<u>HOW TO … ?</u>

…

These tips and tricks will take your LearnToMod Minecraft mods to the next level! They were formerly posted in the LearnToMod forum and expand on the coding fundamentals in the LearnToMod badge tutorials.

…

/// Player events not included in the event blocks

/// How to check if the player has a certain item or block in their inventory?

/// How to use the import and export blocks?

/// How to make something happen when I do a mouse click?

/// How to make it so the player does not get damaged by falling?

/// How to move a drone to a specific set of coordinates?

/// How to change the projectile from a bow?

/// How to achieve a one-hit kill with a specific item?

/// How to change the item/block an entity drops when it dies?

/// How to make something happen when I click on a specific block face (side)?

/// How to make something happen at the location of a projectile hit?

/// How to use loops to read what's in a List?

/// How to customize Minecraft commands?

/// How to run mods of multiple users in the new 1.9 servers?

/// How to make something happen when I chat a specific message?

/// How to use list blocks?

/// How to make something happen when I break a specific block?

///How to make something happen whenever I click with a specific item in hand?

///How to change the item/block drops of the block I break?

///How to make a block “unbreakable”?

///How to get the health percentage of an entity?

///How to apply texture packs to everybody in the server?

///How to create a Vox-L mod

///How to trigger a mod using a chat message in Vox-L?

///How to make an event work for all the players in the server?

///How to pick up items from a larger radius

///How to create and add users to a player list?

///How to check if a player joins your server

///How to debug entity damage by entity event?

///How to turn commands into an easy to use library?

///How to debug player interact event?

///How to debug block break event?

///How to check the type of item/block the player picks up?

///How to add to a list?

///How to make the egg that I throw always hatch?

///How to hatch more than one entity from an egg that I throw?

///How to hatch a different entity from an egg that I throw?

///How to make something happen at the location of the block I break?

///Loading texture packs

///How to get the item's display name?

///I want to create an enchanted item using the .addEnchantment() function of the ItemStack class, not by using the /give command.

/// How to make your own player head

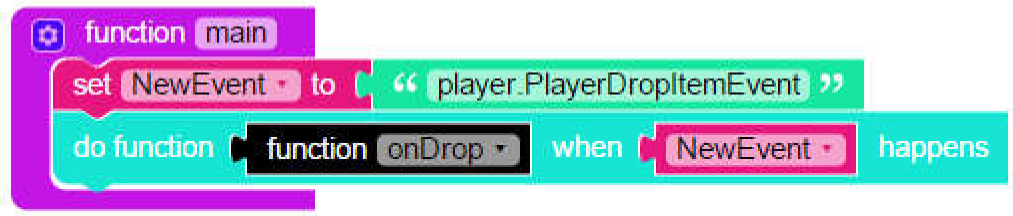
/// Player events not included in the event blocks

carlosiscool

The following table includes a list of events that aren't listed in the blockly event block. Also included int he table is a description of the event as well as what information you can get from the event and what can be set in the event.

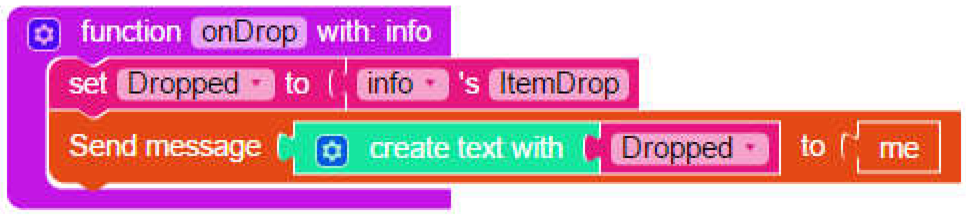
If you want to use one of these events in your mod, create a new variable (NewEvent) set to the name of the event you want to use. Since these are all "player events", every event will start with player. followed by the name of the event, as is written on the table. Then in your event block, set the variable as the event.

For a PlayerDropItemEvent I would use the following:



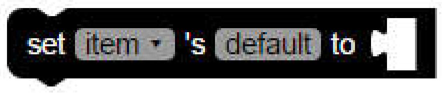
```
function main
    set NewEvent to "player.PlayerDropItemEvent"
    do function function onDrop when NewEvent happens
```

In order to "get" information from the event, create an info parameter in the function being called from the event and set a variable to info's [blank] the blank can be whatever is listed in the "get" part of the table for a particular event. You can get the "player" from every one of the events on the table because they are all player events.



```
function onDrop with: info
    set Dropped to info 's ItemDrop
    Send message create text with Dropped to me
```

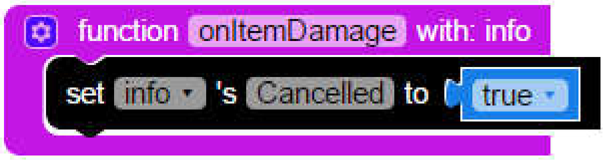
To "set" information from an event, you can use the following block from the misc menu:



```
set item 's default to
```

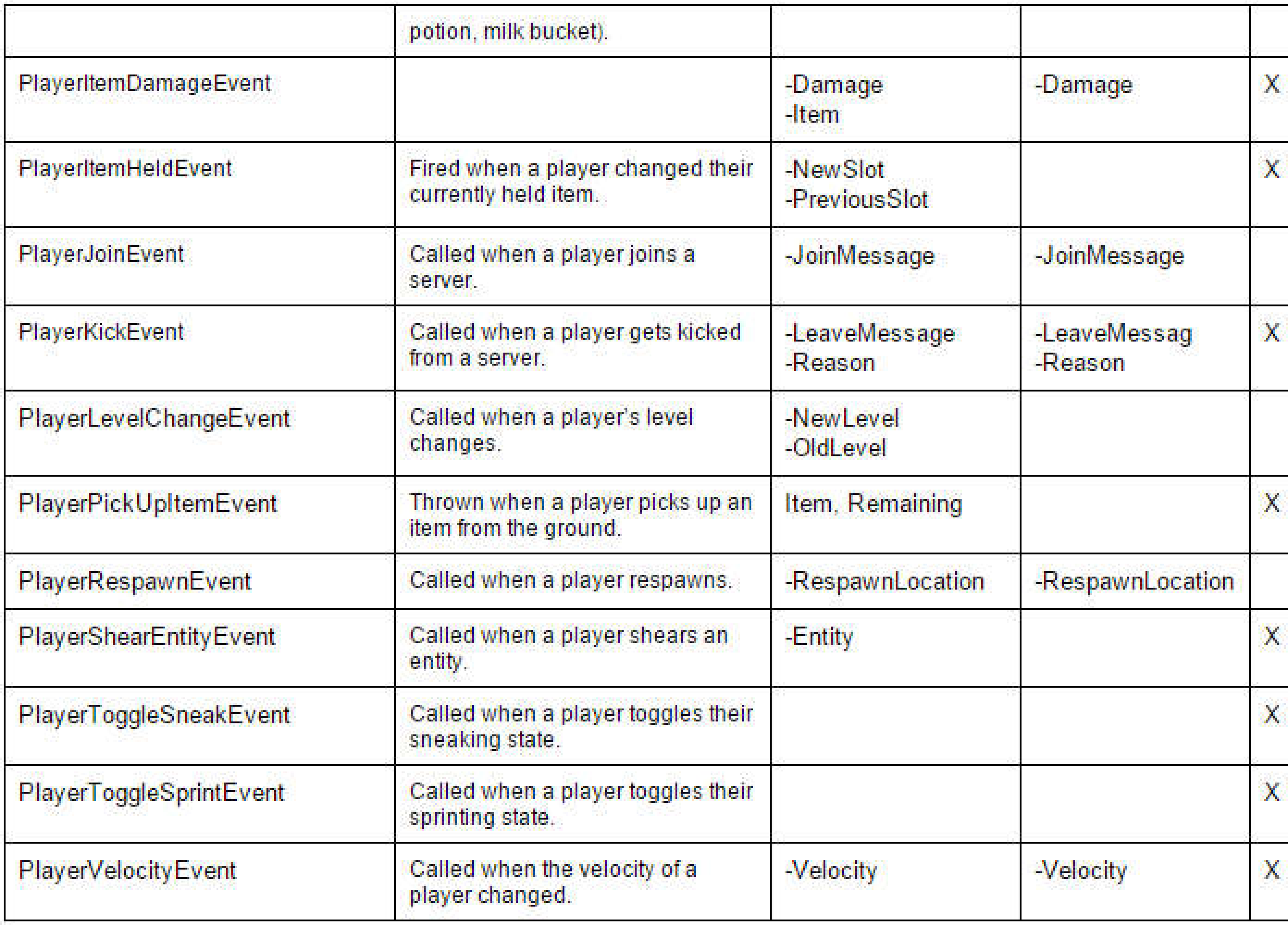
Not all of the events have information that can be set to a new value, but they would follow a similar format to the "get" methods. **Set [info's] [Set Method] to [New Value]**

Lastly, you will notice that there is a column with a C on it, this is to indicate whether the event is cancellable or not. For example, if you wanted to items to not take damage, you could cancel the **PlayerItemDamageEvent** by using the same block used to "set" information and **set [info's] [Cancelled] to [True]**

