Integration Guide for Baedoor Mods

Includes:

Wastelands of Baedoor
Spires of Baedoor
Civilisations of Baedoor

This guide will help you with adding integrations to your mod. Remember that this guide is supposed to help you **only** with publicly accessible integrations/compatibility. In case you want your mod to have specific integration feature, you can message me on Discord (Toma400#9987).

Most of features should have additional MCreator tutorial, as Baedoor mods are created via MCreator and I can lead you easily through it for you to create your own add-on. If you just code mod by yourself, I tried to be as clear as possible to show what should be added or changed for integration to work. Again, if you have any questions, feel free to message me.

Add-on support is now available!
Verified add-ons can be listed in mod's guide as a form of support. For verification, simply contact me and send add-on. If it works correctly, it will be listed.
You can also precise detailed statistics, so it can receive additional note ***vanilla balance*** as quality proof on that field. Vanilla balancing isn't required.

If you integrated your mod correctly and tested features, you can use these official banners/buttons to show your compatibility. You can find them here.

Wastelands of Baedoor

List of available integrations (for mods)

- Mechanics: Non-WoB firearms using WoB experience
- Mechanics: Nether Avoider (spawnproofing)
- Enchantment: Dismantle
- Blocks: Wood Variants*
- Blocks: Lockable Chests (ingots/metals/gems support)*
- Weapons: Firearms
- Weapons: Firearm Table
- Weapons: Sabre
- Weapons: Spear

List of add-on support (for add-ons)

- Weapons: Firearms
- Weapons: Firearm Table
- Weapons: Sabre
- Weapons: Spear

---

* requires contact with author

Mechanics: Non-WoB firearms using WoB experience

If you have your mod that adds any form of guns, you can make it easily use firearm experience feature from WoB, which will be counting experience even without WoB installed (but once WoB is in, you will be able to use it).

It's really simple, honestly: the only that you need to do is just add 1 score for "gun_experience" scoreboard after each successful shooting with your gun.
You can also use that scoreboard to determine some cross-compatibility features, of course. Remember though that WoB is aiming for long time of mastering firearms, so they will need quite a lot of play to be OP by any means.

After you did that, you can contact me to let me know that your mod is compatibile - I will gladly write you in the list of compatibilities :)

Mechanics: Nether Avoider

Nether Avoider block kills various mobs when activated, giving builders possibility to leave places without spawnproofing limiting their creativity.

To use Nether Avoider effect, you can either:

1. add entity to scoreboard “**avoider_killable**” *
2. set logic NBT tag “**avoider_killable**” to true for entity
3. use Forge entity Tag “**avoider_killable**” for entity [in v0.a11]

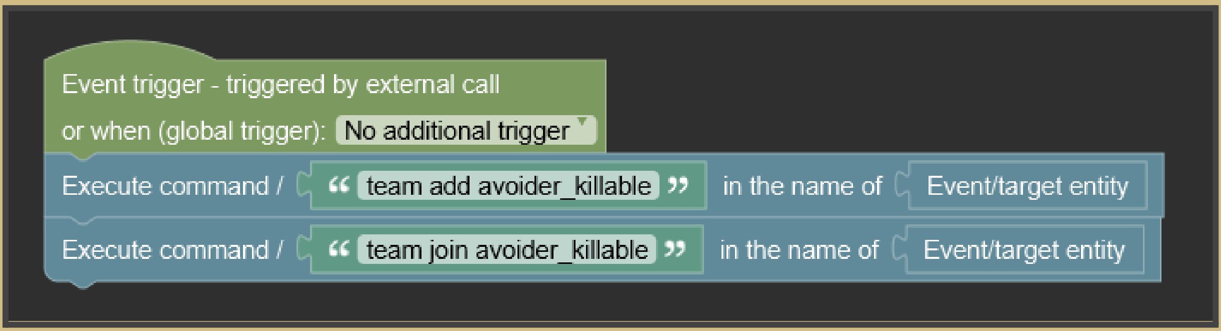
All of these can be used, and scoreboard/NBT tag use can be customised by activating them either after entity spawn (useful for “hostile-only” mobs) or specific event (especially useful if entity is passive by default), it can be also turned off.
Scoreboard method doesn’t remove loot. As such, it can be useful, yet not reliable for performance.

This integration works with both 1.15+ and 1.12.2 versions of WoB. 1.12.2 version supports NBT tag method only.
You can also make custom datapacks; if you made one for yet-not-supported mod and want to share it, contact with me/join our Discord, I will gladly test it and upload on official datapack storage.

* it is recommended to add team before to avoid bugs

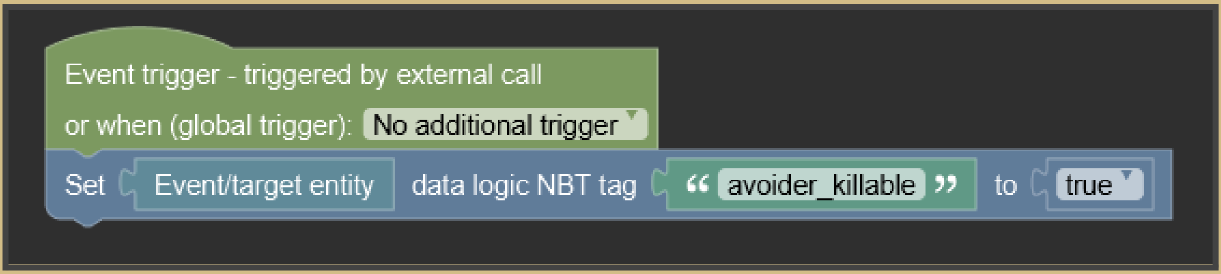
MCreator tutorial

For scoreboard solution you just need to execute two commands. It’s important to add team in case it wasn’t added before (adding existing team will skip that command):

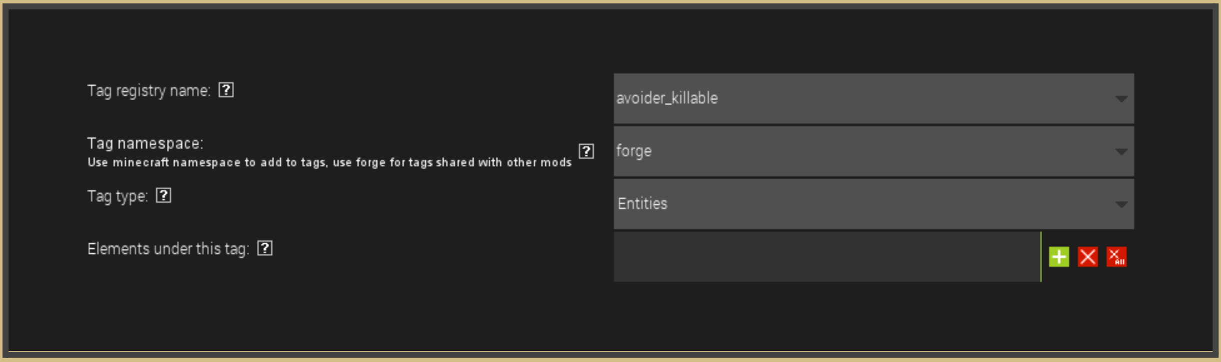


Event trigger - triggered by external call
or when (global trigger): No additional trigger
Execute command / “ team add avoider_killable ” in the name of Event/target entity
Execute command / “ team join avoider_killable ” in the name of Event/target entity

For NBT tag option, you just need to switch it to true:


Event trigger - triggered by external call
or when (global trigger): No additional trigger
Set Event/target entity data logic NBT tag “ avoider_killable ” to true

For Tag solution, you need to use precise Tag registry name and set it to “Entity” type (available since MCreator 2021.1):


Tag registry name: avoider_killable
Tag namespace: Use minecraft namespace to add to tags, use forge for tags shared with other mods: forge
Tag type: Entities
Elements under this tag:

Enchantment: Dismantle

Dismantle enchantment is meant to deal additional damage to any mechanical beings, meaning it’s mostly dedicated for any mods adding machine mobs and bosses.

