

Création des infos depuis serveur WES vers Jeedom

Table des matières

Préambule.....	2
Configuration initiale de Jeedom.....	3
Création des valeurs contenus dans le Serveur Wes.....	3
<i>Création d'un virtuel.....</i>	<i>3</i>
Récupération des valeurs depuis le WES.....	4
<i>Création du scénario de récupération des données.....</i>	<i>4</i>
<i>Ajout du code de traitement des données depuis le WES.....</i>	<i>4</i>
<i>Vérification de l'affichage sur le Dashboard.....</i>	<i>7</i>
<i>Modification du code pour lire d'autres valeurs.....</i>	<i>8</i>
<i>Modification de l'affichage du virtuel sous forme de widget.....</i>	<i>8</i>

Création des infos depuis serveur WES vers Jeedom

Préambule

Je teins à préciser que je ne suis pas l'auteur de ce code et de l'explication originale, j'ai simplement refait quelque chose de plus détaillé à partir du code de « tostaky » et des infos de « philou15 », sur le forum de Jeedom. Le message original est ici, pour ceux intéressés :

<https://www.jeedom.fr/forum/viewtopic.php?f=32&t=8707&hilit=wes#p169454>

Merci à eux pour le partage.

Voici la description des fonctions du code de Tostaky :

Paramétrage du script:

Pour utiliser le script, vous devez renseigner les 2 paramètres situés au début du code python en fonction de votre serveur WES.

\$WesIP: l'adresse IP de votre serveur WES, ou simplement "WES" si votre serveur est accessible par l'url <http://wes>

\$User: contient le user et le mot de passe d'accès à votre serveur. Ils sont séparés par le caractère ":".

Fonction *getWesXml(\$pWesIp, \$pWesUser, \$pWesPage)*:

Cette fonction permet d'interroger le serveur WES pour obtenir l'ensemble des données contenu dans un flux XML (data.cgx).

Fonction *lireValeurXml(\$pXml, \$pXmlNoeud, \$xmlAttribut, \$pId)*:

Cette fonction permet de lire une valeur dans le flux XML du WES pour l'affecter au virtuel de jeedom (via son ID dans Jeedom).

Après ces 2 fonctions, conserver/ajouter la lecture des valeurs dont vous avez besoins:

Ajouter ou supprimer les appels à la fonction *lireValeurXml* en fonction des valeurs que vous voulez enregistrer dans vos virtuels.

Par exemple le code *lireValeurXml(\$xml, 'tic1', 'HCHC', 671)*; va stocker dans le virtuel Jeedom dont l'ID est 671 la valeur de l'index des heures creuses.

Création des infos depuis serveur WES vers Jeedom

Configuration initiale de Jeedom

Installer les plugins « Virtuel » et « script »

Vous devez également connaître votre adresse IP du serveur WES, et être sur que c'est une adresse IP FIXE !

Création des valeurs contenus dans le Serveur Wes

Pour lire les valeurs du serveur Wes dans Jeedom, il faut créer des objets virtuels, chaque objet correspondra à une valeur enregistrée dans le WES, par exemple on crée une valeur virtuelle « temperature_sejour » qui correspondra à la sonde de température du séjour dans le WES.

Création d'un virtuel

Ouvrir le menu « Plugin – Programmation - Virtuel » dans le menu du haut.

On va tomber sur la page principale du plugin virtuel sur laquelle on va créer notre premier objet.

Cliquer sur le bouton « ajouter » et donner un nom explicite, car c'est ce nom là qui sera affiché dans le widget sur la page de Jeedom (sur ma capture d'écran, je l'ai appelé « Sondes séjour », car j'ai une sonde mixte, température, humidité et luminosité).

Ensuite on arrive sur la page principale de configuration.

The screenshot shows the 'Général' configuration page for a virtual object. The 'Nom de l'équipement virtuel' field contains 'Sondes Séjour'. The 'Objet parent' dropdown is set to 'Séjour'. Under 'Catégorie', several checkboxes are visible: Chauffage, Sécurité, Energie, Lumière, Automatisation, Multimedia, and Default. There are two 'Oui' buttons, one for 'Activer' and one for 'Visible'. A 'Commentaire' text area is empty. Below the 'Général' section, there are two buttons: 'Ajouter une info virtuelle' and 'Ajouter une commande virtuelle'.

Vous pouvez cliquer sur le bouton « ajouter une info virtuelle ».