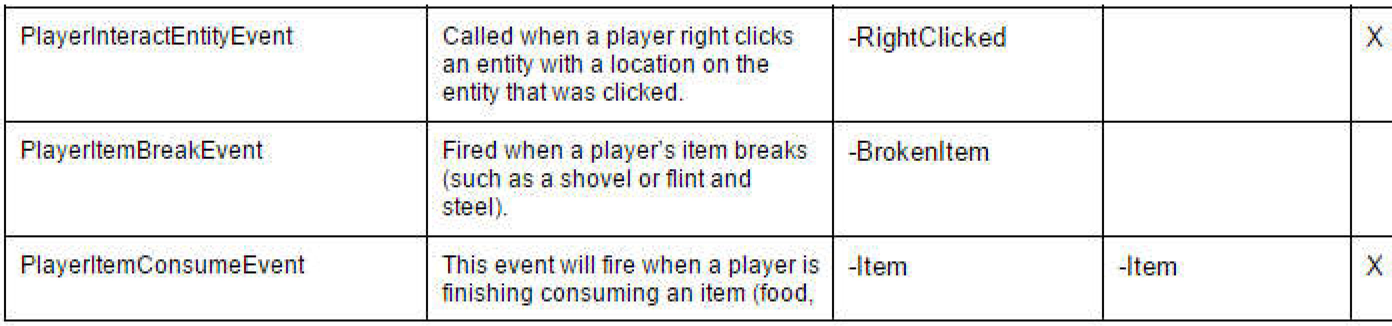


```
function onItemDamage with: info
  set info 's Cancelled to true
```

player.[Event]	Event Description	get	set	C
PlayerArmorStandManipulateEvent	Called when a player interacts with an armor stand and will either swap, retrieve or place an item.	-ArmorStandItem -PlayerItem -RightClicked -Slot		
PlayerBedEnterEvent	This event is fired when the player is almost about to enter the bed.	-Bed		X
PlayerBedLeaveEvent	This event is fired when the player is leading a bed.	-Bed		
PlayerBucketEvent	Called when a player interacts with a bucket.	-BlockClicked -BlockFace, -Bucket -ItemStack	-ItemStack	X
PlayerCommandProcessEvent	This event is called whenever a player runs a command (by placing a slash at the start of their message).	-Message	-Message -Player	
PlayerDropItemEvent	Thrown when a player drops an item from their inventory.	-ItemDrop		X
PlayerEditBookEvent	Called when a player edits or signs a book and quill item.	-NewBookMeta -PreviousBookMeta -Slot	-NewBookMeta -Signing	X
PlayerExpChangeEvent	Called when a player's experience changes naturally.	-Amount	-Amount	
PlayerFishEvent	Thrown when a player is fishing.	-Caught -ExpTpDrop -Hook -State	-ExpToDrop	X
PlayerGameModeChangeEvent	Called when the GameMode of the player is changed.	-NewGameMode		X



	potion, milk bucket).			
PlayerItemDamageEvent		-Damage -Item	-Damage	X
PlayerItemHeldEvent	Fired when a player changed their currently held item.	-NewSlot -PreviousSlot		X
PlayerJoinEvent	Called when a player joins a server.	-JoinMessage	-JoinMessage	
PlayerKickEvent	Called when a player gets kicked from a server.	-LeaveMessage -Reason	-LeaveMessag -Reason	X
PlayerLevelChangeEvent	Called when a player's level changes.	-NewLevel -OldLevel		
PlayerPickUpItemEvent	Thrown when a player picks up an item from the ground.	Item, Remaining		X
PlayerRespawnEvent	Called when a player respawns.	-RespawnLocation	-RespawnLocation	
PlayerShearEntityEvent	Called when a player shears an entity.	-Entity		X
PlayerToggleSneakEvent	Called when a player toggles their sneaking state.			X
PlayerToggleSprintEvent	Called when a player toggles their sprinting state.			X
PlayerVelocityEvent	Called when the velocity of a player changed.	-Velocity	-Velocity	X



PlayerInteractEntityEvent	Called when a player right clicks an entity with a location on the entity that was clicked.	-RightClicked		X
PlayerItemBreakEvent	Fired when a player's item breaks (such as a shovel or flint and steel).	-BrokenItem		
PlayerItemConsumeEvent	This event will fire when a player is finishing consuming an item (food,	-Item	-Item	X

/// How to check if the player has a certain item or block In their inventory?

Mr_Sonny

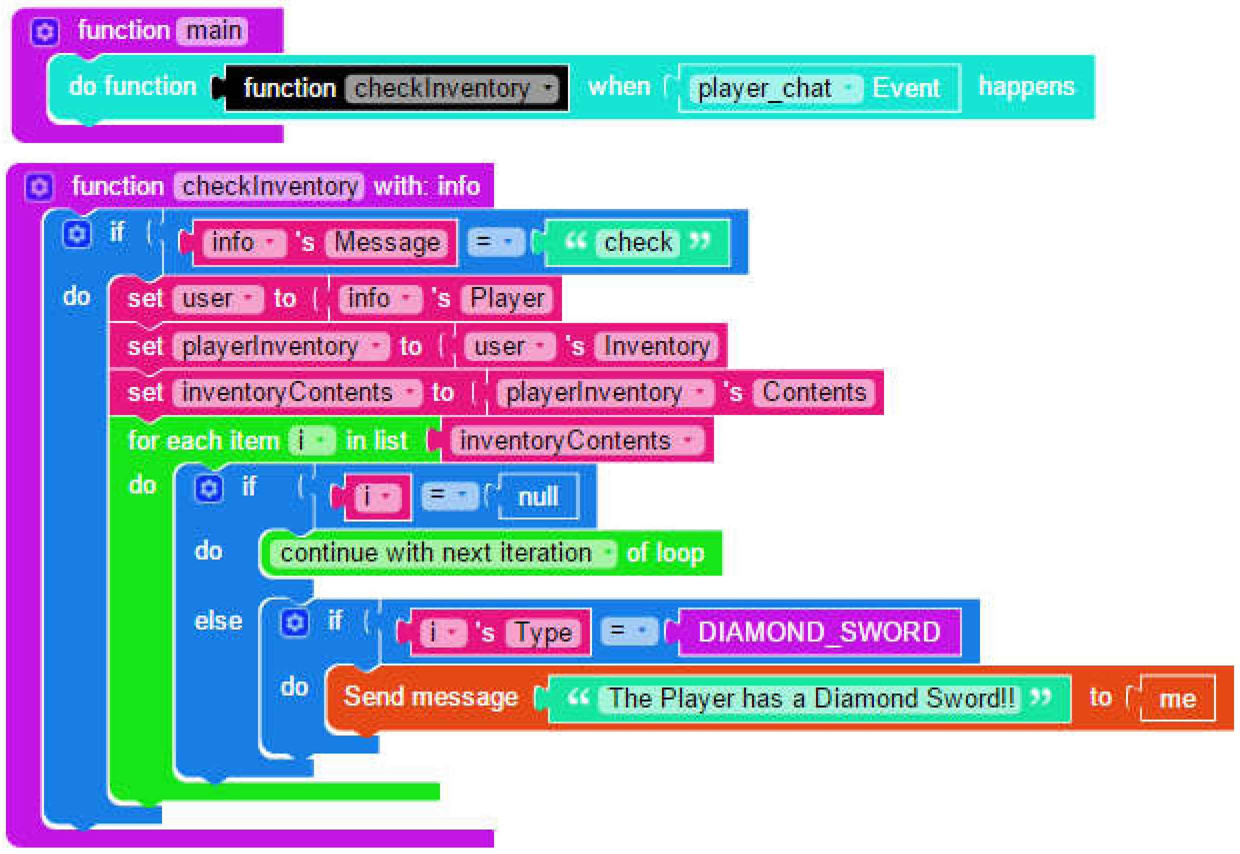
First add an event inside of the main function. You need to add a function reference (to **checkInventory**) and a "**player_chat Event**" (**NOTE**: this event can be changed to something else, depending on how you want to trigger the checkInventory function).

The function that you reference (**checkInventory**) needs to have a parameter (in this case **info**), this will allow the code to retrieve information from the event. Inside of this new function you will need to check if the message that was chatted was **check**; do this by getting an if statement and checking if "**info's Message = 'check**'".

If that returns true you need to create and set three new variables: set "**user**" to "**info's Player**" to retrieve who chatted, then set "**playerInventory**" to "**user's Inventory**" to retrieve the inventory of that player, and finally set "**inventoryContents**" to "**playerInventory's Contents**" to retrieve the list of items in the inventory.

Next you have to loop through each item of the list and check each time if the current item is the one you're looking for (**NOTE**: if the current slot is empty [null] it has to skipped and check the next slot [continue with next iteration]). Make sure you look carefully at the image below, there is an inner if statement in the else section of the outer if statement; and you need to check "**i's Type"**which is the type of item/block in that current slot.

This mod checks the player's inventory when they chat "check", and prints out "The player has a Diamond Sword" if it finds one in their inventory.



