Voici la partie la plus importante dans la domotique : les scénarios. Véritable cerveau de la domotique, c’est ce qui permet d’interagir avec le monde réel de manière “intelligente”.

La page de gestion des Scénarios

Gestion

Pour y accéder, rien de plus simple, il suffit d’aller sur Outils -> Scénarios. Vous y trouverez la liste des scénarios de votre Jeedom ainsi que des fonctions pour les gérer au mieux :

* Ajouter : Permet de créer un scénario. La procédure est décrite dans le chapitre suivant.
* Désactiver scénarios : Permet de désactiver tous les scénarios.
* Voir variables : Permet de voir les variables, leur valeur ainsi que l’endroit où elle sont utilisées. Vous pouvez également y en créer une. Les variables sont décrites dans un chapitre de cette page.
* Vue d’ensemble : Permet d’avoir une vue d’ensemble de tous les scénarios. Vous pouvez changer les valeurs actif, visible, multi lancement, mode synchrone, Log et Timeline (ces paramètres sont décrits dans le chapitre suivant). Vous pouvez également accéder aux logs de chaque scénario et les démarrer individuellement.
* Testeur d’expression : Permet d’executer un test sur une expression de votre choix et d’en afficher le résultat.

Mes scénarios

Vous trouverez dans cette partie la liste des scénarios que vous avez créés. Ils sont classés suivant les groupes que vous avez définis pour chacun d’eux. Chaque scénario est affiché avec son nom et son objet parent. Les scénarios grisés sont ceux qui sont désactivés.

Comme dans de nombreuses pages de Jeedom, mettre la souris à gauche de l’écran permet de faire apparaître un menu d’accès rapide (à partir de votre profil, vous pouvez le laisser toujours visible). Vous pourrez alors chercher votre scénario, mais aussi en ajouter un par ce menu.

Edition d’un scénario

Après avoir cliqué sur Ajouter, vous devez choisir le nom de votre scénario et vous êtes redirigés vers la page de ses paramètres généraux. En haut, on retrouve quelques fonctions utiles pour gérer notre scénario :

* ID : A côté du mot Général, c’est l’identifiant du scénario.
* statut : Etat actuel de votre scénario.
* variables : Permet d’afficher les variables.
* Expression : Permet d’afficher le testeur d’expression.
* Exécuter : Permet de lancer le scénario manuellement (N’oubliez pas de sauvegarder au préalable !). Les déclencheurs ne sont donc pas pris en compte.
* Supprimer : Permet de supprimer le scénario.
* Sauvegarder : Permet de sauvegarder les changements effectués.
* Template : Permet d’accéder aux templates et d’en appliquer un au scénario depuis le market. (expliqué en bas de page).
* Exporter : Permet d’obtenir une version texte du scénario.
* Log : Permet d’afficher les logs du scénario.
* Dupliquer : Permet de copier le scénario pour en créer un nouveau avec un autre nom.
* Liens : Permet de visualiser le graphique des éléments en lien avec le scénario.

Onglet Général

Onglet Scénario

**Blocs Si/Alors/Sinon , Boucle, Dans et A**

Note

Sur les blocs de type Si/Alors/Sinon, des

Pour les conditions, Jeedom essaye de faire en sorte qu’on puisse les écrire le plus possible en langage naturel tout en restant souple. Trois boutons sont disponibles sur la droite de ce type de bloc pour sélectionner un élément à tester :

* . Une fois la commande trouvée, Jeedom ouvre une fenêtre pour vous demander quel test vous souhaitez effectuer sur celle-ci. Si vous choisissez de Ne rien mettre, Jeedom ajoutera la commande sans comparaison. Vous pouvez également choisir et ou ou devant Ensuite pour enchaîner des tests sur différents équipements.
* Rechercher un scénario : Permet de chercher un scénario à tester.
* Rechercher un équipement : Idem pour un équipement.

Tip

Il existe une liste de tags permettant d’avoir accès à, ou bien à l’heure, la date, un nombre aléatoire,…. Voir plus loin les chapitres sur les commandes et les tags.