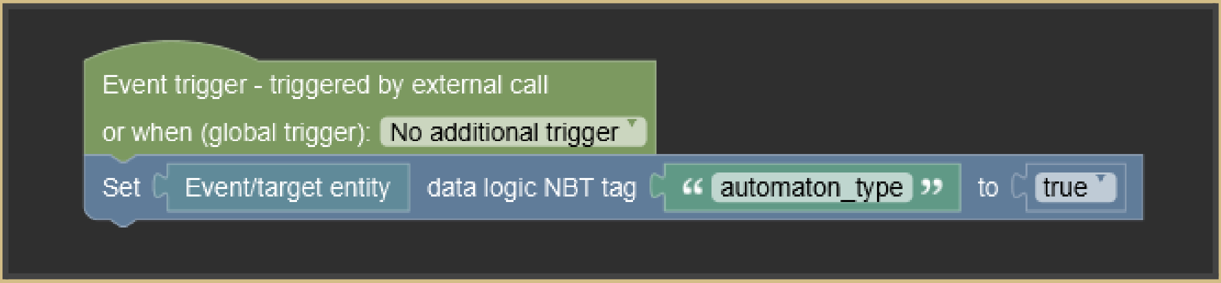
For Dismantle enchantment support, you can either:

1. set logic NBT tag “**automaton_type**” to true for entity
2. use Forge entity Tag “**automaton_type**” for entity [in v0.a11]

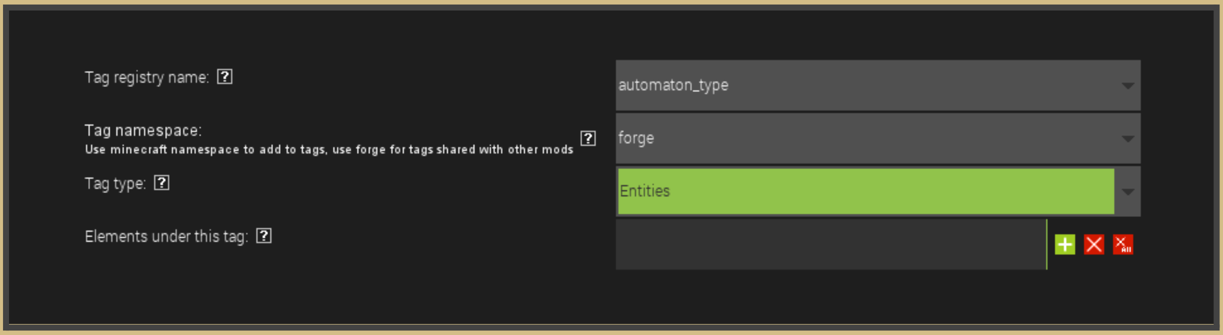
You can also make custom datapacks; if you made one for yet-not-supported mod and want to share it, contact with me/join our Discord, I will gladly test it and upload on official datapack storage.

MCreator tutorial

For NBT tag way, you need to switch its value to true:


Event trigger - triggered by external call
or when (global trigger): No additional trigger
Set Event/target entity data logic NBT tag “ automaton_type ” to true

For entity Tag you need to make sure that registry name and Tag type are set correctly (available since MCreator 2021.1):


Tag registry name: automaton_type
Tag namespace: forge
Use minecraft namespace to add to tags, use forge for tags shared with other mods
Tag type: Entities
Elements under this tag:

Blocks: Wood Variants

Wastelands of Baedoor adds several wood variants for vanilla wood types, such as stairs, slabs, chiseled logs, chiseled slabs, pillars and lamps.

Although there's no dedicated support for adding your own wood type for that feature, so you can't make it entirely by yourself, there's possibility to add integration on that part after messaging me.

What you need to deliver?
Not really much:

- Item registry name/Tag for log and stripped log. Optional, but recommended.
- Texture of log and stripped log. Optional, but recommended.
- Texture of chiseled log/pillar side & top/lamp. This one is especially optional, since I can make one yourself.

After delivering these elements, I will be able to use it to make your own variant. Of course I can also take that from your mod directly, but providing them will save me some effort. And that's appreciated!

Blocks: Lockable Chests

Lockable Chests are chests possible to be locked with key, and usually are highly decorative, made with metals or aesthetical, rare resources.
WoB supports custom lockable chests with your own mod's **metal**, **gem**, aesthetic **stone**, or even **wood**.

Similarly to how it's done with wood variants, you need to contact with me first to add such integration. And what's more important, you will need to deliver texture for chest, made with that template: *link*. Link also contains

model, in case you need it to visualise chest better.
(yes, I can make texture myself, but I find making model-bound textures much more difficult, so I'd prefer if you make it yourself. It will also save my time, I don't have endless amount of it, sadly :P)

After that, let me know what is your material registry name (ID)/tag, and - if you wish so - specific values for hardness and blast resistance (for both open and locked state). If you do not provide these values, they will be considered upon material tier (already existing: **stone - light metal - hard metal - obsidian - netherite - cirtain**) which I will try to choose as close to your material as possible. You can also choose one of these tiers instead of your custom numbers, if it's more appealing for you.

Weapons: Firearms

Firearms are preciously crafted weapons, enabling their holders to destroy with subtle, yet terrifyingly powerful mighty!
Seriously though: guns allow you to shoot, reload and use different mechanics bound to firearms. Use them wisely.

Important issue: firearms, if added as mod feature, will not work unless WoB is installed too. Keep that in mind while developing it.

Using firearms starts, as always, from creating Forge item Tag, this time "wobr_firearms". After that, you will need to create procedure, using quite a lot of INBT tags. Let me list them all. Remember that NBT tags are case-sensitive:

- **ammo_using** - it's text INBT used to recognise type of ammo gun is using. There are four types:
    - **Small_Bullet** - used by revolvers and smaller guns
    - **Large_Bullet** - used by rifles
    - **Slug** - used by shotguns
    - **Gunpowder** - used by specific firearms
- **max_ammo** - maximum capacity of gun's clip
- **rld_type** - text INBT, used to recognising type of reload. For now there are three:
    - **Single** - you reload each bullet, one at the time
    - **Quick Single** - you reload each bullet, one at the time, but you can force reloading next bullet faster
    - **All_In_Once** - you reload whole clip at once

- **rld_delay** - time length (in ticks, 20 ticks = 1 second) of reloading
- **shot_mode** - it contains information whether the gun uses modes or not. Logic INBT, "false" value is default. Optional. Returns **shoot_mode** INBT number tag to know whether it is in standard mode [**0**] or alternative mode [**1**] for you to use in conditions.
- **shot_delay** - delay (in ticks, 20 ticks = 1 second) between each shot
- **shot_pwr** - shot power. Serves as a information how fast bullet will fly, but also partly increases damage.
- **shot_dmg** - shot damage value. Remember that final damage value is result of multiplication **shot_pwr** and **shot_dmg**.
- **shot_knc** - shot knockback. Default value is null (0). Optional.
- **firearm_quality** - number INBT. Used to recognise whether firearm is handmade (2) or uses scheme (1). For better quality, use "1". Other values can freeze the game.

There are also advanced logic INBT you can use, for customised firearm functionality or sounds. All of them are optional (**false** by default):

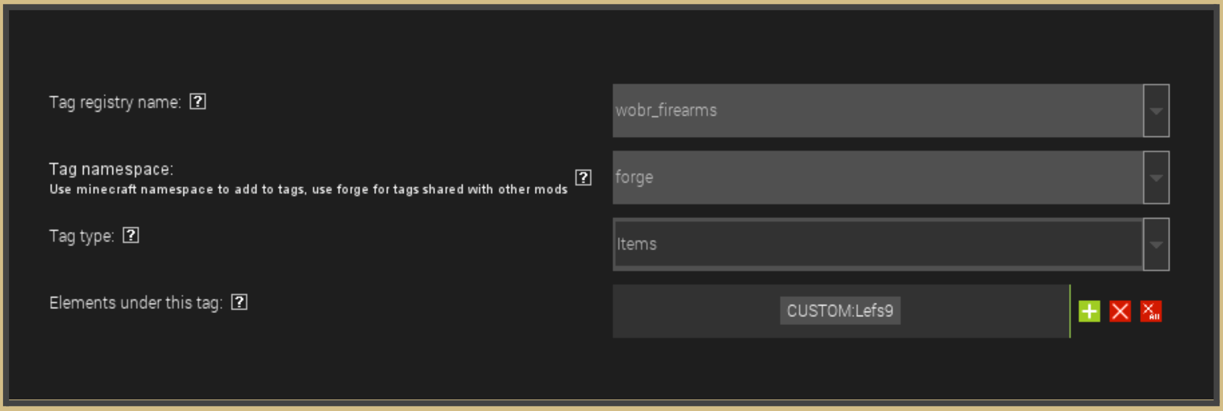
- **rld_sound** - disables default reload sound and gives you returning INBT tag to use as a condition to play your own sound
  - **rld_trig** set to **true**. Remember to set it to false in the end to avoid endless loop.
- **shot_sound** - disables default shot sound and gives you returning INBT tag to use as a condition to play your own sound
  - **shot_trig** set to **true**. Remember to set it to false in the end to avoid endless loop.
- **block_load** - disables reloading and unloading entirely, in case you need to customise it
- **block_shot** - disables shooting entirely, in case you need it for something else
- **notify_shot -** returns INBT tag after successful shot, allowing you to create additional events with that condition
  - **trig_shot** set to **true**. Remember to set it to false in the end to avoid endless loop.
- **notify_jam** - returns INBT tag after failed shot, allowing you to create additional events with that condition
  - **trig_jam** set to **true**. Remember to set it to false in the end to avoid endless loop.