The screenshot shows the 'Virtuel' configuration page. At the top, there are two buttons: 'Ajouter une info virtuelle' and 'Ajouter une commande virtuelle'. Below is a table with the following columns: #, Nom, Sous-Type, Valeur, Unité, and Paramètres. The first row has the following data: # 8, Nom 'Température', Sous-Type 'info', Valeur 'Calcul', Unité '°C', and Paramètres with 'Historiser' and 'Afficher' checked. Below the table, there are buttons for 'Rechercher équipement', 'Valeur retour d'état', and 'Durée avant retour d'état'. At the bottom, there are 'Supprimer' and 'Sauvegarder' buttons.

#	Nom	Sous-Type	Valeur	Unité	Paramètres
8	Température	info	Calcul	°C	<input checked="" type="checkbox"/> Historiser <input checked="" type="checkbox"/> Afficher
		Numérique	<input type="text" value="Rechercher équipement"/> <input type="text" value="Valeur retour d'état"/> <input type="text" value="Durée avant retour d'état"/>		<input checked="" type="checkbox"/> Événement <input type="text" value="Min"/> <input type="text" value="Max"/>

J'ai appelé la première valeur « Température », notez bien le chiffre d'identifiant de la valeur virtuelle, ici **8** (valeur numérique derrière le caractère dièse #), il correspond à l'identifiant unique de la valeur virtuelle à utiliser dans le script plus tard.

On coche « Afficher », « Événement » et « Historique », à noter que la coche événement, est une obligation, sinon

Création des infos depuis serveur WES vers Jeedom

votre valeur reviendra à 0 après avoir été affichée brièvement.

Évidemment comme partout dans Jeedom, on n'oublie pas de cliquer sur le bouton « Sauvegarder ».

Récupération des valeurs depuis le WES

Bon ok, on vient de créer notre objet (ou valeur) virtuel, maintenant il faut chercher la valeur depuis le serveur WES, pour les assigner à notre valeur virtuelle toute fraîchement créée.

Création du scénario de récupération des données

Pour cela on va créer un scénario, aller dans le menu « Général - Scénarios ».

Cliquer sur « Ajouter », et donner le nom de votre scénario (pour exemple « wes-jeedom »), on choisi « type avancé » et on clique sur le bouton « d'accord ».

Scénario (ID : 4) [Executer] [Supprimer] [Sauvegarder] [Template] [Exporter] [Log] [Duplique]

Nom du scénario: wes-jeedom

Mode du scénario: Programmé

Nom à afficher: Nom à afficher

Précédent: 2015-11-21 06:10:00 Prochain: 2015-11-21 06:30:00

Groupe: Groupe du scénario

Programmation: */10****

Objet parent: Séjour

Timeout secondes (0 = illimité): 0

Actif: Oui

Visible: Oui

Exécuter en avant plan: Non

Enchaîner les commandes sans attendre: Non

Pas de log: Non

Etat: Erreur

Ajouter bloc

On voit que dans le nom du scénario il y a notre nom affiché, ensuite on va le configurer comme suit :

- Mode scénario : « Programmé » (déclenché automatiquement selon une action)
- Programmation : « */10**** » (temporisation de 10 min, si vous voulez mettre 5 min, changez la valeur à « */5**** »)
- Objet parent : n'est pas obligatoire, et si vous souhaitez regrouper les virtuels par pièce par exemple, il faudra créer préalablement des objets via le menu « Général - Objets ».

Ajout du code de traitement des données depuis le WES

Bon la on va attaquer le lourd, on va créer le script qui va chercher les infos dans le WES, pour extraire les infos et les insérer dans le virtuel.

On clique sur le bouton « Ajouter bloc », et on choisi « Code » dans la liste de choix, on valide par « Enregistrer ».

Ensuite le plus dur, le code, mais je vais vous faciliter la tâche, un petit copier/coller suffira pour ce que l'on veut faire :-).

