Custom Blocks

This is a startup script.

```
onEvent('block.registry', event => {
  event.create('test_block')
    .material('glass')
    .hardness(0.5)
    .displayName('Test Block') // No longer required in 1.18.2+
    .tagBlock('minecraft:mineable/shovel') // Make it mine faster using a shovel in 1.18.2+
    .tagBlock('minecraft:needs_iron_tool') // Make it require an iron or higher level tool on 1.18.2+
    .requiresTool(true) // Make it require a tool to drop ay loot

  // Block with custom type (see below for list of types for 1.18 (use .type for 1.16))
  event.create('test_block_slab', 'slab').material('glass').hardness(0.5)



  //uses a combo of properties (things you might consider blockstate) and random tick to make the block
eventually change to test_block, but only progresses if waterlogged

event.create('test_block_2').material('glass').hardness(0.2).property(BlockProperties.WATERLOGGED).property(B
lockProperties.AGE_7).randomTick(tick => {
    const block = tick.block
    const properties = block.properties
    const age = Number(properties.age)
    if (properties.waterlogged == 'false') return
    if (age == 7) {
      block.set('kubejs:test_block')
    } else {
      block.set('kubejs:test_block_2',{waterlogged:'true',age:`${age+1}`})
  })
})
```

The texture for this block has to be placed in `kubejs/assets/kubejs/textures/block/test_block.png`.
If you want a custom block model, you can create one in Blockbench and put it in

`kubejs/assets/kubejs/models/block/test_block.json`.

List of available materials - to change break/walk sounds and to *change some properties*.

Materials (1.18.2)
air
wood
stone
metal
grass
dirt
water
lava
leaves
plant
sponge
wool
sand
glass
explosive
ice
snow

Materials (1.18.2)
clay
vegetable
dragon_egg
portal
cake
web
slime
honey
berry_bush
lantern

Other methods block builder supports:

- displayName('name')
  - Not required for 1.18.2+
- material('material')
  - See list above
- type('basic')
  - See available types below.
  - Do not use for 1.18.2, use the syntax in the second example above
- hardness(float)
  - >= 0.0
- resistance(float)
  - >= 0.0
- unbreakable()
  - Sets the resistance to MAX_VALUE and hardness to -1, like bedrock
- lightLevel(int)
  - 0.0 - 1.0

- harvestTool('tool', level)
  - Available tools: pickaxe, axe, hoe, shovel
  - level >= 0
  - Not used in 1.18.2+, see tag in example above
- opaque(boolean)
- fullBlock(boolean)
- requiresTool(boolean)
- renderType('type')
  - Available types: solid, cutout, translucent
  - cutout required for blocks with texture like glass
  - translucent required for blocks like stained glass
- color(tintindex, color)
- textureAll('texturepath')
- texture('side', 'texturepath')
- model('modelpath')
- noItem()
- box(x0, y0, z0, x1, y1, z1, true)
  - 0.0 - 16.0
  - default is (0,0,0,16,16,16, true)
- box(x0, y0, z0, x1, y1, z1, false)
  - Same as above, but in 0.0 - 1.0 scale
  - default is (0,0,0,1,1,1, false)
- noCollision()
- notSolid()
- waterlogged()
- noDrops()
- slipperiness(float)
- speedFactor(float)
- jumpFactor(float)
- randomTick(randomTickEvent => {})
  - see below
- item(itemBuilder => {})
- setLootTableJson(json)
- setBlockstateJson(json)
- setModelJson(json)
- noValidSpawns(boolean)
- suffocating(boolean)
- viewBlocking(boolean)
- redstoneConductor(boolean)
- transparent(boolean)
- defaultCutout()
  - batches a bunch of methods to make blocks such as glass
- defaultTranslucent()
  - similar to defaultCutout() but using translucent layer instead
- tagBlock('forge:something')

    - adds a block tag
- tagItem('forge:something_better')
    - adds an item tag
- tagBoth('forge:something')
    - adds both block and item tag
- property(BlockProperty)
    - See example above, but adds in more "blockstates" to the block
    - Example: BlockProperties.WATERLOGGED
    - You can add as many or few as you desire

RandomTickEvent callback properties

- BlockContainerJS block
- Random random
- LevelJS level
- ServerJS server

Block Properties

The default 1.18 properties are:

- "MAX_RESPAWN_ANCHOR_CHARGES"
- "BAMBOO_LEAVES"
- "HANGING"
- "WEST_WALL"
- "BOTTOM"
- "EYE"
- "HALF"
- "DRAG"
- "MAX_ROTATIONS_16"
- "SOUTH"
- "MIN_RESPAWN_ANCHOR_CHARGES"
- "DISTANCE"
- "LOCKED"
- "EXTENDED"
- "SCULK_SENSOR_PHASE"
- "LEVEL"
- "DOOR_HINGE"
- "STAIRS_SHAPE"
- "EGGS"
- "LAYERS"
- "CONDITIONAL"
- "EAST_WALL"
- "HATCH"
- "ORIENTATION"
- "LEVEL_CAULDRON"

- "RAIL_SHAPE_STRAIGHT"
- "SIGNAL_FIRE"
- "STRUCTUREBLOCK_MODE"
- "PISTON_TYPE"
- "MIN_LEVEL"
- "HAS_BOOK"
- "ATTACH_FACE"
- "WATERLOGGED"
- "FALLING"
- "AGE_25"
- "TRIGGERED"
- "MAX_LEVEL_8"
- "UNSTABLE"
- "CHEST_TYPE"
- "AGE_5"
- "SOUTH_WALL"
- "AGE_7"
- "STABILITY_MAX_DISTANCE"
- "BELL_ATTACHMENT"
- "AGE_1"
- "MAX_LEVEL_3"
- "ATTACHED"
- "AGE_3"
- "STAGE"
- "AGE_2"
- "POWER"
- "MAX_DISTANCE"
- "HAS_BOTTLE_1"
- "HAS_BOTTLE_0"
- "PICKLES"
- "HAS_BOTTLE_2"
- "OPEN"
- "DRIPSTONE_THICKNESS"
- "AGE_15"
- "LEVEL_HONEY"
- "CANDLES"
- "LEVEL_COMPOSTER"
- "LIT"
- "EAST_REDSTONE"
- "OCCUPIED"
- "MODE_COMPARATOR"
- "NORTH_REDSTONE"
- "IN_WALL"
- "SNOWY"
- "DOWN"
- "WEST"

- "NORTH_WALL"
- "MIN_LEVEL_CAULDRON"
- "BED_PART"
- "NORTH"
- "LEVEL_FLOWING"
- "TILT"
- "UP"
- "SOUTH_REDSTONE"
- "MAX_AGE_15"
- "HORIZONTAL_FACING"
- "BITES"
- "SLAB_TYPE"
- "MAX_AGE_2"
- "MAX_AGE_1"
- "ROTATION_16"
- "MAX_AGE_7"
- "STABILITY_DISTANCE"
- "MAX_AGE_5"
- "MAX_AGE_3"
- "MAX_AGE_25"
- "DELAY"
- "AXIS"
- "MAX_LEVEL_15"
- "HORIZONTAL_AXIS"
- "RAIL_SHAPE"
- "MOISTURE"
- "VERTICAL_DIRECTION"
- "DOUBLE_BLOCK_HALF"
- "NOTE"
- "BERRIES"
- "RESPAWN_ANCHOR_CHARGES"
- "EAST"
- "PERSISTENT"
- "HAS_RECORD"
- "FACING_HOPPER"
- "NOTEBLOCK_INSTRUMENT"
- "POWERED"
- "SHORT"
- "VINE_END"
- "WEST_REDSTONE"
- "ENABLED"
- "INVERTED"
- "FACING"
- "DISARMED"

You can make your own also using the following example:

```
const $BooleanProperty = Java.loadClass('net.minecraft.world.level.block.state.properties.BooleanProperty')
const $IntegerProperty = Java.loadClass('net.minecraft.world.level.block.state.properties.IntegerProperty')

onEvent('block.registry', event => {
  event.create('my_block').property($IntegerProperty.create("uses", 0, 2)).property($BooleanProperty.create("empty"))
})
```

Types

- basic
- detector
- slab
- stairs
- fence
- fence_gate
- wall
- wooden_pressure_plate
- stone_pressure_plate
- wooden_button
- stone_button
- falling
- crop

Detector Block Types

The detector block type can be used to run code when the block is powered with redstone signal.

Startup script code:

```
onEvent('block.registry', event => {
 event.create('test_block','detector').detectorId('myDetector')
}
```

Server script code:

```
onEvent('block.detector.myDetector.unpowered', event => { // you can also use powered and changed instead of upowered
      event.block.set('tnt')
}
```

Revision #28
Created 28 June 2020 17:09:30 by Lat
Updated 9 August 2023 03:57:41 by Q6