Une fois la condition renseignée, vous devez utiliser le bouton “ajouter”, à gauche, afin d’ajouter un nouveau bloc ou une action dans le bloc actuel.

**Bloc Code**

Important

Commandes (capteurs et actionneurs):

* cmd::byString($string); : Retourne l’objet commande correspondant.
* $string : Lien vers la commande voulue : #[objet][equipement][commande]# (ex : #[Appartement][Alarme][Actif]#)
* cmd::byId($id); : Retourne l’objet commande correspondant.
* $id : ID de la commande voulue
* $cmd->execCmd($options = null); : Exécute la commande et retourne le résultat.
* $options : Options pour l’exécution de la commande (peut être spécifique au plugin), option de base (sous-type de la commande) : - message : $option = array(‘title’ => ‘titre du message , ‘message’ => ‘Mon message’); - color : $option = array(‘color’ => ‘couleur en hexadécimal’); - slider : $option = array(‘slider’ => ‘valeur voulue de 0 à 100’);

Log :

* log::add(‘filename’,’level’,’message’);
* filename : Nom du fichier de log.
* level : [debug], [info], [error], [event].
* message : Message à écrire dans les logs.

Scénario :

* $scenario->getName(); : Retourne le nom du scénario courant.
* $scenario->getGroup(); : Retourne le groupe du scénario.
* $scenario->getIsActive(); : Retourne l’état du scénario.
* $scenario->setIsActive($active); : Permet d’activer ou non le scénario.
* $active : 1 actif , 0 non actif.
* $scenario->setOnGoing($onGoing); : Permet de dire si le scénario est en cours ou non.
* $onGoing => 1 en cours , 0 arrêté.
* $scenario->save(); : Sauvegarde les modifications.
* $scenario->setData($key, $value); : Sauvegarde une donnée (variable).
* $key : clé de la valeur (int ou string).
* $value : valeur à stocker (int, string, array ou object).
* $scenario->getData($key); : Récupère une donnée (variable).
* $key => clé de la valeur (int ou string).
* $scenario->removeData($key); : Supprime une donnée.
* $scenario->setLog($message); : Ecrit un message dans le log du scénario.
* $scenario->persistLog(); : Force l’écriture du log (sinon il est écrit seulement à la fin du scénario). Attention, ceci peut un peu ralentir le scénario.

Les Actions

Les actions ajoutées dans les blocs ont plusieurs options. Dans l’ordre :

Les déclencheurs

Les fonctions de calcul

Plusieurs fonctions sont disponibles pour les équipements :

* average(commande,période) et averageBetween(commande,start,end)

Donnent la moyenne de la commande sur la période (period=[month,day,hour,min] ou [expression PHP](http://php.net/manual/fr/datetime.formats.relative.php)) ou entre les 2 bornes demandées (sous la forme Y-m-d H:i:s ou [expression PHP](http://php.net/manual/fr/datetime.formats.relative.php)) :