There is also **Ammo** INBT tag which contains information about how much ammunition is loaded inside the firearm. It is overwritten with each reload/unload, yet can be used in certain situations.

You can also condition these core INBT tags themselves, making them more sophisticated, but that can result on richer gameplay.
For example, you can make enchantment for your gun that makes it more reliable, so you switch ***firearm_quality*** *from 2 to 1 if it's applied.*

MCreator tutorial

First of all, create Forge item Tag "wobr_firearms". Make sure each section is set correctly:



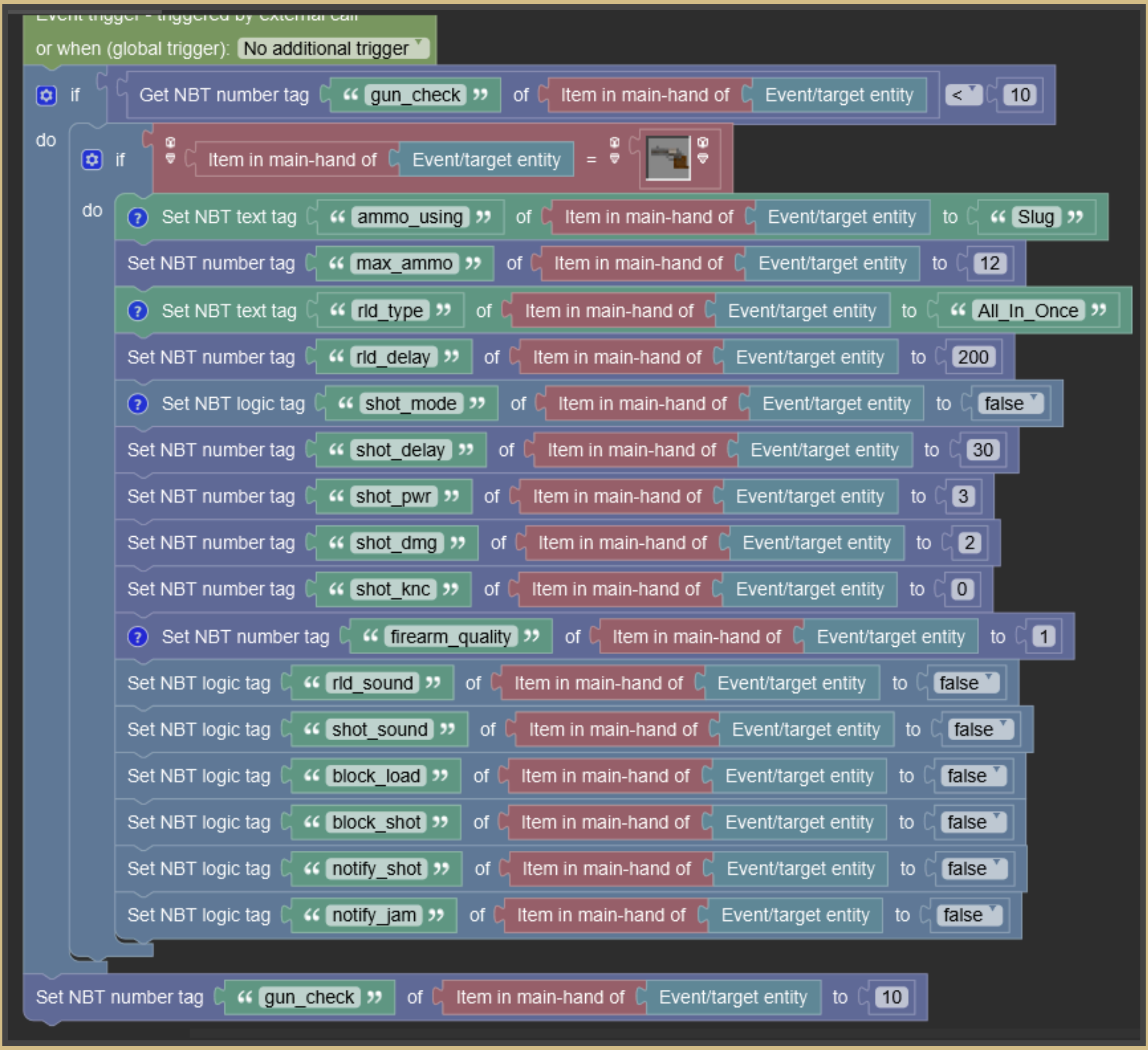
Tag registry name: wobr_firearms

Tag namespace: forge
Use minecraft namespace to add to tags, use forge for tags shared with other mods

Tag type: Items

Elements under this tag: CUSTOM:Lefs9

Next, create procedure setting up all INBT tags listed above (with or without optional ones). Set their values precisely, remembering that NBT tags are case-sensitive. In the end, your procedure can look like that (I used all NBT tags, just for showing it entirely):

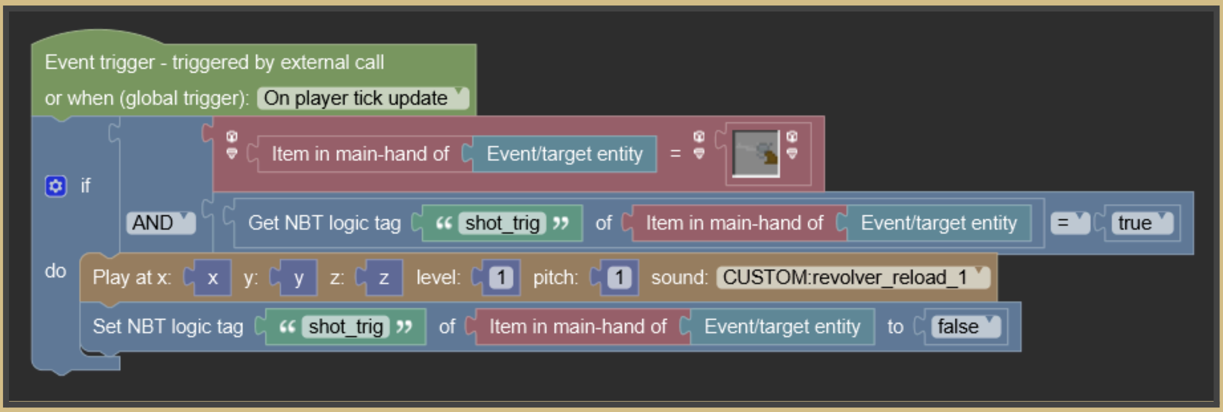


```
Event trigger - triggered by external call
or when (global trigger): No additional trigger
if  Get NBT number tag "gun_check" of Item in main-hand of Event/target entity < 10
do
    if  Item in main-hand of Event/target entity = 
    do
        Set NBT text tag "ammo_using" of Item in main-hand of Event/target entity to "Slug"
        Set NBT number tag "max_ammo" of Item in main-hand of Event/target entity to 12
        Set NBT text tag "rld_type" of Item in main-hand of Event/target entity to "All_In_Once"
        Set NBT number tag "rld_delay" of Item in main-hand of Event/target entity to 200
        Set NBT logic tag "shot_mode" of Item in main-hand of Event/target entity to false
        Set NBT number tag "shot_delay" of Item in main-hand of Event/target entity to 30
        Set NBT number tag "shot_pwr" of Item in main-hand of Event/target entity to 3
        Set NBT number tag "shot_dmg" of Item in main-hand of Event/target entity to 2
        Set NBT number tag "shot_knc" of Item in main-hand of Event/target entity to 0
        Set NBT number tag "firearm_quality" of Item in main-hand of Event/target entity to 1
        Set NBT logic tag "rld_sound" of Item in main-hand of Event/target entity to false
        Set NBT logic tag "shot_sound" of Item in main-hand of Event/target entity to false
        Set NBT logic tag "block_load" of Item in main-hand of Event/target entity to false
        Set NBT logic tag "block_shot" of Item in main-hand of Event/target entity to false
        Set NBT logic tag "notify_shot" of Item in main-hand of Event/target entity to false
        Set NBT logic tag "notify_jam" of Item in main-hand of Event/target entity to false
Set NBT number tag "gun_check" of Item in main-hand of Event/target entity to 10
```

