Halloween Camp

version 1.0

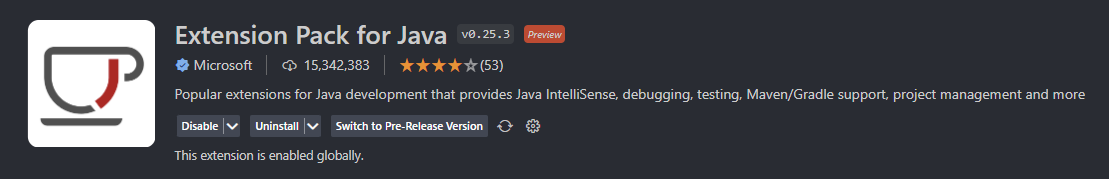




# Prérequis

Pour ce sujet, il faudra installer plusieurs choses dont :

* Le Java Extension Pack sur VSCode (et VSCode si vous ne l’aviez pas):



* Maven: (<https://dlcdn.apache.org/maven/maven-3/3.8.6/binaries/apache-maven-3.8.6-bin.zip>)

Apache Maven Site

* OpenJDK: (<https://jdk.java.net/19/>)



Une fois tout cela installé, il vous faudra une base pour pouvoir commencer :

* Serveur Java Basique (à dézipper sur votre bureau):
* Workspace pour le plugin:

# Consignes

* Si vous avez des questions, demandez à votre voisin de gauche, puis de droite. Coopérez !
* En cas de blocage, et après avoir tenté la consigne précédente, n’hésitez pas à demander à un Cobra, ils sont très gentils et sont là pour vous aider.
* Vous avez entièrement le droit d’utiliser internet comme vous le souhaitez pour vous renseigner, nous le conseillons fortement.
* *N’hésitez pas à faire des bonus et ajouter/modifier des fonctionnalités à votre programme, la seule limite est votre imagination ! (Et 6h)*

* Amusez-vous ! Nous ne sommes pas là pour apprendre mais pour découvrir, vous avez le droit de ne pas réussir à compléter le sujet. 😊
* Si vous voulez un sujet plus simple, vous êtes libre de faire celui en Skript

# Le Serveur

Une fois le serveur dézippé, le fichier « start.bat » sera utilisé pour le lancer. Pour vous connecter au serveur, vous aurez juste besoin de mettre « localhost » comme IP dans le jeu.

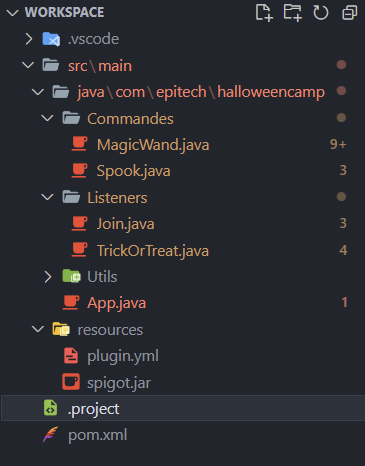
A picture containing graphical user interface

Description automatically generated

À chaque fois que le plugin est compilé, vous devrez le remettre dans le dossier « plugin » du serveur.

# Le WorkSpace

Le Workspace doit ressembler à cela :



Le dossier *« .vscode »* contient des réglages basique pour que vous puissiez coder en java sans trop de problèmes. Le dossier *« resources »* contient le plugin.yml. C’est le fichier où vous alliez écrire des informations sur le plugin, dont le nom, la version, les commandes, les alias… Et le dernier dossier, le dossier *« halloweencamp »* qui lui contiendra tous les fichiers java pour le plugin, et où vous allez coder !

Le fichier *pom.xml* est déjà prêt, c’est le fichier qui va vous permettre de compiler facilement et le *.project* qui contient d’autres informations sur le projet.

Pour compiler le code quand cela est demandé, vous auriez juste besoin de lancer la commande suivante : « & mvn install -f .\pom.xml » (si vous êtes sur linux, remplacez \ par /).

Pour ouvrir un terminal, sur VSCode, cliquez sur « Terminal » en haut de la fenêtre et puis sur « Nouveau Terminal ». Simple.

Bonne Chance !

