMonitor

Interface pour interagir avec les moniteurs de réveil, disponibles par exemple dans le YoctoHub-GSM-3G-EU, le YoctoHub-GSM-3G-NA, le YoctoHub-GSM-4G et le YoctoHub-Wireless-n

La classe `YWakeUpMonitor` prend en charge le contrôle global de toutes les sources de réveil possibles ainsi que les mises en sommeil automatiques.

Pour utiliser les fonctions décrites ici, vous devez inclure:

es	in HTML: <code>&lt;script src="../../lib/yocto_wakeupmonitor.js"&gt;&lt;/script&gt;</code> in node.js: <code>require('yoctolib-es2017/yocto_wakeupmonitor.js');</code>
js	<code>&lt;script type='text/javascript' src='yocto_wakeupmonitor.js'&gt;&lt;/script&gt;</code>
cpp	<code>#include "yocto_wakeupmonitor.h"</code>
m	<code>#import "yocto_wakeupmonitor.h"</code>
pas	<code>uses yocto_wakeupmonitor;</code>
vb	<code>yocto_wakeupmonitor.vb</code>
cs	<code>yocto_wakeupmonitor.cs</code>
java	<code>import com.yoctopuce.YoctoAPI.YWakeUpMonitor;</code>
uwp	<code>import com.yoctopuce.YoctoAPI.YWakeUpMonitor;</code>
py	<code>from yocto_wakeupmonitor import *</code>
php	<code>require_once('yocto_wakeupmonitor.php');</code>
ts	in HTML: <code>import { YWakeUpMonitor } from '../../dist/esm/yocto_wakeupmonitor.js';</code> in Node.js: <code>import { YWakeUpMonitor } from 'yoctolib-cjs/yocto_wakeupmonitor.js';</code>
dnp	<code>import YoctoProxyAPI.YWakeUpMonitorProxy</code>
cp	<code>#include "yocto_wakeupmonitor_proxy.h"</code>
vi	<code>YWakeUpMonitor.vi</code>
ml	<code>import YoctoProxyAPI.YWakeUpMonitorProxy</code>

### Fonction globales

#### **YWakeUpMonitor.FindWakeUpMonitor(func)**

Permet de retrouver un moniteur de réveil d'après un identifiant donné.

#### **YWakeUpMonitor.FindWakeUpMonitorInContext(yctx, func)**

Permet de retrouver un moniteur de réveil d'après un identifiant donné dans un Context YAPI.

#### **YWakeUpMonitor.FirstWakeUpMonitor()**

Commence l'énumération des moniteurs de réveil accessibles par la librairie.

#### **YWakeUpMonitor.FirstWakeUpMonitorInContext(yctx)**

Commence l'énumération des moniteurs de réveil accessibles par la librairie.

#### **YWakeUpMonitor.GetSimilarFunctions()**

Enumère toutes les fonctions de type `WakeUpMonitor` disponibles sur les modules actuellement joignables par la librairie, et retourne leurs identifiants matériels uniques (`hardwareId`).

### Propriétés des objets `YWakeUpMonitorProxy`

#### **wakeupmonitor**→**AdvertisedValue** *[lecture seule]*

Courte chaîne de caractères représentant l'état courant de la fonction.

#### **wakeupmonitor**→**FriendlyName** *[lecture seule]*

Identifiant global de la fonction au format `NOM_MODULE . NOM_FONCTION`.

#### **wakeupmonitor**→**FunctionId** *[lecture seule]*

Identifiant matériel du moniteur de réveil, sans référence au module.

#### **wakeupmonitor**→**HardwareId** *[lecture seule]*

Identifiant matériel unique de la fonction au format SERIAL . FUNCTIONID.

**wakeupmonitor**→**IsOnline** [*lecture seule*]

Vérifie si le module hébergeant la fonction est joignable, sans déclencher d'erreur.

**wakeupmonitor**→**LogicalName** [*modifiable*]

Nom logique de la fonction.

**wakeupmonitor**→**NextWakeUp** [*modifiable*]

Prochaine date/heure de réveil agendée (format UNIX).

**wakeupmonitor**→**PowerDuration** [*modifiable*]

Temp d'éveil maximal en secondes avant de retourner en sommeil automatiquement.

**wakeupmonitor**→**SerialNumber** [*lecture seule*]

Numéro de série du module, préprogrammé en usine.

### Méthodes des objets YWakeUpMonitor

**wakeupmonitor**→**clearCache()**

Invalide le cache.

**wakeupmonitor**→**describe()**

Retourne un court texte décrivant de manière non-ambigüe l'instance du moniteur de réveil au format TYPE ( NAME ) =SERIAL . FUNCTIONID.

**wakeupmonitor**→**get\_advertisedValue()**

Retourne la valeur courante du moniteur de réveil (pas plus de 6 caractères).

**wakeupmonitor**→**get\_errorMessage()**

Retourne le message correspondant à la dernière erreur survenue lors de l'utilisation du moniteur de réveil.

**wakeupmonitor**→**get\_errorType()**

Retourne le code d'erreur correspondant à la dernière erreur survenue lors de l'utilisation du moniteur de réveil.

**wakeupmonitor**→**get\_friendlyName()**

Retourne un identifiant global du moniteur de réveil au format NOM\_MODULE . NOM\_FONCTION.

**wakeupmonitor**→**get\_functionDescriptor()**

Retourne un identifiant unique de type YFUN\_DESCR correspondant à la fonction.

**wakeupmonitor**→**get\_functionId()**

Retourne l'identifiant matériel du moniteur de réveil, sans référence au module.

**wakeupmonitor**→**get\_hardwareId()**

Retourne l'identifiant matériel unique du moniteur de réveil au format SERIAL . FUNCTIONID.

**wakeupmonitor**→**get\_logicalName()**

Retourne le nom logique du moniteur de réveil.

**wakeupmonitor**→**get\_module()**

Retourne l'objet YModule correspondant au module Yoctopuce qui héberge la fonction.

**wakeupmonitor**→**get\_module\_async(callback, context)**

Retourne l'objet YModule correspondant au module Yoctopuce qui héberge la fonction.

**wakeupmonitor**→**get\_nextWakeUp()**

Retourne la prochaine date/heure de réveil agendée (format UNIX).

**wakeupmonitor**→**get\_powerDuration()**

Retourne le temp d'éveil maximal en secondes avant de retourner en sommeil automatiquement.

**wakeupmonitor**→**get\_serialNumber()**

Retourne le numéro de série du module, préprogrammé en usine.

**wakeupmonitor**→**get\_sleepCountdown()**

Retourne le temps avant le prochain sommeil.

**wakeupmonitor**→**get\_userData()**

Retourne le contenu de l'attribut userData, précédemment stocké à l'aide de la méthode set\_userData.

**wakeupmonitor**→**get\_wakeUpReason()**

Renvoie la raison du dernier réveil.

**wakeupmonitor**→**get\_wakeUpState()**

Revoie l'état actuel du moniteur.

**wakeupmonitor**→**isOnline()**

Vérifie si le module hébergeant le moniteur de réveil est joignable, sans déclencher d'erreur.

**wakeupmonitor**→**isOnline\_async(callback, context)**

Vérifie si le module hébergeant le moniteur de réveil est joignable, sans déclencher d'erreur.

**wakeupmonitor**→**isReadOnly()**

Test si la fonction est en lecture seule.

**wakeupmonitor**→**load(msValidity)**

Met en cache les valeurs courantes du moniteur de réveil, avec une durée de validité spécifiée.

**wakeupmonitor**→**loadAttribute(attrName)**

Retourne la valeur actuelle d'un attribut spécifique de la fonction, sous forme de texte, le plus rapidement possible mais sans passer par le cache.

**wakeupmonitor**→**load\_async(msValidity, callback, context)**

Met en cache les valeurs courantes du moniteur de réveil, avec une durée de validité spécifiée.

**wakeupmonitor**→**muteValueCallbacks()**

Désactive l'envoi de chaque changement de la valeur publiée au hub parent.

**wakeupmonitor**→**nextWakeUpMonitor()**

Continue l'énumération des moniteurs de réveil commencée à l'aide de yFirstWakeUpMonitor(). Attention, vous ne pouvez faire aucune supposition sur l'ordre dans lequel les moniteurs de réveil sont retournés.

**wakeupmonitor**→**registerValueCallback(callback)**

Enregistre la fonction de callback qui est appelée à chaque changement de la valeur publiée.

**wakeupmonitor**→**resetSleepCountDown()**

Réinitialise le compteur de mise en sommeil.

**wakeupmonitor**→**set\_logicalName(newval)**

Modifie le nom logique du moniteur de réveil.

**wakeupmonitor**→**set\_nextWakeUp(newval)**

Modifie les jours de la semaine où un réveil doit avoir lieu.

**wakeupmonitor**→**set\_powerDuration(newval)**

Modifie le temps d'éveil maximal en secondes avant de retourner en sommeil automatiquement.

**wakeupmonitor**→**set\_sleepCountdown(newval)**

Modifie le temps avant le prochain sommeil .

**wakeupmonitor**→**set\_userData(data)**

Enregistre un contexte libre dans l'attribut userData de la fonction, afin de le retrouver plus tard à l'aide de la méthode get\_userData.

**wakeupmonitor**→**sleep(secBeforeSleep)**

Déclenche une mise en sommeil jusqu'à la prochaine condition de réveil, l'heure du RTC du module doit impérativement avoir été réglée au préalable.

**wakeupmonitor**→**sleepFor(secUntilWakeUp, secBeforeSleep)**

Déclenche une mise en sommeil pour un temps donné ou jusqu'à la prochaine condition de réveil, l'heure du RTC du module doit impérativement avoir été réglée au préalable.

**wakeupmonitor**→**sleepUntil(wakeUpTime, secBeforeSleep)**

Déclenche une mise en sommeil jusqu'à une date donnée ou jusqu'à la prochaine condition de réveil, l'heure du RTC du module doit impérativement avoir été réglée au préalable.

**wakeupmonitor**→**unmuteValueCallbacks()**

Réactive l'envoi de chaque changement de la valeur publiée au hub parent.

**wakeupmonitor**→**wait\_async(callback, context)**

Attend que toutes les commandes asynchrones en cours d'exécution sur le module soient terminées, et appelle le callback passé en paramètre.

**wakeupmonitor**→**wakeUp()**

Force un réveil.

## YWakeUpMonitor.FindWakeUpMonitor() YWakeUpMonitor.FindWakeUpMonitor()

## YWakeUpMonitor

Permet de retrouver un moniteur de réveil d'après un identifiant donné.

js	function <b>yFindWakeUpMonitor</b> ( <b>func</b> )
cpp	YWakeUpMonitor* <b>FindWakeUpMonitor</b> ( string <b>func</b> )
m	+(YWakeUpMonitor*) <b>FindWakeUpMonitor</b> : (NSString*) <b>func</b>
pas	TYWakeUpMonitor <b>yFindWakeUpMonitor</b> ( <b>func</b> : string): TYWakeUpMonitor
vb	function <b>FindWakeUpMonitor</b> ( ByVal <b>func</b> As String) As YWakeUpMonitor
cs	static YWakeUpMonitor <b>FindWakeUpMonitor</b> ( string <b>func</b> )
java	static YWakeUpMonitor <b>FindWakeUpMonitor</b> ( String <b>func</b> )
uwp	static YWakeUpMonitor <b>FindWakeUpMonitor</b> ( string <b>func</b> )
py	<b>FindWakeUpMonitor</b> ( <b>func</b> )
php	function <b>FindWakeUpMonitor</b> ( <b>\$func</b> )
ts	static <b>FindWakeUpMonitor</b> ( <b>func</b> : string): YWakeUpMonitor
es	static <b>FindWakeUpMonitor</b> ( <b>func</b> )
dnp	static YWakeUpMonitorProxy <b>FindWakeUpMonitor</b> ( string <b>func</b> )
cp	static YWakeUpMonitorProxy * <b>FindWakeUpMonitor</b> ( string <b>func</b> )

L'identifiant peut être spécifié sous plusieurs formes:

- NomLogiqueFonction
- NoSerieModule.IdentifiantFonction
- NoSerieModule.NomLogiqueFonction
- NomLogiqueModule.IdentifiantMatériel
- NomLogiqueModule.NomLogiqueFonction

Cette fonction n'exige pas que le moniteur de réveil soit en ligne au moment où elle est appelée, l'objet retourné sera néanmoins valide. Utiliser la méthode `YWakeUpMonitor.isOnline()` pour tester si le moniteur de réveil est utilisable à un moment donné. En cas d'ambiguïté lorsqu'on fait une recherche par nom logique, aucune erreur ne sera notifiée: la première instance trouvée sera renvoyée. La recherche se fait d'abord par nom matériel, puis par nom logique.

Si un appel à la méthode `is_online()` de cet objet renvoie FAUX alors que vous êtes sûr que le module correspondant est bien branché, vérifiez que vous n'avez pas oublié d'appeler `registerHub()` à l'initialisation de de l'application.

### Paramètres :

**func** une chaîne de caractères qui référence le moniteur de réveil sans ambiguïté, par exemple `YHUBGSM3.wakeUpMonitor`.

### Retourne :

un objet de classe `YWakeUpMonitor` qui permet ensuite de contrôler le moniteur de réveil.

## YWakeUpMonitor.FindWakeUpMonitorInContext() YWakeUpMonitor.FindWakeUpMonitorInContext()

YWakeUpMonitor

Permet de retrouver un moniteur de réveil d'après un identifiant donné dans un Context YAPI.

```
java static YWakeUpMonitor FindWakeUpMonitorInContext( YAPIContext yctx,  
                                                       String func)
```

```
uwp static YWakeUpMonitor FindWakeUpMonitorInContext( YAPIContext yctx,  
                                                       string func)
```

```
ts static FindWakeUpMonitorInContext( yctx: YAPIContext, func: string): YWakeUpMonitor
```

```
es static FindWakeUpMonitorInContext( yctx, func)
```

L'identifiant peut être spécifié sous plusieurs formes:

- NomLogiqueFonction
- NoSerieModule.IdentifiantFonction
- NoSerieModule.NomLogiqueFonction
- NomLogiqueModule.IdentifiantMatériel
- NomLogiqueModule.NomLogiqueFonction

Cette fonction n'exige pas que le moniteur de réveil soit en ligne au moment où elle est appelée, l'objet retourné sera néanmoins valide. Utiliser la méthode `YWakeUpMonitor.isOnline()` pour tester si le moniteur de réveil est utilisable à un moment donné. En cas d'ambiguïté lorsqu'on fait une recherche par nom logique, aucune erreur ne sera notifiée: la première instance trouvée sera renvoyée. La recherche se fait d'abord par nom matériel, puis par nom logique.

### Paramètres :

**yctx** un contexte YAPI

**func** une chaîne de caractères qui référence le moniteur de réveil sans ambiguïté, par exemple `YHUBGSM3.wakeUpMonitor`.

### Retourne :

un objet de classe `YWakeUpMonitor` qui permet ensuite de contrôler le moniteur de réveil.

## YWakeUpMonitor.FirstWakeUpMonitor() YWakeupMonitor.FirstWakeUpMonitor()

## YWakeupMonitor

Commence l'énumération des moniteurs de réveil accessibles par la librairie.

js	function <b>yFirstWakeUpMonitor</b> ( )
cpp	YWakeupMonitor * <b>FirstWakeUpMonitor</b> ( )
m	+(YWakeupMonitor*) <b>FirstWakeUpMonitor</b>
pas	TYWakeupMonitor <b>yFirstWakeUpMonitor</b> ( ): TYWakeupMonitor
vb	function <b>FirstWakeUpMonitor</b> ( ) As YWakeupMonitor
cs	static YWakeupMonitor <b>FirstWakeUpMonitor</b> ( )
java	static YWakeupMonitor <b>FirstWakeUpMonitor</b> ( )
uwp	static YWakeupMonitor <b>FirstWakeUpMonitor</b> ( )
py	<b>FirstWakeUpMonitor</b> ( )
php	function <b>FirstWakeUpMonitor</b> ( )
ts	static <b>FirstWakeUpMonitor</b> ( ): YWakeupMonitor   null
es	static <b>FirstWakeUpMonitor</b> ( )

Utiliser la fonction `YWakeupMonitor.nextWakeUpMonitor()` pour itérer sur les autres moniteurs de réveil.

**Retourne :**

un pointeur sur un objet `YWakeupMonitor`, correspondant au premier moniteur de réveil accessible en ligne, ou `null` si il n'y a pas de moniteurs de réveil disponibles.

## YWakeUpMonitor.FirstWakeUpMonitorInContext() YWakeupMonitor.FirstWakeUpMonitorInContext()

YWakeupMonitor

Commence l'énumération des moniteurs de réveil accessibles par la librairie.

```
java static YWakeUpMonitor FirstWakeUpMonitorInContext( YAPIContext yctx)
uwp static YWakeUpMonitor FirstWakeUpMonitorInContext( YAPIContext yctx)
ts static FirstWakeUpMonitorInContext( yctx: YAPIContext): YWakeUpMonitor | null
es static FirstWakeUpMonitorInContext( yctx)
```

Utiliser la fonction `YWakeupMonitor.nextWakeUpMonitor()` pour itérer sur les autres moniteurs de réveil.

### Paramètres :

**yctx** un contexte YAPI.

### Retourne :

un pointeur sur un objet `YWakeupMonitor`, correspondant au premier moniteur de réveil accessible en ligne, ou `null` si il n'y a pas de moniteurs de réveil disponibles.

## YWakeUpMonitor.GetSimilarFunctions() YWakeupMonitor.GetSimilarFunctions()

## YWakeupMonitor

Enumère toutes les fonctions de type WakeUpMonitor disponibles sur les modules actuellement joignables par la librairie, et retourne leurs identifiants matériels uniques (hardwareId).

```
dnv static new string[] GetSimilarFunctions( )
```

```
cp static vector<string> GetSimilarFunctions( )
```

Chaque chaîne retournée peut être passée en argument à la méthode `YWakeupMonitor.FindWakeUpMonitor` pour obtenir un objet permettant d'interagir avec le module correspondant.

**Retourne :**

un tableau de chaînes de caractères, contenant les identifiants matériels de chaque fonction disponible trouvée.

## wakeupmonitor→AdvertisedValue

YWakeUpMonitor

---

Courte chaîne de caractères représentant l'état courant de la fonction.

`dnv` string **AdvertisedValue**

---

**wakeupmonitor** → **FriendlyName****YWakeUpMonitor**

---

Identifiant global de la fonction au format `NOM_MODULE . NOM_FONCTION`.

`dnsp` string **FriendlyName**

Le chaîne retournée utilise soit les noms logiques du module et de la fonction si ils sont définis, soit respectivement le numéro de série du module et l'identifiant matériel de la fonction (par exemple: `MyCustomName.relay1`)

## wakeupmonitor → FunctionId

YWakeUpMonitor

---

Identifiant matériel du moniteur de réveil, sans référence au module.

`dnsp` string **FunctionId**

Par exemple `relay1`.

---

**wakeupmonitor** → **HardwareId****YWakeUpMonitor**

---

Identifiant matériel unique de la fonction au format `SERIAL.FUNCTIONID`.

`dnf` `string` **HardwareId**

L'identifiant unique est composé du numéro de série du module et de l'identifiant matériel de la fonction (par exemple `RELAYLO1-123456.relay1`).

Vérifie si le module hébergeant la fonction est joignable, sans déclencher d'erreur.

`bool IsOnline`

Si les valeurs des attributs en cache de la fonction sont valides au moment de l'appel, le module est considéré joignable. Cette fonction ne cause en aucun cas d'exception, quelle que soit l'erreur qui pourrait se produire lors de la vérification de joignabilité.

---

**wakeupmonitor**→**LogicalName****YWakeUpMonitor**

---

Nom logique de la fonction.

`dnf` `string LogicalName`

**Modifiable.** Vous pouvez utiliser `yCheckLogicalName()` pour vérifier si votre paramètre est valide. N'oubliez pas d'appeler la méthode `saveToFlash()` du module si le réglage doit être préservé.

## wakeupmonitor → NextWakeUp

YWakeUpMonitor

---

Prochaine date/heure de réveil agendée (format UNIX).

`dnsp` long [NextWakeUp](#)

**Modifiable.** Modifie les jours de la semaine où un réveil doit avoir lieu.

---

**wakeupmonitor** → **PowerDuration****YWakeUpMonitor**

---

Temp d'éveil maximal en secondes avant de retourner en sommeil automatiquement.

`dnsp` **int PowerDuration**

**Modifiable.** Modifie le temps d'éveil maximal en secondes avant de retourner en sommeil automatiquement. N'oubliez pas d'appeler la méthode `saveToFlash()` du module si le réglage doit être préservé.