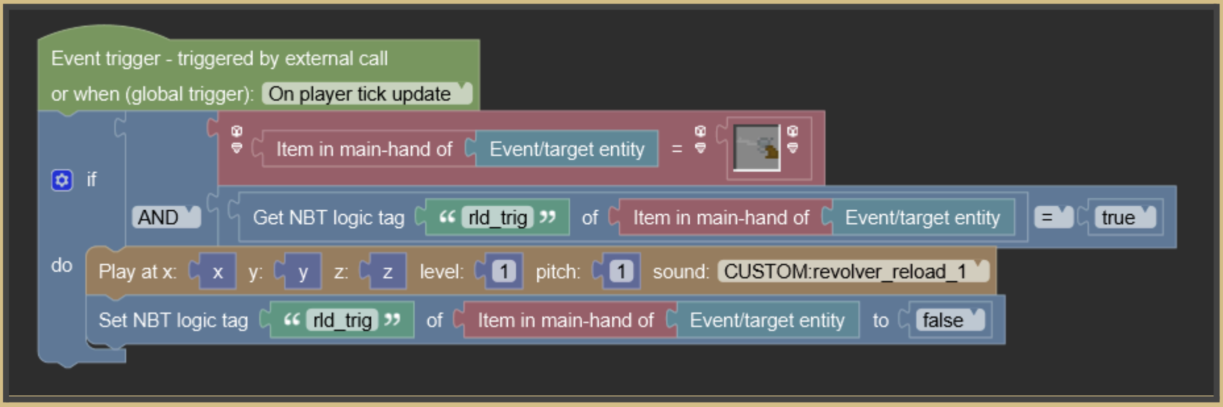
Used "gun_check" NBT tag just to make the procedure happen once. It should be used as early as possible though, so I recommend setting it up on "when in hand" or "when in inventory" triggers.

Sound-related NBT tags are meant to use your own sound after a shot or reloading.
They can be used with separate procedure on "player tick update" trigger. Use conditions to make it appear only when firearm is in hand and you receive INBT tags bound to them.

For reloading, the INBT tag is **rld_trig** set to **true**, and for shooting, the INBT tag is **shot_trig** set to true. In the end of the procedure, set these values to false again, to avoid endless loop.
Examples of sound-adding procedures can be seen below:

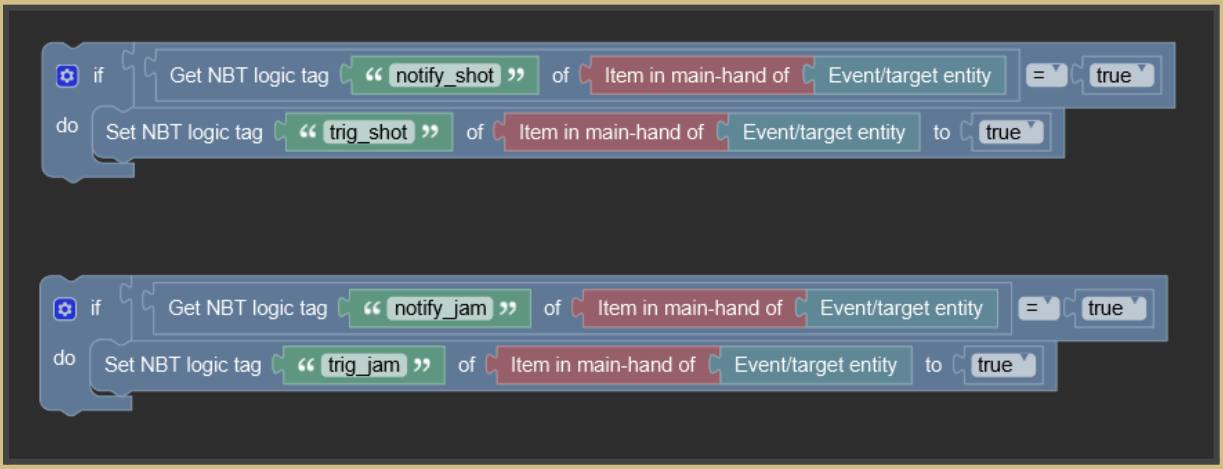


```
Event trigger - triggered by external call
or when (global trigger): On player tick update
if    Item in main-hand of Event/target entity = [item]
  AND Get NBT logic tag "shot_trig" of Item in main-hand of Event/target entity = true
do    Play at x: x y: y z: z level: 1 pitch: 1 sound: CUSTOM:revolver_reload_1
      Set NBT logic tag "shot_trig" of Item in main-hand of Event/target entity to false
```



```
Event trigger - triggered by external call
or when (global trigger): On player tick update
if    Item in main-hand of Event/target entity = [item]
  AND Get NBT logic tag "rld_trig" of Item in main-hand of Event/target entity = true
do    Play at x: x y: y z: z level: 1 pitch: 1 sound: CUSTOM:revolver_reload_1
      Set NBT logic tag "rld_trig" of Item in main-hand of Event/target entity to false
```

Notification-related INBT tags work similarly, giving you true value after a successful shot or failed shot (respectively, **trig_shot** and **trig_jam** NBT tags). It can be seen below:



```
if  Get NBT logic tag "notify_shot" of Item in main-hand of Event/target entity = true
do  Set NBT logic tag "trig_shot" of Item in main-hand of Event/target entity to true

if  Get NBT logic tag "notify_jam" of Item in main-hand of Event/target entity = true
do  Set NBT logic tag "trig_jam" of Item in main-hand of Event/target entity to true
```

Use them as a conditions to create your own elements after each event. Remember that it should be set to **false** eventually, to avoid endless loop.

You can also condition core INBT tags themselves, making them more sophisticated, but that can result on richer gameplay.
*For example, you can make enchantment for your gun that makes it more reliable, so you switch **firearm_quality** from 2 to 1 if it's applied.*

Using Modules [available in v0.a11s4+]

If you want to use WoB’s gun modules for crafting, such as grips, clips and barrels, it is also possible.
It basically slightly changed condition in firearm table (or any other GUI) - instead of needing specific item, use condition needing any item with Forge tag.

List of Forge tags to use, provided by WoB:

- grips:
  - **module_small_grip** - for small grip
  - **module_large_grip** - for large grip
- barrels:
  - **module_short_barrel** - for short barrel
  - **module_long_barrel** - for long barrel
  - **module_double_long_barrel** - for double long barrel
- clips:
  - **module_clip_2_shotgun** - for 2-rounded shotgun clip
  - **module_clip_3_rifle** - for 3-rounded rifle clip
  - **module_clip_6_rifle** - for 6-rounded rifle clip
  - **module_clip_12_rifle** - for 12-rounded rifle clip
  - **module_clip_2_revolver** - for 2-rounded revolver clip
  - **module_clip_4_revolver** - for 4-rounded revolver clip
  - **module_clip_5_revolver** - for 5-rounded revolver clip
  - **module_clip_6_revolver** - for 6-rounded revolver clip
  - **module_clip_7_revolver** - for 7-rounded revolver clip

To use firearm table, look at section below.

Using Custom Ammo [available in v0.a11s5+]

If you want your gun to use custom ammo (ammo is usually ranged item, since it can be shot by default), you can do it! Guns which are compatibile with it will be able to use your custom ammo as they were WoB-related ones.

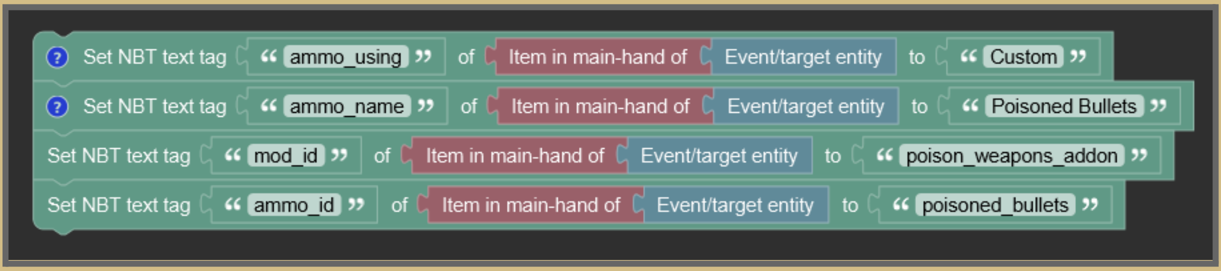
First of all, create Forge item Tag “wobr_ammo” and put all your ammo items you need. It is required for further steps. This is everything you need on ammunition side, though.