* min(commande,période) et minBetween(commande,start,end) : Donnent le minimum de la commande sur la période (period=[month,day,hour,min] ou [expression PHP](http://php.net/manual/fr/datetime.formats.relative.php)) ou entre les 2 bornes demandées (sous la forme Y-m-d H:i:s ou [expression PHP](http://php.net/manual/fr/datetime.formats.relative.php)) :
* max(commande,période) et maxBetween(commande,start,end) : Donnent le maximum de la commande sur la période (period=[month,day,hour,min] ou [expression PHP](http://php.net/manual/fr/datetime.formats.relative.php)) ou entre les 2 bornes demandées (sous la forme Y-m-d H:i:s ou [expression PHP](http://php.net/manual/fr/datetime.formats.relative.php)) :
* duration(commande, valeur, période) et durationbetween(commande,valeur,start,end) : Donnent la durée en minutes pendant laquelle l’équipement avait la valeur choisie sur la période (period=[month,day,hour,min] ou [expression PHP](http://php.net/manual/fr/datetime.formats.relative.php)) ou entre les 2 bornes demandées (sous la forme Y-m-d H:i:s ou [expression PHP](http://php.net/manual/fr/datetime.formats.relative.php)) :
* statistics(commande,calcul,période) et statisticsBetween(commande,calcul,start,end) : Donnent le résultat de différents calculs statistiques (sum, count, std, variance, avg, min, max) sur la période (period=[month,day,hour,min] ou [expression PHP](http://php.net/manual/fr/datetime.formats.relative.php)) ou entre les 2 bornes demandées (sous la forme Y-m-d H:i:s ou [expression PHP](http://php.net/manual/fr/datetime.formats.relative.php)) :
* tendance(commande,période,seuil) : Donne la tendance de la commande sur la période (period=[month,day,hour,min] ou [expression PHP](http://php.net/manual/fr/datetime.formats.relative.php)) :
* stateDuration(commande) : Donne la durée en secondes depuis le dernier changement de valeur. Retourne -1 si aucun historique n’existe ou si la valeur n’existe pas dans l’historique. Retourne -2 si la commande n’est pas historisée.
* lastChangeStateDuration(commande,valeur) : Donne la durée en secondes depuis le dernier changement d’état à la valeur passée en paramètre. Retourne -1 si aucun historique n’existe ou si la valeur n’existe pas dans l’historique. Retourne -2 si la commande n’est pas historisée
* lastStateDuration(commande,valeur) : Donne la durée en secondes pendant laquelle l’équipement a dernièrement eu la valeur choisie. Retourne -1 si aucun historique n’existe ou si la valeur n’existe pas dans l’historique. Retourne -2 si la commande n’est pas historisée
* stateChanges(commande,[valeur], période) et stateChangesBetween(commande, [valeur], start, end) : Donnent le nombre de changements d’état (vers une certaine valeur si indiquée, ou au total sinon) sur la période (period=[month,day,hour,min] ou [expression PHP](http://php.net/manual/fr/datetime.formats.relative.php)) ou entre les 2 bornes demandées (sous la forme Y-m-d H:i:s ou [expression PHP](http://php.net/manual/fr/datetime.formats.relative.php)) :
* lastBetween(commande,start,end) : Donne la dernière valeur enregistrée pour l’équipement entre les 2 bornes demandées (sous la forme Y-m-d H:i:s ou [expression PHP](http://php.net/manual/fr/datetime.formats.relative.php)) :
* variable(mavariable,valeur par défaut) : Récupère la valeur d’une variable ou de la valeur souhaitée par défaut :
* scenario(scenario) : Renvoie le statut du scénario. 1 en cours, 0 si arrêté et -1 si désactivé, -2 si le scénario n’existe pas et -3 si l’état n’est pas cohérent. Pour avoir le nom “humain” du scénario, vous pouvez utiliser le bouton dédié à droite de la recherche de scénario.
* lastScenarioExecution(scenario) : Donne la durée en secondes depuis le dernier lancement du scénario :
* collectDate(cmd,[format]) : Renvoie la date de la dernière donnée pour la commande donnée en paramètre, le 2ème paramètre optionnel permet de spécifier le format de retour (détails [ici](http://php.net/manual/fr/function.date.php)). Un retour de -1 signifie que la commande est introuvable et -2 que la commande n’est pas de type info
* valueDate(cmd,[format]) : Renvoie la date de la dernière donnée pour la commande donnée en paramètre, le 2ème paramètre optionnel permet de spécifier le format de retour (détails [ici](http://php.net/manual/fr/function.date.php)). Un retour de -1 signifie que la commande est introuvable et -2 que la commande n’est pas de type info
* eqEnable(equipement) : Renvoie l’état de l’équipement. -2 si l’équipement est introuvable, 1 si l’équipement est actif et 0 s’il est inactif
* value(cmd) : Renvoie la valeur d’une commande si elle n’est pas donnée automatiquement par Jeedom (cas lors du stockage du nom de la commande dans une variable)
* tag(montag,[defaut]) : Permet de récupérer la valeur d’un tag ou la valeur par défaut si il n’existe pas :
* name(type,commande) : Permet de récuperer le nom de la commande, de l’équipement ou de l’objet. Type vaut soit cmd, eqLogic ou object.
* lastCommunication(equipment,[format]) : Renvoie la date de la dernière communication pour l’équipement donnée en paramètre, le 2ème paramètre optionnel permet de spécifier le format de retour (détails [ici](http://php.net/manual/fr/function.date.php)). Un retour de -1 signifie que l’équipment est introuvable
* color\_gradient(couleur\_debut,couleur\_fin,valuer\_min,valeur\_max,valeur) : Renvoi une couleur calculé par rapport à valeur dans l’intervalle couleur\_debut/couleur\_fin. La valeur doit etre comprise entre valeur\_min et valeur\_max