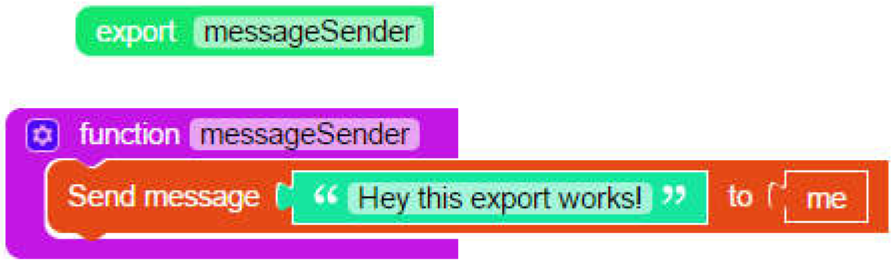
```
function main
  do function  function checkInventory  when  player_chat Event  happens

function checkInventory with: info
  if  info 's Message  =  “ check ”
  do  set user to  info 's Player
      set playerInventory to  user 's Inventory
      set inventoryContents to  playerInventory 's Contents
      for each item i in list  inventoryContents
      do  if  i  =  null
          do  continue with next iteration of loop
          else  if  i 's Type  =  DIAMOND_SWORD
                do  Send message  “ The Player has a Diamond Sword!! ”  to  me
```

/// How to use the import and export blocks?

IronGiant

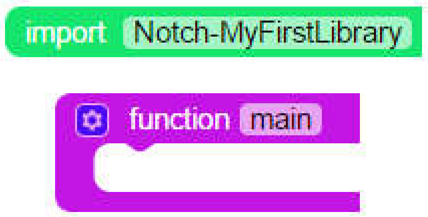
Exporting functions is easy -- and useful! **Export** blocks allow you to copy functions from another mod and place it into your current mod using **import** blocks. Mods that contain no “main” method and just exportable functions are called “libraries”. You can use exports and libraries to keep your big mods clean, simple, and easy-to-read!

First, create two mods, one called “MyFirstLibrary” and the other called “MyTestMod”. Inside “MyFirstLibrary”, make a simple function that sends the message “Hey, this export works!” Then, drag an **export** block from the **Misc** tab above the function and type the function’s name in the box.



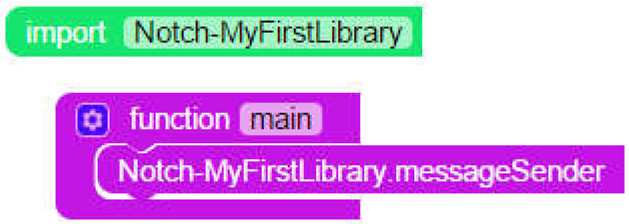
```
export messageSender

function messageSender
    Send message “ Hey this export works! ” to me
```

Now jump out and go over to “MyTestMod”. Make a “main” function, but don’t put anything in it. Drag an **import** block from the **Misc** tab and put it above your main. Now, in order to import the code you exported over in “MyFirstLibrary”, you have to type “-” into the box. So, if my nickname was Notch and I wanted to import “MyFirstLibrary”, it would look something like this:



```
import Notch-MyFirstLibrary

function main
```

Once you’ve imported your library, check out the **Functions** tab. Your “messageSender” function should be there! Grab it and drag it into your main. Run your mod, and you’ll get the “Hey this export works!” message you made in your “MyFirstLibrary” mod.



```
import Notch-MyFirstLibrary

function main
    Notch-MyFirstLibrary.messageSender
```

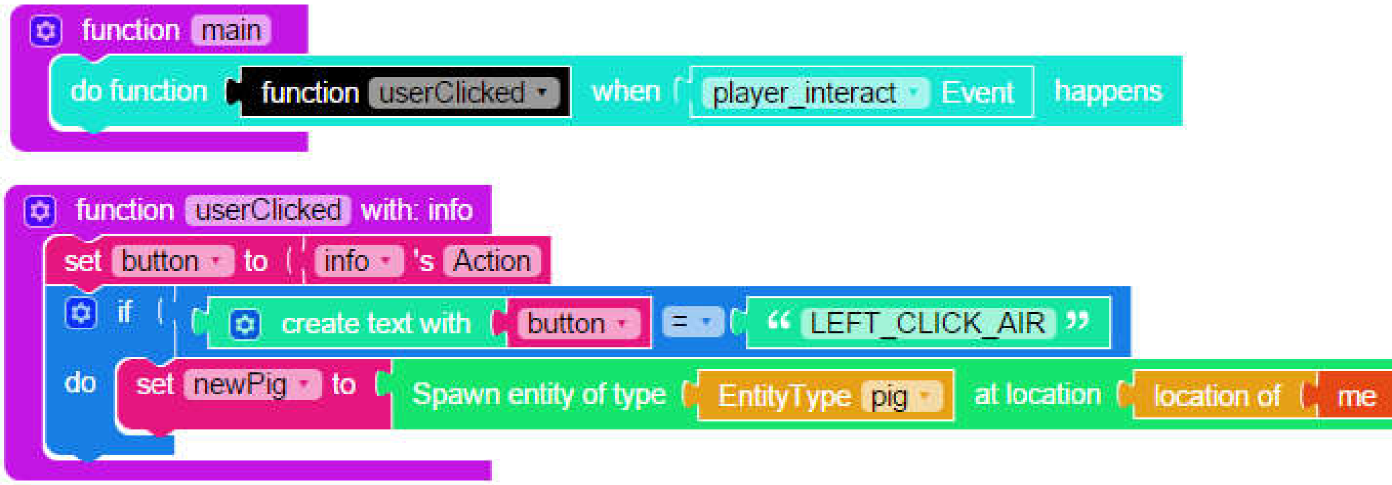
/// How to make something happen when I do a mouse click?

First add an event inside of the main function. You need to add a function reference (to **userClicked**) and a **“player_interact Event”**. The function that you reference (**userClicked**) needs to have a parameter (in this case **info**), this will allow the code to retrieve information from the event.

Inside of this new function you will need one variable: **button.** You need to set button to **“info’s Action**”; which is the action that was performed. There are four possibilities: LEFT_CLICK_AIR, LEFT_CLICK_BLOCK, RIGHT_CLICK_AIR, and RIGHT_CLICK_BLOCK.

Now that you have the necessary information, by using an if statement you can compare that to the desired action, and make something happen accordingly.

In this mod a pig will spawn whenever the player left clicks the air:



```
function main
  do function ( function userClicked ) when ( player_interact Event ) happens

function userClicked with: info
  set button to ( info 's Action )
  if ( ( create text with ( button ) = ( " LEFT_CLICK_AIR " )
  do set newPig to ( Spawn entity of type ( EntityType pig ) at location ( location of ( me
```

Links to similar badges:

https://mod.learntomod.com/badge_builders/3187?sequence=9084&position=76

https://mod.learntomod.com/badge_builders/2808?sequence=9049&position=13

/// How to make it so the player does not get damaged by falling?

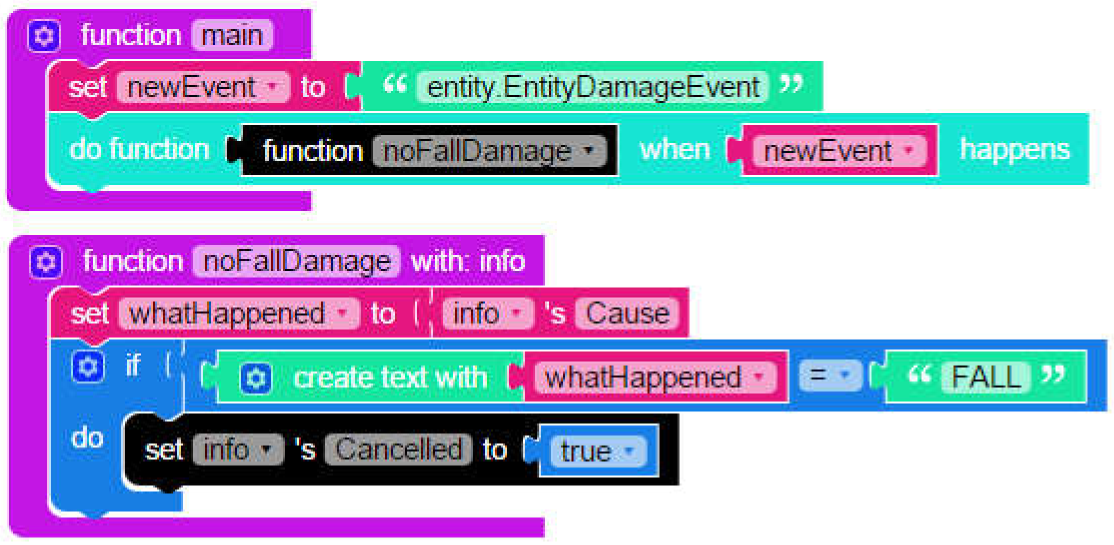
First you need to create a variable (**newEvent**) and set it to “**entity.EntityDamageEvent**”. You need to do this because in the dropdown menu there is not an “entity_damage Event” option. There are hundreds of different events that are not in the dropdown list, and this is how you would add them to your code.

Then add an event inside of the main function. You need to add a function reference (to **noFallDamage**) and add the variables “**newEvent**” to the second slot. The function that you reference (**noFallDamage**) needs to have a parameter (in this case **info**), this will allow the code to retrieve information from the event.

Inside of this new function you will need one variable: **whatHappened**. You have to set it to “**info’s Cause**”; which is the reason why the entity got damaged. There many different reasons like FIRE, POISON, SUFFOCATION, etc..

Now that you have the necessary information, by using an if statement you can compare that to the desired cause, and if this is true then set “**info’s Cancelled**” to “**true**”. This will cancel the damage that you take from falling.