Second part of it is just making gun compatibile with your newly added ammunition. First of all, come back to section with INBT tags. Find "**ammo_using**" and set it to **Custom** text value.
After that, add three new INBT tags (these are needed only with custom ammo):

- **ammo_name** - this should be display name of your custom ammo
- **mod_id** - this should be your mod/add-on ID
- **ammo_id** - this should be your ammo ID (usually the same as display name, with underlines instead of spaces and lower case only)

This is everything! Enjoy your gun using custom bullets!

You can also see example here:



```
Set NBT text tag "ammo_using" of Item in main-hand of Event/target entity to "Custom"
Set NBT text tag "ammo_name" of Item in main-hand of Event/target entity to "Poisoned Bullets"
Set NBT text tag "mod_id" of Item in main-hand of Event/target entity to "poison_weapons_addon"
Set NBT text tag "ammo_id" of Item in main-hand of Event/target entity to "poisoned_bullets"
```

Weapons: Firearm Table

Firearm Table allows you to craft your own firearms using dedicated GUI for various gun parts. With v0.a11 version you will be able to support it with your own mod or add-on.

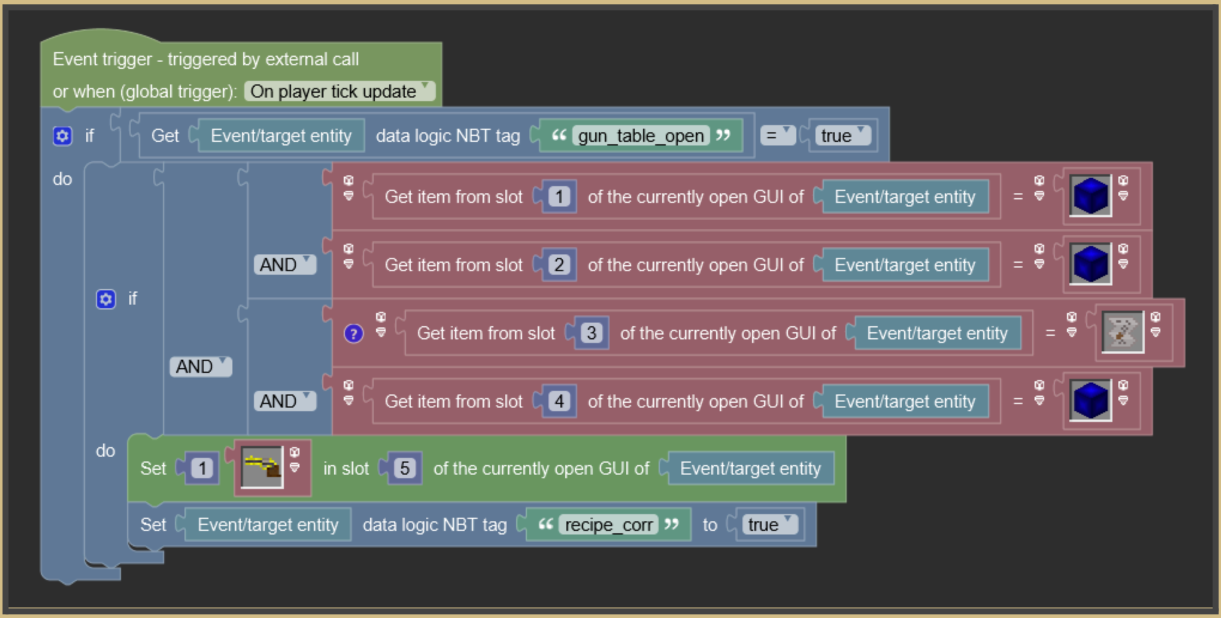
To use it, simply add procedure triggering on "player tick update" activated when player's NBT tag "gun_table_open" is set to *true* value.

You can customise required items in slots 1-4 (respectively: grip slot, clip slot, scheme slot and barrel slot), as well as output (slot 5). Remember that:

- Slot 3 will not remove its contents
- Set slot to require "air" if it doesn't require any particular item (to avoid possible incompatibilities)

Eventually, set player's NBT tag "recipe_corr" to *true* value as a second result of achieving conditions.

Pretty simply just follow the instructions from above, resulting in similar procedure as shown below. Items from slots are customisable, yet remember to use all of them, putting "air" in the ones you don't need. Slot 3 doesn't remove items inside of it.
Make sure all NBT tags names are correctly written.



```
Event trigger - triggered by external call
or when (global trigger): On player tick update
if  Get Event/target entity data logic NBT tag "gun_table_open" = true
do
    if      Get item from slot 1 of the currently open GUI of Event/target entity =
        AND Get item from slot 2 of the currently open GUI of Event/target entity =
        AND Get item from slot 3 of the currently open GUI of Event/target entity =
        AND Get item from slot 4 of the currently open GUI of Event/target entity =
    do
        Set 1 in slot 5 of the currently open GUI of Event/target entity
        Set Event/target entity data logic NBT tag "recipe_corr" to true
```

Weapons: Sabres

Sabres are faster, yet less durable alternatives for swords, having additional mechanics making them tricker, yet even more powerful in long-term use. They can defend you for short amount of time (just like using swords before Minecraft 1.9 update), deal additional damage if you are experienced fighting with it, and harm you sometimes - if not.

Sabres will require WoB for:

- defence mechanics
- experience and damage bonus
- self-harming mechanics

Meaning that sabres without WoB will work, but they will rely on speed/durability ratio, or effects if you add any. You can make them similar to WoB sabres in that scenario by using speed starting from 2.2 and rather weak durability (100-200).

To use sabres mechanics, you need to:

1. use Forge item Tag "**wobr_sabre**" and add your weapons there
2. set item NBT tags to specific numbers:

2.1. **“sabre_defence”** means defence time in ticks. Default is 10.

2.2. **“sabre_cooldown”** means time of cooldown for attacking. Default is 15 (for faster sabres) or 20 (for slower sabres). It should be a bit higher than sabre_defence.

2.3. **“sabre_harm”** is multiplier of self-harm damage. Its default value is **1**, and **0** is not accepted (it will automatically switch to default value). Self-harm with multiplier **1** deals half of sabre damage. Optional.

2.4. **“sabre_def_off”** is logic tag to turn off defence mechanics. Default is false, true will disable defence mechanics. Optional.

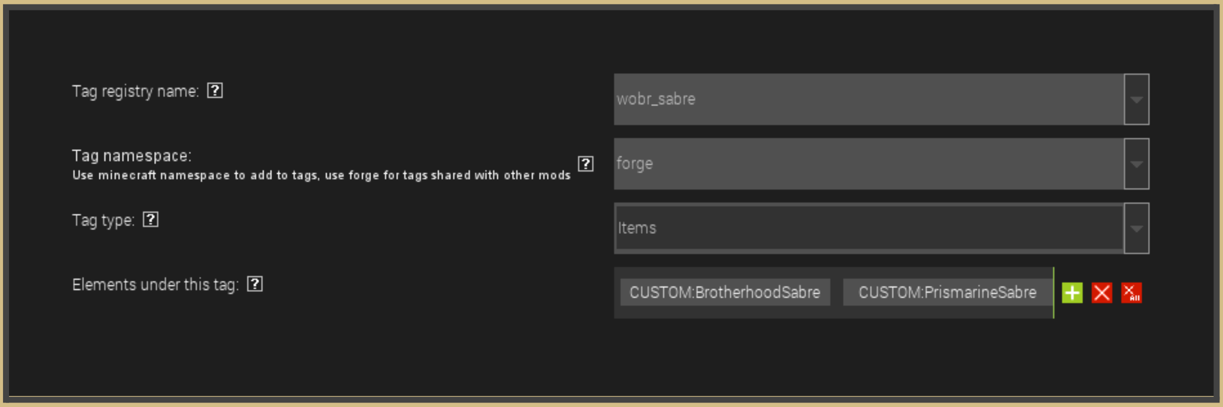
*You can also take sabre experience from scoreboard score **sabre_experience**, in case you want to use it further.*