Les périodes et intervalles de ces fonctions peuvent également s’utiliser avec [des expressions PHP](http://php.net/manual/fr/datetime.formats.relative.php) comme par exemple :

* Now : maintenant
* Today : 00:00 aujourd’hui (permet par exemple d’obtenir des résultats de la journée si entre ‘Today’ et ‘Now’)
* Last Monday : lundi dernier à 00:00
* 5 days ago : il y a 5 jours
* Yesterday noon : hier midi
* Etc.

Voici des exemples pratiques pour comprendre les valeurs retournées par ces différentes fonctions :

| **Prise ayant pour valeurs :** | **000 (pendant 10 minutes) 11 (pendant 1 heure) 000 (pendant 10 minutes)** |
| --- | --- |
| average(prise,période) | Renvoie la moyenne des 0 et 1 (peut |
|  | être influencée par le polling) |
| averageBetween(#[Salle de bain][Hydrometrie][Humidité]#,2015-01-01 00:00:00,2015-01-15 00:00:00) | Renvoie la moyenne de la commande entre le 1 janvier 2015 et le 15 janvier 2015 |
| min(prise,période) | Renvoie 0 : la prise a bien été éteinte dans la période |
| minBetween(#[Salle de bain][Hydrometrie][Humidité]#,2015-01-01 00:00:00,2015-01-15 00:00:00) | Renvoie le minimum de la commande entre le 1 janvier 2015 et le 15 janvier 2015 |
| max(prise,période) | Renvoie 1 : la prise a bien été allumée dans la période |
| maxBetween(#[Salle de bain][Hydrometrie][Humidité]#,2015-01-01 00:00:00,2015-01-15 00:00:00) | Renvoie le maximum de la commande entre le 1 janvier 2015 et le 15 janvier 2015 |
| duration(prise,1,période) | Renvoie 60 : la prise était allumée (à 1) pendant 60 minutes dans la période |
| durationBetween(#[Salon][Prise][Etat]#,0,Last Monday,Now) | Renvoie la durée en minutes pendant laquelle la prise était éteinte depuis lundi dernier. |
| statistics(prise,count,période) | Renvoie 8 : il y a eu 8 remontées d’état dans la période |
| tendance(prise,période,0.1) | Renvoie -1 : tendance à la baisse |
| stateDuration(prise) | Renvoie 600 : la prise est dans son état actuel depuis 600 secondes (10 minutes) |
| lastChangeStateDuration(prise,0) | Renvoie 600 : la prise s’est éteinte (passage à 0) pour la dernière fois il y a 600 secondes (10 minutes) |
| lastChangeStateDuration(prise,1) | Renvoie 4200 : la prise s’est allumée (passage à 1) pour la dernière fois il y a 4200 secondes (1h10) |
| lastStateDuration(prise,0) | Renvoie 600 : la prise est éteinte depuis 600 secondes (10 minutes) |
| lastStateDuration(prise,1) | Renvoie 3600 : la prise a été allumée pour la dernière fois pendant 3600 secondes (1h) |
| stateChanges(prise,période) | Renvoie 3 : la prise a changé 3 fois d’état pendant la période |
| stateChanges(prise,0,période) | Renvoie 2 : la prise s’est éteinte (passage à 0) deux fois pendant la période |
| stateChanges(prise,1,période) | Renvoie 1 : la prise s’est allumée (passage à 1) une fois pendant la période |
| lastBetween(#[Salle de bain][Hydrometrie][Humidité]#,Yesterday,Today) | Renvoie la dernière température enregistrée hier. |
| variable(plop,10) | Renvoie la valeur de la variable plop ou 10 si elle est vide ou n’existe pas |
| scenario(#[Salle de bain][Lumière][Auto]#) | Renvoie 1 en cours, 0 si arreté et -1 si desactivé, -2 si le scénario n’existe pas et -3 si l’état n’est pas cohérent |
| lastScenarioExecution(#[Salle de bain][Lumière][Auto]#) | Renvoie 300 si le scénario s’est lancé pour la dernière fois il y a 5 min |
| collectDate(#[Salle de bain][Hydrometrie][Humidité]#) | Renvoie 2015-01-01 17:45:12 |
| valueDate(#[Salle de bain][Hydrometrie][Humidité]#) | Renvoie 2015-01-01 17:50:12 |
| eqEnable(#[Aucun][Basilique]#) | Renvoie -2 si l’équipement est introuvable, 1 si l’équipement est actif et 0 s’il est inactif |
| tag(montag,toto) | Renvoie la valeur de “montag” si il existe sinon renvoie la valeur “toto” |
| name(eqLogic,#[Salle de bain][Hydrometrie][Humidité]#) | Renvoie Hydrometrie |