**wakeupmonitor** → **SerialNumber**

**YWakeUpMonitor**

---

Numéro de série du module, préprogrammé en usine.

`dnsp` string **SerialNumber**

**wakeupmonitor** → **clearCache()****YWakeUpMonitor**

Invalide le cache.

js	function <b>clearCache</b> ( )
cpp	void <b>clearCache</b> ( )
m	-(void) <b>clearCache</b>
pas	<b>clearCache</b> ( )
vb	procedure <b>clearCache</b> ( )
cs	void <b>clearCache</b> ( )
java	void <b>clearCache</b> ( )
py	<b>clearCache</b> ( )
php	function <b>clearCache</b> ( )
ts	async <b>clearCache</b> ( ): Promise<void>
es	async <b>clearCache</b> ( )

Invalide le cache des valeurs courantes du moniteur de réveil. Force le prochain appel à une méthode `get_xxx()` ou `loadxxx()` pour charger les les données depuis le module.

**wakeupmonitor**→**describe()****YWakeUpMonitor**

Retourne un court texte décrivant de manière non-ambigüe l'instance du moniteur de réveil au format `TYPE (NAME) =SERIAL . FUNCTIONID`.

js	function <b>describe</b> ( )
cpp	string <b>describe</b> ( )
m	-(NSString*) <b>describe</b>
pas	string <b>describe</b> ( ): string
vb	function <b>describe</b> ( ) As String
cs	string <b>describe</b> ( )
java	String <b>describe</b> ( )
py	<b>describe</b> ( )
php	function <b>describe</b> ( )
ts	async <b>describe</b> ( ): Promise<string>
es	async <b>describe</b> ( )

Plus précisément, `TYPE` correspond au type de fonction, `NAME` correspond au nom utilisé lors du premier accès a la fonction, `SERIAL` correspond au numéro de série du module si le module est connecté, ou "unresolved" sinon, et `FUNCTIONID` correspond à l'identifiant matériel de la fonction si le module est connecté. Par exemple, La methode va retourner `Relay(MyCustomName.relay1)=RELAYLO1-123456.relay1` si le module est déjà connecté ou `Relay(BadCustomeName.relay1)=unresolved` si le module n'est pas déjà connecté. Cette methode ne declenche aucune transaction USB ou TCP et peut donc être utilisé dans un debugueur.

**Retourne :**

une chaîne de caractères décrivant le moniteur de réveil (ex:  
`Relay(MyCustomName.relay1)=RELAYLO1-123456.relay1`)

**wakeupmonitor**→**get\_advertisedValue()****YWakeUpMonitor****wakeupmonitor**→**advertisedValue()**

Retourne la valeur courante du moniteur de réveil (pas plus de 6 caractères).

js	function <b>get_advertisedValue</b> ( )
cpp	string <b>get_advertisedValue</b> ( )
m	-(NSString*) <b>advertisedValue</b>
pas	string <b>get_advertisedValue</b> ( ): string
vb	function <b>get_advertisedValue</b> ( ) As String
cs	string <b>get_advertisedValue</b> ( )
java	String <b>get_advertisedValue</b> ( )
uwp	async Task<string> <b>get_advertisedValue</b> ( )
py	<b>get_advertisedValue</b> ( )
php	function <b>get_advertisedValue</b> ( )
ts	async <b>get_advertisedValue</b> ( ): Promise<string>
es	async <b>get_advertisedValue</b> ( )
dnp	string <b>get_advertisedValue</b> ( )
cp	string <b>get_advertisedValue</b> ( )
cmd	YWakeUpMonitor <b>target</b> <b>get_advertisedValue</b>

**Retourne :**

une chaîne de caractères représentant la valeur courante du moniteur de réveil (pas plus de 6 caractères).

En cas d'erreur, déclenche une exception ou retourne `YWakeUpMonitor.AVERTISEDVALUE_INVALID`.

**wakeupmonitor**→**get\_errorMessage()****YWakeUpMonitor****wakeupmonitor**→**errorMessage()**

Retourne le message correspondant à la dernière erreur survenue lors de l'utilisation du moniteur de réveil.

js	function <b>get_errorMessage</b> ( )
cpp	string <b>get_errorMessage</b> ( )
m	-(NSString*) <b>errorMessage</b>
pas	string <b>get_errorMessage</b> ( ): string
vb	function <b>get_errorMessage</b> ( ) As String
cs	string <b>get_errorMessage</b> ( )
java	String <b>get_errorMessage</b> ( )
py	<b>get_errorMessage</b> ( )
php	function <b>get_errorMessage</b> ( )
ts	<b>get_errorMessage</b> ( ): string
es	<b>get_errorMessage</b> ( )

Cette méthode est principalement utile lorsque la librairie Yoctopuce est utilisée en désactivant la gestion des exceptions.

**Retourne :**

une chaîne de caractères correspondant au message de la dernière erreur qui s'est produit lors de l'utilisation du moniteur de réveil.

**wakeupmonitor**→**get\_errorType()****YWakeUpMonitor****wakeupmonitor**→**errorType()**

Retourne le code d'erreur correspondant à la dernière erreur survenue lors de l'utilisation du moniteur de réveil.

js	function <b>get_errorType</b> ( )
cpp	YRETCODE <b>get_errorType</b> ( )
m	-(YRETCODE) errorType
pas	YRETCODE <b>get_errorType</b> ( ): YRETCODE
vb	function <b>get_errorType</b> ( ) As YRETCODE
cs	YRETCODE <b>get_errorType</b> ( )
java	int <b>get_errorType</b> ( )
py	<b>get_errorType</b> ( )
php	function <b>get_errorType</b> ( )
ts	<b>get_errorType</b> ( ): number
es	<b>get_errorType</b> ( )

Cette méthode est principalement utile lorsque la librairie Yoctopuce est utilisée en désactivant la gestion des exceptions.

**Retourne :**

un nombre correspondant au code de la dernière erreur qui s'est produit lors de l'utilisation du moniteur de réveil.

**wakeupmonitor**→**get\_friendlyName()****YWakeUpMonitor****wakeupmonitor**→**friendlyName()**

Retourne un identifiant global du moniteur de réveil au format `NOM_MODULE . NOM_FONCTION`.

js	function <b>get_friendlyName</b> ( )
cpp	string <b>get_friendlyName</b> ( )
m	-(NSString*) friendlyName
cs	string <b>get_friendlyName</b> ( )
java	String <b>get_friendlyName</b> ( )
py	<b>get_friendlyName</b> ( )
php	function <b>get_friendlyName</b> ( )
ts	async <b>get_friendlyName</b> ( ): Promise<string>
es	async <b>get_friendlyName</b> ( )
dnf	string <b>get_friendlyName</b> ( )
cp	string <b>get_friendlyName</b> ( )

Le chaîne retournée utilise soit les noms logiques du module et du moniteur de réveil si ils sont définis, soit respectivement le numéro de série du module et l'identifiant matériel du moniteur de réveil (par exemple: `MyCustomName.relay1`)

**Retourne :**

une chaîne de caractères identifiant le moniteur de réveil en utilisant les noms logiques (ex: `MyCustomName.relay1`)

En cas d'erreur, déclenche une exception ou retourne `YWakeupMonitor.FRIENDLYNAME_INVALID`.

**wakeupmonitor**→**get\_functionDescriptor()****YWakeUpMonitor****wakeupmonitor**→**functionDescriptor()**

Retourne un identifiant unique de type YFUN\_DESCR correspondant à la fonction.

js	function <b>get_functionDescriptor()</b>
cpp	YFUN_DESCR <b>get_functionDescriptor()</b>
m	-(YFUN_DESCR) functionDescriptor
pas	YFUN_DESCR <b>get_functionDescriptor()</b> : YFUN_DESCR
vb	function <b>get_functionDescriptor()</b> As YFUN_DESCR
cs	YFUN_DESCR <b>get_functionDescriptor()</b>
java	String <b>get_functionDescriptor()</b>
py	<b>get_functionDescriptor()</b>
php	function <b>get_functionDescriptor()</b>
ts	async <b>get_functionDescriptor()</b> : Promise<string>
es	async <b>get_functionDescriptor()</b>

Cet identifiant peut être utilisé pour tester si deux instance de YFunction référencent physiquement la même fonction sur le même module.

**Retourne :**

un identifiant de type YFUN\_DESCR.

Si la fonction n'a jamais été contactée, la valeur retournée sera Y\$CLASSNAME\$.FUNCTIONDESCRIPTOR\_INVALID

**wakeupmonitor**→**get\_functionId()****YWakeUpMonitor****wakeupmonitor**→**functionId()**

Retourne l'identifiant matériel du moniteur de réveil, sans référence au module.

js	function <b>get_functionId</b> ( )
cpp	string <b>get_functionId</b> ( )
m	-(NSString*) <b>functionId</b>
vb	function <b>get_functionId</b> ( ) As String
cs	string <b>get_functionId</b> ( )
java	String <b>get_functionId</b> ( )
py	<b>get_functionId</b> ( )
php	function <b>get_functionId</b> ( )
ts	async <b>get_functionId</b> ( ): Promise<string>
es	async <b>get_functionId</b> ( )
dnp	string <b>get_functionId</b> ( )
cp	string <b>get_functionId</b> ( )

Par exemple `relay1`.

**Retourne :**

une chaîne de caractères identifiant le moniteur de réveil (ex: `relay1`)

En cas d'erreur, déclenche une exception ou retourne `YWakeupMonitor.FUNCTIONID_INVALID`.

**wakeupmonitor**→**get\_hardwareId()****YWakeUpMonitor****wakeupmonitor**→**hardwareId()**

Retourne l'identifiant matériel unique du moniteur de réveil au format `SERIAL.FUNCTIONID`.

js	function <b>get_hardwareId</b> ( )
cpp	string <b>get_hardwareId</b> ( )
m	-(NSString*) <b>hardwareId</b>
vb	function <b>get_hardwareId</b> ( ) As String
cs	string <b>get_hardwareId</b> ( )
java	String <b>get_hardwareId</b> ( )
py	<b>get_hardwareId</b> ( )
php	function <b>get_hardwareId</b> ( )
ts	async <b>get_hardwareId</b> ( ): Promise<string>
es	async <b>get_hardwareId</b> ( )
dnp	string <b>get_hardwareId</b> ( )
cp	string <b>get_hardwareId</b> ( )

L'identifiant unique est composé du numéro de série du module et de l'identifiant matériel du moniteur de réveil (par exemple `RELAYLO1-123456.relay1`).

**Retourne :**

une chaîne de caractères identifiant le moniteur de réveil (ex: `RELAYLO1-123456.relay1`)

En cas d'erreur, déclenche une exception ou retourne `YWakeUpMonitor.HARDWAREID_INVALID`.

**wakeupmonitor**→**get\_logicalName()****YWakeUpMonitor****wakeupmonitor**→**logicalName()**

Retourne le nom logique du moniteur de réveil.

js	function <b>get_logicalName</b> ( )
cpp	string <b>get_logicalName</b> ( )
m	-(NSString*) logicalName
pas	string <b>get_logicalName</b> ( ): string
vb	function <b>get_logicalName</b> ( ) As String
cs	string <b>get_logicalName</b> ( )
java	String <b>get_logicalName</b> ( )
uwp	async Task<string> <b>get_logicalName</b> ( )
py	<b>get_logicalName</b> ( )
php	function <b>get_logicalName</b> ( )
ts	async <b>get_logicalName</b> ( ): Promise<string>
es	async <b>get_logicalName</b> ( )
dnp	string <b>get_logicalName</b> ( )
cp	string <b>get_logicalName</b> ( )
cmd	YWakeUpMonitor <b>target</b> <b>get_logicalName</b>

**Retourne :**

une chaîne de caractères représentant le nom logique du moniteur de réveil.

En cas d'erreur, déclenche une exception ou retourne `YWakeUpMonitor.LOGICALNAME_INVALID`.

**wakeupmonitor**→**get\_module()****YWakeUpMonitor****wakeupmonitor**→**module()**

Retourne l'objet `YModule` correspondant au module Yoctopuce qui héberge la fonction.

js	function <b>get_module</b> ( )
cpp	<code>YModule *</code> <b>get_module</b> ( )
m	-( <code>YModule*</code> ) module
pas	<code>TYModule</code> <b>get_module</b> ( ): <code>TYModule</code>
vb	function <b>get_module</b> ( ) As <code>YModule</code>
cs	<code>YModule</code> <b>get_module</b> ( )
java	<code>YModule</code> <b>get_module</b> ( )
py	<b>get_module</b> ( )
php	function <b>get_module</b> ( )
ts	async <b>get_module</b> ( ): <code>Promise&lt;YModule&gt;</code>
es	async <b>get_module</b> ( )
dnp	<code>YModuleProxy</code> <b>get_module</b> ( )
cp	<code>YModuleProxy *</code> <b>get_module</b> ( )

Si la fonction ne peut être trouvée sur aucun module, l'instance de `YModule` retournée ne sera pas joignable.

**Retourne :**

une instance de `YModule`

**wakeupmonitor**→**get\_module\_async()****YWakeUpMonitor****wakeupmonitor**→**module\_async()**

Retourne l'objet `YModule` correspondant au module Yoctopuce qui héberge la fonction.

```
js fonction get_module_async( callback, context)
```

Si la fonction ne peut être trouvée sur aucun module, l'instance de `YModule` retournée ne sera pas joignable.

Cette version asynchrone n'existe qu'en Javascript. Elle utilise une fonction de callback plutôt qu'une simple valeur de retour, pour éviter de bloquer la VM Javascript de Firefox, qui n'implémente pas le passage de contrôle entre threads durant les appels d'entrée/sortie bloquants.

**Paramètres :**

**callback** fonction de callback qui sera appelée dès que le résultat sera connu. La fonction callback reçoit trois arguments: le contexte fourni par l'appelant, l'objet fonction concerné et l'instance demandée de `YModule`

**context** contexte fourni par l'appelant, et qui sera passé tel-quel à la fonction de callback

**Retourne :**

rien du tout : le résultat sera passé en paramètre à la fonction de callback.

**wakeupmonitor**→**get\_nextWakeUp()****YWakeUpMonitor****wakeupmonitor**→**nextWakeUp()**

Retourne la prochaine date/heure de réveil agendée (format UNIX).

js	function <b>get_nextWakeUp</b> ( )
cpp	s64 <b>get_nextWakeUp</b> ( )
m	-(s64) nextWakeUp
pas	int64 <b>get_nextWakeUp</b> ( ): int64
vb	function <b>get_nextWakeUp</b> ( ) As Long
cs	long <b>get_nextWakeUp</b> ( )
java	long <b>get_nextWakeUp</b> ( )
uwp	async Task<long> <b>get_nextWakeUp</b> ( )
py	<b>get_nextWakeUp</b> ( )
php	function <b>get_nextWakeUp</b> ( )
ts	async <b>get_nextWakeUp</b> ( ): Promise<number>
es	async <b>get_nextWakeUp</b> ( )
dnp	long <b>get_nextWakeUp</b> ( )
cp	s64 <b>get_nextWakeUp</b> ( )
cmd	YWakeUpMonitor <b>target</b> <b>get_nextWakeUp</b>

**Retourne :**

un entier représentant la prochaine date/heure de réveil agendée (format UNIX)

En cas d'erreur, déclenche une exception ou retourne `YWakeUpMonitor.NEXTWAKEUP_INVALID`.

wakeupmonitor→get\_powerDuration()

YWakeUpMonitor

wakeupmonitor→powerDuration()

Retourne le temp d'éveil maximal en secondes avant de retourner en sommeil automatiquement.

js	function <b>get_powerDuration</b> ( )
cpp	int <b>get_powerDuration</b> ( )
m	-(int) powerDuration
pas	LongInt <b>get_powerDuration</b> ( ): LongInt
vb	function <b>get_powerDuration</b> ( ) As Integer
cs	int <b>get_powerDuration</b> ( )
java	int <b>get_powerDuration</b> ( )
uwp	async Task<int> <b>get_powerDuration</b> ( )
py	<b>get_powerDuration</b> ( )
php	function <b>get_powerDuration</b> ( )
ts	async <b>get_powerDuration</b> ( ): Promise<number>
es	async <b>get_powerDuration</b> ( )
dnp	int <b>get_powerDuration</b> ( )
cp	int <b>get_powerDuration</b> ( )
cmd	YWakeupMonitor <b>target</b> <b>get_powerDuration</b>

**Retourne :**

un entier représentant le temp d'éveil maximal en secondes avant de retourner en sommeil automatiquement

En cas d'erreur, déclenche une exception ou retourne `YWakeupMonitor.POWERDURATION_INVALID`.

**wakeupmonitor**→**get\_serialNumber()****YWakeUpMonitor****wakeupmonitor**→**serialNumber()**

Retourne le numéro de série du module, préprogrammé en usine.

js	function <b>get_serialNumber</b> ( )
cpp	string <b>get_serialNumber</b> ( )
m	-(NSString*) <b>serialNumber</b>
pas	string <b>get_serialNumber</b> ( ): string
vb	function <b>get_serialNumber</b> ( ) As String
cs	string <b>get_serialNumber</b> ( )
java	String <b>get_serialNumber</b> ( )
uwp	async Task<string> <b>get_serialNumber</b> ( )
py	<b>get_serialNumber</b> ( )
php	function <b>get_serialNumber</b> ( )
ts	async <b>get_serialNumber</b> ( ): Promise<string>
es	async <b>get_serialNumber</b> ( )
dnp	string <b>get_serialNumber</b> ( )
cp	string <b>get_serialNumber</b> ( )
cmd	YWakeUpMonitor <b>target</b> <b>get_serialNumber</b>

**Retourne :**

: une chaîne de caractères représentant le numéro de série du module, préprogrammé en usine.

En cas d'erreur, déclenche une exception ou retourne YFunction.SERIALNUMBER\_INVALID.

**wakeupmonitor**→**get\_sleepCountdown()****YWakeUpMonitor****wakeupmonitor**→**sleepCountdown()**

Retourne le temps avant le prochain sommeil.

js	function <b>get_sleepCountdown</b> ( )
cpp	int <b>get_sleepCountdown</b> ( )
m	-(int) sleepCountdown
pas	LongInt <b>get_sleepCountdown</b> ( ): LongInt
vb	function <b>get_sleepCountdown</b> ( ) As Integer
cs	int <b>get_sleepCountdown</b> ( )
java	int <b>get_sleepCountdown</b> ( )
uwp	async Task<int> <b>get_sleepCountdown</b> ( )
py	<b>get_sleepCountdown</b> ( )
php	function <b>get_sleepCountdown</b> ( )
ts	async <b>get_sleepCountdown</b> ( ): Promise<number>
es	async <b>get_sleepCountdown</b> ( )
dnp	int <b>get_sleepCountdown</b> ( )
cp	int <b>get_sleepCountdown</b> ( )
cmd	YWakeUpMonitor <b>target</b> <b>get_sleepCountdown</b>

**Retourne :**

un entier représentant le temps avant le prochain sommeil

En cas d'erreur, déclenche une exception ou retourne `YWakeUpMonitor.SLEEP_COUNTDOWN_INVALID`.

**wakeupmonitor**→**get\_userData()****YWakeUpMonitor****wakeupmonitor**→**userData()**

Retourne le contenu de l'attribut `userData`, précédemment stocké à l'aide de la méthode `set_userData`.

js	function <b>get_userData</b> ( )
cpp	void * <b>get_userData</b> ( )
m	-(id) <code>userData</code>
pas	Tobject <b>get_userData</b> ( ): Tobject
vb	function <b>get_userData</b> ( ) As Object
cs	object <b>get_userData</b> ( )
java	Object <b>get_userData</b> ( )
py	<b>get_userData</b> ( )
php	function <b>get_userData</b> ( )
ts	async <b>get_userData</b> ( ): Promise<object null>
es	async <b>get_userData</b> ( )

Cet attribut n'es pas utilisé directement par l'API. Il est à la disposition de l'appelant pour stocker un contexte.