Copiez le texte ci dessous (attention à bien copier, sans oublier les accolades en fin de code, et ne pas rajouter d'espaces supplémentaires) :

```
$WesIP="VOTRE IP";
$User="VOTRE USER:VOTRE MOT DE PASSE";
$dataPage="data.cgx";

function getWesXml($pUser, $pWesIP, $pDataPage) {

try {
// Url a requeter
$wesUrl="http://".$pUser."@".$pWesIP."/".$pDataPage;
```

Création des infos depuis serveur WES vers Jeedom

```
// Tableau contenant les options de téléchargement
$options=array(
    CURLOPT_URL=>$wesUrl,
    CURLOPT_RETURNTRANSFER=>true,
    CURLOPT_HEADER=>false,
    CURLOPT_CONNECTTIMEOUT=>2000,
    CURLOPT_FRESH_CONNECT=>true,
    CURLOPT_USERPWD=>$pUser
);

// Création d'une nouvelle ressource cURL
$CURL=curl_init();

// Configuration des options de téléchargement
curl_setopt_array($CURL, $options);

// Exécution de la requête
$content=curl_exec($CURL);
/*
if(!curl_errno($CURL))
{
    $info = curl_getinfo($CURL);
    log::add("WES", "INFO", "Appel data.cgx OK:".$info["http_code"],"");
}
*/
// Fermeture de la session cURL
curl_close($CURL);

if($content == "") {log::add("WES", "INFO", "HTTP FAILED!","");}
else {
    $xml = new DomDocument();
    $xml->loadXML($content);
}

return $xml;
} catch (Exception $e) {
    log::add("WES", "INFO", "HTTP erreur:".$e->getMessage(),"");
    return;
}
}

# Lecture d'une valeur d'un flux XML et mise à jour de l'objet virtuel de JEEDOM
# Paramètres:
# - $pXml: contient l'ensemble du document XML à lire.
# - $pXmlNoeud: contient le nom du noeud XML à lire.
# - $pXmlAttribut: contient le nom de l'attribut XML à lire.
# - $pId: Identifiant du virtuel Jeedom dans lequel la valeur de l'attribut XML doit être stockée.
# - $pUnite: Divise la valeur par ce chiffre si $pUnite est > à zéro.
function lireValeurXml($pXml, $pXmlNoeud, $pXmlAttribut, $pId, $pUnite) {

    try {
        $valeur = $pXml->getElementsByTagName($pXmlNoeud)->item(0)->getElementsByTagName($pXmlAttribut)->item(0)->nodeValue;

        if ($pUnite > 0)
            $valeur = $valeur / $pUnite;

        $cmd=cmd::byId($pId);
        $cmd->setValue($valeur);
        $cmd->event($valeur);
    }
}
```

Création des infos depuis serveur WES vers Jeedom

```
} catch (Exception $e) {
    log::add("WES", "INFO", "XML erreur on Id:". $pId. ":". $e->getMessage(), "");
}
}

# Lecture des données
$xml = getWesXml($User, $WesIP, $dataPage);

if($xml===null)
{
    log::add("WES", "INFO", "XML is null,");
}
else {
    # Temperatures
    lireValeurXml($xml, 'temp', 'SONDE1', 525, 0);
    lireValeurXml($xml, 'temp', 'SONDE2', 526, 0);

    # Relais
    # lireValeurXml($xml, 'relais', 'RELAIS1', 536, 0);
    # lireValeurXml($xml, 'relais', 'RELAIS2', 537, 0);

#####
#####
# Téléinfo

#####
#####

# Puissance apparente
lireValeurXml($xml, 'tic1', 'PAP', 674, 0);

# Pincas ampèremétrique
lireValeurXml($xml, 'pince', 'I1', 677, 0);
lireValeurXml($xml, 'pince', 'I2', 678, 0);

# Tarif en cours
lireValeurXml($xml, 'tic1', 'PTEC', 673, 0);

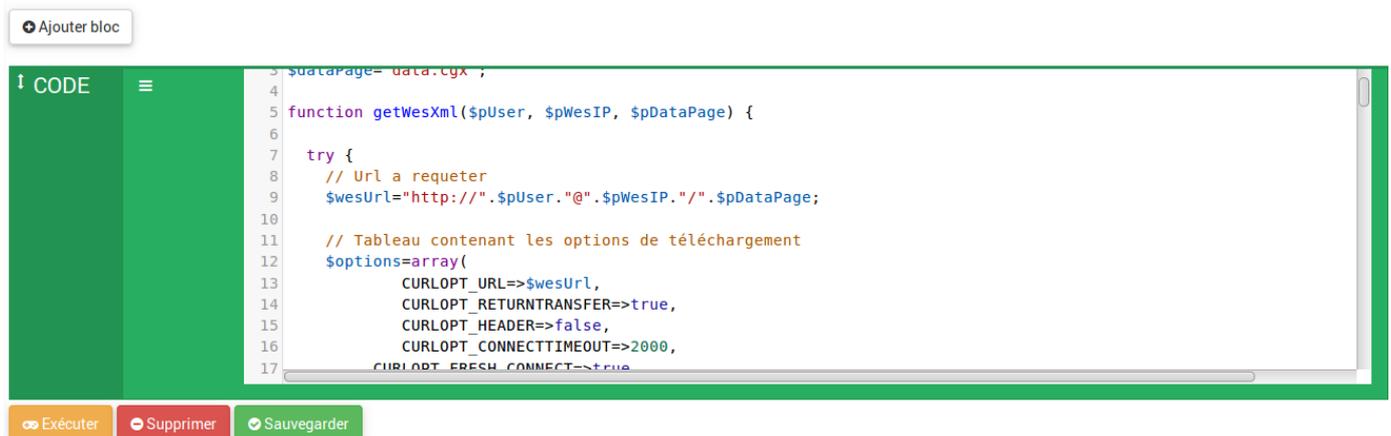
# Index heure creuse
lireValeurXml($xml, 'tic1', 'HCHC', 671, 1000);
# Index heure pleine
lireValeurXml($xml, 'tic1', 'HCHP', 672, 1000);

# Abonnement
lireValeurXml($xml, 'tic1', 'ISOUSC', 675, 0);

# Intensité consommée
lireValeurXml($xml, 'tic1', 'IINST', 676, 0);
}
```