Les fonctions mathématiques

Une boîte à outils de fonctions génériques peut également servir à effectuer des conversions ou des calculs :

* rand(1,10) : Donne un nombre aléatoire de 1 à 10.
* randText(texte1;texte2;texte…​..) : Permet de retourner un des textes aléatoirement (séparer les texte par un ; ). Il n’y a pas de limite dans le nombre de texte.
* randomColor(min,max) : Donne une couleur aléatoire compris entre 2 bornes ( 0 => rouge, 50 => vert, 100 => bleu).
* trigger(commande) : Permet de connaître le déclencheur du scénario ou de savoir si c’est bien la commande passée en paramètre qui a déclenché le scénario.
* triggerValue(commande) : Permet de connaître la valeur du déclencheur du scénario.
* round(valeur,[decimal]) : Donne un arrondi au-dessus, [decimal] nombre de décimales après la virgule.
* odd(valeur) : Permet de savoir si un nombre est impair ou non. Renvoie 1 si impair 0 sinon.
* median(commande1,commande2…​.commandeN) : Renvoie la médiane des valeurs.
* time\_op(time,value) : Permet de faire des opérations sur le temps, avec time=temps (ex : 1530) et value=valeur à ajouter ou à soustraire en minutes.
* time\_between(time,start,end) : Permet de tester si un temps est entre deux valeurs avec time=temps (ex : 1530), start=temps, end=temps. Les valeurs start et end peuvent être à cheval sur minuit.
* time\_diff(date1,date1[,format]) : Permet de connaître la différence entre 2 dates (les dates doivent être au format AAAA/MM/JJ HH:MM:SS). Par défaut (si vous ne mettez rien pour format), la méthode retourne le nombre total de jours. Vous pouvez lui demander en secondes (s), minutes (m), heures (h). Exemple en secondes time\_diff(2018-02-02 14:55:00,2018-02-25 14:55:00,s)
* formatTime(time) : Permet de formater le retour d’une chaine #time#.
* floor(time/60) : Permet de convertir des secondes en minutes, ou des minutes en heures (floor(time/3600) pour des secondes en heures)

Et les exemples pratiques :

| **Exemple de fonction** | **Résultat retourné** |
| --- | --- |
| randText(il fait #[salon][oeil][température]#;La température est de #[salon][oeil][température]#;Actuellement on a #[salon][oeil][température]#) | la fonction retournera un de ces textes aléatoirement à chaque exécution. |
| randomColor(40,60) | Retourne une couleur aléatoire proche du vert. |
| trigger(#[Salle de bain][Hydrometrie][Humidité]#) | 1 si c’est bien #[Salle de bain][Hydrometrie][Humidité]# qui a déclenché le scénario sinon 0 |
| triggerValue(#[Salle de bain][Hydrometrie][Humidité]#) | 80 si l’hydrométrie de #[Salle de bain][Hydrometrie][Humidité]# est de 80 %. |
| round(#[Salle de bain][Hydrometrie][Humidité]# / 10) | Renvoie 9 si le pourcentage d’humidité et 85 |
| odd(3) | Renvoie 1 |
| median(15,25,20) | Renvoie 20 |
| time\_op(#time#, -90) | s’il est 16h50, renvoie : 1650 - 0130 = 1520 |
| formatTime(1650) | Renvoie 16h50 |
| floor(130/60) | Renvoie 2 (minutes si 130s, ou heures si 130m) |