# Lancement du serveur !

Lancez le serveur en ouvrant le fichier « start.bat » et laissez le terminal ouvert. Il vous sera utile (il est tout simplement nécéssaire de le laisser ouvert pour garder le serveur ouvert).

Qr code

Description automatically generated

Si la dernière ligne du terminal, dit quelque chose de similaire à cela, c’est que tout s’est bien passé !



Tu as juste à te connecter une première fois et lancer la commande « op (ton nom) » sur le terminal pour te donner toutes permissions



# Le code !🧑‍💻

Maintenant que tout est prêt, tu peux commencer ! Lance VSCode, et c’est parti !

1. **Commande /spook**

Nous allons commencer par la commande /spook. Créer une commande en Java, c’est long. C’est pour cela qu’on vous a donné la base.

Dirige-toi dans le dossier « Commandes » à l’intérieur du dossier « halloweencamp ». À l’intérieur, tu trouveras un fichier « Spook.java », ouvres le.

C’est en dessous du commentaire que nous allons coder la fin de cette fonction.  
Actuellement, cette fonction stock juste le joueur dans une variable.

Text

Description automatically generated

Donc, il faudrait faire un check d’argument, pour voir si le premier argument est bon, comme ceci.



Maintenant, il faudrait peut-être vérifier si le joueur est en ligne… Ça ne sert à rien de faire peur a un joueur qui n’existe pas…



Et maintenant, si tout se passe bien, on fait peur au joueur en lui envoyant l’effet du Elder Gardien !



Maintenant que la commande est faite, il faut la définir quelque part. C’est dans « App.java » et dans « plugin.yml » que tu vas la mettre.

Ouvre « App.java » et modifie les points d’interrogations par « spook », pour que quand /spook est exécuté dans le jeu, que ça exécute le code que l’on vient de faire.



Dirige-toi maintenant dans « plugin.yml » qui est dans « resources », et modifie les points d’interrogations par la même commande.



Tu peux maintenant compiler le code pour tester ta commande !

Text

Description automatically generated

Si le terminal te renvoie BUILD SUCCESS, c’est que tout s’est bien passé et que tu peux tester ton plugin !

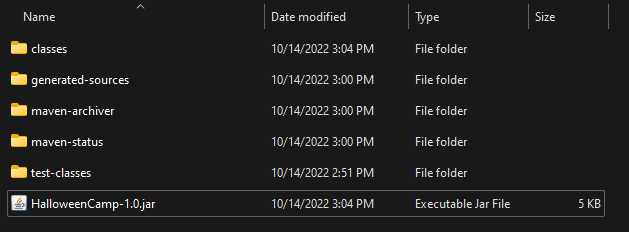
Text

Description automatically generated

Dans ton workspace, tu as dû voir un dossier « target » qui est apparu. Ce dossier contient ton plugin compilé

Graphical user interface, text, application

Description automatically generated



Ce fichier « HalloweenCamp-1.0.jar », tu peux le mettre dans le dossier « plugin » de ton serveur.

A screenshot of a computer

Description automatically generated with medium confidence

Tu reprends le terminal que tu as, bien sûr, laissé ouvert. Et tu écris « restart ». Le serveur va se redémarrer et va lancer ton plugin au redémarrage !

Text

Description automatically generated

Si tu vois ça au démarrage, nikel ! Tout s’est bien passé !



Maintenant c’est l’heure de vérité… connecte toi au serveur… et lances la commande /spook !



Si tout marche bien, bravo ! Sinon… appelle un Cobra, ils vont t’aider à voir où est le problème.

1. **La citrouille sur la tête**

Bon, j’avoue, la commande /spook n’a pas été super simple. Maintenant on va essayer de faire quelque chose d’un peu plus facile.

Dans cette partie, on va faire en sorte que quand un joueur rejoint le serveur, s’il n’a rien sur la tête ça lui met une Jack O’ Lantern. Pour ça, on va devoir créer un Listener. Un Listener va écouter un événement donné et exécuter une fonction quand cette action est faite. Par exemple, nous allons utiliser le Listener pour regarder quand un joueur rejoint le serveur avec « PlayerJoinEvent ».