Coller le tout dans le cadre « Code » sous Jeedom.

Création des infos depuis serveur WES vers Jeedom



```
3 $dataPage= data.Cgx ;
4
5 function getWesXml($pUser, $pWesIP, $pDataPage) {
6
7     try {
8         // Url a requester
9         $wesUrl="http://".$pUser."@".$pWesIP."/".$pDataPage;
10
11         // Tableau contenant les options de téléchargement
12         $options=array(
13             CURLOPT_URL=>$wesUrl,
14             CURLOPT_RETURNTRANSFER=>true,
15             CURLOPT_HEADER=>false,
16             CURLOPT_CONNECTTIMEOUT=>2000,
17             CURLOPT_FRESH_CONNECT=>true
```

Ensuite vous aurez sûrement fait attention, mais il faut rentrer plusieurs infos pour que le code fonctionne :

- Sur la première ligne remplacer `$WesIP = "wes"` ; par l'adresse IP de votre WES comme par exemple « `$WesIP = "192.168.0.100"` ; »
- Sur la 2ème ligne, il faut noter l'identifiant/mot de passe que vous utilisez pour vous connecter au serveur WES, et mettre ces infos à la place de « `$User = "monUser : monMotDePasse"` ; », attention à bien garder le double point entre l'identifiant et le mot de passe.
- Ensuite on descend plus bas, jusqu'à la première ligne où l'on lit les valeurs

```
# Temperatures
lireValeurXml($xml, 'temp', 'SONDE1', 525, 0);
lireValeurXml($xml, 'temp', 'SONDE2', 526, 0);
```

Les valeurs qui nous intéressent, sont les 4èmes (ici 525 et 526), c'est le moment de ressortir votre cahier et l'identifiant de votre valeur virtuelle créée avant, pour mon exemple je remplace donc 525, par 8.

Ensuite on oublie pas d'enregistrer et le tour est joué.

Vérification de l'affichage sur le Dashboard

On peut tester, en cliquant sur le bouton orange « Tester » en haut de la page et aller sur la vue Dashboard pour voir si une valeur est présente, chez moi c'est bien le cas, mais la valeur n'est pas la bonne (il est affiché 54°C). Tout simplement car j'ai des sondes combinées humidité, température et luminosité, chez moi, et si l'on regarde sous l'interface de WES, la première sonde qui est affichée est celle d'humidité.