In this mod whenever an entity falls from a high place it will not get damaged:



```
function main
  set newEvent to “ entity.EntityDamageEvent ”
  do function function noFallDamage when newEvent happens

function noFallDamage with: info
  set whatHappened to info 's Cause
  if create text with whatHappened = “ FALL ”
  do
    set info 's Cancelled to true
```

NOTE:

Instead of using the word "FALL", you can use any of the following:

BLOCK_EXPLOSION

Damage caused by being in the area when a block explodes.

CONTACT

Damage caused when an entity contacts a block such as a Cactus.

CUSTOM

Custom damage.

DRAGON_BREATH

Damage caused by a dragon breathing fire.

DROWNING

Damage caused by running out of air while in water

ENTITY_ATTACK

Damage caused when an entity attacks another entity.

ENTITY_EXPLOSION

Damage caused by being in the area when an entity, such as a Creeper, explodes.

FALL

Damage caused when an entity falls a distance greater than 3 blocks

FALLING_BLOCK

Damage caused by being hit by a falling block which deals damage

FIRE

Damage caused by direct exposure to fire

FIRE_TICK

Damage caused due to burns caused by fire

FLY_INTO_WALL

Damage caused when an entity runs into a wall.

HOT_FLOOR

Damage caused when an entity steps on Material.MAGMA.

LAVA

Damage caused by direct exposure to lava

LIGHTNING

Damage caused by being struck by lightning

MAGIC

Damage caused by being hit by a damage potion or spell

MELTING

Damage caused due to a snowman melting

POISON

Damage caused due to an ongoing poison effect

PROJECTILE

Damage caused when attacked by a projectile.

STARVATION

Damage caused by starving due to having an empty hunger bar

SUFFOCATION

Damage caused by being put in a block

SUICIDE

Damage caused by committing suicide using the command "/kill"

THORNS

Damage caused in retaliation to another attack by the Thorns enchantment.

VOID

Damage caused by falling into the void

WITHER

Damage caused by Wither potion effect

/// How to move a drone to a specific set of coordinates?

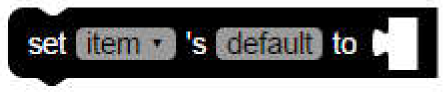
carlosiscool

First you will need to create a Drone (**d**) and get the drone's location with a separate variable (**d_loc**). The location of an entity is composed of the world, world name, x-position, y-position, z-position, pitch and yaw. If you decide to create text with this variable, you will receive something like:

Location {world = CraftWorld { name = current_world}, x = 22232.0, y = 3.0, z = -733.0, pitch = 0.0, yaw = 0.0}

In this case, you only really care about **d_loc's X, Y and Z** coordinates.

Under the Misc tab you will see a block that looks like this:



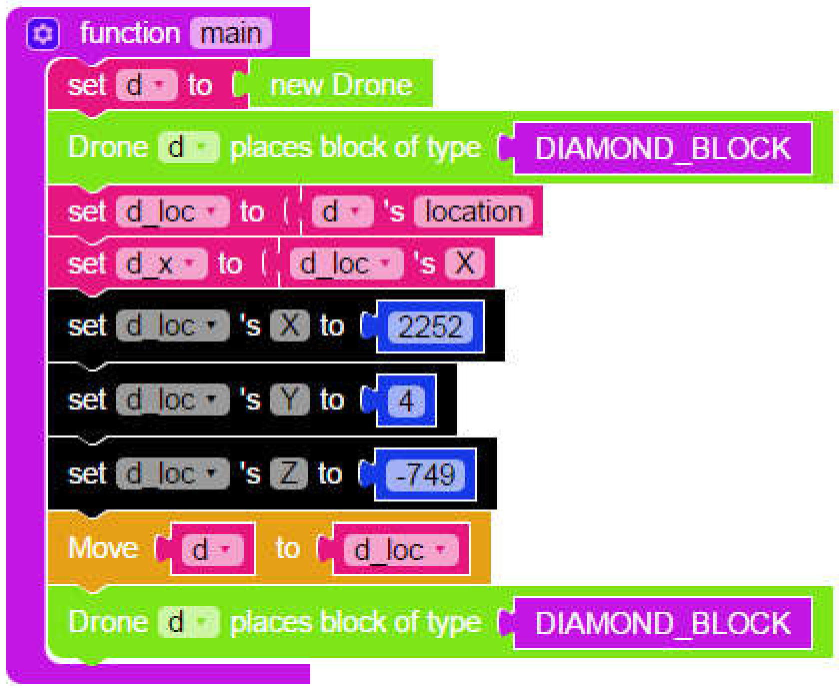
```
set item ▾ 's default to
```

You can use this block to set **d_loc's new X, Y and Z coordinates**. Under the d_loc variable, place three of the "set item's__ to__" block. For each of those blocks you'll be setting d_loc's X, Y or Z to the new desired coordinates.

Once d_loc has updated coordinates, teleport you drone (**d**) to the new location (**d_loc**) using the "move__ to__" block.

Since the tp command does not work with drones, you can set new coordinate values to the drones location and move the drone to the new location. This is the equivalent of saying /tp droneName [x] [y] [z]

In the example below, I spawned at the coordinates **2252, 4, -749**, I moved away from that location and ran my code which placed a diamond block at the block I was looking at, my code updated the drones location to my original coordinates, moved the drone to that location and emerald block there. I know this works because I can tp myself to those coordinates and make sure that an emeral block was placed at those exact coordinates.



```
function main
    set d ▾ to new Drone
    Drone d ▾ places block of type DIAMOND_BLOCK
    set d_loc ▾ to d ▾ 's location
    set d_x ▾ to d_loc ▾ 's X
    set d_loc ▾ 's X to 2252
    set d_loc ▾ 's Y to 4
    set d_loc ▾ 's Z to -749
    Move d ▾ to d_loc ▾
    Drone d ▾ places block of type DIAMOND_BLOCK
```

/// How to change the projectile from a bow?

(Mr_Sonny)

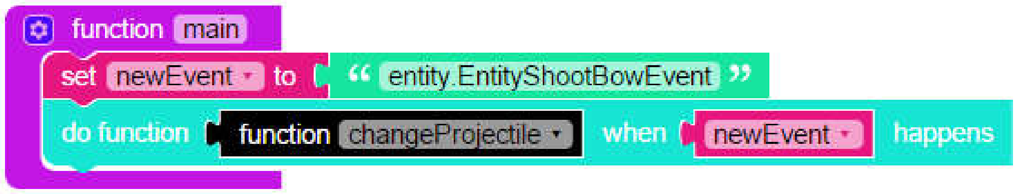
First you need to create a variable (**newEvent**) and set it to “**entity.EntityShootBowEvent**”. You need to do this because in the dropdown menu there is not an “entity_shoot_bow Event” option. There are hundreds of different events that are not in the dropdown list, and this is how you would add them to your code.

Then add an event inside of the main function. You need to add a function reference (to **changeProjectile**) and add the variable “**newEvent**” to the second slot. The function that you reference; **changeProjectile**; needs to have a parameter (in this case **info**), this will allow the code to retrieve information from the event.

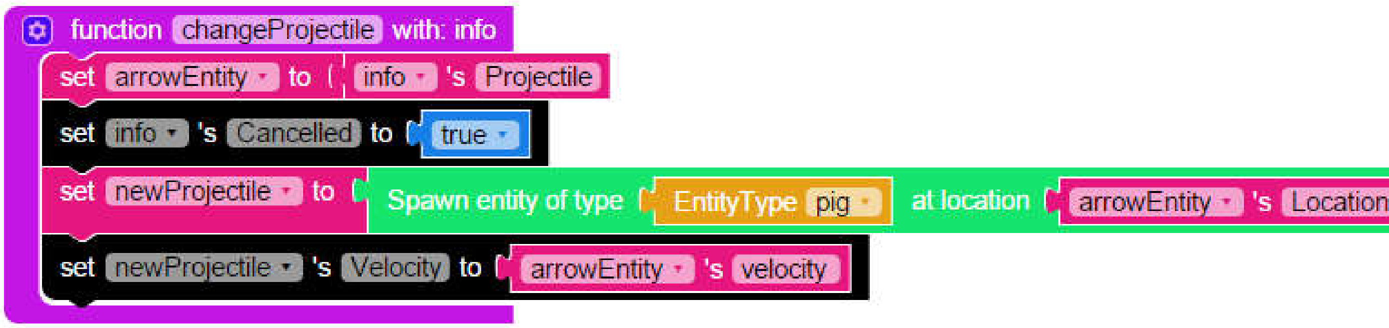
Inside of this new function you need to create a varible called **arrowEntity** and set it to "**info's Projectile**"; this will save all of the arrow's information for later use. Next, you need to cancel the event; stop the arrow from spawning; to achieve this set **info's Cancelled** to **true**.

Then create a new variable, **newProjectile**, to spawn the entity that you want to get shot, and for the location set it to "**arrowEntity's Location**". Finally give the new entity the original arrow's velocity by setting the **newProjectile's Velocity** to **arrowEntity's Velocity**.

In this mod whenever the player shoots a bow, the arrow will be replaced by a pig.



```
function main
  set newEvent to “ entity.EntityShootBowEvent ”
  do function function changeProjectile when newEvent happens
```



```
function changeProjectile with: info
  set arrowEntity to info 's Projectile
  set info 's Cancelled to true
  set newProjectile to Spawn entity of type EntityType pig at location arrowEntity 's Location
  set newProjectile 's Velocity to arrowEntity 's velocity
```

NOTE from Brodosaur: To do this with a fireball:

summon a fireball at the location of a drone in front of "me", then set the velocity to 2, 2, and 2 (X Y Z). Since the fireball will only want to go in one direction, you will need to combine this with the "EXPLODE ON PROJECTILE LOCATION" mod available further down in this HOW TO article.