Redirige-toi dans le dossier « Listeners » et ouvre le fichier « Join.java ». Il doit ressembler à ça :



D’abord on va stocker le joueur dans une variable pour que ça soit plus simple de l’appeler :



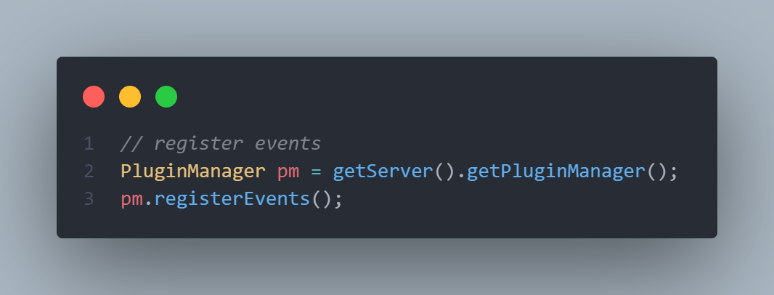
Maintenant on va regarder s’il a rien à la place de son casque en récupérant son inventaire avec getInventory() et puis getHelmet() pour récupérer le Helmet Slot :



Et si le joueur à rien, on set le helmet du joueur en Jack o lantern en faisant getInventory().setHelmet(new ItemStack(Material.JACK\_O\_LANTERN)) (vu que le helmet doit être un ItemStack).



Maintenant on enregistre l’événement dans « App.java ». À l’intérieur des parenthèses de « pm.registerEvents(); », il faut appeler l’événement et en deuxième argument, mettre l’application, dans ce cas on peut juste mettre « this ».



Un événement on l’appelle en mettant « new » et puis le nom de la fonction (d’habitude, le nom du fichier, dans ce cas, « Join() ». Voici ce que cela donne une fois finis :



Maintenant tu peux compiler, redémarrer et tester ! 😊 (Voir page 9-11) Si tout s’est bien passé, tu devrais recevoir une Jack O’ Lantern quand tu te connectes !

A picture containing text, clock

Description automatically generated

1. **Un objet ou un sort !**

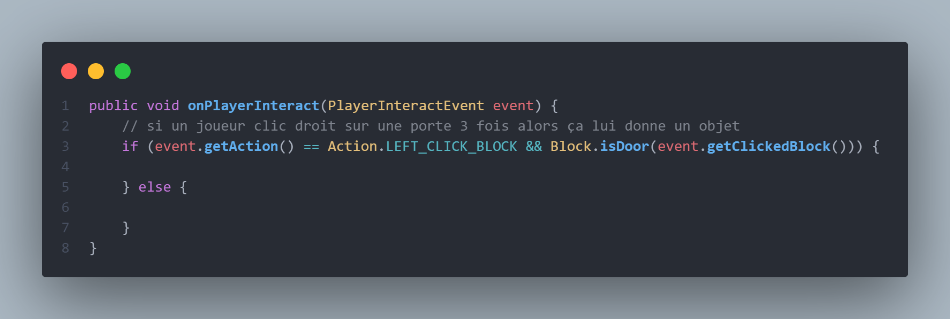
Pour la troisième tache, tu devras faire en sorte que quand tu frappes à une porte 3 fois, tu récupères un item. En vrai c’est très facile.  
Dirige-toi vers le dossier « Listener » encore une fois, dedans tu trouveras « TrickOrTreat.java » :



Dans le dossier « Utils » une fonction a été crée pour te simplifier la vie. Tu auras juste à utiliser « Block.isDoor(*bloc ici*) » pour check si le bloc qui a été donné est bien une porte.

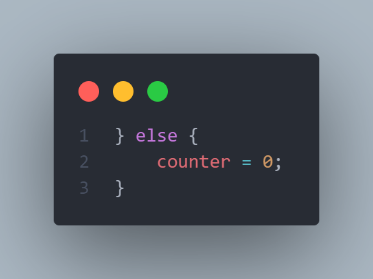
Donc, on va d’abord rajouter un « if » simple pour savoir si l’interaction est un clic gauche et si le bloc cliqué est une porte. As-tu une idée de comment le faire ? Un indice, pour récupérer l’action tu peux utiliser « event.getAction() » et il doit être égal à « Action.LEFT\_CLICK\_BLOCK ». Pour récupérer le bloc c’est « event.getClickedBlock() ».