Les commandes spécifiques

En plus des commandes domotiques, vous avez accès aux actions suivantes :

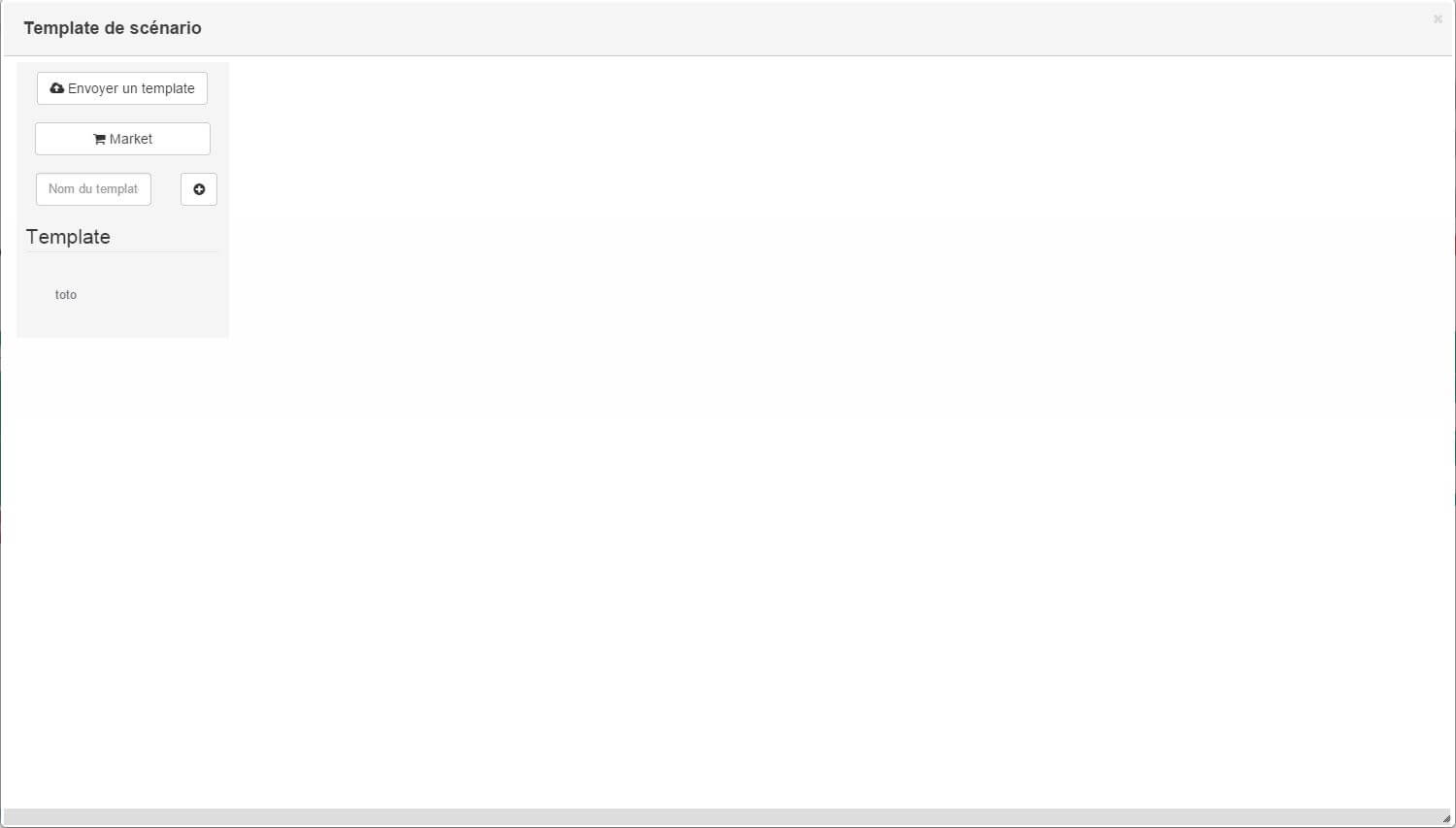
* Pause (sleep) : Pause de x seconde(s).
* variable (variable) : Création/modification d’une variable ou de la valeur d’une variable.
* Supprimer variable (delete\_variable) : Permet de supprimer une variable
* Scénario (scenario) : Permet de contrôler des scénarios. La partie tags permet d’envoyer des tags au scénario, ex : montag=2 (attention il ne faut utiliser que des lettre de a à z. Pas de majuscules, pas d’accents et pas de caractères spéciaux). On récupère le tag dans le scénario cible avec la fonction tag(montag). La commande “Remise à zéro des SI” permet de remettre à zéro le status des “SI” (ce status est utilisé pour la non répétition des actions d’un “SI” si on passe pour la 2ème fois consécutive dedans)
* Stop (stop) : Arrête le scénario.
* Attendre (wait) : Attend jusqu’à ce que la condition soit valide (maximum 2h), le timeout est en seconde(s).
* Aller au design (gotodesign) : Change le design affiché sur tous les navigateurs par le design demandé.
* Ajouter un log (log) : Permet de rajouter un message dans les logs.
* Créer un message (message) : Permet d’ajouter un message dans le centre de messages.
* Activer/Désactiver Masquer/afficher un équipement (equipement) : Permet de modifier les propriétés d’un équipement visible/invisible, actif/inactif.