**Retourne :**

l'objet stocké précédemment par l'appelant.

wakeupmonitor → `get_wakeUpReason()`

YWakeUpMonitor

wakeupmonitor → `wakeUpReason()`

Renvoie la raison du dernier réveil.

js	function <code>get_wakeUpReason()</code>
cpp	<code>Y_WAKEUPREASON_enum</code> <code>get_wakeUpReason()</code>
m	<code>-(Y_WAKEUPREASON_enum)</code> <code>wakeUpReason</code>
pas	Integer <code>get_wakeUpReason()</code> : Integer
vb	function <code>get_wakeUpReason()</code> As Integer
cs	int <code>get_wakeUpReason()</code>
java	int <code>get_wakeUpReason()</code>
uwp	async Task<int> <code>get_wakeUpReason()</code>
py	<code>get_wakeUpReason()</code>
php	function <code>get_wakeUpReason()</code>
ts	async <code>get_wakeUpReason()</code> : Promise<YWakeUpMonitor_WakeUpReason>
es	async <code>get_wakeUpReason()</code>
dnp	int <code>get_wakeUpReason()</code>
cp	int <code>get_wakeUpReason()</code>
cmd	YWakeUpMonitor <b>target</b> <code>get_wakeUpReason</code>

**Retourne :**

```

une valeur parmi YWakeUpMonitor.WAKEUPREASON_USBPOWER,
YWakeUpMonitor.WAKEUPREASON_EXTPOWER,
YWakeUpMonitor.WAKEUPREASON_ENDOFSLEEP,
YWakeUpMonitor.WAKEUPREASON_EXTSIG1,
YWakeUpMonitor.WAKEUPREASON_SCHEDULE1           et
YWakeUpMonitor.WAKEUPREASON_SCHEDULE2

```

En cas d'erreur, déclenche une exception ou retourne `YWakeUpMonitor.WAKEUPREASON_INVALID`.

**wakeupmonitor**→**get\_wakeUpState()****YWakeUpMonitor****wakeupmonitor**→**wakeUpState()**

Revoie l'état actuel du moniteur.

js	function <b>get_wakeUpState</b> ( )
cpp	Y_WAKEUPSTATE_enum <b>get_wakeUpState</b> ( )
m	-(Y_WAKEUPSTATE_enum) wakeUpState
pas	Integer <b>get_wakeUpState</b> ( ): Integer
vb	function <b>get_wakeUpState</b> ( ) As Integer
cs	int <b>get_wakeUpState</b> ( )
java	int <b>get_wakeUpState</b> ( )
uwp	async Task<int> <b>get_wakeUpState</b> ( )
py	<b>get_wakeUpState</b> ( )
php	function <b>get_wakeUpState</b> ( )
ts	async <b>get_wakeUpState</b> ( ): Promise<YWakeUpMonitor_WakeUpState>
es	async <b>get_wakeUpState</b> ( )
dnp	int <b>get_wakeUpState</b> ( )
cp	int <b>get_wakeUpState</b> ( )
cmd	YWakeUpMonitor <b>target</b> <b>get_wakeUpState</b>

**Retourne :**

soit `YWakeUpMonitor.WAKEUPSTATE_SLEEPING`, soit  
`YWakeUpMonitor.WAKEUPSTATE_AWAKE`

En cas d'erreur, déclenche une exception ou retourne  
`YWakeUpMonitor.WAKEUPSTATE_INVALID`.

**wakeupmonitor**→**isOnline()****YWakeUpMonitor**

Vérifie si le module hébergeant le moniteur de réveil est joignable, sans déclencher d'erreur.

js	function <b>isOnline</b> ( )
cpp	bool <b>isOnline</b> ( )
m	-(BOOL) <b>isOnline</b>
pas	boolean <b>isOnline</b> ( ): boolean
vb	function <b>isOnline</b> ( ) As Boolean
cs	bool <b>isOnline</b> ( )
java	boolean <b>isOnline</b> ( )
py	<b>isOnline</b> ( )
php	function <b>isOnline</b> ( )
ts	async <b>isOnline</b> ( ): Promise<boolean>
es	async <b>isOnline</b> ( )
dnp	bool <b>isOnline</b> ( )
cp	bool <b>isOnline</b> ( )

Si les valeurs des attributs en cache du moniteur de réveil sont valides au moment de l'appel, le module est considéré joignable. Cette fonction ne cause en aucun cas d'exception, quelle que soit l'erreur qui pourrait se produire lors de la vérification de joignabilité.

**Retourne :**

`true` si le moniteur de réveil est joignable, `false` sinon

**wakeupmonitor**→**isOnline\_async()****YWakeUpMonitor**

Vérifie si le module hébergeant le moniteur de réveil est joignable, sans déclencher d'erreur.

```
js fonction isOnline_async( callback, context)
```

Si les valeurs des attributs en cache du moniteur de réveil sont valides au moment de l'appel, le module est considéré joignable. Cette fonction ne cause en aucun cas d'exception, quelle que soit l'erreur qui pourrait se produire lors de la vérification de joignabilité.

Cette version asynchrone n'existe qu'en Javascript. Elle utilise une fonction de callback plutôt qu'une simple valeur de retour, pour éviter de bloquer la machine virtuelle Javascript avec une attente active.

**Paramètres :**

**callback** fonction de callback qui sera appelée dès que le résultat sera connu. La fonction callback reçoit trois arguments: le contexte fourni par l'appelant, l'objet fonction concerné et le résultat booléen

**context** contexte fourni par l'appelant, et qui sera passé tel-quel à la fonction de callback

**Retourne :**

rien du tout : le résultat sera passé en paramètre à la fonction de callback.

## wakeupmonitor→isReadOnly()

## YWakeUpMonitor

Test si la fonction est en lecture seule.

cpp	bool <b>isReadOnly</b> ( )
m	-(bool) <b>isReadOnly</b>
pas	boolean <b>isReadOnly</b> ( ): boolean
vb	function <b>isReadOnly</b> ( ) As Boolean
cs	bool <b>isReadOnly</b> ( )
java	boolean <b>isReadOnly</b> ( )
uwp	async Task<bool> <b>isReadOnly</b> ( )
py	<b>isReadOnly</b> ( )
php	function <b>isReadOnly</b> ( )
ts	async <b>isReadOnly</b> ( ): Promise<boolean>
es	async <b>isReadOnly</b> ( )
dnp	bool <b>isReadOnly</b> ( )
cp	bool <b>isReadOnly</b> ( )
cmd	YWakeUpMonitor <b>target isReadOnly</b>

Retourne vrais si la fonction est protégé en ecriture ou que la fontion n'est pas disponible.

**Retourne :**

true si la fonction est protégé en ecriture ou que la fontion n'est pas disponible

**wakeupmonitor**→**load()****YWakeUpMonitor**

Met en cache les valeurs courantes du moniteur de réveil, avec une durée de validité spécifiée.

js	function <b>load</b> ( <b>msValidity</b> )
cpp	YRETCODE <b>load</b> ( int <b>msValidity</b> )
m	-(YRETCODE) <b>load</b> : (u64) <b>msValidity</b>
pas	YRETCODE <b>load</b> ( <b>msValidity</b> : u64): YRETCODE
vb	function <b>load</b> ( ByVal <b>msValidity</b> As Long) As YRETCODE
cs	YRETCODE <b>load</b> ( ulong <b>msValidity</b> )
java	int <b>load</b> ( long <b>msValidity</b> )
py	<b>load</b> ( <b>msValidity</b> )
php	function <b>load</b> ( <b>\$msValidity</b> )
ts	async <b>load</b> ( <b>msValidity</b> : number): Promise<number>
es	async <b>load</b> ( <b>msValidity</b> )

Par défaut, lorsqu'on accède à un module, tous les attributs des fonctions du module sont automatiquement mises en cache pour la durée standard (5 ms). Cette méthode peut être utilisée pour marquer occasionnellement les données cachées comme valides pour une plus longue période, par exemple dans le but de réduire le trafic réseau.

**Paramètres :**

**msValidity** un entier correspondant à la durée de validité attribuée aux les paramètres chargés, en millisecondes

**Retourne :**

YAPI . SUCCESS si l'opération se déroule sans erreur.

En cas d'erreur, déclenche une exception ou retourne un code d'erreur négatif.

## wakeupmonitor → loadAttribute()

## YWakeUpMonitor

Retourne la valeur actuelle d'un attribut spécifique de la fonction, sous forme de texte, le plus rapidement possible mais sans passer par le cache.

js	function <b>loadAttribute</b> ( <b>attrName</b> )
cpp	string <b>loadAttribute</b> ( string <b>attrName</b> )
m	-(NSString*) <b>loadAttribute</b> : (NSString*) <b>attrName</b>
pas	string <b>loadAttribute</b> ( <b>attrName</b> : string): string
vb	function <b>loadAttribute</b> ( ByVal <b>attrName</b> As String) As String
cs	string <b>loadAttribute</b> ( string <b>attrName</b> )
java	String <b>loadAttribute</b> ( String <b>attrName</b> )
uwp	async Task<string> <b>loadAttribute</b> ( string <b>attrName</b> )
py	<b>loadAttribute</b> ( <b>attrName</b> )
php	function <b>loadAttribute</b> ( <b>\$attrName</b> )
ts	async <b>loadAttribute</b> ( <b>attrName</b> : string): Promise<string>
es	async <b>loadAttribute</b> ( <b>attrName</b> )
dnp	string <b>loadAttribute</b> ( string <b>attrName</b> )
cp	string <b>loadAttribute</b> ( string <b>attrName</b> )

**Paramètres :**

**attrName** le nom de l'attribut désiré

**Retourne :**

une chaîne de caractères représentant la valeur actuelle de l'attribut.

En cas d'erreur, déclenche une exception ou retourne un chaîne vide.

**wakeupmonitor**→**load\_async()****YWakeUpMonitor**

Met en cache les valeurs courantes du moniteur de réveil, avec une durée de validité spécifiée.

```
js function load_async( msValidity, callback, context)
```

Par défaut, lorsqu'on accède à un module, tous les attributs des fonctions du module sont automatiquement mises en cache pour la durée standard (5 ms). Cette méthode peut être utilisée pour marquer occasionnellement les données cachées comme valides pour une plus longue période, par exemple dans le but de réduire le trafic réseau.

Cette version asynchrone n'existe qu'en Javascript. Elle utilise une fonction de callback plutôt qu'une simple valeur de retour, pour éviter de bloquer la machine virtuelle Javascript avec une attente active.

**Paramètres :**

- msValidity** un entier correspondant à la durée de validité attribuée aux les paramètres chargés, en millisecondes
- callback** fonction de callback qui sera appelée dès que le résultat sera connu. La fonction callback reçoit trois arguments: le contexte fourni par l'appelant, l'objet fonction concerné et le code d'erreur (ou YAPI . SUCCESS)
- context** contexte fourni par l'appelant, et qui sera passé tel-quel à la fonction de callback

**Retourne :**

rien du tout : le résultat sera passé en paramètre à la fonction de callback.

## wakeupmonitor → muteValueCallbacks()

## YWakeUpMonitor

Désactive l'envoi de chaque changement de la valeur publiée au hub parent.

js	function <b>muteValueCallbacks</b> ( )
cpp	int <b>muteValueCallbacks</b> ( )
m	-(int) <b>muteValueCallbacks</b>
pas	LongInt <b>muteValueCallbacks</b> ( ): LongInt
vb	function <b>muteValueCallbacks</b> ( ) As Integer
cs	int <b>muteValueCallbacks</b> ( )
java	int <b>muteValueCallbacks</b> ( )
uwp	async Task<int> <b>muteValueCallbacks</b> ( )
py	<b>muteValueCallbacks</b> ( )
php	function <b>muteValueCallbacks</b> ( )
ts	async <b>muteValueCallbacks</b> ( ): Promise<number>
es	async <b>muteValueCallbacks</b> ( )
dnp	int <b>muteValueCallbacks</b> ( )
cp	int <b>muteValueCallbacks</b> ( )
cmd	YWakeUpMonitor <b>target</b> <b>muteValueCallbacks</b>

Vous pouvez utiliser cette fonction pour économiser la bande passante et le CPU sur les machines de faible puissance, ou pour éviter le déclenchement de callbacks HTTP. N'oubliez pas d'appeler la méthode `saveToFlash()` du module si le réglage doit être préservé.

**Retourne :**

YAPI . SUCCESS si l'opération se déroule sans erreur.

En cas d'erreur, déclenche une exception ou retourne un code d'erreur négatif.

**wakeupmonitor**→**nextWakeUpMonitor()****YWakeUpMonitor**

Continue l'énumération des moniteurs de réveil commencée à l'aide de `yFirstWakeUpMonitor()`. Attention, vous ne pouvez faire aucune supposition sur l'ordre dans lequel les moniteurs de réveil sont retournés.

js	function <b>nextWakeUpMonitor()</b>
cpp	<code>YWakeUpMonitor *</code> <b>nextWakeUpMonitor()</b>
m	<code>-(nullable YWakeUpMonitor*)</code> <b>nextWakeUpMonitor</b>
pas	<code>TYWakeUpMonitor</code> <b>nextWakeUpMonitor()</b> : <code>TYWakeUpMonitor</code>
vb	function <b>nextWakeUpMonitor()</b> As <code>YWakeUpMonitor</code>
cs	<code>YWakeUpMonitor</code> <b>nextWakeUpMonitor()</b>
java	<code>YWakeUpMonitor</code> <b>nextWakeUpMonitor()</b>
uwp	<code>YWakeUpMonitor</code> <b>nextWakeUpMonitor()</b>
py	<b>nextWakeUpMonitor()</b>
php	function <b>nextWakeUpMonitor()</b>
ts	<b>nextWakeUpMonitor()</b> : <code>YWakeUpMonitor</code>   null
es	<b>nextWakeUpMonitor()</b>

Si vous souhaitez retrouver un moniteur de réveil spécifique, utilisez `WakeUpMonitor.findWakeUpMonitor()` avec un `hardwareID` ou un nom logique.

**Retourne :**

un pointeur sur un objet `YWakeUpMonitor` accessible en ligne, ou `null` lorsque l'énumération est terminée.

## wakeupmonitor → registerValueCallback()

## YWakeUpMonitor

Enregistre la fonction de callback qui est appelée à chaque changement de la valeur publiée.

js	function <b>registerValueCallback</b> ( <b>callback</b> )
cpp	int <b>registerValueCallback</b> ( YWakeUpMonitorValueCallback <b>callback</b> )
m	-(int) <b>registerValueCallback</b> : (YWakeUpMonitorValueCallback _Nullable) <b>callback</b>
pas	LongInt <b>registerValueCallback</b> ( <b>callback</b> : TYWakeUpMonitorValueCallback): LongInt
vb	function <b>registerValueCallback</b> ( ByVal <b>callback</b> As YWakeUpMonitorValueCallback) As Integer
cs	int <b>registerValueCallback</b> ( ValueCallback <b>callback</b> )
java	int <b>registerValueCallback</b> ( UpdateCallback <b>callback</b> )
uwp	async Task<int> <b>registerValueCallback</b> ( ValueCallback <b>callback</b> )
py	<b>registerValueCallback</b> ( <b>callback</b> )
php	function <b>registerValueCallback</b> ( <b>\$callback</b> )
ts	async <b>registerValueCallback</b> ( <b>callback</b> : YWakeUpMonitorValueCallback   null): Promise<number>
es	async <b>registerValueCallback</b> ( <b>callback</b> )

Ce callback n'est appelé que durant l'exécution de `ySleep` ou `yHandleEvents`. Cela permet à l'appelant de contrôler quand les callback peuvent se produire. Il est important d'appeler l'une de ces deux fonctions périodiquement pour garantir que les callback ne soient pas appelés trop tard. Pour désactiver un callback, il suffit d'appeler cette méthode en lui passant un pointeur nul.

**Paramètres :**

**callback** la fonction de callback à rappeler, ou un pointeur nul. La fonction de callback doit accepter deux arguments: l'object fonction dont la valeur a changé, et la chaîne de caractère décrivant la nouvelle valeur publiée.

**wakeupmonitor**→**resetSleepCountDown()****YWakeUpMonitor**

Réinitialise le compteur de mise en sommeil.

js	function <b>resetSleepCountDown</b> ( )
cpp	int <b>resetSleepCountDown</b> ( )
m	-(int) <b>resetSleepCountDown</b>
pas	LongInt <b>resetSleepCountDown</b> ( ): LongInt
vb	function <b>resetSleepCountDown</b> ( ) As Integer
cs	int <b>resetSleepCountDown</b> ( )
java	int <b>resetSleepCountDown</b> ( )
uwp	async Task<int> <b>resetSleepCountDown</b> ( )
py	<b>resetSleepCountDown</b> ( )
php	function <b>resetSleepCountDown</b> ( )
ts	async <b>resetSleepCountDown</b> ( ): Promise<number>
es	async <b>resetSleepCountDown</b> ( )
dnp	int <b>resetSleepCountDown</b> ( )
cp	int <b>resetSleepCountDown</b> ( )
cmd	YWakeUpMonitor <b>target</b> <b>resetSleepCountDown</b>

**Retourne :**

YAPI.SUCCESS si l'opération se déroule sans erreur. En cas d'erreur, déclenche une exception ou retourne un code d'erreur négatif.

**wakeupmonitor**→**set\_logicalName()****YWakeUpMonitor****wakeupmonitor**→**setLogicalName()**

Modifie le nom logique du moniteur de réveil.

js	function <b>set_logicalName</b> ( <b>newval</b> )
cpp	int <b>set_logicalName</b> ( string <b>newval</b> )
m	-(int) setLogicalName : (NSString*) <b>newval</b>
pas	integer <b>set_logicalName</b> ( <b>newval</b> : string): integer
vb	function <b>set_logicalName</b> ( ByVal <b>newval</b> As String) As Integer
cs	int <b>set_logicalName</b> ( string <b>newval</b> )
java	int <b>set_logicalName</b> ( String <b>newval</b> )
uwp	async Task<int> <b>set_logicalName</b> ( string <b>newval</b> )
py	<b>set_logicalName</b> ( <b>newval</b> )
php	function <b>set_logicalName</b> ( <b>\$newval</b> )
ts	async <b>set_logicalName</b> ( <b>newval</b> : string): Promise<number>
es	async <b>set_logicalName</b> ( <b>newval</b> )
dnp	int <b>set_logicalName</b> ( string <b>newval</b> )
cp	int <b>set_logicalName</b> ( string <b>newval</b> )
cmd	YWakeUpMonitor <b>target</b> <b>set_logicalName</b> <b>newval</b>

Vous pouvez utiliser `yCheckLogicalName()` pour vérifier si votre paramètre est valide. N'oubliez pas d'appeler la méthode `saveToFlash()` du module si le réglage doit être préservé.

**Paramètres :**

**newval** une chaîne de caractères représentant le nom logique du moniteur de réveil.

**Retourne :**

YAPI . SUCCESS si l'appel se déroule sans erreur.

En cas d'erreur, déclenche une exception ou retourne un code d'erreur négatif.

**wakeupmonitor**→**set\_nextWakeUp()****YWakeUpMonitor****wakeupmonitor**→**setNextWakeUp()**

Modifie les jours de la semaine où un réveil doit avoir lieu.

js	function <b>set_nextWakeUp</b> ( <b>newval</b> )
cpp	int <b>set_nextWakeUp</b> ( s64 <b>newval</b> )
m	-(int) setNextWakeUp : (s64) <b>newval</b>
pas	integer <b>set_nextWakeUp</b> ( <b>newval</b> : int64): integer
vb	function <b>set_nextWakeUp</b> ( ByVal <b>newval</b> As Long) As Integer
cs	int <b>set_nextWakeUp</b> ( long <b>newval</b> )
java	int <b>set_nextWakeUp</b> ( long <b>newval</b> )
uwp	async Task<int> <b>set_nextWakeUp</b> ( long <b>newval</b> )
py	<b>set_nextWakeUp</b> ( <b>newval</b> )
php	function <b>set_nextWakeUp</b> ( \$ <b>newval</b> )
ts	async <b>set_nextWakeUp</b> ( <b>newval</b> : number): Promise<number>
es	async <b>set_nextWakeUp</b> ( <b>newval</b> )
dnp	int <b>set_nextWakeUp</b> ( long <b>newval</b> )
cp	int <b>set_nextWakeUp</b> ( s64 <b>newval</b> )
cmd	YWakeUpMonitor <b>target</b> <b>set_nextWakeUp</b> <b>newval</b>

**Paramètres :**

**newval** un entier représentant les jours de la semaine où un réveil doit avoir lieu

**Retourne :**

YAPI . SUCCESS si l'opération se déroule sans erreur.

En cas d'erreur, déclenche une exception ou retourne un code d'erreur négatif.

wakeupmonitor → **set\_powerDuration()**

YWakeUpMonitor

wakeupmonitor → **setPowerDuration()**

Modifie le temps d'éveil maximal en secondes avant de retourner en sommeil automatiquement.

js	function <b>set_powerDuration</b> ( <b>newval</b> )
cpp	int <b>set_powerDuration</b> ( int <b>newval</b> )
m	-(int) <b>setPowerDuration</b> : (int) <b>newval</b>
pas	integer <b>set_powerDuration</b> ( <b>newval</b> : LongInt): integer
vb	function <b>set_powerDuration</b> ( ByVal <b>newval</b> As Integer) As Integer
cs	int <b>set_powerDuration</b> ( int <b>newval</b> )
java	int <b>set_powerDuration</b> ( int <b>newval</b> )
uwp	async Task<int> <b>set_powerDuration</b> ( int <b>newval</b> )
py	<b>set_powerDuration</b> ( <b>newval</b> )
php	function <b>set_powerDuration</b> ( <b>\$newval</b> )
ts	async <b>set_powerDuration</b> ( <b>newval</b> : number): Promise<number>
es	async <b>set_powerDuration</b> ( <b>newval</b> )
dnp	int <b>set_powerDuration</b> ( int <b>newval</b> )
cp	int <b>set_powerDuration</b> ( int <b>newval</b> )
cmd	YWakeUpMonitor <b>target set_powerDuration newval</b>

N'oubliez pas d'appeler la méthode `saveToFlash()` du module si le réglage doit être préservé.

