

Your First Script

Writing Your First Script

If you have launched the game at least once before you will find

`kubejs/server_scripts/example_server_script.js` It looks like this:

```
// priority: 0

settings.logAddedRecipes = true
settings.logRemovedRecipes = true
settings.logSkippedRecipes = false
settings.logErroringRecipes = true

console.info('Hello, World! (You will see this line every time server resources reload)')

onEvent('recipes', event => {
  // Change recipes here
})

onEvent('item.tags', event => {
  // Get the #forge:cobblestone tag collection and add Diamond Ore to it
  // event.get('forge:cobblestone').add('minecraft:diamond_ore')

  // Get the #forge:cobblestone tag collection and remove Mossy Cobblestone from it
  // event.get('forge:cobblestone').remove('minecraft:mossy_cobblestone')
})
```

Lets break it down:

- `// priority: 0`
 - Makes it so that if you have multiple server scripts, this script gets loaded first
 - If you have only one `server_script`, this has no effect
- `settings.logAddedRecipes = true`
`settings.logRemovedRecipes = true`
`settings.logSkippedRecipes = false`
`settings.logErroringRecipes = true`

- sets settings for what messages are logged
- You can remove all four of these lines if you want and it will only change what is put into the logs
- `console.info('Hello, World! (You will see this line every time server resources reload)')`
 - Prints the message in the log
 - This line is useless other than example and should be removed eventually
- `onEvent('recipes', event => {`
 - This makes an event listener for the `recipes` event, and will run the code inside when and only when the `recipes` event is triggered
 - This is triggered when server resources reload
 - Which happens when the world load or the `/reload` command is used
- `// Change recipes here`
 - comment, an code in a line following `//` will be considered a comment and will not be run
 - Used for taking notes as you write the code
- `})`
 - Indicates the end of the 'recipes' event listener
- `onEvent('item.tags', event => {`
 - `// Get the #forge:cobblestone tag collection and add Diamond Ore to it`
 - `// event.get('forge:cobblestone').add('minecraft:diamond_ore')`
 - `// Get the #forge:cobblestone tag collection and remove Mossy Cobblestone from it`
 - `// event.get('forge:cobblestone').remove('minecraft:mossy_cobblestone')`
- `})`
 - Same thing as the other one but for the `item.tags` event
 - You can find the list of all event [here](#)

Finally Writing Code For Real

Lets start off by adding a recipe to craft flint from three gravel.

To do so, insert this code right after the **recipes event**.

```
event.shapeless("flint", ["gravel", "gravel", "gravel"])
```

It should look like this:

```
// priority: 0

settings.logAddedRecipes = true
settings.logRemovedRecipes = true
settings.logSkippedRecipes = false
settings.logErroringRecipes = true
```

```
console.info('Hello, World! (You will see this line every time server resources reload)')
```

```
onEvent('recipes', event => {  
  // Change recipes here  
  event.shapeless("flint", ["gravel", "gravel", "gravel"])  
})  
  
onEvent('item.tags', event => {  
  // Get the #forge:cobblestone tag collection and add Diamond Ore to it  
  // event.get('forge:cobblestone').add('minecraft:diamond_ore')  
  
  // Get the #forge:cobblestone tag collection and remove Mossy Cobblestone from it  
  // event.get('forge:cobblestone').remove('minecraft:mossy_cobblestone')  
})
```

Now lets test it!

Run the command `/reload` in game, then try crafting three gravel together in any order.

But how does it work?

- event
 - This is a variable that created with the arrow expression in `onEvent('recipes', event => {`
- .
 - The dot operator is used for calling a method of an object
 - In this case event is the object and shapeless is the method
- shapeless(
 - This is the method that is called by the dot operator on the event
 - It is taking two arguments, that being an item result and a array input
- "
 - Indicates the start of a string
- flint
 - The contents of the string
 - You can use `create:flour` , if it is from a different mod (`flint` is the same as `minecraft:flint` , and both are valid)
- "
 - Signifies the end of the string.
 - A string is simply a sequence of characters, or letters
 - You can read more about strings in JS [here](#).
- ,
 - separates different arguments in the method.

- [
 - Signifies the start of the array.
 - An array holds multiple values or any type, including other arrays.
 - You can read more about arrays in JS [here](#).
- "gravel", "gravel", "gravel"
 - The contents of the array
 - Arrays can hold an indefinite number of elements
-]
 - Closing the array
-)
 - Closing the method

There you go! You can make custom shapeless recipes!

If you want to make other types of recipes, learn about it [here](#), and if you have an addon that adds more recipe types, look at its mod page, or [here](#).

Revision #3

Created 7 October 2022 19:59:03 by Q6

Updated 27 January 2023 21:18:26 by Q6