--

MCreator tutorial

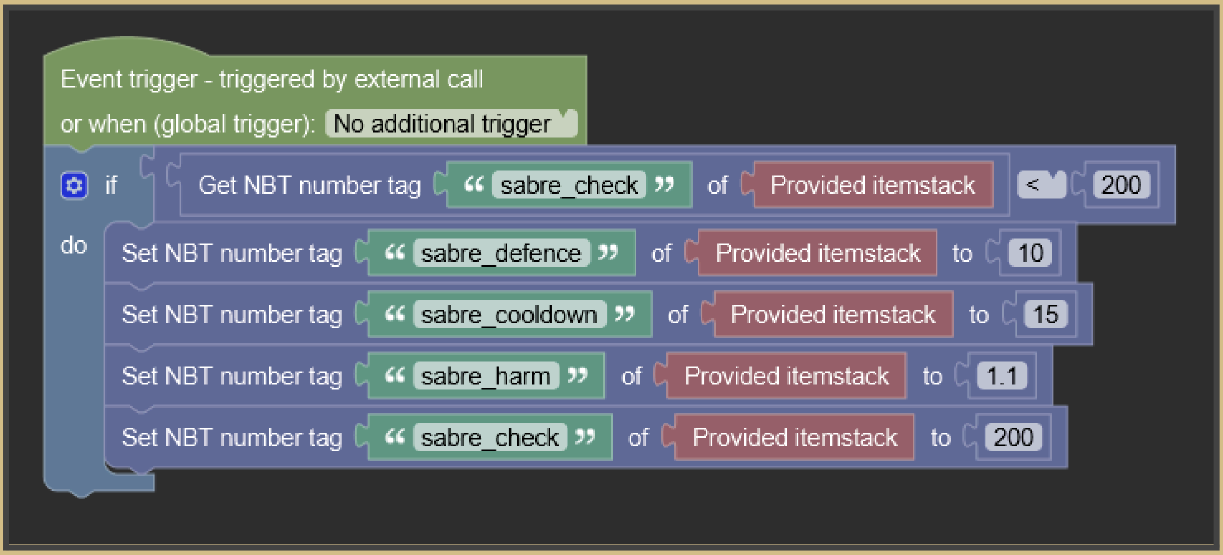
This tutorial only shows exemplary way to create integration feature. It can be done in different ways.

Firstly, create Forge item Tag and add your sabres to it. Make sure registry name and namespace/type are correctly set.



Tag registry name:	wobr_sabre
Tag namespace: Use minecraft namespace to add to tags, use forge for tags shared with other mods	forge
Tag type:	Items
Elements under this tag:	CUSTOM:BrotherhoodSabre, CUSTOM:PrismarineSabre

Secondly, make procedure setting up item NBT tags.



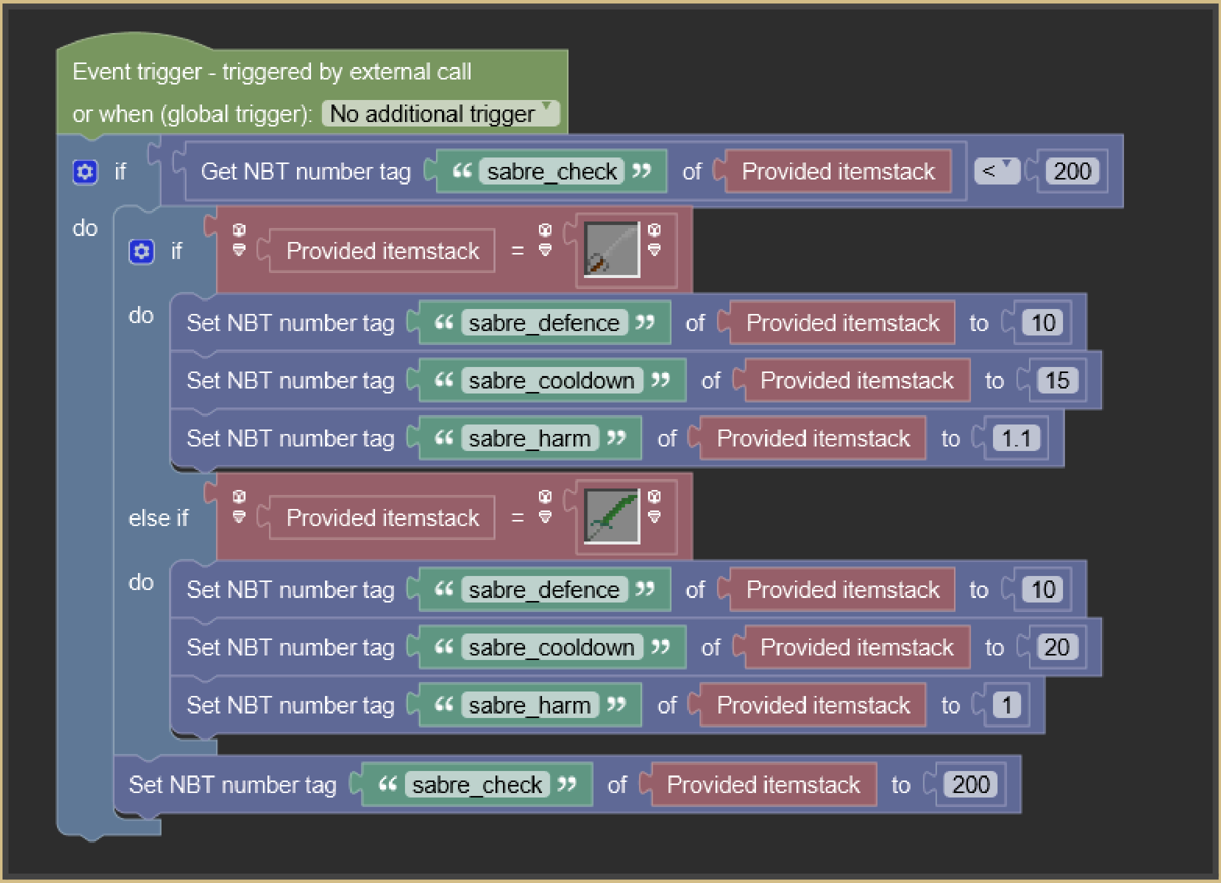
```
Event trigger - triggered by external call
or when (global trigger): No additional trigger
if Get NBT number tag "sabre_check" of Provided itemstack < 200
do Set NBT number tag "sabre_defence" of Provided itemstack to 10
   Set NBT number tag "sabre_cooldown" of Provided itemstack to 15
   Set NBT number tag "sabre_harm" of Provided itemstack to 1.1
   Set NBT number tag "sabre_check" of Provided itemstack to 200
```

“Sabre_check” NBT tag is my solution for not calling “set NBT number tag” procedure blocks everytime, so it can be event applied once. You can solve that in any way you like.

In that menu you can decide how long defence will be, how long cooldown for attacking player will get and how big amplifier for harming with sabre should be. You can also switch off the defence mechanics.
Default values are:

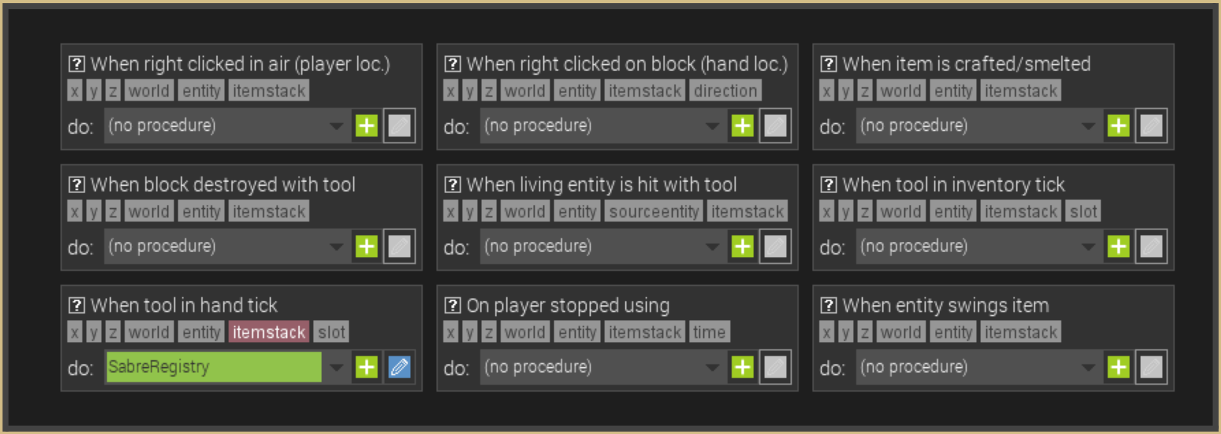
10 for **sabre_defence**
15 or 20 for **sabre_cooldown**
1 for **sabre_harm** (0 for **sabre_harm** is not usable).
false for **sabre_def_off**

