ScreenJS

Download: CurseForge

> The custom ContainerMenu event is a startup event.

Custom Container menus are created in a startup script. They cannot be reloaded without restarting the game. The event is not cancellable.

for block entities:

```
StartupEvents.registry('menu', event => {
    event.create('example_block' /*name can be anything*/, 'block_entity')
        .addSlot(-10, -10) // adds a slot into this x,y position on the texture
        .addSlot(10, 200)
        .loop(builder /*this builder*/=> {
            for(let x = 0; x < 9; x++) {
                for (let y = 0; y < 4; y++) {
                    builder.addSlot(x * 18 /*<- the width of a slot, remember to add this*/, y * 18, x + y * 4, 0)
                }
            }
        })
        .addOutputSlot(118, 118, 0, 0, 1, 'minecraft:smelting') // adds a slot you can't put an item into, but can pull an item from
  □□□□□□□□□□□□□□□□// LAST PARAMETER CAN BE NULL FOR NO OUTPUT HANDLING
  □□.inputSlotIndices(0) // sets a list of ITEM HANDLER indexes to handle as slotChanged callback input
        .playerInventoryY(100) // marks the start of the player's inventory on the texture
        .tintColor(0xFF00FF00) // a color to tint the whole inventory texture, in hexadecimal [a, r, g, b]
        .progressDrawable(50, 50, new Rectangle(0, 0, 10, 30), 'forge:textures/white.png', 'up', 'energy') // displays an energy bar from the blockentity's FE capability
  □□.slotChanged((menu, level, player, itemHandlers) => {
      □□console.info('' + player)
    □})
```

```
        .setBlockEntity('kubejs:example_block') // the block entity type that should open this
GUI on right-click
})
```

for any block:

```
StartupEvents.registry('menu', event => {
    event.create('grass_block' /*name can be anything*/, 'block')
        /*default parameter set*/
  □□.addItemHandler(9) // adds an item handler.
  □□.addItemHandler(1)
  □□.inputSlotIndices(0)
        .setBlock('minecraft:grass_block') // the block that should open this GUI on right-
click
})
```

for entities:

```
StartupEvents.registry('menu', event => {
    event.create('snow_golem' /*name can be anything*/, 'entity')
        /*default parameter set*/
        .setEntity('minecraft:snow_golem') // the enity type that should open this GUI on
right-click
})
```

and lastly, for completely separate 'basic' GUIs:

```
StartupEvents.registry('menu', event => {
    event.create('name_here' /*name can be anything*/)
        /*default parameter set*/
})
```

valid menu types:

- basic (this is the default)
- block_entity
- block
- entity

methods the menu builder supports:

- `addSlot(x, y, slotIndex, containerIndex)`
- `addOutputSlot(x, y, slotIndex, inputContainerIndex, outputContainerIndex, recipeType)`
- `loop(builder => ...)`
- `inputSlotIndices(int[] indexes)`
- `tintColor(color)`
- `drawable(screenX, screenY, rectangle, textureLocation)`
- `progressDrawable(x, y, rectangle, textureLocation, direction, type)`
- `fluidDrawable(x, y, rectangle, textureLocation, direction, tankIndex)`
- `customDrawable(x, y, rectangle, textureLocation, direction, (menu, screen, drawable, direction) => ...)`
- `backroundTexture(texture, rectangle)`
- `quickMoveFunc((player, slotIndex, menu) => ... return item)`
- `slotChanged((menu, level, player, itemHandler) => ...)`
- `validityFunc((player, pos) => ... return boolean)`
- `disablePlayerInventory()`
- `playerInventoryY(yPos)`
- `button(rectangle, textComponent, button => ...)`

default available types:

- PROGRESS
- FUEL
- ENERGY

default available move directions:

- UP
- DOWN
- LEFT
- RIGHT

available types:

- `Rectangle(x, y, u ,v)`
- `MenuUtils` (contains `progress(max, current, length)` for custom bars)
- `RecipeWrapper` (forge `IItemHandlerModifiable` wrapper for recipes)
- `CraftingWrapper` (ScreenJS wrapper class used for `crafting` recipes)

Custom Key Binds

ScreenJS can do custom key bindings! examples & available methods below:

> The custom KeyBind event is a Client event.

```
// client_scripts

KeybindEvents.register(event => {
    event.register(new KeyBind("open_menu" /* name */, InputConstants.KEY_G /* key index,
opengl spec */, "screenjs" /* category name */), (action, modifiers /* modifiers as per OpenGL
spec */) => {
        if (action == 1) { // action == 1 is PRESS
            Minecraft.instance.gui.setOverlayMessage(Text.string('AAA').yellow(), false) //
vanilla method
            MenuScreens.create('kubejs:separate', Minecraft.instance, 1000,
Text.string('AAA').yellow()) // opens a GUI container, preferably of type 'basic'
        } else if (action == 0) { // action == 0 is RELEASE
            Minecraft.instance.gui.setOverlayMessage(Text.string('BBB').yellow(), true)
        } else { // action == 2 is REPEAT (after a second of PRESS)
            Minecraft.instance.gui.setOverlayMessage(Text.string('REPEAT').red(), false)
        }
    })
})
```

available methods:

- `register`

available types:

- `KeyBind(name, keyIndex, category)`
- `KeyAction(action, modifiers)`
- `InputConstants`
- `Minecraft` (client main class)
-

Revision #20
Created 15 February 2023 15:15:54 by Lat
Updated 28 February 2023 07:42:16 by Screret