Voici une copie d'écran de la vue du serveur WES

Création des infos depuis serveur WES vers Jeedom

Configuration sonde(s) 1 wire (Température / Humidité)

Activer lecture des sondes ON OFF

Activer fichier CSV (1enr/minute) ON OFF

0 sonde(s) de température & 2 sonde(s) d'humidité détectée(s)

N°	Identifiant	Nom	Type	Valeurs lues	valeurs corrigées	Déecté	Graph. 1 à 4
1	26 E4 50 BF 01 00 00 A1	hum_sejour	DS2438	58.44 %	58.44 %	<input checked="" type="radio"/>	2
2	26 E4 50 BF 01 00 00 A1	temp_sejou	DS2438	20.91 °C	20.91 °C	<input checked="" type="radio"/>	1
3	26 E4 50 BF 01 00 00 A1	lum_sejour	DS2438	0.00 L	0.00 L	<input checked="" type="radio"/>	3
4	26 0D 9B BF 01 00 00 4C	hum_lea	DS2438	62.18 %	62.18 %	<input checked="" type="radio"/>	2
5	26 0D 9B BF 01 00 00 4C	temp_lea	DS2438	20.62 °C	20.62 °C	<input checked="" type="radio"/>	1
6	26 0D 9B BF 01 00 00 4C	lum_lea	DS2438	0.00 L	0.00 L	<input checked="" type="radio"/>	3

Donc ici on voit bien que la première valeur du tableau est bien l'humidité donc ça colle.

Modification du code pour lire d'autres valeurs

On va retourner sur notre scénario, en cliquant sur le menu « Général - Scénario », puis en cliquant sur le scénario « WES - Jeedom ».

On retourne dans le code ou il y a la lecture de notre sonde, normalement la ligne devrait être comme ceci si vous avez bien tout suivi :

```
# Temperatures
lireValeurXml($xml, 'temp', 'SONDE1', 8, 0);
lireValeurXml($xml, 'temp', 'SONDE2', 526, 0);
```

Et on change simplement par :

```
# Temperatures
lireValeurXml($xml, 'temp', 'SONDE1', 525, 0);
lireValeurXml($xml, 'temp', 'SONDE2', 8, 0);
```

Comme d'habitude, on enregistre, on test le scénario et on va sur le dashboard, et oh miracle ça fonctionne, on a bien la valeur de la température au lieu de l'humidité qui s'est affichée.

C'est tout simplement car la numérotation des sondes est fonction du tableau dans le WES, si je reviens sur ma copie du tableau du WES chez moi (plus haut), si j'avais voulu afficher la valeur de la luminosité du séjour, j'aurais affecté mon virtuel jeedom (#8 rappelez vous), sur ma sonde N°3 (jetez un œil sur le tableau du WES qui se trouve dans le menu « Températures - Configuration »), comme ceci :

```
# Temperatures
lireValeurXml($xml, 'temp', 'SONDE1', 525, 0);
lireValeurXml($xml, 'temp', 'SONDE2', 526, 0);
lireValeurXml($xml, 'temp', 'SONDE3', 8, 0);
```

Modification de l'affichage du virtuel sous forme de widget

Bon ok, on vient de créer notre objet (ou valeur) virtuel, ça fonctionne, on sait comment modifier et ajouter des

Création des infos depuis serveur WES vers Jeedom

valeurs, etc.. maintenant il faut l'afficher de façon « classe ».

On clique sur le virtuel pour pouvoir afficher sa page de configuration, et dans la ligne du virtuel (#8), on voit avoir un bouton engrenages tout à droite du cadre



On clique sur ce bouton, pour afficher les paramètres avancés du virtuel.

On a une nouvelle fenêtre avec 3 onglets qui apparaissent, celui qui nous intéresse est le 3ème (Affichage avancé)

Configuration commande

[Informations](#) [Configuration avancée](#) [Affichage avancé](#) [Appliquer à](#) [Enregistrer](#)

Widget

	Dashboard et design	Vue	Mobile
Widget	thermometre		gauge
Visible	<input checked="" type="checkbox"/> Oui		<input checked="" type="checkbox"/> Oui
Afficher le nom	<input checked="" type="checkbox"/> Oui	<input checked="" type="checkbox"/> Oui	
Afficher les statistiques	<input checked="" type="checkbox"/> Oui	<input checked="" type="checkbox"/> Oui	<input checked="" type="checkbox"/> Oui
Retour à la ligne forcé avant le widget	<input checked="" type="checkbox"/> Non	après le widget	<input checked="" type="checkbox"/> Non

C'est dans la partie Widget que ca se passe, je vous laisse essayer les différents paramètres pour voir ce que ca donne, n'oubliez pas, comme d'habitude, d'enregistrer avant de quitter la page pour jeter un coup d'oeil sur le dashboard et voir à quoi ca ressemble.