Voici ce que ça doit donner : As-tu réussi ? 👀

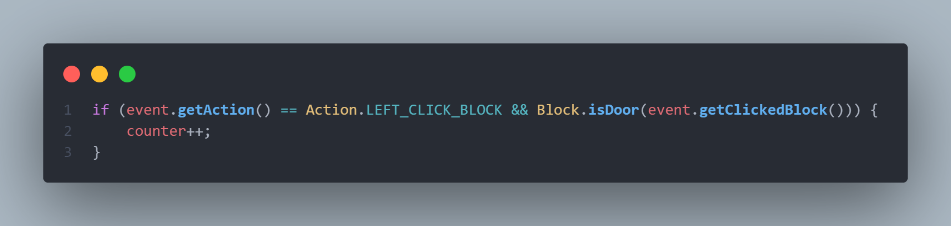


Oui, c’est moche.

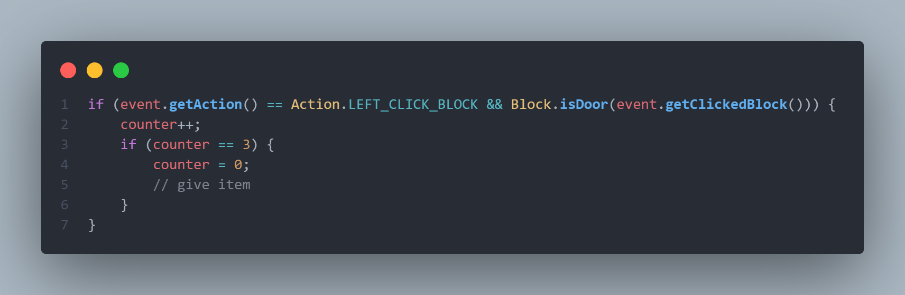
Bref, si la condition est fausse, on veut réinitialiser le compteur à 0, au cas où il a été incrémenté. Facile.



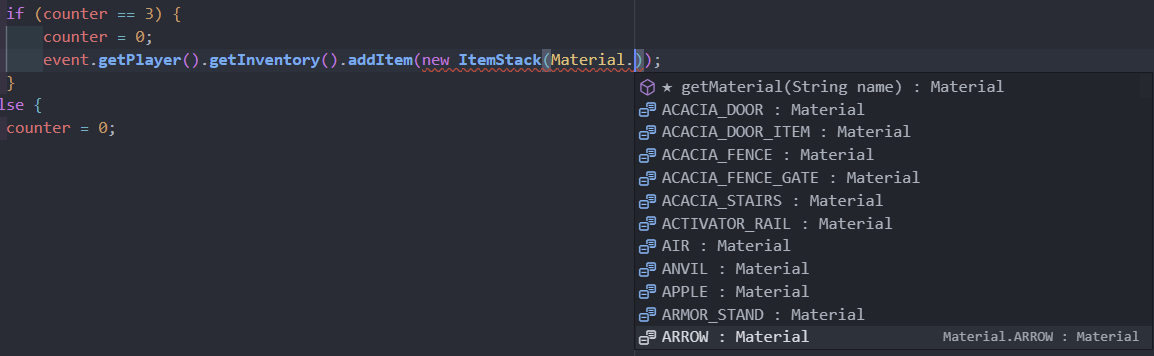
Et si la condition est vraie, on veut incrémenter le compteur de 1.



Maintenant le compteur augmente, cool. Mais il faudrait vérifier s’il atteint 3 pour pouvoir donner un objet et réinitialiser le compteur… Un simple if et c’est fait ^^



Pour donner un objet, c’est similaire au casque de tout à l’heure. On récupère le joueur (event.getPlayer()), son inventaire (getInventory()) sauf que cette fois au lieu de faire setHelmet() on va tout simplement faire addItem(). Tu peux mettre n’importe quel item, si tu n’as pas d’idée, mets « Material.DIAMOND » :



Si ton code ressemble à peu près à ça, tu peux passer à l’étape suivante !