* Faire une demande (ask) : Permet d’indiquer à Jeedom qu’il faut poser une question à l’utilisateur. La réponse est stockée dans une variable, il suffit ensuite de tester sa valeur. Pour le moment, seuls les plugins sms et slack sont compatibles. Attention, cette fonction est bloquante. Tant qu’il n’y a pas de réponse ou que le timeout n’est pas atteint, le scénario attend.
* Arrêter Jeedom (jeedom\_poweroff) : Demande à Jeedom de s’éteindre.
* Redémarrer Jeedom (jeedom\_reboot) : Demande à Jeedom de redémarrer.
* Retourner un texte/une donnée (scenario\_return) : Retourne un texte ou une valeur pour une interaction par exemple.
* Icône (icon) : Permet de changer l’icône de représentation du scénario.
* Alerte (alert) : Permet d’afficher un petit message d’alerte sur tous les navigateurs qui ont une page Jeedom ouverte. Vous pouvez, en plus, choisir 4 niveaux d’alerte.
* Pop-up (popup) : Permet d’afficher un pop-up qui doit absolument être validé sur tous les navigateurs qui ont une page jeedom ouverte.
* Rapport (report) : Permet d’exporter une vue au format (PDF,PNG, JPEG ou SVG) et de l’envoyer par le biais d’une commande de type message. Attention, si votre accès Internet est en HTTPS non-signé, cette fonctionalité ne marchera pas. Il faut du HTTP ou HTTPS signé.
* Supprimer bloc DANS/A programmé (remove\_inat) : Permet de supprimer la programmation de tous les blocs DANS et A du scénario.
* Evènement (event) : Permet de pousser une valeur dans une commande de type information de manière arbitraire
* Tag (tag) : Permet d’ajouter/modifier un tag (le tag n’existe que pendant l’exécution en cours du scénario à la difference des variables qui survivent à la fin du scénario)