You can also specify the procedure for various weapons, so you can use one procedure for them all:



```
Event trigger - triggered by external call
or when (global trigger): No additional trigger
if   Get NBT number tag "sabre_check" of Provided itemstack < 200
do
    if   Provided itemstack = 
    do   Set NBT number tag "sabre_defence" of Provided itemstack to 10
         Set NBT number tag "sabre_cooldown" of Provided itemstack to 15
         Set NBT number tag "sabre_harm" of Provided itemstack to 1.1
    else if   Provided itemstack = 
    do   Set NBT number tag "sabre_defence" of Provided itemstack to 10
         Set NBT number tag "sabre_cooldown" of Provided itemstack to 20
         Set NBT number tag "sabre_harm" of Provided itemstack to 1
    Set NBT number tag "sabre_check" of Provided itemstack to 200
```

After creating that procedure, you can now bind it to your sabre:



```
When right clicked in air (player loc.)     When right clicked on block (hand loc.)          When item is crafted/smelted
x y z world entity itemstack                x y z world entity itemstack direction           x y z world entity itemstack
do: (no procedure)                          do: (no procedure)                               do: (no procedure)

When block destroyed with tool              When living entity is hit with tool              When tool in inventory tick
x y z world entity itemstack                x y z world entity sourceentity itemstack        x y z world entity itemstack slot
do: (no procedure)                          do: (no procedure)                               do: (no procedure)

When tool in hand tick                      On player stopped using                          When entity swings item
x y z world entity itemstack slot           x y z world entity itemstack time                x y z world entity itemstack
do: SabreRegistry                           do: (no procedure)                               do: (no procedure)
```

Remember that you can use other triggers, but you will need these values before using sabre to defence/attack.

Weapons: Spears

Spears are defensive weapons that are meant to be used with shields. While having shield in off-hand, they will equip you with resistance effect for a while if you use shield. This event has a cooldown, though.

Another functionality of shields is rage mode, which decreases cooldown and increases resistance power if you have 2 hearts or less.

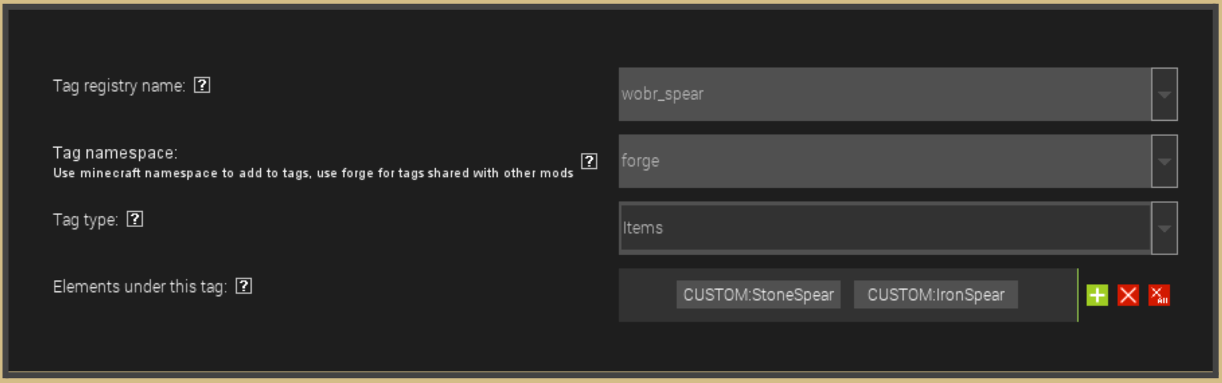
To include your weapon to spear type, use Forge item Tag “wobr_spear” and define its INBT tags:

- **spr_def_power** - given resistance effect level +1. Default value: 2
- **spr_def_time** - time of resistance effect
- **spr_cooldown** - time in ticks for defence mode cooldown. Default value: 301 (15 seconds). Optional.
- **spr_rg_power** - rage bonus for resistance effect. Default value: 1. Optional.
- **spr_rg_factor** - rage divisor for cooldown. Default value: 2. Optional.

If you are not sure about some values, leave them with default value. Values marked as optional don’t need to be set.

MCreator tutorial

First, you need to bind all your spears to Forge tag:



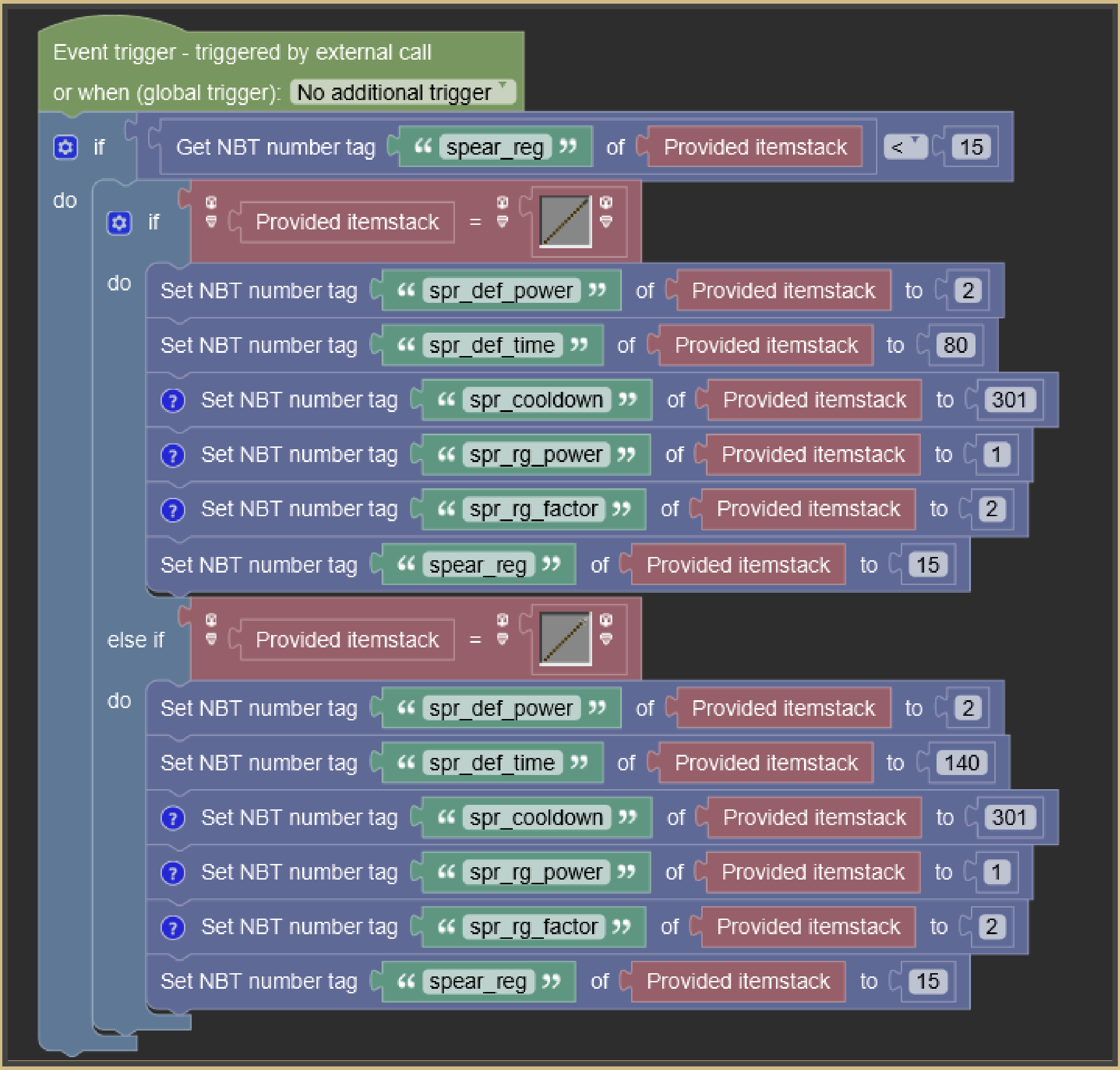
Tag registry name: wobr_spear

Tag namespace: forge
Use minecraft namespace to add to tags, use forge for tags shared with other mods