Maintenant tu peux retourner sur « App.java » et enregistrer l’événement (cf Page 13).

Si tu as tout bien fait, tu devrais avoir cela :



Compile, et test ! (Voir page 9-11, si tu es toujours perdu)

Tu peux te rendre aux coordonnées -2292 102 144.0 (en utilisant la commande « /tp -2292 102 144.0 » si tu n’as pas été téléporté au bon endroit



Si ça ne marche pas, tu peux toujours demander de l’aide à un Cobra !

1. **Un dernier pour la route ?**

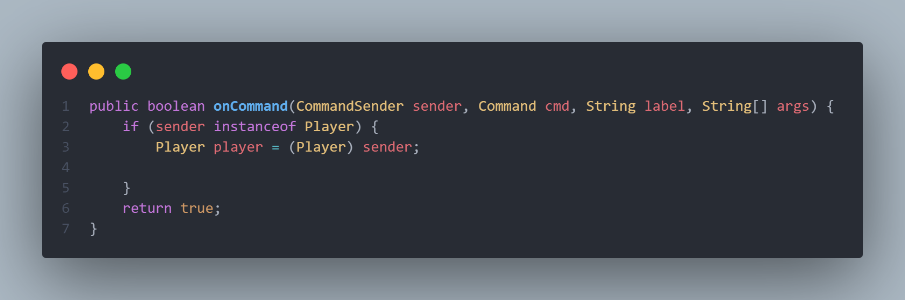
Bon, j’espère que ce n’était pas trop facile… Sinon, accroches toi bien. Pour cette dernière tache, on va faire une baguette magique qui spawn des chauves-souris et une sorcière une fois utilisée.

Dirige-toi vers le dossier « Commandes » encore une fois, dedans tu trouveras « MagicWand.java » :



La première partie, en dessous de « @Override » c’est la commande, qui va te donner une Baguette Magique, et la deuxième partie, « @EventHandler » c’est le Listener pour quand tu clic droit avec la Baguette Magique.

Commençons par la première partie :

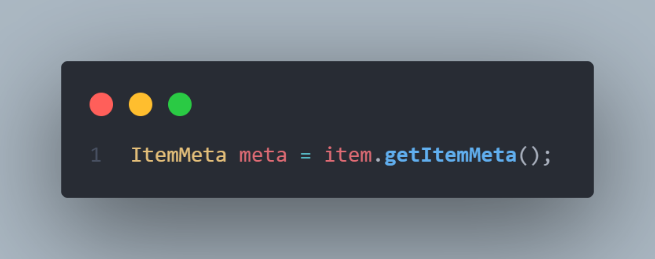


Ici, nous allons juste donner un bâton avec un nom spécial lors de l’exécution de la commande, rien de bien méchant.

Pour cela nous allons créer l’item et la mettre dans une variable en utilisant le code ci-dessous :



Puis, on va récupérer l’ItemMeta, qui contiens le nom de l’item, son lore, ses enchantements, tag NBT et d’autres infos. On va juste se concentrer sur le nom de l’item.



Maintenant, avec l’ItemMeta, on va pouvoir changer le nom de l’item (ou plutôt le set dans ce cas-là). Pour ceci, nous allons utiliser la fonction « setDisplayName() » - ce qui donne meta.setDisplayName(ChatColor.DARK\_PURPLE + ‘‘Baguette Magique’’) (pour donner une couleur a notre nom).

Après avoir modifié meta, on set l’ItemMeta d’item à notre nouveau meta. Le code doit ressembler à ceci :