**Paramètres :**

**newval** un entier représentant le temps d'éveil maximal en secondes avant de retourner en sommeil automatiquement

**Retourne :**

YAPI . SUCCESS si l'opération se déroule sans erreur.

En cas d'erreur, déclenche une exception ou retourne un code d'erreur négatif.

**wakeupmonitor**→**set\_sleepCountdown()****YWakeUpMonitor****wakeupmonitor**→**setSleepCountdown()**

Modifie le temps avant le prochain sommeil .

js	function <b>set_sleepCountdown</b> ( <b>newval</b> )
cpp	int <b>set_sleepCountdown</b> ( int <b>newval</b> )
m	-(int) <b>setSleepCountdown</b> : (int) <b>newval</b>
pas	integer <b>set_sleepCountdown</b> ( <b>newval</b> : LongInt): integer
vb	function <b>set_sleepCountdown</b> ( ByVal <b>newval</b> As Integer) As Integer
cs	int <b>set_sleepCountdown</b> ( int <b>newval</b> )
java	int <b>set_sleepCountdown</b> ( int <b>newval</b> )
uwp	async Task<int> <b>set_sleepCountdown</b> ( int <b>newval</b> )
py	<b>set_sleepCountdown</b> ( <b>newval</b> )
php	function <b>set_sleepCountdown</b> ( <b>\$newval</b> )
ts	async <b>set_sleepCountdown</b> ( <b>newval</b> : number): Promise<number>
es	async <b>set_sleepCountdown</b> ( <b>newval</b> )
dnp	int <b>set_sleepCountdown</b> ( int <b>newval</b> )
cp	int <b>set_sleepCountdown</b> ( int <b>newval</b> )
cmd	YWakeUpMonitor <b>target</b> <b>set_sleepCountdown</b> <b>newval</b>

**Paramètres :**

**newval** un entier représentant le temps avant le prochain sommeil

**Retourne :**

YAPI . SUCCESS si l'opération se déroule sans erreur.

En cas d'erreur, déclenche une exception ou retourne un code d'erreur négatif.

**wakeupmonitor**→**set\_userData()****YWakeUpMonitor****wakeupmonitor**→**setUserData()**

Enregistre un contexte libre dans l'attribut `userData` de la fonction, afin de le retrouver plus tard à l'aide de la méthode `get_userData`.

js	function <b>set_userData</b> ( <b>data</b> )
cpp	void <b>set_userData</b> ( void * <b>data</b> )
m	-(void) <b>setUserData</b> : (id) <b>data</b>
pas	<b>set_userData</b> ( <b>data</b> : Tobject)
vb	procedure <b>set_userData</b> ( ByVal <b>data</b> As Object)
cs	void <b>set_userData</b> ( object <b>data</b> )
java	void <b>set_userData</b> ( Object <b>data</b> )
py	<b>set_userData</b> ( <b>data</b> )
php	function <b>set_userData</b> ( <b>\$data</b> )
ts	async <b>set_userData</b> ( <b>data</b> : object null): Promise<void>
es	async <b>set_userData</b> ( <b>data</b> )

Cet attribut n'es pas utilisé directement par l'API. Il est à la disposition de l'appelant pour stocker un contexte.

**Paramètres :**

**data** objet quelconque à mémoriser

**wakeupmonitor**→**sleep()****YWakeUpMonitor**

Déclenche une mise en sommeil jusqu'à la prochaine condition de réveil, l'heure du RTC du module doit impérativement avoir été réglée au préalable.

js	function <b>sleep</b> ( <b>secBeforeSleep</b> )
cpp	int <b>sleep</b> ( int <b>secBeforeSleep</b> )
m	-(int) <b>sleep</b> : (int) <b>secBeforeSleep</b>
pas	LongInt <b>sleep</b> ( <b>secBeforeSleep</b> : LongInt): LongInt
vb	function <b>sleep</b> ( ByVal <b>secBeforeSleep</b> As Integer) As Integer
cs	int <b>sleep</b> ( int <b>secBeforeSleep</b> )
java	int <b>sleep</b> ( int <b>secBeforeSleep</b> )
uwp	async Task<int> <b>sleep</b> ( int <b>secBeforeSleep</b> )
py	<b>sleep</b> ( <b>secBeforeSleep</b> )
php	function <b>sleep</b> ( <b>\$secBeforeSleep</b> )
ts	async <b>sleep</b> ( <b>secBeforeSleep</b> : number): Promise<number>
es	async <b>sleep</b> ( <b>secBeforeSleep</b> )
dnp	int <b>sleep</b> ( int <b>secBeforeSleep</b> )
cp	int <b>sleep</b> ( int <b>secBeforeSleep</b> )
cmd	YWakeUpMonitor <b>target sleep secBeforeSleep</b>

**Paramètres :**

**secBeforeSleep** nombre de seconde avant la mise en sommeil

**Retourne :**

YAPI . SUCCESS si l'opération se déroule sans erreur.

En cas d'erreur, déclenche une exception ou retourne un code d'erreur négatif.

## wakeupmonitor → sleepFor()

## YWakeUpMonitor

Déclenche une mise en sommeil pour un temps donné ou jusqu'à la prochaine condition de réveil, l'heure du RTC du module doit impérativement avoir été réglée au préalable.

js	function <b>sleepFor</b> ( <b>secUntilWakeUp</b> , <b>secBeforeSleep</b> )
cpp	int <b>sleepFor</b> ( int <b>secUntilWakeUp</b> , int <b>secBeforeSleep</b> )
m	-(int) <b>sleepFor</b> : (int) <b>secUntilWakeUp</b> : (int) <b>secBeforeSleep</b>
pas	LongInt <b>sleepFor</b> ( <b>secUntilWakeUp</b> : LongInt, <b>secBeforeSleep</b> : LongInt): LongInt
vb	function <b>sleepFor</b> ( ByVal <b>secUntilWakeUp</b> As Integer, ByVal <b>secBeforeSleep</b> As Integer) As Integer
cs	int <b>sleepFor</b> ( int <b>secUntilWakeUp</b> , int <b>secBeforeSleep</b> )
java	int <b>sleepFor</b> ( int <b>secUntilWakeUp</b> , int <b>secBeforeSleep</b> )
uwp	async Task<int> <b>sleepFor</b> ( int <b>secUntilWakeUp</b> , int <b>secBeforeSleep</b> )
py	<b>sleepFor</b> ( <b>secUntilWakeUp</b> , <b>secBeforeSleep</b> )
php	function <b>sleepFor</b> ( <b>\$secUntilWakeUp</b> , <b>\$secBeforeSleep</b> )
ts	async <b>sleepFor</b> ( <b>secUntilWakeUp</b> : number, <b>secBeforeSleep</b> : number): Promise<number>
es	async <b>sleepFor</b> ( <b>secUntilWakeUp</b> , <b>secBeforeSleep</b> )
dnp	int <b>sleepFor</b> ( int <b>secUntilWakeUp</b> , int <b>secBeforeSleep</b> )
cp	int <b>sleepFor</b> ( int <b>secUntilWakeUp</b> , int <b>secBeforeSleep</b> )
cmd	YWakeUpMonitor <b>target sleepFor secUntilWakeUp secBeforeSleep</b>

Le compte à rebours avant la mise en sommeil peut être annulé grâce à `resetSleepCountDown`.

**Paramètres :**

- secUntilWakeUp** nombre de secondes avant le prochain réveil
- secBeforeSleep** nombre de secondes avant la mise en sommeil

**Retourne :**

YAPI . SUCCESS si l'opération se déroule sans erreur.

En cas d'erreur, déclenche une exception ou retourne un code d'erreur négatif.

**wakeupmonitor**→**sleepUntil()****YWakeUpMonitor**

Déclenche une mise en sommeil jusqu'à une date donnée ou jusqu'à la prochaine condition de réveil, l'heure du RTC du module doit impérativement avoir été réglée au préalable.

js	function <b>sleepUntil</b> ( <b>wakeUpTime</b> , <b>secBeforeSleep</b> )
cpp	int <b>sleepUntil</b> ( int <b>wakeUpTime</b> , int <b>secBeforeSleep</b> )
m	-(int) <b>sleepUntil</b> : (int) <b>wakeUpTime</b> : (int) <b>secBeforeSleep</b>
pas	LongInt <b>sleepUntil</b> ( <b>wakeUpTime</b> : LongInt, <b>secBeforeSleep</b> : LongInt): LongInt
vb	function <b>sleepUntil</b> ( ByVal <b>wakeUpTime</b> As Integer, ByVal <b>secBeforeSleep</b> As Integer) As Integer
cs	int <b>sleepUntil</b> ( int <b>wakeUpTime</b> , int <b>secBeforeSleep</b> )
java	int <b>sleepUntil</b> ( int <b>wakeUpTime</b> , int <b>secBeforeSleep</b> )
uwp	async Task<int> <b>sleepUntil</b> ( int <b>wakeUpTime</b> , int <b>secBeforeSleep</b> )
py	<b>sleepUntil</b> ( <b>wakeUpTime</b> , <b>secBeforeSleep</b> )
php	function <b>sleepUntil</b> ( \$ <b>wakeUpTime</b> , \$ <b>secBeforeSleep</b> )
ts	async <b>sleepUntil</b> ( <b>wakeUpTime</b> : number, <b>secBeforeSleep</b> : number): Promise<number>
es	async <b>sleepUntil</b> ( <b>wakeUpTime</b> , <b>secBeforeSleep</b> )
dnf	int <b>sleepUntil</b> ( int <b>wakeUpTime</b> , int <b>secBeforeSleep</b> )
cp	int <b>sleepUntil</b> ( int <b>wakeUpTime</b> , int <b>secBeforeSleep</b> )
cmd	YWakeUpMonitor <b>target</b> <b>sleepUntil</b> <b>wakeUpTime</b> <b>secBeforeSleep</b>

Le compte à rebours avant la mise en sommeil peut être annulé grâce à `resetSleepCountDown`.

**Paramètres :**

**wakeUpTime** date/heure du réveil (format UNIX)  
**secBeforeSleep** nombre de secondes avant la mise en sommeil

**Retourne :**

YAPI . SUCCESS si l'opération se déroule sans erreur.

En cas d'erreur, déclenche une exception ou retourne un code d'erreur négatif.

## wakeupmonitor → unmuteValueCallbacks()

## YWakeUpMonitor

Réactive l'envoi de chaque changement de la valeur publiée au hub parent.

js	function <b>unmuteValueCallbacks</b> ( )
cpp	int <b>unmuteValueCallbacks</b> ( )
m	-(int) <b>unmuteValueCallbacks</b>
pas	LongInt <b>unmuteValueCallbacks</b> ( ): LongInt
vb	function <b>unmuteValueCallbacks</b> ( ) As Integer
cs	int <b>unmuteValueCallbacks</b> ( )
java	int <b>unmuteValueCallbacks</b> ( )
uwp	async Task<int> <b>unmuteValueCallbacks</b> ( )
py	<b>unmuteValueCallbacks</b> ( )
php	function <b>unmuteValueCallbacks</b> ( )
ts	async <b>unmuteValueCallbacks</b> ( ): Promise<number>
es	async <b>unmuteValueCallbacks</b> ( )
dnp	int <b>unmuteValueCallbacks</b> ( )
cp	int <b>unmuteValueCallbacks</b> ( )
cmd	YWakeUpMonitor <b>target unmuteValueCallbacks</b>

Cette fonction annule un précédent appel à `muteValueCallbacks()`. N'oubliez pas d'appeler la méthode `saveToFlash()` du module si le réglage doit être préservé.

**Retourne :**

YAPI . SUCCESS si l'opération se déroule sans erreur.

En cas d'erreur, déclenche une exception ou retourne un code d'erreur négatif.

**wakeupmonitor**→**wait\_async()****YWakeUpMonitor**

Attend que toutes les commandes asynchrones en cours d'exécution sur le module soient terminées, et appelle le callback passé en paramètre.

```
js fonction wait_async( callback, context)
```

```
ts wait_async( callback: Function, context: object)
```

```
es wait_async( callback, context)
```

La fonction callback peut donc librement utiliser des fonctions synchrones ou asynchrones, sans risquer de bloquer la machine virtuelle Javascript.

**Paramètres :**

**callback** fonction de callback qui sera appelée dès que toutes les commandes en cours d'exécution sur le module seront terminées. La fonction callback reçoit deux arguments: le contexte fourni par l'appelant et l'objet fonction concerné.

**context** contexte fourni par l'appelant, et qui sera passé tel-qu'el à la fonction de callback

**Retourne :**

rien du tout.

## wakeupmonitor → wakeup()

## YWakeUpMonitor

Force un réveil.

js	function <b>wakeup</b> ( )
cpp	int <b>wakeup</b> ( )
m	-(int) <b>wakeup</b>
pas	LongInt <b>wakeup</b> ( ): LongInt
vb	function <b>wakeup</b> ( ) As Integer
cs	int <b>wakeup</b> ( )
java	int <b>wakeup</b> ( )
uwp	async Task<int> <b>wakeup</b> ( )
py	<b>wakeup</b> ( )
php	function <b>wakeup</b> ( )
ts	async <b>wakeup</b> ( ): Promise<number>
es	async <b>wakeup</b> ( )
dnp	int <b>wakeup</b> ( )
cp	int <b>wakeup</b> ( )
cmd	YWakeUpMonitor <b>target wakeup</b>

## 8.7. La classe YWakeUpSchedule

Interface pour interagir avec les réveils agendés, disponibles par exemple dans le YoctoHub-GSM-3G-EU, le YoctoHub-GSM-3G-NA, le YoctoHub-GSM-4G et le YoctoHub-Wireless-n

La classe `YWakeUpSchedule` implémente une condition de réveil. Le réveil est spécifié par un ensemble de mois et/ou jours et/ou heures et/ou minutes où il doit se produire.

Pour utiliser les fonctions décrites ici, vous devez inclure:

es	in HTML: <code>&lt;script src="../../lib/yocto_wakeupschedule.js"&gt;&lt;/script&gt;</code> in node.js: <code>require('yoctolib-es2017/yocto_wakeupschedule.js');</code>
js	<code>&lt;script type='text/javascript' src='yocto_wakeupschedule.js'&gt;&lt;/script&gt;</code>
cpp	<code>#include "yocto_wakeupschedule.h"</code>
m	<code>#import "yocto_wakeupschedule.h"</code>
pas	<code>uses yocto_wakeupschedule;</code>
vb	<code>yocto_wakeupschedule.vb</code>
cs	<code>yocto_wakeupschedule.cs</code>
java	<code>import com.yoctopuce.YoctoAPI.YWakeUpSchedule;</code>
uwp	<code>import com.yoctopuce.YoctoAPI.YWakeUpSchedule;</code>
py	<code>from yocto_wakeupschedule import *</code>
php	<code>require_once('yocto_wakeupschedule.php');</code>
ts	in HTML: <code>import { YWakeUpSchedule } from '../../dist/esm/yocto_wakeupschedule.js';</code> in Node.js: <code>import { YWakeUpSchedule } from 'yoctolib-cjs/yocto_wakeupschedule.js';</code>
dnp	<code>import YoctoProxyAPI.YWakeUpScheduleProxy</code>
cp	<code>#include "yocto_wakeupschedule_proxy.h"</code>
vi	<code>YWakeUpSchedule.vi</code>
ml	<code>import YoctoProxyAPI.YWakeUpScheduleProxy</code>

### Fonction globales

#### **YWakeUpSchedule.FindWakeUpSchedule(func)**

Permet de retrouver un réveil agendé d'après un identifiant donné.

#### **YWakeUpSchedule.FindWakeUpScheduleInContext(yctx, func)**

Permet de retrouver un réveil agendé d'après un identifiant donné dans un Context YAPI.

#### **YWakeUpSchedule.FirstWakeUpSchedule()**

Commence l'énumération des réveils agendés accessibles par la librairie.

#### **YWakeUpSchedule.FirstWakeUpScheduleInContext(yctx)**

Commence l'énumération des réveils agendés accessibles par la librairie.

#### **YWakeUpSchedule.GetSimilarFunctions()**

Enumère toutes les fonctions de type `WakeUpSchedule` disponibles sur les modules actuellement joignables par la librairie, et retourne leurs identifiants matériels uniques (`hardwareId`).

### Propriétés des objets `YWakeUpScheduleProxy`

#### **wakeupschedule→AdvertisedValue** *[lecture seule]*

Courte chaîne de caractères représentant l'état courant de la fonction.

#### **wakeupschedule→FriendlyName** *[lecture seule]*

Identifiant global de la fonction au format `NOM_MODULE . NOM_FONCTION`.

#### **wakeupschedule→FunctionId** *[lecture seule]*

Identifiant matériel du réveil agendé, sans référence au module.

#### **wakeupschedule→HardwareId** *[lecture seule]*

Identifiant matériel unique de la fonction au format SERIAL . FUNCTIONID.
<b>wakeupschedule→Hours</b> <i>[modifiable]</i> S heures où le réveil est actif..
<b>wakeupschedule→IsOnline</b> <i>[lecture seule]</i> Vérifie si le module hébergeant la fonction est joignable, sans déclencher d'erreur.
<b>wakeupschedule→LogicalName</b> <i>[modifiable]</i> Nom logique de la fonction.
<b>wakeupschedule→MinutesA</b> <i>[modifiable]</i> S minutes de l'intervall 00-29 de chaque heures où le réveil est actif.
<b>wakeupschedule→MinutesB</b> <i>[modifiable]</i> S minutes de l'intervall 30-59 de chaque heure où le réveil est actif.
<b>wakeupschedule→MonthDays</b> <i>[modifiable]</i> S jours du mois où le réveil est actif.
<b>wakeupschedule→Months</b> <i>[modifiable]</i> S mois où le réveil est actif.
<b>wakeupschedule→NextOccurence</b> <i>[lecture seule]</i> Date/heure de la prochaine occurrence de réveil.
<b>wakeupschedule→SerialNumber</b> <i>[lecture seule]</i> Numéro de série du module, préprogrammé en usine.
<b>wakeupschedule→WeekDays</b> <i>[modifiable]</i> S jours de la semaine où le réveil est actif.
<b>Méthodes des objets YWakeUpSchedule</b>
<b>wakeupschedule→clearCache()</b> Invalide le cache.
<b>wakeupschedule→describe()</b> Retourne un court texte décrivant de manière non-ambigüe l'instance du réveil agendé au format TYPE ( NAME ) =SERIAL . FUNCTIONID.
<b>wakeupschedule→get_advertisedValue()</b> Retourne la valeur courante du réveil agendé (pas plus de 6 caractères).
<b>wakeupschedule→get_errorMessage()</b> Retourne le message correspondant à la dernière erreur survenue lors de l'utilisation du réveil agendé.
<b>wakeupschedule→get_errorType()</b> Retourne le code d'erreur correspondant à la dernière erreur survenue lors de l'utilisation du réveil agendé.
<b>wakeupschedule→get_friendlyName()</b> Retourne un identifiant global du réveil agendé au format NOM_MODULE . NOM_FONCTION.
<b>wakeupschedule→get_functionDescriptor()</b> Retourne un identifiant unique de type YFUN_DESCR correspondant à la fonction.
<b>wakeupschedule→get_functionId()</b> Retourne l'identifiant matériel du réveil agendé, sans référence au module.
<b>wakeupschedule→get_hardwareId()</b> Retourne l'identifiant matériel unique du réveil agendé au format SERIAL . FUNCTIONID.
<b>wakeupschedule→get_hours()</b> Retourne les heures où le réveil est actif..
<b>wakeupschedule→get_logicalName()</b> Retourne le nom logique du réveil agendé.

**wakeupschedule**→**get\_minutes()**

Retourne toutes les minutes de chaque heure où le réveil est actif.

**wakeupschedule**→**get\_minutesA()**

Retourne les minutes de l'intervall 00-29 de chaque heures où le réveil est actif.

**wakeupschedule**→**get\_minutesB()**

Retourne les minutes de l'intervall 30-59 de chaque heure où le réveil est actif.

**wakeupschedule**→**get\_module()**

Retourne l'objet `YModule` correspondant au module Yoctopuce qui héberge la fonction.