/// How to achieve a one-hit kill with a specific item?

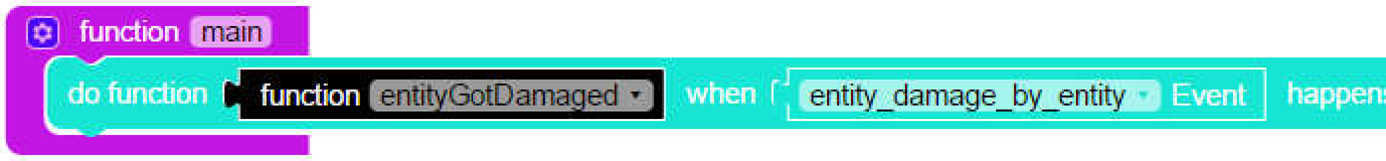
(Mr_Sonny)

First add an event inside of the main function. You need to add a function reference (to **entityGotDamaged**) and a "**entity_damage_by_entity Event**". The function that you reference (entityGotDamaged) needs to have a parameter (in this case info), this will allow the code to retrieve information from the event.

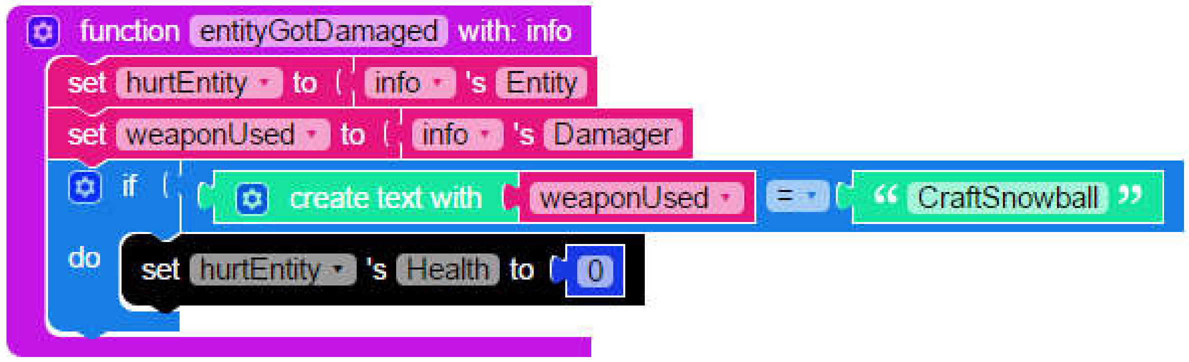
Inside of this new function you will need two variables: one to get the entity that is being hurt and another one to get the item/weapon that is damaging (**hurtEntity**, **weaponUsed**). You need to set hurtEntity to "**info's Entity**", and weaponUsed to "**info's Damager**".

Now that you have the necessary information, by using an if statement you can compare that to the desired item/weapon, and make something happen accordingly. **NOTE**: make sure to use the item's minecraft name; in this case CraftSnowball.

In this mod any entity that gets hit with a snowball will die instantly:



```
function main
  do function ( function entityGotDamaged ) when ( entity_damage_by_entity Event ) happens
```



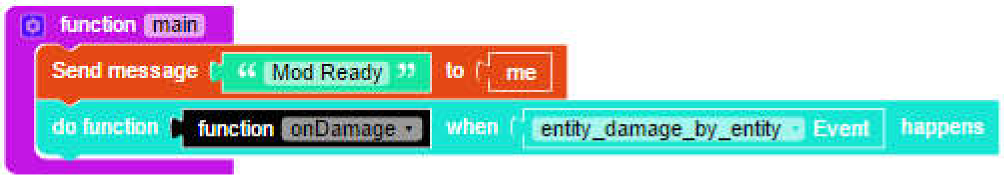
```
function entityGotDamaged with: info
  set hurtEntity to ( info 's Entity )
  set weaponUsed to ( info 's Damager )
  if ( ( create text with ( weaponUsed ) ) = ( " CraftSnowball " ) )
  do
    set hurtEntity 's Health to ( 0 )
```

Link to similiar badge: https://mod.learntomod.com/badge_builders/2819?sequence=9084&position=1

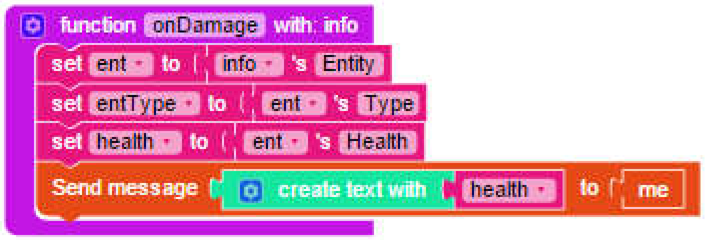
NOTE:

Only "entities" can be damagers (snowball, arrow, egg, player, creeper, etc

To check the health of any entity (including the player) that is damaged:



```
function main
  Send message " Mod Ready " to me
  do function function onDamage when entity_damage_by_entity Event happens
```



```
function onDamage with: info
  set ent to info 's Entity
  set entType to ent 's Type
  set health to ent 's Health
  Send message create text with health to me
```

/// How to change the item/block an entity drops when it dies?

(Mr_Sonny)

First add an event inside of the main function. You need to add a function reference (to **entityDied**) and a "**entity_death Event**". The function that you reference (**entityDied**) needs to have a parameter (in this case **info**), this will allow the code to retrieve information from the event.

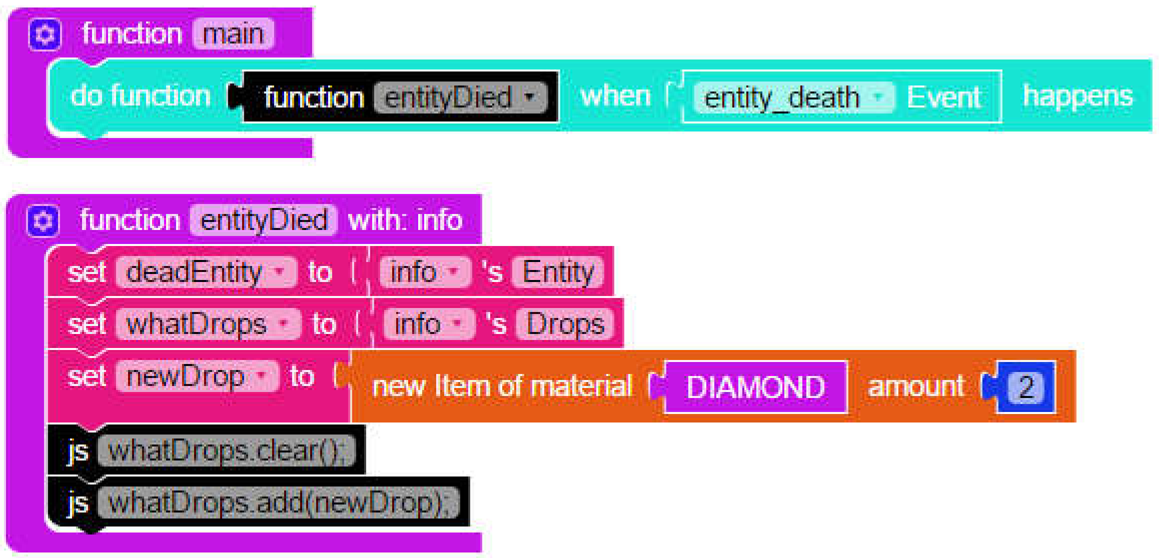
Inside of this new function you will need three variables: one to get the entity that died, one to get the regular itemstack drop, and the last one to hold the new itemstack (**deadEntity**, **whatDrops**, and **newDrop**).

Now that you have the necessary information you need two js blocks to enter the following code:

whatDrops.clear(); -- this will get rid of the regular things that drop

whatDrops.add(newDrop); -- this will add the new itemstack to the items that drop

