

# KubeJS Create

[Create](#) integration for KubeJS. This mod allows you to add and properly edit recipes of Create mod in KubeJS scripts. All supported recipe types and examples are below. See [Recipes](#) page for more info.

## Simple Recipe Types

- createCrushing
- createCutting
- createMilling
- createBasin
- createMixing
  - supports *.heated()* and *.superheated()*
- createCompacting
  - supports *.heated()* and *.superheated()*
  - Can have any number of inputs
  - Used basin
- createPressing
  - Only has one item input
  - Used on any surface
- createSandpaperPolishing
- createSplashing
  - AKA Bulk Washing
- createDeploying
- createFilling
- createEmptying
- createHaunting

Bulk Smoking and Bulk Blasting recipes are auto generated from vanilla smelting, smoking, and blasting recipes.

- Bulk Smoking is vanilla smoking.
- Bulk Blasting is vanilla smelting (as long as there is not a smoking recipe) or vanilla blasting.

## Syntax

**event.recipes.create.mixing(output[], input[])**

or

**event.recipes.createMixing(output[], input[])**

**Output** can be an item, fluid, or an array of multiple.

**Input** can be an ingredient, fluid, or an array of multiple.

## Examples

```
onEvent('recipes', event => {
  event.recipes.createCrushing([
    2x bone_meal',
    Item.of('5x bone_meal').withChance(0.5)
  ], 'bone_block')

  event.recipes.create.mixing(Fluid.of('create:builders_tea',500),[
    Fluid.of('milk',250),
    Fluid.of('water',250),
    '#leaves'
  ]).heated()

  event.recipes.createFilling('create:blaze_cake', [
    'create:blaze_cake_base',
    Fluid.of('minecraft:lava', 250)
  ])

  event.recipes.createEmptying([
    'minecraft:glass_bottle',
    Fluid.of('create:honey', 250)
  ], 'minecraft:honey_bottle')
})
```

# Mechanical Crafter

## Syntax

**event.recipes.create.mechanicalCrafting(output, pattern[], {patternKey: input})**

or

**event.recipes.createMechanicalCrafting(output, pattern[], {patternKey: input})**

This recipe type is the same as regular crafting table shaped recipe, however the pattern can be up to 9x9, instead of 3x3.

## Examples

```
onEvent('recipes', event => {
  event.recipes.createMechanicalCrafting('minecraft:piston', [
    'CCCCC',
    'CPIPC',
    'CPRPC'
  ], {
    C: '#forge:cobblestone',
    P: '#minecraft:planks',
    R: '#forge:dusts/redstone',
    I: '#forge:ingots/iron'
  })
})
```

# Sequenced Assembly

## Syntax

**event.recipes.create.sequencedAssembly(output[], input, sequence[]).transitionalItem(transitionalItem).loops(loops)**

or

**event.recipes.createSequencedAssembly(output[], input, sequence[]).transitionalItem(transitionalItem).loops(loops)**

**Output** is an item or an array of items.

If it is an array:

- The first item is the real output, the remainder are scrap.
- Only one item is chosen, with equal chance of each.
- You can use `Item.of('create:shaft').withChance(2)` to double the chance of that specific item to being chosen.

**Input** is an ingredient.

**Transitional Item** is any item\* and is used during the intermediate stages of the assembly.

**Sequence** is an array of recipes.

- The only legal recipes are:
  - createCutting
  - createPressing
  - createDeploying
  - createFilling
- The transitional item needs to be the output of each of these recipes.
- The transitional item needs to be the an input of each of these recipes.