Tag type: Items

Elements under this tag: CUSTOM:StoneSpear CUSTOM:IronSpear

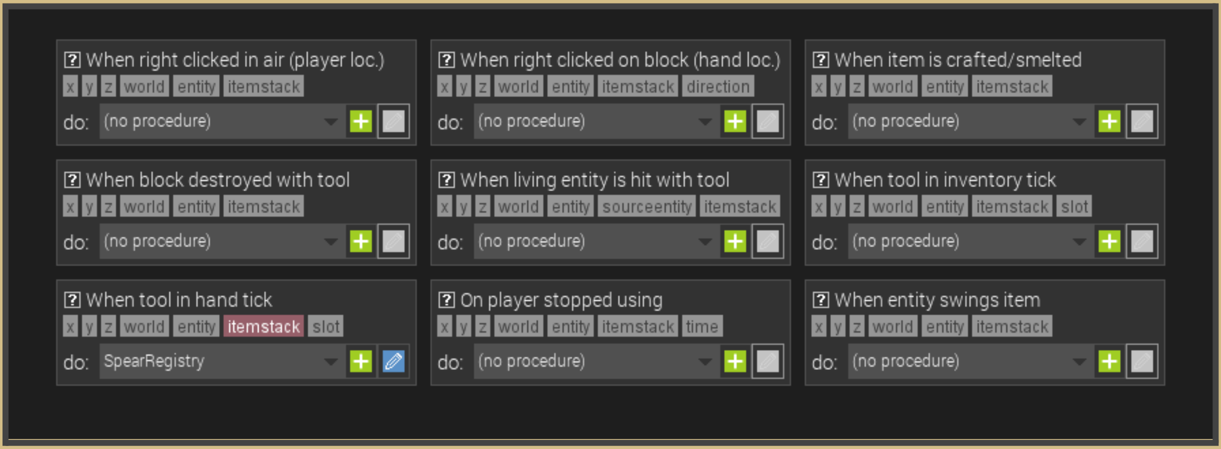
Next, make procedure defining all spear’s stats. Just use INBT tags with number values, list of which are above. For example, WoB’s procedure looks like this:



```
Event trigger - triggered by external call
or when (global trigger): No additional trigger
if  Get NBT number tag "spear_reg" of Provided itemstack < 15
do
    if  Provided itemstack = [item]
    do
        Set NBT number tag "spr_def_power" of Provided itemstack to 2
        Set NBT number tag "spr_def_time" of Provided itemstack to 80
        Set NBT number tag "spr_cooldown" of Provided itemstack to 301
        Set NBT number tag "spr_rg_power" of Provided itemstack to 1
        Set NBT number tag "spr_rg_factor" of Provided itemstack to 2
        Set NBT number tag "spear_reg" of Provided itemstack to 15
    else if  Provided itemstack = [item]
    do
        Set NBT number tag "spr_def_power" of Provided itemstack to 2
        Set NBT number tag "spr_def_time" of Provided itemstack to 140
        Set NBT number tag "spr_cooldown" of Provided itemstack to 301
        Set NBT number tag "spr_rg_power" of Provided itemstack to 1
        Set NBT number tag "spr_rg_factor" of Provided itemstack to 2
        Set NBT number tag "spear_reg" of Provided itemstack to 15
```

I used NBT tag "spear_reg" INBT tag just to call this procedure once. You don't have to, yet remember it's lag-friendly solution.

After defining statistics, bind that procedure to trigger you want to. Personally I'd suggest "when tool in hand tick":



```
When right clicked in air (player loc.)    x y z world entity itemstack                  do: (no procedure)
When right clicked on block (hand loc.)    x y z world entity itemstack direction        do: (no procedure)
When item is crafted/smelted               x y z world entity itemstack                  do: (no procedure)
When block destroyed with tool             x y z world entity itemstack                  do: (no procedure)
When living entity is hit with tool        x y z world entity sourceentity itemstack     do: (no procedure)
When tool in inventory tick                x y z world entity itemstack slot             do: (no procedure)
When tool in hand tick                     x y z world entity itemstack slot             do: SpearRegistry
On player stopped using                    x y z world entity itemstack time             do: (no procedure)
When entity swings item                    x y z world entity itemstack                  do: (no procedure)
```

Spires of Baedoor

List of available integrations (for mods)

Integrating gems for scrolls

If you create scrolls, you usually need to use gems to make them usable - being it different variant, or even the basic one. Of course Spires of Baedoor uses some resources for this purpose, but with your mod, you can expand it more with your own gems.

You can add your gems to SoB list by just adding them to its own Forge item tag:
❈ scroll_gem_self - for self-applicable scrolls (cheap gems)
❈ scroll_gem_target - for scrolls used on other entities (medium rare gems)
❈ scroll_gem_field - for scrolls using field effects (rare gems)

Civilisations of Baedoor

List of available integrations (for mods)

- ❖ Generating mobs/structures/blocks in CoB biomes

Generating mobs/structures/blocks in CoB biomes

If you want your custom elements (mobs/structures/blocks) to spawn in specific dimension or biome from Civilisations, you can do it via conditions related to biome temperature. CoB biomes have exclusive temperature values using six digits after comma, meaning you will spawn your stuff only in these biomes specifically.

DUNES DIMENSION

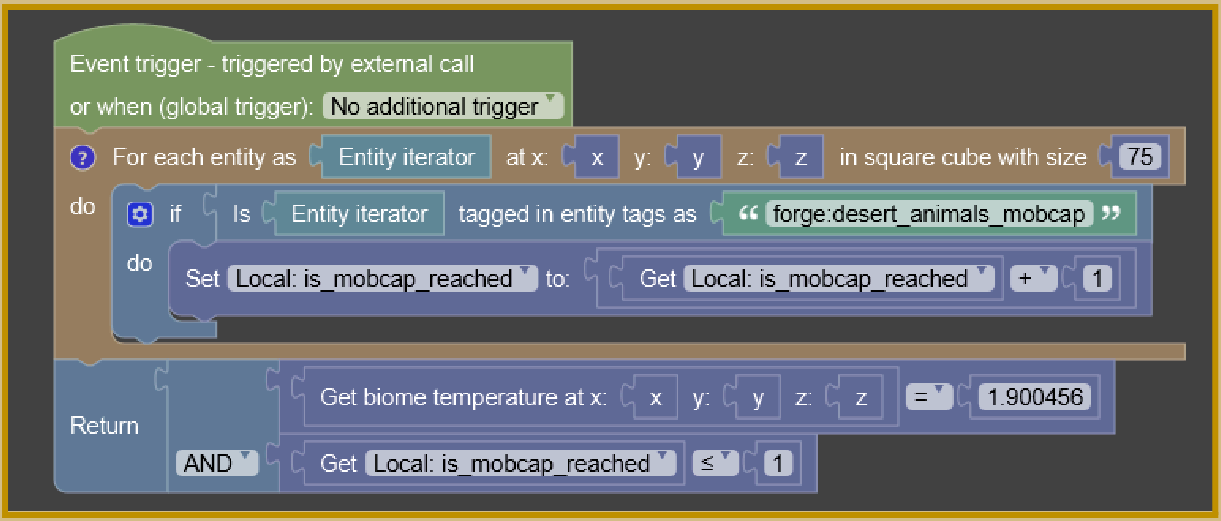
❈ Dry Lands - 1.900456
❈ Eotic Lakes - 1.800368
❈ Tertenic Dunes - 1.900582
❈ Tertenic Dunes Edge - 1.900544

Mobs

It is important to remember that CoB use their own mobcaps system, based on Forge tags to determine whether mob spawn or not. You can set the animal to “ambient” this way without fear that it will flood the world endlessly.
If you do so, follow these steps:

To use mobcap, simply create and add your mob to one of the following Forge entity Tags:
❈ desert_animals_mobcap - for desert animals in Dunes

After that, condition your mob to appear only in specific temperatures for biomes (you can specify more than one). To use mobcap, make simple counter like shown below:



```
Event trigger - triggered by external call
or when (global trigger): No additional trigger
For each entity as Entity iterator at x: x y: y z: z in square cube with size 75
do
    if Is Entity iterator tagged in entity tags as "forge:desert_animals_mobcap"
    do
        Set Local: is_mobcap_reached to: Get Local: is_mobcap_reached + 1
Return Get biome temperature at x: x y: y z: z = 1.900456
AND Get Local: is_mobcap_reached ≤ 1
```

You need to create local numerical variable for it, in my case it is called "is_mobcap_reached".
Example seen on screenshot will spawn your mob only in Dry Lands (temperature on 1.900456), when there is up to 1 mob in range of 75 blocks.

You can download .ptpl example file from here, so it will serve you as a template.

If you want to increase the rates of spawn, either raise maximum of this variable (**1** on the end) or lower the iteration cube (**75** by default).