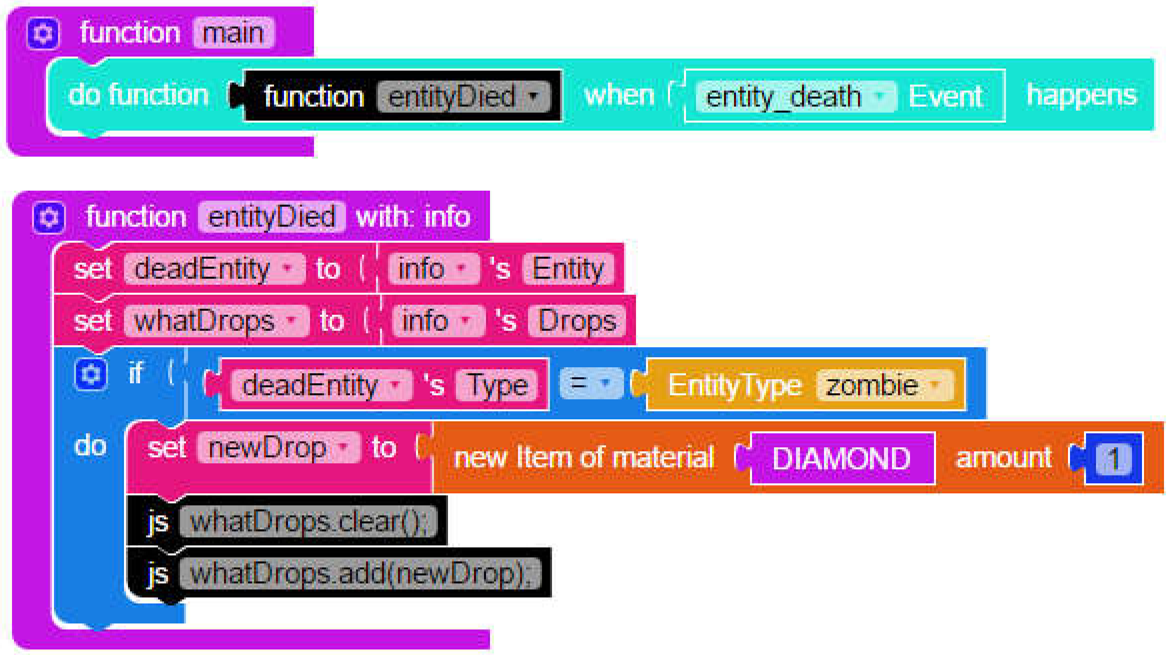
In this mod whenever an entity dies it will drop 3 diamonds:



```
function main
    do function ( function entityDied ) when ( entity_death Event ) happens

function entityDied with: info
    set deadEntity to ( info 's Entity )
    set whatDrops to ( info 's Drops )
    set newDrop to ( new Item of material ( DIAMOND ) amount ( 2 ) )
    js whatDrops.clear();
    js whatDrops.add(newDrop);
```

link to similar badge: https://mod.learntomod.com/badge_builders/2796?sequence=9018&position=6

To make a zombie drop diamonds:



```
function main
    do function ( function entityDied ) when ( entity_death Event ) happens

function entityDied with: info
    set deadEntity to ( info 's Entity )
    set whatDrops to ( info 's Drops )
    if ( ( deadEntity 's Type ) = ( EntityType zombie ) )
    do
        set newDrop to ( new Item of material ( DIAMOND ) amount ( 1 ) )
        js whatDrops.clear();
        js whatDrops.add(newDrop);
```

/// How to make something happen when I click on a specific block face (side)?

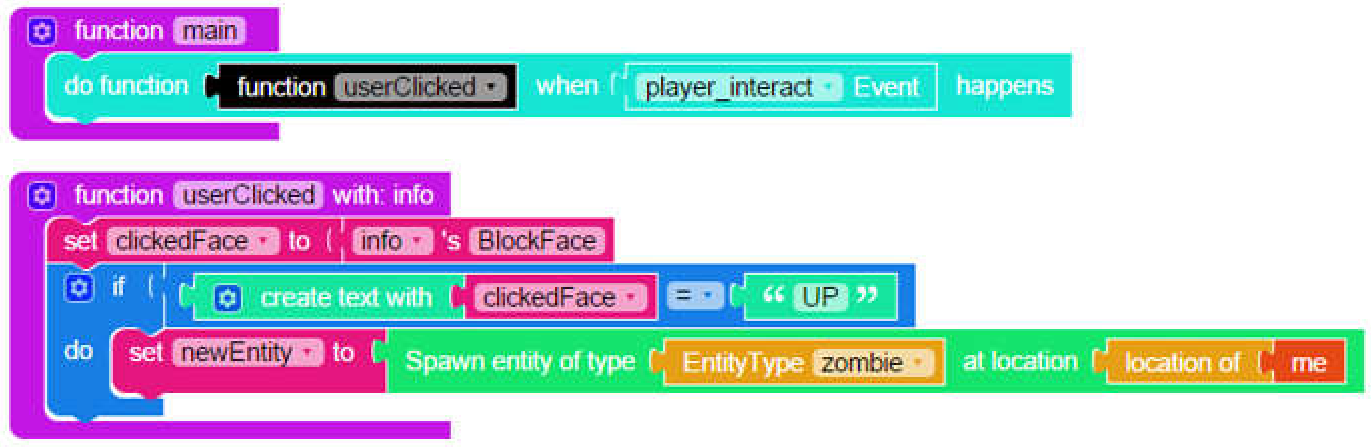
(Mr_Sonny)

First add an event inside of the main function. You need to add a function reference (to **userClicked**) and a “**player_interact Event**”. The function that you reference (userClicked) needs to have a parameter (in this case **info**), this will allow the code to retrieve information from the event.

Inside of this new function you will need one variable: **clickedFace**. You need to set clickedFace to “**info’s BlockFace**”; which is the face (side) of the block that the player clicked on. There are six possibilities: UP, DOWN, NORTH, EAST, SOUTH, WEST.

Now that you have the necessary information, by using an if statement you can compare that to the desired face, and make something happen accordingly.

In this mod a zombie will spawn whenever the player clicks on the UP (top) face of a block:



```
function main
    do function  function userClicked  when  player_interact Event  happens

function userClicked with: info
    set clickedFace to  info 's BlockFace
    if  create text with clickedFace  =  “ UP ”
    do  set newEntity to  Spawn entity of type  EntityType zombie  at location  location of  me
```

link to similar badge: https://mod.learntomod.com/badge_builders/2808?sequence=9049&position=1

/// How to make something happen at the location of a projectile hit?

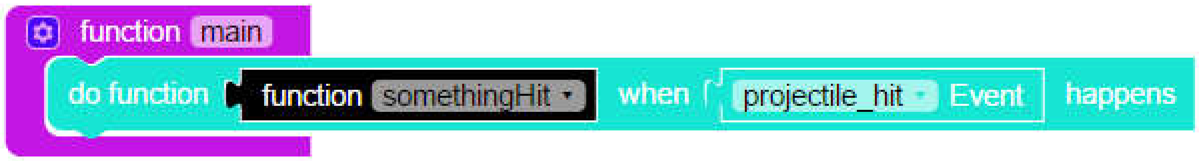
Mr_Sonny

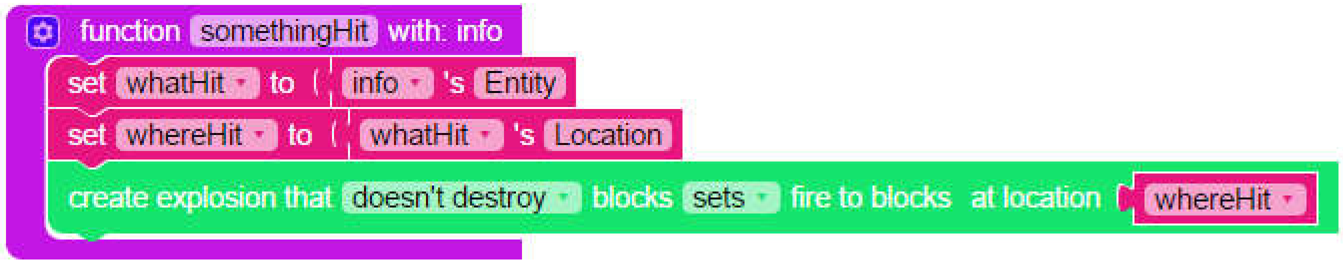
First add an event inside of the main function. You need to add a function reference (to **somethingHit**) and a “**projectile_hit Event**”. The function that you reference (**somethingHit**) needs to have a parameter (in this case **info**), this will allow the code to retrieve information from the event.

Inside of this new function you will need two variables: one to get the entity (projectile) that hit the ground, and another one to get the location of where it hit (**whatHit** and **whereHit**). You need to set whatHit to “**info’s Entity**”, and whereHit to “**whatHit’s Location**”.

Now that you have the necessary information, you can make something specific happen at that location.

In this mod whenever a projectile hits the ground an explosion will happen:


function main
do function function somethingHit when projectile_hit Event happens


function somethingHit with: info
set whatHit to info 's Entity
set whereHit to whatHit 's Location
create explosion that doesn't destroy blocks sets fire to blocks at location whereHit

Link to similiar badges:

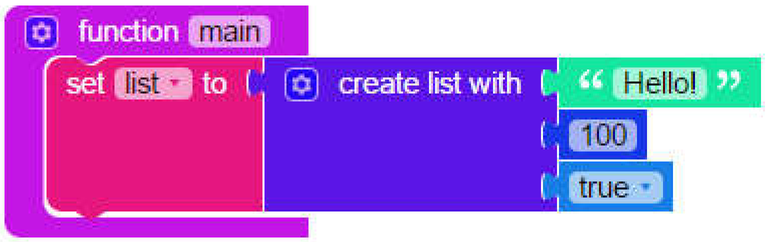
https://mod.learntomod.com/badge_builders/2804?sequence=9018&position=14

https://mod.learntomod.com/badge_builders/2805?sequence=9018&position=15

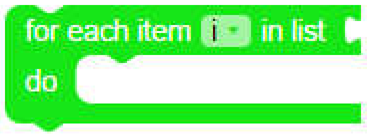
/// How to use loops to read what's in a list?