**Loops** is the number of time that the recipes repeats. Calling `.loops()` is optional, and defaults to 4.

## Examples

```
onEvent('recipes', event => {
  event.recipes.createSequencedAssembly([ // start the recipe
    Item.of('create:precision_mechanism').withChance(130.0), // this is the item that will appear in JEI as the result
    Item.of('create:golden_sheet').withChance(8.0), // the rest of these items will part of the scrap
    Item.of('create:andesite_alloy').withChance(8.0),
    Item.of('create:cogwheel').withChance(5.0),
    Item.of('create:shaft').withChance(2.0),
    Item.of('create:crushed_gold_ore').withChance(2.0),
    Item.of('2x gold_nugget').withChance(2.0),
    'iron_ingot',
    'clock'
  ], 'create:golden_sheet', [ // 'create:golden_sheet' is the input
    // the transitional item set by "transitionalItem('create:incomplete_large_cogwheel')" is the item used during the
    intermediate stages of the assembly
    event.recipes.createDeploying('create:incomplete_precision_mechanism', ['create:incomplete_precision_mecha
nism', 'create:cogwheel']),
    // like a normal recipe function, is used as a sequence step in this array. Input and output have the transitional
    item
    event.recipes.createDeploying('create:incomplete_precision_mechanism', ['create:incomplete_precision_mecha
nism', 'create:large_cogwheel']),
    event.recipes.createDeploying('create:incomplete_precision_mechanism', ['create:incomplete_precision_mecha
nism', 'create:iron_nugget'])
  ]).transitionalItem('create:incomplete_precision_mechanism').loops(5) // set the transitional item and the loops
  (amount of repetitions)

  // for this code to work, kubejs:incomplete_spore_blossom need to be added to the game
  let inter = 'kubejs:incomplete_spore_blossom' // making a varriable to store the transition item makes the code
  more readable
```

```

event.recipes.createSequencedAssembly([
  item.of('spore_blossom').withChance(16.0), // this is the item that will appear in JEI as the result
  item.of('flowering_azalea_leaves').withChance(16.0), // the rest of these items will part of the scrap
  item.of('azalea_leaves').withChance(2.0),
  'oak_leaves',
  'spruce_leaves',
  'birch_leaves',
  'jungle_leaves',
  'acacia_leaves',
  'dark_oak_leaves'
], 'flowering_azalea_leaves', [ // 'flowering_azalea_leaves' is the input
  // the transitional item is a varriable, that is "kubejs:incomplete_spore_blossom", and is used during the
  intermediate stages of the assembly
  event.recipes.createPressing(inter, inter),
  // like a normal recipe function, is used as a sequence step in this array. Input and output have the transitional
  item
  event.recipes.createDeploying(inter, [inter, 'minecraft:hanging_roots']),
  event.recipes.createFilling(inter, [inter, Fluid.of('minecraft:water', 420)]),
  event.recipes.createDeploying(inter, [inter, 'minecraft:moss_carpet']),
  event.recipes.createCutting(inter, inter)
]).transitionalItem(inter).loops(2) // set the transitional item and the loops (amount of repetitions)
})

```

## Transitional Items

As mentioned earlier, any item can be a transition item. However, this is not completely recommended.

If you wish to make your own transitional item, its best if you make the type

`create:sequenced_assembly`.

### 1.16 syntax

```

onEvent('item.registry', event => {
  event.create('incomplete_spore_blossom').displayName('Incomplete Spore Blossom').type('create:sequenced_assembly')
})

```

### 1.18 syntax

```
onEvent('item.registry', event => {  
  event.create('incomplete_spore_blossom', 'create:sequenced_assembly')  
})
```

# Mysterious Conversion

Mysterious Conversion recipes are client side only, so the only way to add them currently is using reflection.

## Example

Goes inside of **client scripts** and **not in an event**.

```
//makes the varriables used  
let MysteriousItemConversionCategory =  
java('com.simibubi.create.compat.jei.category.MysteriousItemConversionCategory')  
let ConversionRecipe = java('com.simibubi.create.compat.jei.ConversionRecipe')  
  
//adds in the recipes  
MysteriousItemConversionCategory.RECIPES.add(ConversionRecipe.create('minecraft:apple', 'minecraft:carrot'))  
  
MysteriousItemConversionCategory.RECIPES.add(ConversionRecipe.create('minecraft:golden_apple',  
'minecraft:golden_carrot'))
```

# Preventing Recipe Auto-Generation

If you don't want a smelting, blasting, smoking, crafting, or stone-cutting to get an auto-generated counter part, then include `manual_only` at the end of the recipe id.

## Example

```
onEvent('recipes', event => {  
  event.shapeless('wet_sponge', ['water_bucket', 'sponge']).id('kubejs:moisting_the_sponge_manual_only')  
})
```

Other types of prevention, can be done in the create config (the goggles button leads you there).

If it is not in the config, then you can not change it.

---

Revision #10

Created 29 September 2022 23:51:55 by Q6

Updated 10 July 2023 16:19:23 by Lexxie