**wakeupschedule**→**get\_module\_async(callback, context)**

Retourne l'objet `YModule` correspondant au module Yoctopuce qui héberge la fonction.

**wakeupschedule**→**get\_monthDays()**

Retourne les jours du mois où le réveil est actif.

**wakeupschedule**→**get\_months()**

Retourne les mois où le réveil est actif.

**wakeupschedule**→**get\_nextOccurence()**

Retourne la date/heure de la prochaine occurrence de réveil.

**wakeupschedule**→**get\_serialNumber()**

Retourne le numéro de série du module, préprogrammé en usine.

**wakeupschedule**→**get\_userData()**

Retourne le contenu de l'attribut `userData`, précédemment stocké à l'aide de la méthode `set_userData`.

**wakeupschedule**→**get\_weekDays()**

Retourne les jours de la semaine où le réveil est actif.

**wakeupschedule**→**isOnline()**

Vérifie si le module hébergeant le réveil agendé est joignable, sans déclencher d'erreur.

**wakeupschedule**→**isOnline\_async(callback, context)**

Vérifie si le module hébergeant le réveil agendé est joignable, sans déclencher d'erreur.

**wakeupschedule**→**isReadOnly()**

Test si la fonction est en lecture seule.

**wakeupschedule**→**load(msValidity)**

Met en cache les valeurs courantes du réveil agendé, avec une durée de validité spécifiée.

**wakeupschedule**→**loadAttribute(attrName)**

Retourne la valeur actuelle d'un attribut spécifique de la fonction, sous forme de texte, le plus rapidement possible mais sans passer par le cache.

**wakeupschedule**→**load\_async(msValidity, callback, context)**

Met en cache les valeurs courantes du réveil agendé, avec une durée de validité spécifiée.

**wakeupschedule**→**muteValueCallbacks()**

Désactive l'envoi de chaque changement de la valeur publiée au hub parent.

**wakeupschedule**→**nextWakeUpSchedule()**

Continue l'énumération des réveils agendés commencée à l'aide de `yFirstWakeUpSchedule()`. Attention, vous ne pouvez faire aucune supposition sur l'ordre dans lequel les réveils agendés sont retournés.

**wakeupschedule**→**registerValueCallback(callback)**

Enregistre la fonction de callback qui est appelée à chaque changement de la valeur publiée.

**wakeupschedule**→**set\_hours(newval)**

Modifie les heures où un réveil doit avoir lieu.

**wakeupschedule**→**set\_logicalName(newval)**

Modifie le nom logique du réveil agendé.

**wakeupschedule→set\_minutes(bitmap)**

Modifie toutes les minutes où un réveil doit avoir lieu

**wakeupschedule→set\_minutesA(newval)**

Modifie les minutes de l'intervall 00-29 où un réveil doit avoir lieu.

**wakeupschedule→set\_minutesB(newval)**

Modifie les minutes de l'intervall 30-59 où un réveil doit avoir lieu.

**wakeupschedule→set\_monthDays(newval)**

Modifie les jours du mois où un réveil doit avoir lieu.

**wakeupschedule→set\_months(newval)**

Modifie les mois où un réveil doit avoir lieu.

**wakeupschedule→set\_userData(data)**

Enregistre un contexte libre dans l'attribut userData de la fonction, afin de le retrouver plus tard à l'aide de la méthode `get_userdata`.

**wakeupschedule→set\_weekDays(newval)**

Modifie les jours de la semaine où un réveil doit avoir lieu.

**wakeupschedule→unmuteValueCallbacks()**

Réactive l'envoi de chaque changement de la valeur publiée au hub parent.

**wakeupschedule→wait\_async(callback, context)**

Attend que toutes les commandes asynchrones en cours d'exécution sur le module soient terminées, et appelle le callback passé en paramètre.

## YWakeUpSchedule.FindWakeUpSchedule() YWakeupSchedule.FindWakeUpSchedule()

## YWakeupSchedule

Permet de retrouver un réveil agendé d'après un identifiant donné.

js	function <b>yFindWakeUpSchedule( func)</b>
cpp	YWakeupSchedule* <b>FindWakeUpSchedule( string func)</b>
m	+(YWakeupSchedule*) <b>FindWakeUpSchedule : (NSString*) func</b>
pas	TYWakeupSchedule <b>yFindWakeUpSchedule( func: string): TYWakeupSchedule</b>
vb	function <b>FindWakeUpSchedule( ByVal func As String) As YWakeupSchedule</b>
cs	static YWakeupSchedule <b>FindWakeUpSchedule( string func)</b>
java	static YWakeupSchedule <b>FindWakeUpSchedule( String func)</b>
uwp	static YWakeupSchedule <b>FindWakeUpSchedule( string func)</b>
py	<b>FindWakeUpSchedule( func)</b>
php	function <b>FindWakeUpSchedule( \$func)</b>
ts	static <b>FindWakeUpSchedule( func: string): YWakeupSchedule</b>
es	static <b>FindWakeUpSchedule( func)</b>
dnp	static YWakeupScheduleProxy <b>FindWakeUpSchedule( string func)</b>
cp	static YWakeupScheduleProxy * <b>FindWakeUpSchedule( string func)</b>

L'identifiant peut être spécifié sous plusieurs formes:

- NomLogiqueFonction
- NoSerieModule.IdentifiantFonction
- NoSerieModule.NomLogiqueFonction
- NomLogiqueModule.IdentifiantMatériel
- NomLogiqueModule.NomLogiqueFonction

Cette fonction n'exige pas que le réveil agendé soit en ligne au moment où elle est appelée, l'objet retourné sera néanmoins valide. Utiliser la méthode `YWakeupSchedule.isOnline()` pour tester si le réveil agendé est utilisable à un moment donné. En cas d'ambiguïté lorsqu'on fait une recherche par nom logique, aucune erreur ne sera notifiée: la première instance trouvée sera renvoyée. La recherche se fait d'abord par nom matériel, puis par nom logique.

Si un appel à la méthode `is_online()` de cet objet renvoie FAUX alors que vous êtes sûr que le module correspondant est bien branché, vérifiez que vous n'avez pas oublié d'appeler `registerHub()` à l'initialisation de de l'application.

### Paramètres :

**func** une chaîne de caractères qui référence le réveil agendé sans ambiguïté, par exemple `YHUBGSM3.wakeupSchedule1`.

### Retourne :

un objet de classe `YWakeupSchedule` qui permet ensuite de contrôler le réveil agendé.

## YWakeUpSchedule.FindWakeUpScheduleInContext() YWakeUpSchedule.FindWakeUpScheduleInContext()

## YWakeUpSchedule

Permet de retrouver un réveil agendé d'après un identifiant donné dans un Contexte YAPI.

```

java static YWakeUpSchedule FindWakeUpScheduleInContext( YAPIContext yctx,
                                                         String func)
uwp  static YWakeUpSchedule FindWakeUpScheduleInContext( YAPIContext yctx,
                                                         string func)
ts   static FindWakeUpScheduleInContext( yctx: YAPIContext, func: string): YWakeUpSchedule
es   static FindWakeUpScheduleInContext( yctx, func)

```

L'identifiant peut être spécifié sous plusieurs formes:

- NomLogiqueFonction
- NoSerieModule.IdentifiantFonction
- NoSerieModule.NomLogiqueFonction
- NomLogiqueModule.IdentifiantMatériel
- NomLogiqueModule.NomLogiqueFonction

Cette fonction n'exige pas que le réveil agendé soit en ligne au moment où elle est appelée, l'objet retourné sera néanmoins valide. Utiliser la méthode `YWakeUpSchedule.isOnline()` pour tester si le réveil agendé est utilisable à un moment donné. En cas d'ambiguïté lorsqu'on fait une recherche par nom logique, aucune erreur ne sera notifiée: la première instance trouvée sera renvoyée. La recherche se fait d'abord par nom matériel, puis par nom logique.

### Paramètres :

**yctx** un contexte YAPI

**func** une chaîne de caractères qui référence le réveil agendé sans ambiguïté, par exemple `YHUBGSM3.wakeUpSchedule1`.

### Retourne :

un objet de classe `YWakeUpSchedule` qui permet ensuite de contrôler le réveil agendé.

## YWakeUpSchedule.FirstWakeUpSchedule() YWakeupSchedule.FirstWakeUpSchedule()

## YWakeupSchedule

Commence l'énumération des réveils agendés accessibles par la librairie.

js	function <b>yFirstWakeUpSchedule</b> ( )
cpp	YWakeupSchedule * <b>FirstWakeUpSchedule</b> ( )
m	+(YWakeupSchedule*) <b>FirstWakeUpSchedule</b>
pas	TYWakeupSchedule <b>yFirstWakeUpSchedule</b> ( ): TYWakeupSchedule
vb	function <b>FirstWakeUpSchedule</b> ( ) As YWakeupSchedule
cs	static YWakeupSchedule <b>FirstWakeUpSchedule</b> ( )
java	static YWakeupSchedule <b>FirstWakeUpSchedule</b> ( )
uwp	static YWakeupSchedule <b>FirstWakeUpSchedule</b> ( )
py	<b>FirstWakeUpSchedule</b> ( )
php	function <b>FirstWakeUpSchedule</b> ( )
ts	static <b>FirstWakeUpSchedule</b> ( ): YWakeupSchedule   null
es	static <b>FirstWakeUpSchedule</b> ( )

Utiliser la fonction `YWakeupSchedule.nextWakeUpSchedule( )` pour itérer sur les autres réveils agendés.

### Retourne :

un pointeur sur un objet `YWakeupSchedule`, correspondant au premier réveil agendé accessible en ligne, ou `null` si il n'y a pas de réveils agendés disponibles.

## YWakeUpSchedule.FirstWakeUpScheduleInContext() YWakeUpSchedule.FirstWakeUpScheduleInContext()

## YWakeUpSchedule

Commence l'énumération des réveils agendés accessibles par la librairie.

```
java static YWakeUpSchedule FirstWakeUpScheduleInContext( YAPIContext yctx)
uwp static YWakeUpSchedule FirstWakeUpScheduleInContext( YAPIContext yctx)
ts static FirstWakeUpScheduleInContext( yctx: YAPIContext): YWakeUpSchedule | null
es static FirstWakeUpScheduleInContext( yctx)
```

Utiliser la fonction `YWakeUpSchedule.nextWakeUpSchedule()` pour itérer sur les autres réveils agendés.

### Paramètres :

`yctx` un contexte YAPI.

### Retourne :

un pointeur sur un objet `YWakeUpSchedule`, correspondant au premier réveil agendé accessible en ligne, ou `null` si il n'y a pas de réveils agendés disponibles.

## YWakeUpSchedule.GetSimilarFunctions() YWakeupSchedule.GetSimilarFunctions()

## YWakeupSchedule

Enumère toutes les fonctions de type WakeUpSchedule disponibles sur les modules actuellement joignables par la librairie, et retourne leurs identifiants matériels uniques (hardwareId).

```
dnv static new string[] GetSimilarFunctions( )
```

```
cp static vector<string> GetSimilarFunctions( )
```

Chaque chaîne retournée peut être passée en argument à la méthode YWakeUpSchedule.FindWakeUpSchedule pour obtenir un objet permettant d'interagir avec le module correspondant.

**Retourne :**

un tableau de chaînes de caractères, contenant les identifiants matériels de chaque fonction disponible trouvée.

**wakeupschedule**→**AdvertisedValue**

**YWakeUpSchedule**

---

Courte chaîne de caractères représentant l'état courant de la fonction.

`dnv` string **AdvertisedValue**

---

**wakeupschedule**→**FriendlyName****YWakeUpSchedule**

---

Identifiant global de la fonction au format NOM\_MODULE . NOM\_FONCTION.

`dnf` string **FriendlyName**

Le chaîne retournée utilise soit les noms logiques du module et de la fonction si ils sont définis, soit respectivement le numéro de série du module et l'identifiant matériel de la fonction (par exemple: `MyCustomName.relay1`)

**wakeupschedule** → **FunctionId**

**YWakeUpSchedule**

---

Identifiant matériel du réveil agendé, sans référence au module.

`dnsp` string **FunctionId**

Par exemple `relay1`.

---

**wakeupschedule** → **HardwareId****YWakeUpSchedule**

---

Identifiant matériel unique de la fonction au format `SERIAL.FUNCTIONID`.

`dnf` `string` **HardwareId**

L'identifiant unique est composé du numéro de série du module et de l'identifiant matériel de la fonction (par exemple `RELAYLO1-123456.relay1`).

## wakeupschedule→Hours

## YWakeUpSchedule

---

S heures où le réveil est actif..

`dnsp` `int Hours`

**Modifiable.** Modifie les heures où un réveil doit avoir lieu. N'oubliez pas d'appeler la méthode `saveToFlash()` du module si le réglage doit être préservé.

---

**wakeupschedule→IsOnline****YWakeUpSchedule**

---

Vérifie si le module hébergeant la fonction est joignable, sans déclencher d'erreur.

`dnf` `bool IsOnline`

Si les valeurs des attributs en cache de la fonction sont valides au moment de l'appel, le module est considéré joignable. Cette fonction ne cause en aucun cas d'exception, quelle que soit l'erreur qui pourrait se produire lors de la vérification de joignabilité.

wakeupschedule→LogicalName

YWakeUpSchedule

---

Nom logique de la fonction.

`dnf` string LogicalName

**Modifiable.** Vous pouvez utiliser `yCheckLogicalName()` pour vérifier si votre paramètre est valide. N'oubliez pas d'appeler la méthode `saveToFlash()` du module si le réglage doit être préservé.

---

**wakeupschedule** → **MinutesA****YWakeUpSchedule**

---

S minutes de l'intervall 00-29 de chaque heures où le réveil est actif.

`dnf` `int` **MinutesA**

**Modifiable.** Modifie les minutes de l'intervall 00-29 où un réveil doit avoir lieu. N'oubliez pas d'appeler la méthode `saveToFlash()` du module si le réglage doit être préservé.

## wakeupschedule → MinutesB

## YWakeUpSchedule

---

S minutes de l'interval 30-59 de chaque heure où le réveil est actif.

`dnf` `int MinutesB`

**Modifiable.** Modifie les minutes de l'interval 30-59 où un réveil doit avoir lieu. N'oubliez pas d'appeler la méthode `saveToFlash()` du module si le réglage doit être préservé.

---

**wakeupschedule** → **MonthDays****YWakeUpSchedule**

---

S jours du mois où le réveil est actif.

`dnf` `int` **MonthDays**

**Modifiable.** Modifie les jours du mois où un réveil doit avoir lieu. N'oubliez pas d'appeler la méthode `saveToFlash()` du module si le réglage doit être préservé.

## wakeupschedule→Months

## YWakeUpSchedule

---

S mois où le réveil est actif.

`dnsp` `int Months`

**Modifiable.** Modifie les mois où un réveil doit avoir lieu. N'oubliez pas d'appeler la méthode `saveToFlash()` du module si le réglage doit être préservé.

---

**wakeupschedule** → **NextOccurence**

**YWakeUpSchedule**

---

Date/heure de la prochaine occurrence de réveil.

`long` **NextOccurence**

**wakeupschedule** → **SerialNumber**

**YWakeUpSchedule**

---

Numéro de série du module, préprogrammé en usine.

`dnsp` string **SerialNumber**

---

**wakeupschedule** → **WeekDays****YWakeUpSchedule**

---

S jours de la semaine où le réveil est actif.

`dnf` `int WeekDays`

**Modifiable.** Modifie les jours de la semaine où un réveil doit avoir lieu. N'oubliez pas d'appeler la méthode `saveToFlash()` du module si le réglage doit être préservé.

## wakeupschedule→clearCache()

## YWakeUpSchedule

Invalide le cache.

js	function <b>clearCache</b> ( )
cpp	void <b>clearCache</b> ( )
m	-(void) <b>clearCache</b>
pas	<b>clearCache</b> ( )
vb	procedure <b>clearCache</b> ( )
cs	void <b>clearCache</b> ( )
java	void <b>clearCache</b> ( )
py	<b>clearCache</b> ( )
php	function <b>clearCache</b> ( )
ts	async <b>clearCache</b> ( ): Promise<void>
es	async <b>clearCache</b> ( )

Invalide le cache des valeurs courantes du réveil agendé. Force le prochain appel à une méthode `get_xxx()` ou `loadxxx()` pour charger les les données depuis le module.

**wakeupschedule→describe()****YWakeUpSchedule**

Retourne un court texte décrivant de manière non-ambigüe l'instance du réveil agendé au format `TYPE (NAME) =SERIAL.FUNCTIONID`.

js	function <b>describe</b> ( )
cpp	string <b>describe</b> ( )
m	-(NSString*) <b>describe</b>
pas	string <b>describe</b> ( ): string
vb	function <b>describe</b> ( ) As String
cs	string <b>describe</b> ( )
java	String <b>describe</b> ( )
py	<b>describe</b> ( )
php	function <b>describe</b> ( )
ts	async <b>describe</b> ( ): Promise<string>
es	async <b>describe</b> ( )

Plus précisément, `TYPE` correspond au type de fonction, `NAME` correspond au nom utilisé lors du premier accès à la fonction, `SERIAL` correspond au numéro de série du module si le module est connecté, ou "unresolved" sinon, et `FUNCTIONID` correspond à l'identifiant matériel de la fonction si le module est connecté. Par exemple, La méthode va retourner `Relay(MyCustomName.relay1)=RELAYLO1-123456.relay1` si le module est déjà connecté ou `Relay(BadCustomeName.relay1)=unresolved` si le module n'est pas déjà connecté. Cette méthode ne déclenche aucune transaction USB ou TCP et peut donc être utilisé dans un débogueur.

**Retourne :**

une chaîne de caractères décrivant le réveil agendé (ex:  
`Relay(MyCustomName.relay1)=RELAYLO1-123456.relay1`)

**wakeupschedule**→**get\_advertisedValue()****YWakeUpSchedule****wakeupschedule**→**advertisedValue()**

Retourne la valeur courante du réveil agendé (pas plus de 6 caractères).

js	function <b>get_advertisedValue</b> ( )
cpp	string <b>get_advertisedValue</b> ( )
m	-(NSString*) advertisedValue
pas	string <b>get_advertisedValue</b> ( ): string
vb	function <b>get_advertisedValue</b> ( ) As String
cs	string <b>get_advertisedValue</b> ( )
java	String <b>get_advertisedValue</b> ( )
uwp	async Task<string> <b>get_advertisedValue</b> ( )
py	<b>get_advertisedValue</b> ( )
php	function <b>get_advertisedValue</b> ( )
ts	async <b>get_advertisedValue</b> ( ): Promise<string>
es	async <b>get_advertisedValue</b> ( )
dnp	string <b>get_advertisedValue</b> ( )
cp	string <b>get_advertisedValue</b> ( )
cmd	YWakeUpSchedule <b>target</b> <b>get_advertisedValue</b>

**Retourne :**

une chaîne de caractères représentant la valeur courante du réveil agendé (pas plus de 6 caractères).

En cas d'erreur, déclenche une exception ou retourne `YWakeUpSchedule.ADVERTISEDVALUE_INVALID`.

**wakeupschedule**→**get\_errorMessage()****YWakeUpSchedule****wakeupschedule**→**errorMessage()**

Retourne le message correspondant à la dernière erreur survenue lors de l'utilisation du réveil agendé.

js	function <b>get_errorMessage</b> ( )
cpp	string <b>get_errorMessage</b> ( )
m	-(NSString*) <b>errorMessage</b>
pas	string <b>get_errorMessage</b> ( ): string
vb	function <b>get_errorMessage</b> ( ) As String
cs	string <b>get_errorMessage</b> ( )
java	String <b>get_errorMessage</b> ( )
py	<b>get_errorMessage</b> ( )
php	function <b>get_errorMessage</b> ( )
ts	<b>get_errorMessage</b> ( ): string
es	<b>get_errorMessage</b> ( )

Cette méthode est principalement utile lorsque la librairie Yoctopuce est utilisée en désactivant la gestion des exceptions.

**Retourne :**

une chaîne de caractères correspondant au message de la dernière erreur qui s'est produit lors de l'utilisation du réveil agendé.

**wakeupschedule**→**get\_errorType()****YWakeUpSchedule****wakeupschedule**→**errorType()**

Retourne le code d'erreur correspondant à la dernière erreur survenue lors de l'utilisation du réveil agendé.

js	function <b>get_errorType</b> ( )
cpp	YRETCODE <b>get_errorType</b> ( )
m	-(YRETCODE) errorType
pas	YRETCODE <b>get_errorType</b> ( ): YRETCODE
vb	function <b>get_errorType</b> ( ) As YRETCODE
cs	YRETCODE <b>get_errorType</b> ( )
java	int <b>get_errorType</b> ( )
py	<b>get_errorType</b> ( )
php	function <b>get_errorType</b> ( )
ts	<b>get_errorType</b> ( ): number
es	<b>get_errorType</b> ( )

Cette méthode est principalement utile lorsque la librairie Yoctopuce est utilisée en désactivant la gestion des exceptions.