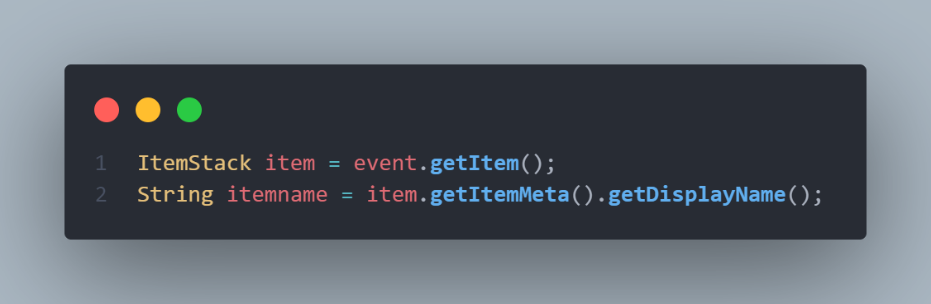
Et maintenant, on a juste besoin de donner l’item au joueur : player.getInventory().addItem(item), comme ce qui a été fait précédemment pour trick or treat. Votre fonction doit donc ressembler à ceci :



Pour la deuxième partie, nous allons faire en sorte que quand l’objet utilisé est une baguette magique, alors on fait apparaitre une sorcière entourée de chauve-souris :



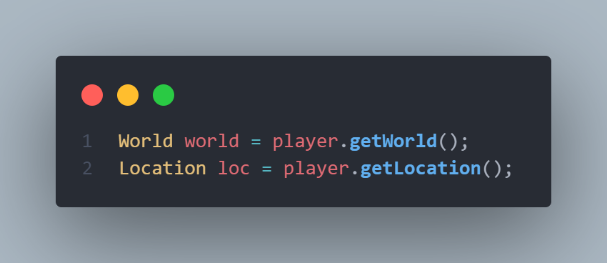
Pour faire ça proprement, nous allons stocker (dans le if) l’item utilisé (ainsi que son nom) dans une variable :



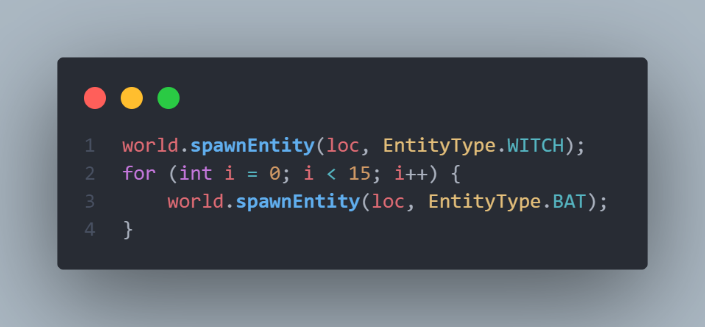
Juste en dessous, nous allons vérifier que le type de l’objet est bien un bâton avec « item.getType() » ainsi que son nom est bien « ChatColor.DARK\_PURPLE + ‘‘Baguette Magique’’ » avec « itemname.equals() ». Voici à quoi doit ressembler votre joli if.



Et dans ce if, nous allons mettre deux nouvelles variables qui seront utile pour spawn les mobs : La localisation du joueur (en utilisant player.getLocation()) et le monde (en utilisant player.getWorld())



Maintenant, vous pouvez spawn les mobs ! (world.spawnEntity(localisation, EntityType.MOB))



Dans cet exemple, une sorcière va être spawn ainsi que 15 chauve-souris lors de l’utilisation de la baguette magique.



Maintenant il faut juste enregistrer la commande ainsi que l’événement dans « App.java » !



N’oubliez pas d’aussi mettre la commande dans « plugin.yml » !



Compile, et test ! (Voir page 9-11, si tu es ENCORE perdu)





1. **Des Bonus ?**

On espère que vous aviez aimé ce sujet. Si vous voulez faire des bonus, voici quelques idées :

• Des messages en plus

• Faire spawn la sorcière 2 blocs en face du joueur et non sur le joueur

• Spawn un montant random de chauves-souris quand la Baguette Magique est utilisée

• Donner un item au hasard lors du trick or treat au lieu d’un item prédéfinis

• Réparer les erreurs potentiel qui peuvent apparaitre sur la console (notamment lors de l’utilisation de la baguette)

• Une chance sur deux de donner un effet de nausée ou blindness lors d’un trick or treat

• Un fichier de config pour le nombre de fois ou il faut toquer ou changer le nom de la baguette magique

• Plus d’effet lors d’un /spook

• Des permissions

• Des alias

Si tu veux d’autres idées et/ou projets (ou refaire un autre plugin (cette fois pas sur un thème halloween)), demande à un Cobra !