IronGiant

First you need a list full of things inside of it. Make that now! It should look something like this:

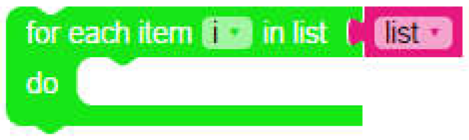


Once you've got it, grab the **For Each Item In List** block from the **Loops** tab and drag it below your list. It'll look something like this:

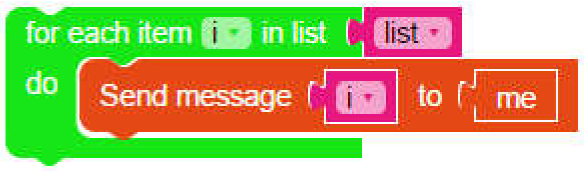


This block lets you move through each slot in your list and do whatever you want with the item in the slot. It creates a new variable called "i" that represents whatever's in the current slot of your list. This whole process is called "iterating", and what we're going to be doing is "iterating through a list" and sending ourselves a message of the list's contents.

To start, plug your list variable into the loop. This will tell the loop to start iterating over your list.



Within the loop, put a **Send Message** block that sends the “i” variable to “me”.



Once you’ve got it all, run your mod. You’ll see that contents of your list printed out for all to see! Think of the other possible things you can do now that you know how to access all the contents of your list. You can make some very powerful programs!

/// How to customize Minecraft commands?

Mr_Sonny

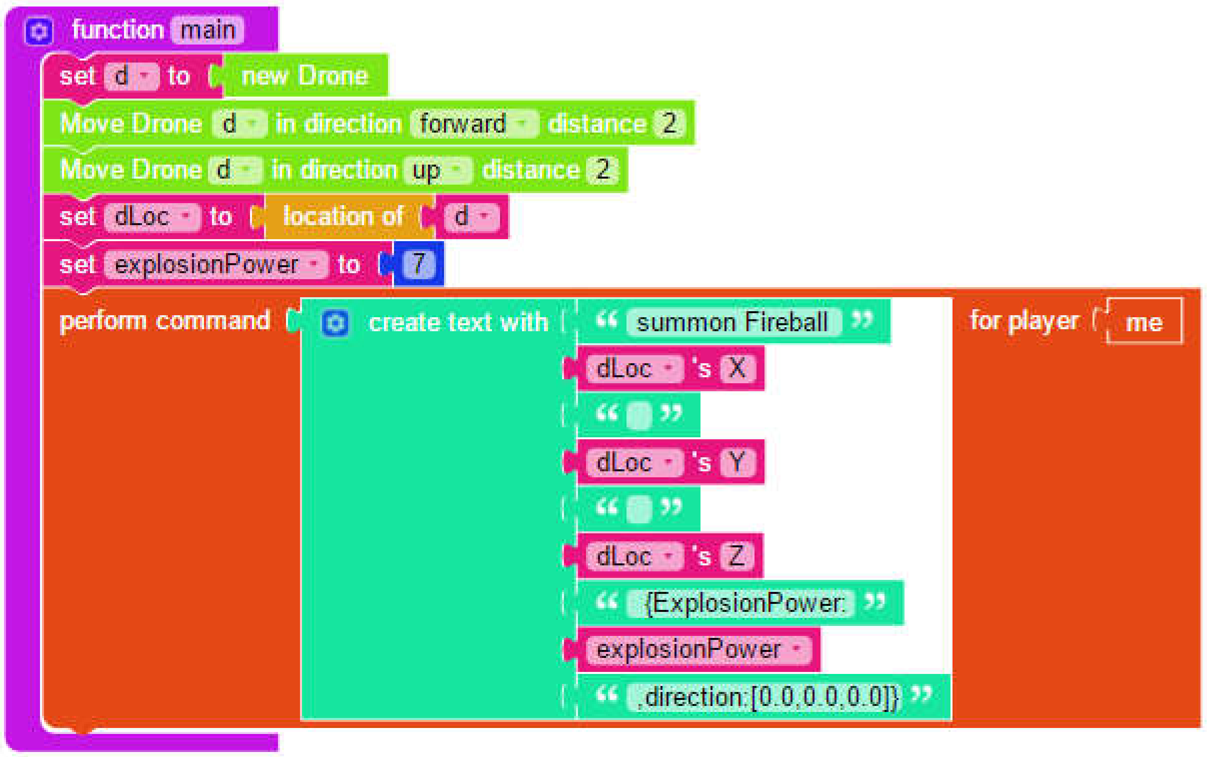
You need to choose a command that you want to customize. I will select a simple one that only has a few customizable parts (**summon**), and one that has many inputs (**give**).

For the first one I'm going to summon a **FIreball**, I want to spawn it at a drone's location, and I also want to set it's explosion power. First, create a drone an move it the the desired location, save that location in another variable, and then set **explostionPower** to the number you want.

Get a perform command block and attach a "create text with" block with 9 items. The first item will simply be the command: "**summon Fireball** " (space at the end). Then you attach the coordinates of where you want it to spawn (x, y, and z coordinates). You have to attach a text block in between those values, so you can add a space (you have to literally click on the block and hit the space bar).

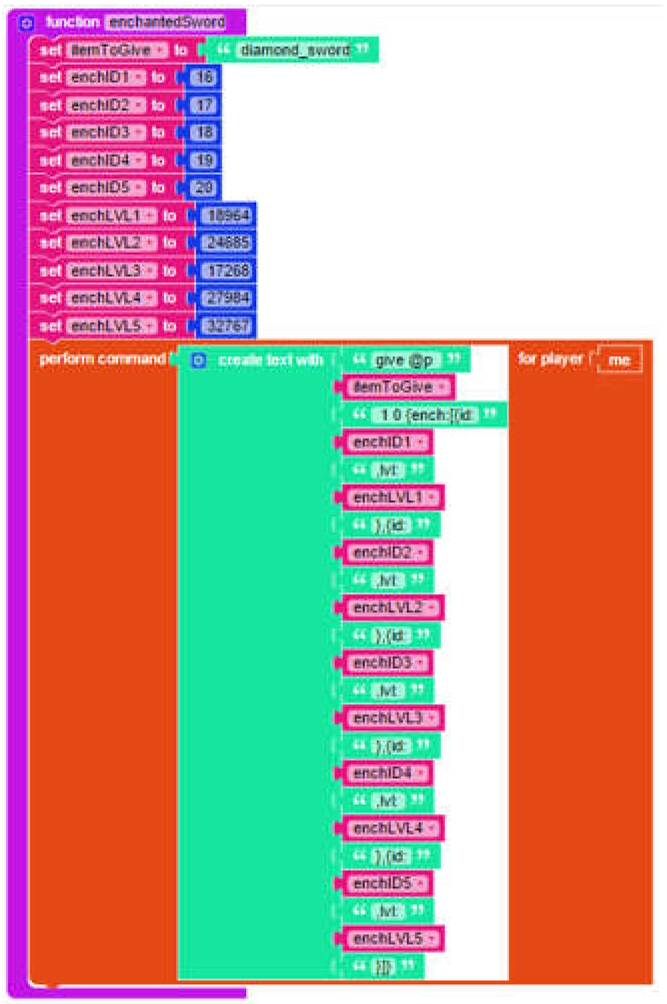
Finally, type in " **{ExplosionPower:** (space at the beginning) in a text block, after that add the variable, and finally type in "**,direction:[0.0,0.0,0.0]}**" (no spaces) in the last input.

NOTE: It is very important that you keep track of the spaces, and parentheses/brackets; if you're missing one of them the command won't work.



```
function main
  set d to new Drone
  Move Drone d in direction forward distance 2
  Move Drone d in direction up distance 2
  set dLoc to location of d
  set explosionPower to 7
  perform command create text with " summon Fireball "
                                   dLoc 's X
                                   "  "
                                   dLoc 's Y
                                   "  "
                                   dLoc 's Z
                                   " {ExplosionPower: "
                                   explosionPower
                                   " ,direction:[0.0,0.0,0.0]} "
                  for player me
```

The second example gives the player a Diamond Sword with five different enchantments of different levels:



```
function enchantedSword
  set itemToGive to "diamond_sword"
  set enchID1 to 16
  set enchID2 to 17
  set enchID3 to 18
  set enchID4 to 19
  set enchID5 to 20
  set enchLVL1 to 18964
  set enchLVL2 to 24685
  set enchLVL3 to 17268
  set enchLVL4 to 27984
  set enchLVL5 to 32767
  perform command  create text with  "give @p "        for player "me"
                                     itemToGive
                                     " 1 0 {ench:[{id:"
                                     enchID1
                                     ",lvl:"
                                     enchLVL1
                                     "},{id:"
                                     enchID2
                                     ",lvl:"
                                     enchLVL2
                                     "},{id:"
                                     enchID3
                                     ",lvl:"
                                     enchLVL3
                                     "},{id:"
                                     enchID4
                                     ",lvl:"
                                     enchLVL4
                                     "},{id:"
                                     enchID5
                                     ",lvl:"
                                     enchLVL5
                                     "}]}"
```

Full mod posted here: https://mod.learntomod.com/program_profiles/1088981

Link to similiar badge: https://mod.learntomod.com/badge_builders/3188?sequence=9084&position=8

/// How to run mods of multiple users in the new 1.9 servers?

Mr_Sonny

Essentially the solution is to have the server owner (**user A**) load another user's mods (**user B**), and either give them the mod book, or the command to run the mods he/she has loaded.

Here are the steps (After successfully joining the server):

Server joiner (**user B**) tells server owner (**user A**) where to find their mod by sharing the program_profile link (the url to view the program, which should be something like 'https://mod.learntomod.com/program_profiles/')

Server owner (**user A**) goes to the program_profile page, and clicks the '**edit**' button, which will copy said mod.