**Retourne :**

un nombre correspondant au code de la dernière erreur qui s'est produit lors de l'utilisation du réveil agendé.

**wakeupschedule**→**get\_friendlyName()****YWakeUpSchedule****wakeupschedule**→**friendlyName()**

Retourne un identifiant global du réveil agendé au format `NOM_MODULE . NOM_FONCTION`.

js	function <b>get_friendlyName</b> ( )
cpp	string <b>get_friendlyName</b> ( )
m	-(NSString*) friendlyName
cs	string <b>get_friendlyName</b> ( )
java	String <b>get_friendlyName</b> ( )
py	<b>get_friendlyName</b> ( )
php	function <b>get_friendlyName</b> ( )
ts	async <b>get_friendlyName</b> ( ): Promise<string>
es	async <b>get_friendlyName</b> ( )
dnp	string <b>get_friendlyName</b> ( )
cp	string <b>get_friendlyName</b> ( )

Le chaîne retournée utilise soit les noms logiques du module et du réveil agendé si ils sont définis, soit respectivement le numéro de série du module et l'identifiant matériel du réveil agendé (par exemple: `MyCustomName.relay1`)

**Retourne :**

une chaîne de caractères identifiant le réveil agendé en utilisant les noms logiques (ex: `MyCustomName.relay1`)

En cas d'erreur, déclenche une exception ou retourne `YWakeUpSchedule.FRIENDLYNAME_INVALID`.

**wakeupschedule**→**get\_functionDescriptor()****YWakeUpSchedule****wakeupschedule**→**functionDescriptor()**

Retourne un identifiant unique de type YFUN\_DESCR correspondant à la fonction.

js	function <b>get_functionDescriptor()</b>
cpp	YFUN_DESCR <b>get_functionDescriptor()</b>
m	-(YFUN_DESCR) <b>functionDescriptor</b>
pas	YFUN_DESCR <b>get_functionDescriptor()</b> : YFUN_DESCR
vb	function <b>get_functionDescriptor()</b> As YFUN_DESCR
cs	YFUN_DESCR <b>get_functionDescriptor()</b>
java	String <b>get_functionDescriptor()</b>
py	<b>get_functionDescriptor()</b>
php	function <b>get_functionDescriptor()</b>
ts	async <b>get_functionDescriptor()</b> : Promise<string>
es	async <b>get_functionDescriptor()</b>

Cet identifiant peut être utilisé pour tester si deux instance de YFunction référencent physiquement la même fonction sur le même module.

**Retourne :**

un identifiant de type YFUN\_DESCR.

Si la fonction n'a jamais été contactée, la valeur retournée sera Y\$CLASSNAME\$.FUNCTIONDESCRIPTOR\_INVALID

**wakeupschedule**→**get\_functionId()****YWakeUpSchedule****wakeupschedule**→**functionId()**

Retourne l'identifiant matériel du réveil agendé, sans référence au module.

js	function <b>get_functionId</b> ( )
cpp	string <b>get_functionId</b> ( )
m	-(NSString*) <b>functionId</b>
vb	function <b>get_functionId</b> ( ) As String
cs	string <b>get_functionId</b> ( )
java	String <b>get_functionId</b> ( )
py	<b>get_functionId</b> ( )
php	function <b>get_functionId</b> ( )
ts	async <b>get_functionId</b> ( ): Promise<string>
es	async <b>get_functionId</b> ( )
dnp	string <b>get_functionId</b> ( )
cp	string <b>get_functionId</b> ( )

Par exemple `relay1`.

**Retourne :**

une chaîne de caractères identifiant le réveil agendé (ex: `relay1`)

En cas d'erreur, déclenche une exception ou retourne `YWakeUpSchedule.FUNCTIONID_INVALID`.

**wakeupschedule**→**get\_hardwareId()****YWakeUpSchedule****wakeupschedule**→**hardwareId()**

Retourne l'identifiant matériel unique du réveil agendé au format `SERIAL.FUNCTIONID`.

js	function <b>get_hardwareId</b> ( )
cpp	string <b>get_hardwareId</b> ( )
m	-(NSString*) <b>hardwareId</b>
vb	function <b>get_hardwareId</b> ( ) As String
cs	string <b>get_hardwareId</b> ( )
java	String <b>get_hardwareId</b> ( )
py	<b>get_hardwareId</b> ( )
php	function <b>get_hardwareId</b> ( )
ts	async <b>get_hardwareId</b> ( ): Promise<string>
es	async <b>get_hardwareId</b> ( )
dnp	string <b>get_hardwareId</b> ( )
cp	string <b>get_hardwareId</b> ( )

L'identifiant unique est composé du numéro de série du module et de l'identifiant matériel du réveil agendé (par exemple `RELAYLO1-123456.relay1`).

**Retourne :**

une chaîne de caractères identifiant le réveil agendé (ex: `RELAYLO1-123456.relay1`)

En cas d'erreur, déclenche une exception ou retourne `YWakeUpSchedule.HARDWAREID_INVALID`.

**wakeupschedule**→**get\_hours()****YWakeUpSchedule****wakeupschedule**→**hours()**

Retourne les heures où le réveil est actif..

js	function <b>get_hours</b> ( )
cpp	int <b>get_hours</b> ( )
m	-(int) hours
pas	LongInt <b>get_hours</b> ( ): LongInt
vb	function <b>get_hours</b> ( ) As Integer
cs	int <b>get_hours</b> ( )
java	int <b>get_hours</b> ( )
uwp	async Task<int> <b>get_hours</b> ( )
py	<b>get_hours</b> ( )
php	function <b>get_hours</b> ( )
ts	async <b>get_hours</b> ( ): Promise<number>
es	async <b>get_hours</b> ( )
dnp	int <b>get_hours</b> ( )
cp	int <b>get_hours</b> ( )
cmd	YWakeUpSchedule <b>target</b> <b>get_hours</b>

**Retourne :**

un entier représentant les heures où le réveil est actif

En cas d'erreur, déclenche une exception ou retourne `YWakeUpSchedule.HOURS_INVALID`.

**wakeupschedule**→**get\_logicalName()****YWakeUpSchedule****wakeupschedule**→**logicalName()**

Retourne le nom logique du réveil agendé.

js	function <b>get_logicalName</b> ( )
cpp	string <b>get_logicalName</b> ( )
m	-(NSString*) logicalName
pas	string <b>get_logicalName</b> ( ): string
vb	function <b>get_logicalName</b> ( ) As String
cs	string <b>get_logicalName</b> ( )
java	String <b>get_logicalName</b> ( )
uwp	async Task<string> <b>get_logicalName</b> ( )
py	<b>get_logicalName</b> ( )
php	function <b>get_logicalName</b> ( )
ts	async <b>get_logicalName</b> ( ): Promise<string>
es	async <b>get_logicalName</b> ( )
dnp	string <b>get_logicalName</b> ( )
cp	string <b>get_logicalName</b> ( )
cmd	YWakeUpSchedule <b>target</b> <b>get_logicalName</b>

**Retourne :**

une chaîne de caractères représentant le nom logique du réveil agendé.

En cas d'erreur, déclenche une exception ou retourne `YWakeUpSchedule.LOGICALNAME_INVALID`.

**wakeupschedule**→**get\_minutes()****YWakeUpSchedule****wakeupschedule**→**minutes()**

Retourne toutes les minutes de chaque heure où le réveil est actif.

js	function <b>get_minutes</b> ( )
cpp	s64 <b>get_minutes</b> ( )
m	-(s64) minutes
pas	int64 <b>get_minutes</b> ( ): int64
vb	function <b>get_minutes</b> ( ) As Long
cs	long <b>get_minutes</b> ( )
java	long <b>get_minutes</b> ( )
uwp	async Task<long> <b>get_minutes</b> ( )
py	<b>get_minutes</b> ( )
php	function <b>get_minutes</b> ( )
ts	async <b>get_minutes</b> ( ): Promise<number>
es	async <b>get_minutes</b> ( )
dnp	long <b>get_minutes</b> ( )
cp	s64 <b>get_minutes</b> ( )
cmd	YWakeUpSchedule <b>target</b> <b>get_minutes</b>

**wakeupschedule**→**get\_minutesA()****YWakeUpSchedule****wakeupschedule**→**minutesA()**

Retourne les minutes de l'intervall 00-29 de chaque heures où le réveil est actif.

js	function <b>get_minutesA</b> ( )
cpp	int <b>get_minutesA</b> ( )
m	-(int) minutesA
pas	LongInt <b>get_minutesA</b> ( ): LongInt
vb	function <b>get_minutesA</b> ( ) As Integer
cs	int <b>get_minutesA</b> ( )
java	int <b>get_minutesA</b> ( )
uwp	async Task<int> <b>get_minutesA</b> ( )
py	<b>get_minutesA</b> ( )
php	function <b>get_minutesA</b> ( )
ts	async <b>get_minutesA</b> ( ): Promise<number>
es	async <b>get_minutesA</b> ( )
dnp	int <b>get_minutesA</b> ( )
cp	int <b>get_minutesA</b> ( )
cmd	YWakeUpSchedule <b>target</b> <b>get_minutesA</b>

**Retourne :**

un entier représentant les minutes de l'intervall 00-29 de chaque heures où le réveil est actif

En cas d'erreur, déclenche une exception ou retourne `YWakeUpSchedule.MINUTESA_INVALID`.

**wakeupschedule**→**get\_minutesB()****YWakeUpSchedule****wakeupschedule**→**minutesB()**

Retourne les minutes de l'intervall 30-59 de chaque heure où le réveil est actif.

js	function <b>get_minutesB</b> ( )
cpp	int <b>get_minutesB</b> ( )
m	-(int) minutesB
pas	LongInt <b>get_minutesB</b> ( ): LongInt
vb	function <b>get_minutesB</b> ( ) As Integer
cs	int <b>get_minutesB</b> ( )
java	int <b>get_minutesB</b> ( )
uwp	async Task<int> <b>get_minutesB</b> ( )
py	<b>get_minutesB</b> ( )
php	function <b>get_minutesB</b> ( )
ts	async <b>get_minutesB</b> ( ): Promise<number>
es	async <b>get_minutesB</b> ( )
dnp	int <b>get_minutesB</b> ( )
cp	int <b>get_minutesB</b> ( )
cmd	YWakeUpSchedule <b>target</b> <b>get_minutesB</b>

**Retourne :**

un entier représentant les minutes de l'intervall 30-59 de chaque heure où le réveil est actif

En cas d'erreur, déclenche une exception ou retourne `YWakeUpSchedule.MINUTESB_INVALID`.

**wakeupschedule**→**get\_module()****wakeupschedule**→**module()**

Retourne l'objet `YModule` correspondant au module Yoctopuce qui héberge la fonction.

js	function <b>get_module</b> ( )
cpp	<code>YModule *</code> <b>get_module</b> ( )
m	-( <code>YModule*</code> ) <b>module</b>
pas	<code>TYModule</code> <b>get_module</b> ( ): <code>TYModule</code>
vb	function <b>get_module</b> ( ) As <code>YModule</code>
cs	<code>YModule</code> <b>get_module</b> ( )
java	<code>YModule</code> <b>get_module</b> ( )
py	<b>get_module</b> ( )
php	function <b>get_module</b> ( )
ts	async <b>get_module</b> ( ): <code>Promise&lt;YModule&gt;</code>
es	async <b>get_module</b> ( )
dnp	<code>YModuleProxy</code> <b>get_module</b> ( )
cp	<code>YModuleProxy *</code> <b>get_module</b> ( )

Si la fonction ne peut être trouvée sur aucun module, l'instance de `YModule` retournée ne sera pas joignable.

**Retourne :**

une instance de `YModule`

---

**wakeupschedule**→**get\_module\_async()****YWakeUpSchedule****wakeupschedule**→**module\_async()**

---

Retourne l'objet `YModule` correspondant au module Yoctopuce qui héberge la fonction.

```
js function get_module_async( callback, context)
```

Si la fonction ne peut être trouvée sur aucun module, l'instance de `YModule` retournée ne sera pas joignable.

Cette version asynchrone n'existe qu'en Javascript. Elle utilise une fonction de callback plutôt qu'une simple valeur de retour, pour éviter de bloquer la VM Javascript de Firefox, qui n'implémente pas le passage de contrôle entre threads durant les appels d'entrée/sortie bloquants.

**Paramètres :**

**callback** fonction de callback qui sera appelée dès que le résultat sera connu. La fonction callback reçoit trois arguments: le contexte fourni par l'appelant, l'objet fonction concerné et l'instance demandée de `YModule`

**context** contexte fourni par l'appelant, et qui sera passé tel-quel à la fonction de callback

**Retourne :**

rien du tout : le résultat sera passé en paramètre à la fonction de callback.

**wakeupschedule**→**get\_monthDays()****YWakeUpSchedule****wakeupschedule**→**monthDays()**

Retourne les jours du mois où le réveil est actif.

js	function <b>get_monthDays</b> ( )
cpp	int <b>get_monthDays</b> ( )
m	-(int) monthDays
pas	LongInt <b>get_monthDays</b> ( ): LongInt
vb	function <b>get_monthDays</b> ( ) As Integer
cs	int <b>get_monthDays</b> ( )
java	int <b>get_monthDays</b> ( )
uwp	async Task<int> <b>get_monthDays</b> ( )
py	<b>get_monthDays</b> ( )
php	function <b>get_monthDays</b> ( )
ts	async <b>get_monthDays</b> ( ): Promise<number>
es	async <b>get_monthDays</b> ( )
dnp	int <b>get_monthDays</b> ( )
cp	int <b>get_monthDays</b> ( )
cmd	YWakeUpSchedule <b>target</b> <b>get_monthDays</b>

**Retourne :**

un entier représentant les jours du mois où le réveil est actif

En cas d'erreur, déclenche une exception ou retourne `YWakeUpSchedule.MONTHDAYS_INVALID`.

**wakeupschedule**→**get\_months()****YWakeUpSchedule****wakeupschedule**→**months()**

Retourne les mois où le réveil est actif.

js	function <b>get_months</b> ( )
cpp	int <b>get_months</b> ( )
m	-(int) months
pas	LongInt <b>get_months</b> ( ): LongInt
vb	function <b>get_months</b> ( ) As Integer
cs	int <b>get_months</b> ( )
java	int <b>get_months</b> ( )
uwp	async Task<int> <b>get_months</b> ( )
py	<b>get_months</b> ( )
php	function <b>get_months</b> ( )
ts	async <b>get_months</b> ( ): Promise<number>
es	async <b>get_months</b> ( )
dnp	int <b>get_months</b> ( )
cp	int <b>get_months</b> ( )
cmd	YWakeUpSchedule <b>target</b> <b>get_months</b>

**Retourne :**

un entier représentant les mois où le réveil est actif

En cas d'erreur, déclenche une exception ou retourne `YWakeUpSchedule.MONTHS_INVALID`.

wakeupschedule→get\_nextOccurence()

YWakeUpSchedule

wakeupschedule→nextOccurence()

Retourne la date/heure de la prochaine occurrence de réveil.

js	function <b>get_nextOccurence</b> ( )
cpp	s64 <b>get_nextOccurence</b> ( )
m	-(s64) nextOccurence
pas	int64 <b>get_nextOccurence</b> ( ): int64
vb	function <b>get_nextOccurence</b> ( ) As Long
cs	long <b>get_nextOccurence</b> ( )
java	long <b>get_nextOccurence</b> ( )
uwp	async Task<long> <b>get_nextOccurence</b> ( )
py	<b>get_nextOccurence</b> ( )
php	function <b>get_nextOccurence</b> ( )
ts	async <b>get_nextOccurence</b> ( ): Promise<number>
es	async <b>get_nextOccurence</b> ( )
dnp	long <b>get_nextOccurence</b> ( )
cp	s64 <b>get_nextOccurence</b> ( )
cmd	YWakeUpSchedule <b>target</b> <b>get_nextOccurence</b>

**Retourne :**

un entier représentant la date/heure de la prochaine occurrence de réveil

En cas d'erreur, déclenche une exception ou retourne `YWakeUpSchedule.NEXTOCCURENCE_INVALID`.

**wakeupschedule**→**get\_serialNumber()****YWakeUpSchedule****wakeupschedule**→**serialNumber()**

Retourne le numéro de série du module, préprogrammé en usine.

js	function <b>get_serialNumber</b> ( )
cpp	string <b>get_serialNumber</b> ( )
m	-(NSString*) <b>serialNumber</b>
pas	string <b>get_serialNumber</b> ( ): string
vb	function <b>get_serialNumber</b> ( ) As String
cs	string <b>get_serialNumber</b> ( )
java	String <b>get_serialNumber</b> ( )
uwp	async Task<string> <b>get_serialNumber</b> ( )
py	<b>get_serialNumber</b> ( )
php	function <b>get_serialNumber</b> ( )
ts	async <b>get_serialNumber</b> ( ): Promise<string>
es	async <b>get_serialNumber</b> ( )
dnp	string <b>get_serialNumber</b> ( )
cp	string <b>get_serialNumber</b> ( )
cmd	YWakeUpSchedule <b>target</b> <b>get_serialNumber</b>

**Retourne :**

: une chaîne de caractères représentant le numéro de série du module, préprogrammé en usine.

En cas d'erreur, déclenche une exception ou retourne YFunction.SERIALNUMBER\_INVALID.

**wakeupschedule**→**get\_userData()****YWakeUpSchedule****wakeupschedule**→**userData()**

Retourne le contenu de l'attribut `userData`, précédemment stocké à l'aide de la méthode `set_userData`.

js	function <b>get_userData</b> ( )
cpp	void * <b>get_userData</b> ( )
m	-(id) <code>userData</code>
pas	Tobject <b>get_userData</b> ( ): Tobject
vb	function <b>get_userData</b> ( ) As Object
cs	object <b>get_userData</b> ( )
java	Object <b>get_userData</b> ( )
py	<b>get_userData</b> ( )
php	function <b>get_userData</b> ( )
ts	async <b>get_userData</b> ( ): Promise<object null>
es	async <b>get_userData</b> ( )

Cet attribut n'est pas utilisé directement par l'API. Il est à la disposition de l'appelant pour stocker un contexte.

**Retourne :**

l'objet stocké précédemment par l'appelant.

**wakeupschedule**→**get\_weekDays()****YWakeUpSchedule****wakeupschedule**→**weekDays()**

Retourne les jours de la semaine où le réveil est actif.

js	function <b>get_weekDays</b> ( )
cpp	int <b>get_weekDays</b> ( )
m	-(int) weekDays
pas	LongInt <b>get_weekDays</b> ( ): LongInt
vb	function <b>get_weekDays</b> ( ) As Integer
cs	int <b>get_weekDays</b> ( )
java	int <b>get_weekDays</b> ( )
uwp	async Task<int> <b>get_weekDays</b> ( )
py	<b>get_weekDays</b> ( )
php	function <b>get_weekDays</b> ( )
ts	async <b>get_weekDays</b> ( ): Promise<number>
es	async <b>get_weekDays</b> ( )
dnp	int <b>get_weekDays</b> ( )
cp	int <b>get_weekDays</b> ( )
cmd	YWakeUpSchedule <b>target</b> <b>get_weekDays</b>

**Retourne :**

un entier représentant les jours de la semaine où le réveil est actif

En cas d'erreur, déclenche une exception ou retourne `YWakeUpSchedule.WEEKDAYS_INVALID`.

## wakeupschedule→isOnline()

## YWakeUpSchedule

Vérifie si le module hébergeant le réveil agendé est joignable, sans déclencher d'erreur.

js	function <b>isOnline</b> ( )
cpp	bool <b>isOnline</b> ( )
m	-(BOOL) <b>isOnline</b>
pas	boolean <b>isOnline</b> ( ): boolean
vb	function <b>isOnline</b> ( ) As Boolean
cs	bool <b>isOnline</b> ( )
java	boolean <b>isOnline</b> ( )
py	<b>isOnline</b> ( )
php	function <b>isOnline</b> ( )
ts	async <b>isOnline</b> ( ): Promise<boolean>
es	async <b>isOnline</b> ( )
dnp	bool <b>isOnline</b> ( )
cp	bool <b>isOnline</b> ( )

Si les valeurs des attributs en cache du réveil agendé sont valides au moment de l'appel, le module est considéré joignable. Cette fonction ne cause en aucun cas d'exception, quelle que soit l'erreur qui pourrait se produire lors de la vérification de joignabilité.