Template de scénario

Cette fonctionalité permet de transformer un scénario en template pour par exemple l’appliquer sur un autre Jeedom ou le partager sur le Market. C’est aussi à partir de là que vous pouvez récupérer un scénario du Market.

scenario15

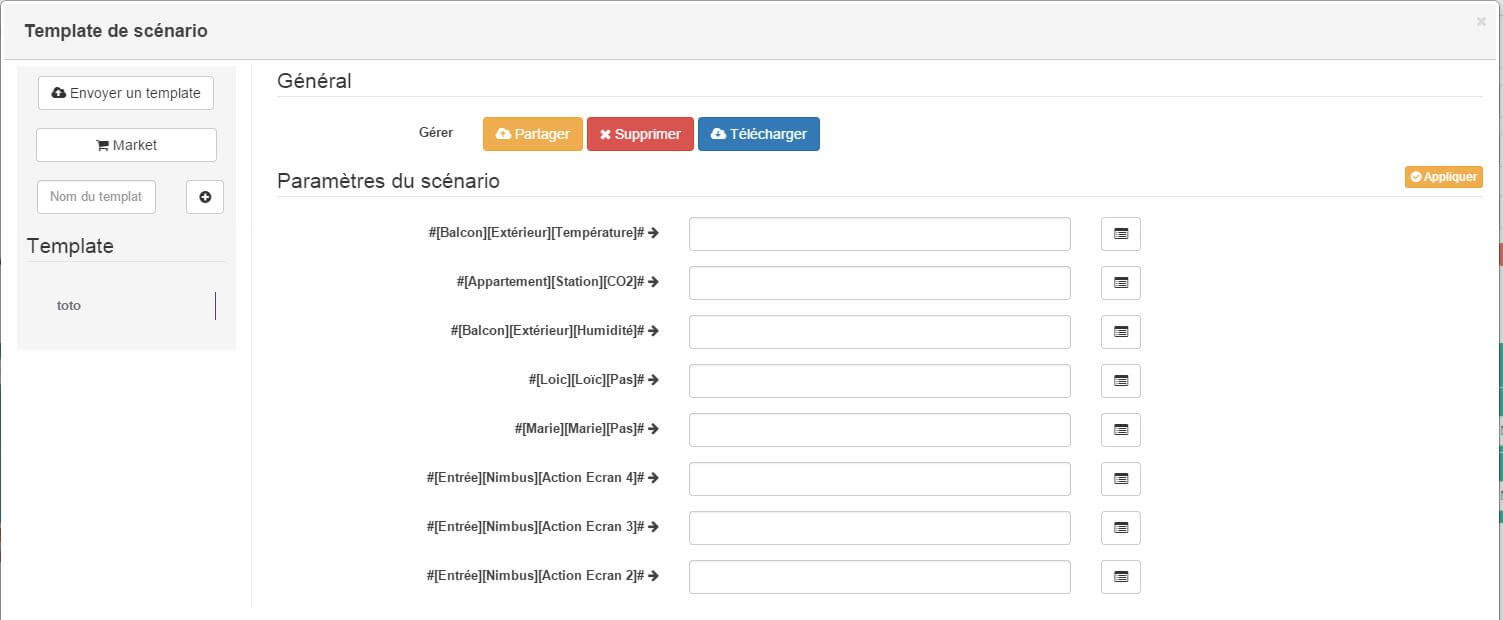
Vous verrez alors cette fenêtre :



A partir de celle-ci, vous avez la possibilité :

* D’envoyer un template à Jeedom (fichier JSON préalablement récupéré),
* De consulter la liste des scénarios disponibles sur le Market,
* De créer un template à partir du scénario courant (n’oubliez pas de donner un nom),
* De consulter les templates actuellement présents sur votre Jeedom.

Par un clic sur un template, vous obtenez :



En haut, vous pouvez :

* Partager : partager le template sur le Market,
* Supprimer : supprimer le template,
* Télécharger : récupérer le template sous forme de fichier JSON pour le renvoyer sur un autre Jeedom par exemple.

En-dessous, vous avez la partie pour appliquer votre template au scénario courant.

Etant donné que d’un Jeedom à l’autre ou d’une installation à une autre, les commandes peuvent être différentes, Jeedom vous demande la correspondance des commandes entre celles présentes lors de la création du template et celles présentes chez vous. Il vous suffit de remplir la correspondance des commandes puis de faire appliquer.

Ajout de fonction php

IMPORTANT

L’ajout de fonction PHP est reservé aux utilisateurs avancés. La moindre erreur peut faire planter votre Jeedom

Mise en place

Aller dans la configuration de Jeedom, puis OS/DB et lancer l’éditeur de fichier.

Allez dans le dossier data puis php et cliquez sur le fichier user.function.class.php.

C’est dans cette class que vous devez ajouter vos fonctions, vous y trouverez un exemple de fonction basique.