**Retourne :**

true si le réveil agendé est joignable, false sinon

**wakeupschedule**→**isOnline\_async()****YWakeUpSchedule**

Vérifie si le module hébergeant le réveil agendé est joignable, sans déclencher d'erreur.

```
js fonction isOnline_async( callback, context)
```

Si les valeurs des attributs en cache du réveil agendé sont valides au moment de l'appel, le module est considéré joignable. Cette fonction ne cause en aucun cas d'exception, quelle que soit l'erreur qui pourrait se produire lors de la vérification de joignabilité.

Cette version asynchrone n'existe qu'en Javascript. Elle utilise une fonction de callback plutôt qu'une simple valeur de retour, pour éviter de bloquer la machine virtuelle Javascript avec une attente active.

**Paramètres :**

**callback** fonction de callback qui sera appelée dès que le résultat sera connu. La fonction callback reçoit trois arguments: le contexte fourni par l'appelant, l'objet fonction concerné et le résultat booléen

**context** contexte fourni par l'appelant, et qui sera passé tel-quel à la fonction de callback

**Retourne :**

rien du tout : le résultat sera passé en paramètre à la fonction de callback.

## wakeupschedule→isReadOnly()

## YWakeUpSchedule

Test si la fonction est en lecture seule.

cpp	bool <b>isReadOnly</b> ( )
m	-(bool) <b>isReadOnly</b>
pas	boolean <b>isReadOnly</b> ( ): boolean
vb	function <b>isReadOnly</b> ( ) As Boolean
cs	bool <b>isReadOnly</b> ( )
java	boolean <b>isReadOnly</b> ( )
uwp	async Task<bool> <b>isReadOnly</b> ( )
py	<b>isReadOnly</b> ( )
php	function <b>isReadOnly</b> ( )
ts	async <b>isReadOnly</b> ( ): Promise<boolean>
es	async <b>isReadOnly</b> ( )
dnp	bool <b>isReadOnly</b> ( )
cp	bool <b>isReadOnly</b> ( )
cmd	YWakeUpSchedule <b>target isReadOnly</b>

Retourne vrai si la fonction est protégé en écriture ou que la fonction n'est pas disponible.

**Retourne :**

true si la fonction est protégé en écriture ou que la fonction n'est pas disponible

**wakeupschedule→load()****YWakeUpSchedule**

Met en cache les valeurs courantes du réveil agendé, avec une durée de validité spécifiée.

js	function <b>load</b> ( <b>msValidity</b> )
cpp	YRETCODE <b>load</b> ( int <b>msValidity</b> )
m	-(YRETCODE) <b>load</b> : (u64) <b>msValidity</b>
pas	YRETCODE <b>load</b> ( <b>msValidity</b> : u64): YRETCODE
vb	function <b>load</b> ( ByVal <b>msValidity</b> As Long) As YRETCODE
cs	YRETCODE <b>load</b> ( ulong <b>msValidity</b> )
java	int <b>load</b> ( long <b>msValidity</b> )
py	<b>load</b> ( <b>msValidity</b> )
php	function <b>load</b> ( <b>\$msValidity</b> )
ts	async <b>load</b> ( <b>msValidity</b> : number): Promise<number>
es	async <b>load</b> ( <b>msValidity</b> )

Par défaut, lorsqu'on accède à un module, tous les attributs des fonctions du module sont automatiquement mises en cache pour la durée standard (5 ms). Cette méthode peut être utilisée pour marquer occasionnellement les données cachées comme valides pour une plus longue période, par exemple dans le but de réduire le trafic réseau.

**Paramètres :**

**msValidity** un entier correspondant à la durée de validité attribuée aux les paramètres chargés, en millisecondes

**Retourne :**

YAPI . SUCCESS si l'opération se déroule sans erreur.

En cas d'erreur, déclenche une exception ou retourne un code d'erreur négatif.

## wakeupschedule→loadAttribute()

## YWakeUpSchedule

Retourne la valeur actuelle d'un attribut spécifique de la fonction, sous forme de texte, le plus rapidement possible mais sans passer par le cache.

js	function <b>loadAttribute</b> ( <b>attrName</b> )
cpp	string <b>loadAttribute</b> ( string <b>attrName</b> )
m	-(NSString*) <b>loadAttribute</b> : (NSString*) <b>attrName</b>
pas	string <b>loadAttribute</b> ( <b>attrName</b> : string): string
vb	function <b>loadAttribute</b> ( ByVal <b>attrName</b> As String) As String
cs	string <b>loadAttribute</b> ( string <b>attrName</b> )
java	String <b>loadAttribute</b> ( String <b>attrName</b> )
uwp	async Task<string> <b>loadAttribute</b> ( string <b>attrName</b> )
py	<b>loadAttribute</b> ( <b>attrName</b> )
php	function <b>loadAttribute</b> ( <b>\$attrName</b> )
ts	async <b>loadAttribute</b> ( <b>attrName</b> : string): Promise<string>
es	async <b>loadAttribute</b> ( <b>attrName</b> )
dnp	string <b>loadAttribute</b> ( string <b>attrName</b> )
cp	string <b>loadAttribute</b> ( string <b>attrName</b> )

**Paramètres :**

**attrName** le nom de l'attribut désiré

**Retourne :**

une chaîne de caractères représentant la valeur actuelle de l'attribut.

En cas d'erreur, déclenche une exception ou retourne un chaîne vide.

**wakeupschedule→load\_async()****YWakeUpSchedule**

Met en cache les valeurs courantes du réveil agendé, avec une durée de validité spécifiée.

```
js function load_async( msValidity, callback, context)
```

Par défaut, lorsqu'on accède à un module, tous les attributs des fonctions du module sont automatiquement mises en cache pour la durée standard (5 ms). Cette méthode peut être utilisée pour marquer occasionnellement les données cachées comme valides pour une plus longue période, par exemple dans le but de réduire le trafic réseau.

Cette version asynchrone n'existe qu'en Javascript. Elle utilise une fonction de callback plutôt qu'une simple valeur de retour, pour éviter de bloquer la machine virtuelle Javascript avec une attente active.

**Paramètres :**

- msValidity** un entier correspondant à la durée de validité attribuée aux les paramètres chargés, en millisecondes
- callback** fonction de callback qui sera appelée dès que le résultat sera connu. La fonction callback reçoit trois arguments: le contexte fourni par l'appelant, l'objet fonction concerné et le code d'erreur (ou `YAPI . SUCCESS`)
- context** contexte fourni par l'appelant, et qui sera passé tel-quel à la fonction de callback

**Retourne :**

rien du tout : le résultat sera passé en paramètre à la fonction de callback.

**wakeupschedule**→**muteValueCallbacks()****YWakeUpSchedule**

Désactive l'envoi de chaque changement de la valeur publiée au hub parent.

js	function <b>muteValueCallbacks</b> ( )
cpp	int <b>muteValueCallbacks</b> ( )
m	-(int) <b>muteValueCallbacks</b>
pas	LongInt <b>muteValueCallbacks</b> ( ): LongInt
vb	function <b>muteValueCallbacks</b> ( ) As Integer
cs	int <b>muteValueCallbacks</b> ( )
java	int <b>muteValueCallbacks</b> ( )
uwp	async Task<int> <b>muteValueCallbacks</b> ( )
py	<b>muteValueCallbacks</b> ( )
php	function <b>muteValueCallbacks</b> ( )
ts	async <b>muteValueCallbacks</b> ( ): Promise<number>
es	async <b>muteValueCallbacks</b> ( )
dnp	int <b>muteValueCallbacks</b> ( )
cp	int <b>muteValueCallbacks</b> ( )
cmd	YWakeUpSchedule <b>target</b> <b>muteValueCallbacks</b>

Vous pouvez utiliser cette fonction pour économiser la bande passante et le CPU sur les machines de faible puissance, ou pour éviter le déclenchement de callbacks HTTP. N'oubliez pas d'appeler la méthode `saveToFlash()` du module si le réglage doit être préservé.

**Retourne :**

YAPI . SUCCESS si l'opération se déroule sans erreur.

En cas d'erreur, déclenche une exception ou retourne un code d'erreur négatif.

**wakeupschedule**→**nextWakeUpSchedule()****YWakeUpSchedule**

Continue l'énumération des réveils agendés commencée à l'aide de `yFirstWakeUpSchedule()`. Attention, vous ne pouvez faire aucune supposition sur l'ordre dans lequel les réveils agendés sont retournés.

js	function <b>nextWakeUpSchedule()</b>
cpp	YWakeupSchedule * <b>nextWakeUpSchedule()</b>
m	-(nullable YWakeupSchedule*) <b>nextWakeUpSchedule</b>
pas	TYWakeupSchedule <b>nextWakeUpSchedule()</b> : TYWakeupSchedule
vb	function <b>nextWakeUpSchedule()</b> As YWakeupSchedule
cs	YWakeupSchedule <b>nextWakeUpSchedule()</b>
java	YWakeupSchedule <b>nextWakeUpSchedule()</b>
uwp	YWakeupSchedule <b>nextWakeUpSchedule()</b>
py	<b>nextWakeUpSchedule()</b>
php	function <b>nextWakeUpSchedule()</b>
ts	<b>nextWakeUpSchedule()</b> : YWakeupSchedule   null
es	<b>nextWakeUpSchedule()</b>

Si vous souhaitez retrouver un réveil agendé spécifique, utilisez `WakeUpSchedule.findWakeUpSchedule()` avec un `hardwareID` ou un nom logique.

**Retourne :**

un pointeur sur un objet `YWakeupSchedule` accessible en ligne, ou `null` lorsque l'énumération est terminée.

## wakeupschedule→registerValueCallback()

## YWakeUpSchedule

Enregistre la fonction de callback qui est appelée à chaque changement de la valeur publiée.

js	function <b>registerValueCallback</b> ( <b>callback</b> )
cpp	int <b>registerValueCallback</b> ( YWakeUpScheduleValueCallback <b>callback</b> )
m	-(int) <b>registerValueCallback</b> : (YWakeUpScheduleValueCallback _Nullable) <b>callback</b>
pas	LongInt <b>registerValueCallback</b> ( <b>callback</b> : TYWakeUpScheduleValueCallback): LongInt
vb	function <b>registerValueCallback</b> ( ByVal <b>callback</b> As YWakeUpScheduleValueCallback) As Integer
cs	int <b>registerValueCallback</b> ( ValueCallback <b>callback</b> )
java	int <b>registerValueCallback</b> ( UpdateCallback <b>callback</b> )
uwp	async Task<int> <b>registerValueCallback</b> ( ValueCallback <b>callback</b> )
py	<b>registerValueCallback</b> ( <b>callback</b> )
php	function <b>registerValueCallback</b> ( <b>\$callback</b> )
ts	async <b>registerValueCallback</b> ( <b>callback</b> : YWakeUpScheduleValueCallback   null): Promise<number>
es	async <b>registerValueCallback</b> ( <b>callback</b> )

Ce callback n'est appelé que durant l'exécution de `ySleep` ou `yHandleEvents`. Cela permet à l'appelant de contrôler quand les callback peuvent se produire. Il est important d'appeler l'une de ces deux fonctions périodiquement pour garantir que les callback ne soient pas appelés trop tard. Pour désactiver un callback, il suffit d'appeler cette méthode en lui passant un pointeur nul.

**Paramètres :**

**callback** la fonction de callback à rappeler, ou un pointeur nul. La fonction de callback doit accepter deux arguments: l'object fonction dont la valeur a changé, et la chaîne de caractère décrivant la nouvelle valeur publiée.

wakeupschedule→set\_hours()

YWakeUpSchedule

wakeupschedule→setHours()

Modifie les heures où un réveil doit avoir lieu.

js	function <b>set_hours</b> ( <b>newval</b> )
cpp	int <b>set_hours</b> ( int <b>newval</b> )
m	-(int) setHours : (int) <b>newval</b>
pas	integer <b>set_hours</b> ( <b>newval</b> : LongInt): integer
vb	function <b>set_hours</b> ( ByVal <b>newval</b> As Integer) As Integer
cs	int <b>set_hours</b> ( int <b>newval</b> )
java	int <b>set_hours</b> ( int <b>newval</b> )
uwp	async Task<int> <b>set_hours</b> ( int <b>newval</b> )
py	<b>set_hours</b> ( <b>newval</b> )
php	function <b>set_hours</b> ( <b>\$newval</b> )
ts	async <b>set_hours</b> ( <b>newval</b> : number): Promise<number>
es	async <b>set_hours</b> ( <b>newval</b> )
dnp	int <b>set_hours</b> ( int <b>newval</b> )
cp	int <b>set_hours</b> ( int <b>newval</b> )
cmd	YWakeUpSchedule <b>target set_hours newval</b>

N'oubliez pas d'appeler la méthode `saveToFlash()` du module si le réglage doit être préservé.

#### Paramètres :

**newval** un entier représentant les heures où un réveil doit avoir lieu

#### Retourne :

YAPI . SUCCESS si l'opération se déroule sans erreur.

En cas d'erreur, déclenche une exception ou retourne un code d'erreur négatif.

wakeupschedule→set\_logicalName()

YWakeUpSchedule

wakeupschedule→setLogicalName()

Modifie le nom logique du réveil agendé.

js	function <b>set_logicalName</b> ( <b>newval</b> )
cpp	int <b>set_logicalName</b> ( string <b>newval</b> )
m	-(int) setLogicalName : (NSString*) <b>newval</b>
pas	integer <b>set_logicalName</b> ( <b>newval</b> : string): integer
vb	function <b>set_logicalName</b> ( ByVal <b>newval</b> As String) As Integer
cs	int <b>set_logicalName</b> ( string <b>newval</b> )
java	int <b>set_logicalName</b> ( String <b>newval</b> )
uwp	async Task<int> <b>set_logicalName</b> ( string <b>newval</b> )
py	<b>set_logicalName</b> ( <b>newval</b> )
php	function <b>set_logicalName</b> ( <b>\$newval</b> )
ts	async <b>set_logicalName</b> ( <b>newval</b> : string): Promise<number>
es	async <b>set_logicalName</b> ( <b>newval</b> )
dnp	int <b>set_logicalName</b> ( string <b>newval</b> )
cp	int <b>set_logicalName</b> ( string <b>newval</b> )
cmd	YWakeUpSchedule <b>target set_logicalName newval</b>

Vous pouvez utiliser `yCheckLogicalName( )` pour vérifier si votre paramètre est valide. N'oubliez pas d'appeler la méthode `saveToFlash( )` du module si le réglage doit être préservé.

#### Paramètres :

**newval** une chaîne de caractères représentant le nom logique du réveil agendé.

#### Retourne :

YAPI . SUCCESS si l'appel se déroule sans erreur.

En cas d'erreur, déclenche une exception ou retourne un code d'erreur négatif.

**wakeupschedule**→**set\_minutes()****YWakeUpSchedule****wakeupschedule**→**setMinutes()**

Modifie toutes les minutes où un réveil doit avoir lieu

js	function <b>set_minutes</b> ( <b>bitmap</b> )
cpp	int <b>set_minutes</b> ( s64 <b>bitmap</b> )
m	-(int) setMinutes : (s64) <b>bitmap</b>
pas	LongInt <b>set_minutes</b> ( <b>bitmap</b> : int64): LongInt
vb	function <b>set_minutes</b> ( ByVal <b>bitmap</b> As Long) As Integer
cs	int <b>set_minutes</b> ( long <b>bitmap</b> )
java	int <b>set_minutes</b> ( long <b>bitmap</b> )
uwp	async Task<int> <b>set_minutes</b> ( long <b>bitmap</b> )
py	<b>set_minutes</b> ( <b>bitmap</b> )
php	function <b>set_minutes</b> ( <b>\$bitmap</b> )
ts	async <b>set_minutes</b> ( <b>bitmap</b> : number): Promise<number>
es	async <b>set_minutes</b> ( <b>bitmap</b> )
dnp	int <b>set_minutes</b> ( long <b>bitmap</b> )
cp	int <b>set_minutes</b> ( s64 <b>bitmap</b> )
cmd	YWakeUpSchedule <b>target</b> <b>set_minutes</b> <b>bitmap</b>

**Paramètres :****bitmap** Minutes 00-59 de chaque heure où le réveil est actif.**Retourne :**

YAPI . SUCCESS si l'opération se déroule sans erreur.

En cas d'erreur, déclenche une exception ou retourne un code d'erreur négatif.

**wakeupschedule**→**set\_minutesA()****wakeupschedule**→**setMinutesA()**

Modifie les minutes de l'intervall 00-29 où un réveil doit avoir lieu.

js	function <b>set_minutesA</b> ( <b>newval</b> )
cpp	int <b>set_minutesA</b> ( int <b>newval</b> )
m	-(int) setMinutesA : (int) <b>newval</b>
pas	integer <b>set_minutesA</b> ( <b>newval</b> : LongInt): integer
vb	function <b>set_minutesA</b> ( ByVal <b>newval</b> As Integer) As Integer
cs	int <b>set_minutesA</b> ( int <b>newval</b> )
java	int <b>set_minutesA</b> ( int <b>newval</b> )
uwp	async Task<int> <b>set_minutesA</b> ( int <b>newval</b> )
py	<b>set_minutesA</b> ( <b>newval</b> )
php	function <b>set_minutesA</b> ( <b>\$newval</b> )
ts	async <b>set_minutesA</b> ( <b>newval</b> : number): Promise<number>
es	async <b>set_minutesA</b> ( <b>newval</b> )
dnp	int <b>set_minutesA</b> ( int <b>newval</b> )
cp	int <b>set_minutesA</b> ( int <b>newval</b> )
cmd	YWakeUpSchedule <b>target set_minutesA newval</b>

N'oubliez pas d'appeler la méthode `saveToFlash()` du module si le réglage doit être préservé.

**Paramètres :**

**newval** un entier représentant les minutes de l'intervall 00-29 où un réveil doit avoir lieu

**Retourne :**

YAPI . SUCCESS si l'opération se déroule sans erreur.

En cas d'erreur, déclenche une exception ou retourne un code d'erreur négatif.

wakeupschedule→set\_minutesB()

YWakeUpSchedule

wakeupschedule→setMinutesB()

Modifie les minutes de l'intervall 30-59 où un réveil doit avoir lieu.

js	function <b>set_minutesB</b> ( <b>newval</b> )
cpp	int <b>set_minutesB</b> ( int <b>newval</b> )
m	-(int) setMinutesB : (int) <b>newval</b>
pas	integer <b>set_minutesB</b> ( <b>newval</b> : LongInt): integer
vb	function <b>set_minutesB</b> ( ByVal <b>newval</b> As Integer) As Integer
cs	int <b>set_minutesB</b> ( int <b>newval</b> )
java	int <b>set_minutesB</b> ( int <b>newval</b> )
uwp	async Task<int> <b>set_minutesB</b> ( int <b>newval</b> )
py	<b>set_minutesB</b> ( <b>newval</b> )
php	function <b>set_minutesB</b> ( <b>\$newval</b> )
ts	async <b>set_minutesB</b> ( <b>newval</b> : number): Promise<number>
es	async <b>set_minutesB</b> ( <b>newval</b> )
dnp	int <b>set_minutesB</b> ( int <b>newval</b> )
cp	int <b>set_minutesB</b> ( int <b>newval</b> )
cmd	YWakeUpSchedule <b>target set_minutesB newval</b>

N'oubliez pas d'appeler la méthode `saveToFlash()` du module si le réglage doit être préservé.

#### Paramètres :

**newval** un entier représentant les minutes de l'intervall 30-59 où un réveil doit avoir lieu

#### Retourne :

YAPI . SUCCESS si l'opération se déroule sans erreur.

En cas d'erreur, déclenche une exception ou retourne un code d'erreur négatif.

**wakeupschedule**→**set\_monthDays()****wakeupschedule**→**setMonthDays()**

Modifie les jours du mois où un réveil doit avoir lieu.

js	function <b>set_monthDays</b> ( <b>newval</b> )
cpp	int <b>set_monthDays</b> ( int <b>newval</b> )
m	-(int) setMonthDays : (int) <b>newval</b>
pas	integer <b>set_monthDays</b> ( <b>newval</b> : LongInt): integer
vb	function <b>set_monthDays</b> ( ByVal <b>newval</b> As Integer) As Integer
cs	int <b>set_monthDays</b> ( int <b>newval</b> )
java	int <b>set_monthDays</b> ( int <b>newval</b> )
uwp	async Task<int> <b>set_monthDays</b> ( int <b>newval</b> )
py	<b>set_monthDays</b> ( <b>newval</b> )
php	function <b>set_monthDays</b> ( <b>\$newval</b> )
ts	async <b>set_monthDays</b> ( <b>newval</b> : number): Promise<number>
es	async <b>set_monthDays</b> ( <b>newval</b> )
dnp	int <b>set_monthDays</b> ( int <b>newval</b> )
cp	int <b>set_monthDays</b> ( int <b>newval</b> )
cmd	YWakeUpSchedule <b>target set_monthDays newval</b>

N'oubliez pas d'appeler la méthode `saveToFlash()` du module si le réglage doit être préservé.

**Paramètres :**

**newval** un entier représentant les jours du mois où un réveil doit avoir lieu

**Retourne :**

YAPI . SUCCESS si l'opération se déroule sans erreur.

En cas d'erreur, déclenche une exception ou retourne un code d'erreur négatif.

wakeupschedule→set\_months()

YWakeUpSchedule

wakeupschedule→setMonths()

Modifie les mois où un réveil doit avoir lieu.

js	function <b>set_months</b> ( <b>newval</b> )
cpp	int <b>set_months</b> ( int <b>newval</b> )
m	-(int) setMonths : (int) <b>newval</b>
pas	integer <b>set_months</b> ( <b>newval</b> : LongInt): integer
vb	function <b>set_months</b> ( ByVal <b>newval</b> As Integer) As Integer
cs	int <b>set_months</b> ( int <b>newval</b> )
java	int <b>set_months</b> ( int <b>newval</b> )
uwp	async Task<int> <b>set_months</b> ( int <b>newval</b> )
py	<b>set_months</b> ( <b>newval</b> )
php	function <b>set_months</b> ( <b>\$newval</b> )
ts	async <b>set_months</b> ( <b>newval</b> : number): Promise<number>
es	async <b>set_months</b> ( <b>newval</b> )
dnp	int <b>set_months</b> ( int <b>newval</b> )
cp	int <b>set_months</b> ( int <b>newval</b> )
cmd	YWakeUpSchedule <b>target set_months newval</b>

N'oubliez pas d'appeler la méthode `saveToFlash()` du module si le réglage doit être préservé.

#### Paramètres :

**newval** un entier représentant les mois où un réveil doit avoir lieu

#### Retourne :

YAPI . SUCCESS si l'opération se déroule sans erreur.

En cas d'erreur, déclenche une exception ou retourne un code d'erreur négatif.

**wakeupschedule**→**set\_userdata()****YWakeUpSchedule****wakeupschedule**→**setUserData()**

Enregistre un contexte libre dans l'attribut `userData` de la fonction, afin de le retrouver plus tard à l'aide de la méthode `get_userdata`.

js	function <b>set_userdata</b> ( <b>data</b> )
cpp	void <b>set_userdata</b> ( void * <b>data</b> )
m	-(void) <b>setUserData</b> : (id) <b>data</b>
pas	<b>set_userdata</b> ( <b>data</b> : Tobject)
vb	procedure <b>set_userdata</b> ( ByVal <b>data</b> As Object)
cs	void <b>set_userdata</b> ( object <b>data</b> )
java	void <b>set_userdata</b> ( Object <b>data</b> )
py	<b>set_userdata</b> ( <b>data</b> )
php	function <b>set_userdata</b> ( <b>\$data</b> )
ts	async <b>set_userdata</b> ( <b>data</b> : object null): Promise<void>
es	async <b>set_userdata</b> ( <b>data</b> )

Cet attribut n'es pas utilisé directement par l'API. Il est à la disposition de l'appelant pour stocker un contexte.

**Paramètres :**

**data** objet quelconque à mémoriser

wakeupschedule→set\_weekDays()

YWakeUpSchedule

wakeupschedule→setWeekDays()

Modifie les jours de la semaine où un réveil doit avoir lieu.

js	function <b>set_weekDays</b> ( <b>newval</b> )
cpp	int <b>set_weekDays</b> ( int <b>newval</b> )
m	-(int) setWeekDays : (int) <b>newval</b>
pas	integer <b>set_weekDays</b> ( <b>newval</b> : LongInt): integer
vb	function <b>set_weekDays</b> ( ByVal <b>newval</b> As Integer) As Integer
cs	int <b>set_weekDays</b> ( int <b>newval</b> )
java	int <b>set_weekDays</b> ( int <b>newval</b> )
uwp	async Task<int> <b>set_weekDays</b> ( int <b>newval</b> )
py	<b>set_weekDays</b> ( <b>newval</b> )
php	function <b>set_weekDays</b> ( <b>\$newval</b> )
ts	async <b>set_weekDays</b> ( <b>newval</b> : number): Promise<number>
es	async <b>set_weekDays</b> ( <b>newval</b> )
dnp	int <b>set_weekDays</b> ( int <b>newval</b> )
cp	int <b>set_weekDays</b> ( int <b>newval</b> )
cmd	YWakeUpSchedule <b>target set_weekDays newval</b>

N'oubliez pas d'appeler la méthode `saveToFlash()` du module si le réglage doit être préservé.

#### Paramètres :

**newval** un entier représentant les jours de la semaine où un réveil doit avoir lieu

#### Retourne :

YAPI . SUCCESS si l'opération se déroule sans erreur.

En cas d'erreur, déclenche une exception ou retourne un code d'erreur négatif.

## wakeupschedule→unmuteValueCallbacks()

## YWakeUpSchedule

Réactive l'envoi de chaque changement de la valeur publiée au hub parent.

js	function <b>unmuteValueCallbacks</b> ( )
cpp	int <b>unmuteValueCallbacks</b> ( )
m	-(int) <b>unmuteValueCallbacks</b>
pas	LongInt <b>unmuteValueCallbacks</b> ( ): LongInt
vb	function <b>unmuteValueCallbacks</b> ( ) As Integer
cs	int <b>unmuteValueCallbacks</b> ( )
java	int <b>unmuteValueCallbacks</b> ( )
uwp	async Task<int> <b>unmuteValueCallbacks</b> ( )
py	<b>unmuteValueCallbacks</b> ( )
php	function <b>unmuteValueCallbacks</b> ( )
ts	async <b>unmuteValueCallbacks</b> ( ): Promise<number>
es	async <b>unmuteValueCallbacks</b> ( )
dnp	int <b>unmuteValueCallbacks</b> ( )
cp	int <b>unmuteValueCallbacks</b> ( )
cmd	YWakeUpSchedule <b>target unmuteValueCallbacks</b>

Cette fonction annule un précédent appel à `muteValueCallbacks()`. N'oubliez pas d'appeler la méthode `saveToFlash()` du module si le réglage doit être préservé.

**Retourne :**

YAPI . SUCCESS si l'opération se déroule sans erreur.

En cas d'erreur, déclenche une exception ou retourne un code d'erreur négatif.

**wakeupschedule**→**wait\_async()****YWakeUpSchedule**

Attend que toutes les commandes asynchrones en cours d'exécution sur le module soient terminées, et appelle le callback passé en paramètre.

```
js fonction wait_async( callback, context)
```

```
ts wait_async( callback: Function, context: object)
```

```
es wait_async( callback, context)
```

La fonction callback peut donc librement utiliser des fonctions synchrones ou asynchrones, sans risquer de bloquer la machine virtuelle Javascript.

**Paramètres :**

**callback** fonction de callback qui sera appelée dès que toutes les commandes en cours d'exécution sur le module seront terminées. La fonction callback reçoit deux arguments: le contexte fourni par l'appelant et l'objet fonction concerné.

**context** contexte fourni par l'appelant, et qui sera passé tel-qu'el à la fonction de callback

**Retourne :**

rien du tout.



## 9. Problèmes courants

### 9.1. Par où commencer ?

Si c'est la première fois que vous utilisez un module Yoctopuce et ne savez pas trop par où commencer, allez donc jeter un coup d'œil sur le blog de Yoctopuce. Il y a une section dédiée aux débutants <sup>1</sup>.

### 9.2. Linux et USB

Pour fonctionner correctement sous Linux la librairie a besoin d'avoir accès en écriture à tous les périphériques USB Yoctopuce. Or, par défaut, sous Linux les droits d'accès des utilisateurs non-root à USB sont limités à la lecture. Afin d'éviter de devoir lancer les exécutable en tant que root, il faut créer une nouvelle règle *udev* pour autoriser un ou plusieurs utilisateurs à accéder en écriture aux périphériques Yoctopuce.

Pour ajouter une règle *udev* à votre installation, il faut ajouter un fichier avec un nom au format "`##-nomArbitraire.rules`" dans le répertoire `/etc/udev/rules.d`. Lors du démarrage du système, *udev* va lire tous les fichiers avec l'extension `.rules` de ce répertoire en respectant l'ordre alphabétique (par exemple, le fichier `51-custom.rules` sera interprété APRES le fichier `50-udev-default.rules`).

Le fichier `50-udev-default` contient les règles *udev* par défaut du système. Pour modifier le comportement par défaut du système, il faut donc créer un fichier qui commence par un nombre plus grand que 50, qui définira un comportement plus spécifique que le défaut du système. Notez que pour ajouter une règle vous aurez besoin d'avoir un accès root sur le système.

Dans le répertoire `udev_conf` de l'archive du *VirtualHub*<sup>2</sup> pour Linux, vous trouverez deux exemples de règles qui vous éviteront de devoir partir de rien.

#### Exemple 1: 51-yoctopuce.rules

Cette règle va autoriser tous les utilisateurs à accéder en lecture et en écriture aux périphériques Yoctopuce USB. Les droits d'accès pour tous les autres périphériques ne seront pas modifiés. Si ce scénario vous convient il suffit de copier le fichier `51-yoctopuce_all.rules` dans le répertoire `/etc/udev/rules.d` et de redémarrer votre système.

---

<sup>1</sup> voir: [http://www.yoctopuce.com/FR/blog\\_by\\_categories/pour-les-debutants](http://www.yoctopuce.com/FR/blog_by_categories/pour-les-debutants)

<sup>2</sup> <http://www.yoctopuce.com/EN/virtualhub.php>

```
# udev rules to allow write access to all users
# for Yoctopuce USB devices
SUBSYSTEM=="usb", ATTR{idVendor}=="24e0", MODE="0666"
```

### Exemple 2: 51-yoctopuce\_group.rules

Cette règle va autoriser le groupe "yoctogroup" à accéder en lecture et écriture aux périphériques Yoctopuce USB. Les droits d'accès pour tous les autres périphériques ne seront pas modifiés. Si ce scénario vous convient il suffit de copier le fichier "51-yoctopuce\_group.rules" dans le répertoire "/etc/udev/rules.d" et de redémarrer votre système.

```
# udev rules to allow write access to all users of "yoctogroup"
# for Yoctopuce USB devices
SUBSYSTEM=="usb", ATTR{idVendor}=="24e0", MODE="0666", GROUP="yoctogroup"
```

## 9.3. Plateformes ARM: HF et EL

Sur ARM il existe deux grandes familles d'exécutables: HF (Hard Float) et EL (EABI Little Endian). Ces deux familles ne sont absolument pas compatibles entre elles. La capacité d'une machine ARM à faire tourner des exécutables de l'une ou l'autre de ces familles dépend du hardware et du système d'exploitation. Les problèmes de compatibilité entre ArmHL et ArmEL sont assez difficiles à diagnostiquer, souvent même l'OS se révèle incapable de distinguer un exécutable HF d'un exécutable EL.

Tous les binaires Yoctopuce pour ARM sont fournis pré-compilée pour ArmHF et ArmEL, si vous ne savez à quelle famille votre machine ARM appartient, essayez simplement de lancer un exécutable de chaque famille.

## 9.4. Les exemples de programmation n'ont pas l'air de marcher

La plupart des exemples de programmation de l'API Yoctopuce sont des programmes en ligne de commande et ont besoin de quelques paramètres pour fonctionner. Vous devez les lancer depuis l'invite de commande de votre système d'exploitation ou configurer votre IDE pour qu'il passe les paramètres corrects au programme <sup>3</sup>.

## 9.5. Module alimenté mais invisible pour l'OS

Si votre YoctoHub-GSM-2G est branché par USB et que sa LED bleue s'allume, mais que le module n'est pas vu par le système d'exploitation, vérifiez que vous utilisez bien un vrai câble USB avec les fils pour les données, et non pas un câble de charge. Les câbles de charge n'ont que les fils d'alimentation.

## 9.6. Another process named xxx is already using yAPI

Si lors de l'initialisation de l'API Yoctopuce, vous obtenez le message d'erreur "*Another process named xxx is already using yAPI*", cela signifie qu'une autre application est déjà en train d'utiliser les modules Yoctopuce USB. Sur une même machine, un seul processus à la fois peut accéder aux modules Yoctopuce par USB. Cette limitation peut facilement être contournée en utilisant un VirtualHub et le mode réseau <sup>4</sup>.

<sup>3</sup> voir: <http://www.yoctopuce.com/FR/article/a-propos-des-programmes-d-exemples>

<sup>4</sup> voir: <http://www.yoctopuce.com/FR/article/message-d-erreur-another-process-is-already-using-yapi>

## 9.7. Déconnexions, comportement erratique

Si votre YoctoHub-GSM-2G se comporte de manière erratique et/ou se déconnecte du bus USB sans raison apparente, vérifiez qu'il est alimenté correctement. Évitez les câbles d'une longueur supérieure à 2 mètres. Au besoin, intercalez un hub USB alimenté <sup>5 6</sup>.

## 9.8. RegisterHub d'un VirtualHub déconnecte le précédent

Si lorsque vous faire un `YAPI.RegisterHub` d'un VirtualHub la connexion avec un autre virtualHub précédemment enregistré tombe, vérifiez que les machines qui hébergent ces VirtualHubs on bien un *hostname* différent. Ce cas de figure est très courant avec les machines dont le système d'exploitation est installé avec une image monolithique, comme les Raspberry-PI par exemple. L'API Yoctopuce utilise les numéros de série Yoctopuce pour communiquer et le numéro de série d'un VirtualHub est créé à la volée à partir du *hostname* de la machine qui l'héberge.

## 9.9. Commandes ignorées

Si vous avez l'impression que des commandes envoyées à un module Yoctopuce sont ignorées, typiquement lorsque vous avez écrit un programme qui sert à configurer ce modules Yoctopuce et qui envoie donc beaucoup de commandes, vérifiez que vous avez bien mis un `YAPI.FreeAPI()` à la fin du programme. Les commandes sont envoyées aux modules de manière asynchrone grâce à un processus qui tourne en arrière plan. Lorsque le programme se termine, ce processus est tué, même s'il n'a pas eu le temps de tout envoyer. En revanche `API.FreeAPI()` attend que la file d'attente des commandes à envoyer soit vide avant de libérer les ressources utilisées par l'API et rendre la main.

## 9.10. Impossible de contacter les sous-devices par USB

Le but du YoctoHub-GSM-2G est de fournir une connectivité réseau aux sous-modules qui lui sont connecté, il ne se comporte pas comme un hub USB. Le port USB du YoctoHub-GSM-2G ne sert qu'à l'alimenter et le configurer. Pour accéder aux modules connectés au hub, vous devez impérativement passer par une connexion réseau.

## 9.11. Network Readiness coincé à3- LAN ready

Vérifiez que votre connexion Internet sortante fonctionne et que vous n'avez pas un callback invalide défini dans la configuration to YoctoHub-GSM-2G.

## 9.12. Module endommagé

Yoctopuce s'efforce de réduire la production de déchets électroniques. Si vous avez l'impression que votre YoctoHub-GSM-2G ne fonctionne plus, commencez par contacter le support Yoctopuce par e-mail pour poser un diagnostic. Même si c'est suite à une mauvaise manipulation que le module a été endommagé, il se peut que Yoctopuce puisse le réparer, et ainsi éviter de créer un déchet électronique.



**Déchets d'équipements électriques et électroniques (DEEE)** Si voulez vraiment vous débarasser de votre YoctoHub-GSM-2G, ne le jetez pas à la poubelle, mais ramenez-le à l'un des points de collecte proposé dans votre région afin qu'il soit envoyé à un centre de recyclage ou de traitement spécialisé.



<sup>5</sup> voir: <http://www.yoctopuce.com/FR/article/cables-usb-la-taille-compte>

<sup>6</sup> voir: <http://www.yoctopuce.com/FR/article/combien-de-capteurs-usb-peut-on-connecter>



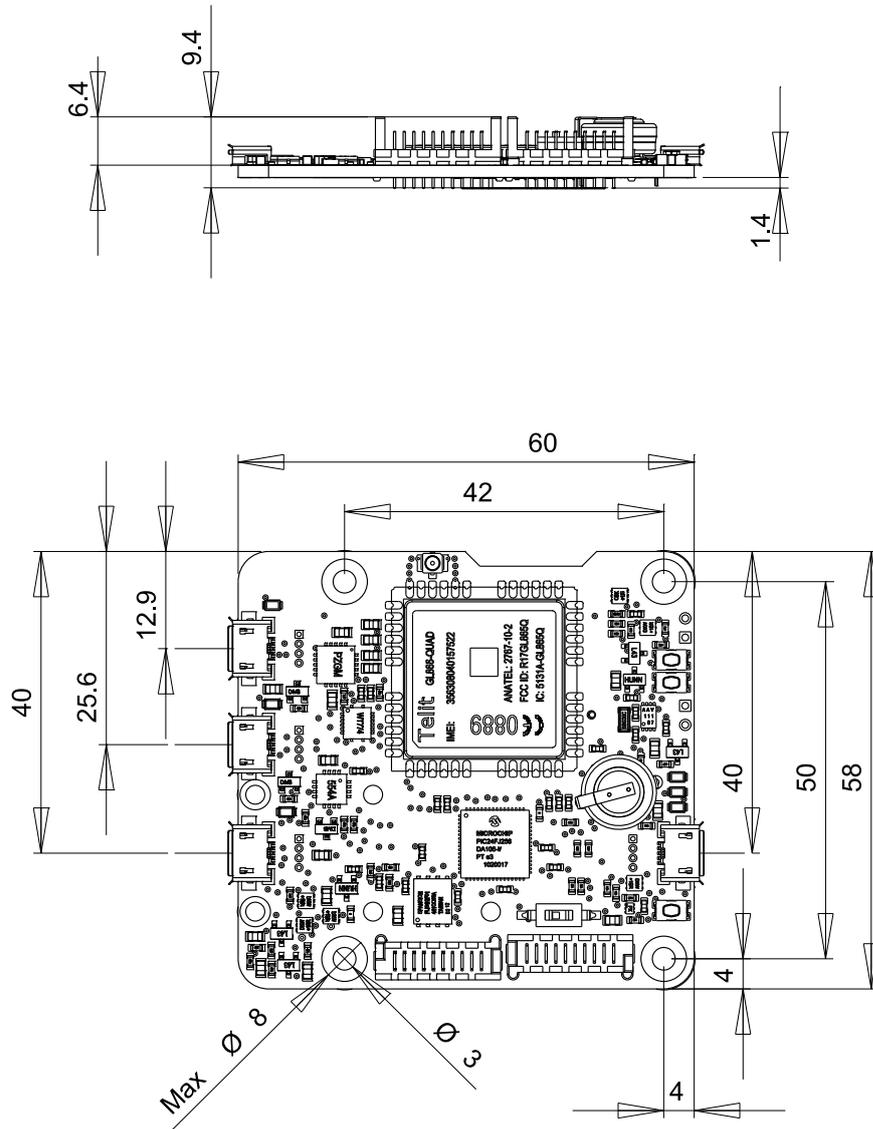
## 10. Caractéristiques

Vous trouverez résumées ci-dessous les principales caractéristiques techniques de votre module YoctoHub-GSM-2G

Identifiant produit	YHUBGSM1
Révision matérielle <sup>†</sup>	Rev. C
Connecteur USB	micro-B
Epaisseur	9.5 mm
Largeur	58 mm
Longueur	60 mm
Poids	34 g
Chipset	Telit GL865-QUAD
Fréquence	850, 900, 1800, 1900 MHz
Classe de protection selon IEC 61140	classe III
Temp. de fonctionnement normale	5...40 °C
Temp. de fonctionnement étendue <sup>‡</sup>	-20...70 °C
Consommation USB	100 mA
Connection réseau	GSM 2G (EDGE)
Conformité RoHS	RoHS III (2011/65/UE+2015/863)
USB Vendor ID	0x24E0
USB Device ID	0x003E
Boîtier recommandé	YoctoBox-HubWlan-Transp
Code tarifaire harmonisé	9032.9000
Fabriqué en	Suisse

<sup>†</sup> Ces spécifications correspondent à la révision matérielle actuelle du produit. Les spécifications des versions antérieures peuvent être inférieures.

<sup>‡</sup> La plage de température étendue est définie d'après les spécifications des composants et testée sur une durée limitée (1h). En cas d'utilisation prolongée hors de la plage de température standard, il est recommandé procéder à des tests extensifs avant la mise en production.



All dimensions are in mm  
 Toutes les dimensions sont en mm

# YoctoHub-GSM

A4

Scale  
**1:1**  
 Echelle