User A clicks the mod button to deploy to server

User A sees (mod loaded ...) the output in Minecraft server chat

User A gives the freshly loaded mod book to **user B**

NOTE: The 'me' block, will reference the server owner (user A), not server joiner (user B). The work around for this is to use the 'player named x' block rather than the 'me' block.

/// How to make something happen when I chat a specific message?

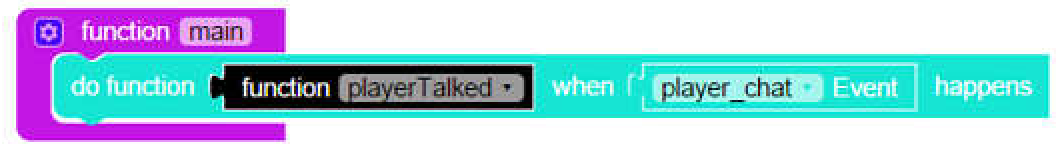
Mr_Sonny

First add an event inside of the main function. You need to add a function reference (to playerTalked) and a "**player_chat Event**". The function that you reference (**playerTalked**) needs to have a parameter (in this case **info**), this will allow the code to retrieve information from the event.

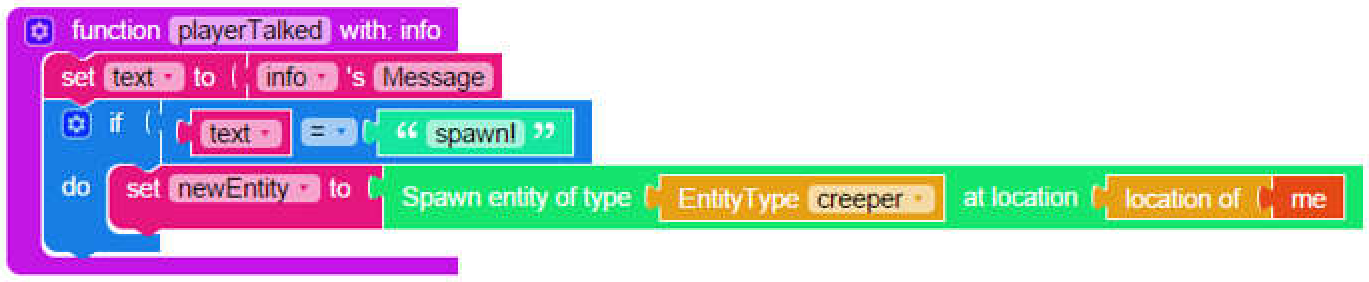
Inside of this new function you will need one variable: **text**. You need to set text to "**info's Message**"; which is the words/letters that the player chats.

Now that you have the necessary information, by using an if statement you can compare that to the desired message, and make something happen accordingly.

In this mod a creeper will spawn whenever the player chats "spawn!":



```
function main
  do function ( function playerTalked ) when ( player_chat Event ) happens
```



```
function playerTalked with: info
  set text to ( info 's Message )
  if ( text = "spawn!" )
  do set newEntity to ( Spawn entity of type ( EntityType creeper ) at location ( location of ( me ) ) )
```

Links to similar badges:

https://mod.learntomod.com/badge_builders/293?sequence=3619&position=209

https://mod.learntomod.com/badge_builders/300?sequence=3619&position=215

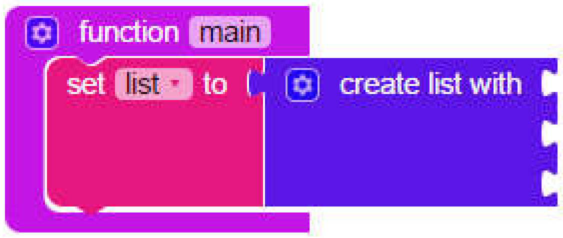
https://mod.learntomod.com/badge_builders/2702?sequence=825&position=2

https://mod.learntomod.com/badge_builders/2777?sequence=9013&position=2

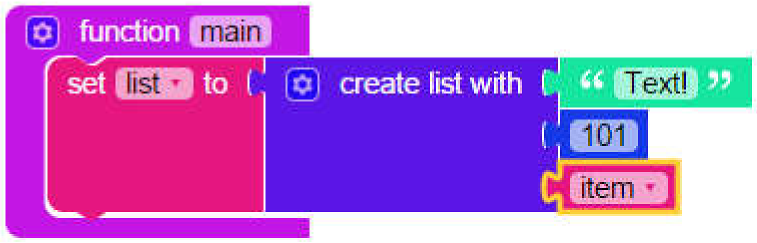
/// How to use list blocks?

IronGiant

First you need to grab a new variable block from the **Variables** tab and name it whatever you want. Then head over to the **Lists** tab and grab the second **Create Empty List** block you see. Plug it into your variable.

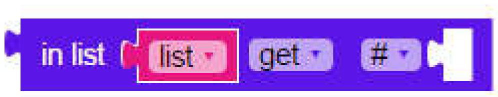


See those open slots? You can plug in whatever you want in there: text, numbers, variables, anything!



Sidebar: You can add more slots by clicking the gearbox to the left of the "create list with" text!

Now that you've got a list filed with useful things, let's learn how to use them! You can use anything that you've put in your list by going to the **Lists** tab and grabbing this block:



You can use this block to give any variable any value that you've stored in your list. After the "#" you can put what slot in the list you want to return. For instance, if we put in 1, we would fetch the first entry in the list, "Text!". You can use lists in conjunction with the **In List Get** block to store lots of data and access it quickly and efficiently!

/// How to make something happen when I break a specific block?

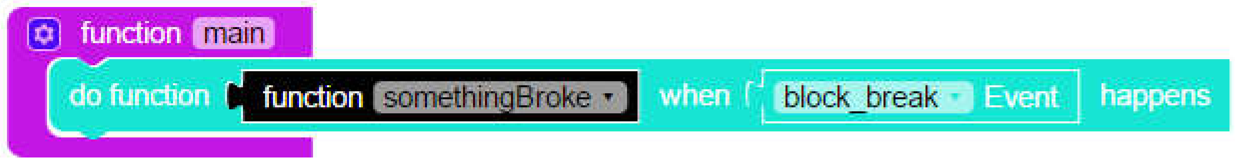
Mr_Sonny

First add an event inside of the main function. You need to add a function reference (to **somethingBroke**) and a "**block_break Event**". The function that you reference (**somethingBroke**) needs to have a parameter (in this case **info**), this will allow the code to retrieve information from the event.

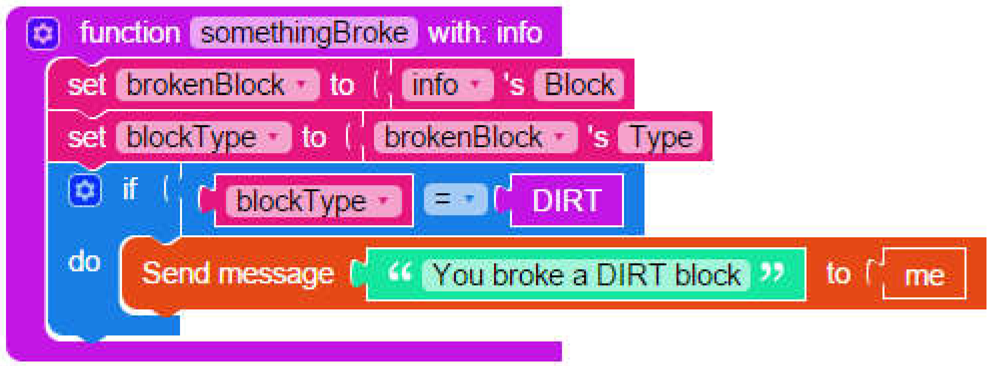
Inside of this new function you need two variables: one to get the block that was broken, and another one to get the type of the block (**brokenBlock, blockType**). You need to set brokenBlock to "**info's Block**", and blockType to "**brokenBlock's Type**".

Now that you have the block's type, by using an if statement you can compare that to a certain block type, and make something happen accordingly.

In this mod a simple message gets sent; "**You broke a DIRT block**"; whenever a dirt block gets broken:



```
function main
    do function ( function somethingBroke ▾ when ( block_break ▾ Event happens
```



```
function somethingBroke with: info
    set brokenBlock ▾ to ( info ▾ 's Block
    set blockType ▾ to ( brokenBlock ▾ 's Type
    if ( ( blockType ▾ = ▾ ( DIRT
    do  Send message ( " You broke a DIRT block " to ( me
```

To check if you broke a diamond block with a diamond pickaxe:

using the same blockBreak event, get the "Player", then get the "ItemInHand" of the player, then get the "Type" of the ItemInHand. Once you have that use an if statement and compare:

if (blockType = Diamond_Block AND ItemType = Diamond_Pickaxe)

NOTE: make sure you create variables and use the correct ones in the if statement. The code above is just an example.

Links to similiar badges:

https://mod.learntomod.com/badge_builders/292?sequence=3619&position=208

https://mod.learntomod.com/badge_builders/297?sequence=3619&position=212

https://mod.learntomod.com/badge_builders/299?sequence=3619&position=214

https://mod.learntomod.com/badge_builders/300?sequence=3619&position=215

///How to make something happen whenever I click with a specific item in hand?

Mr_Sonny

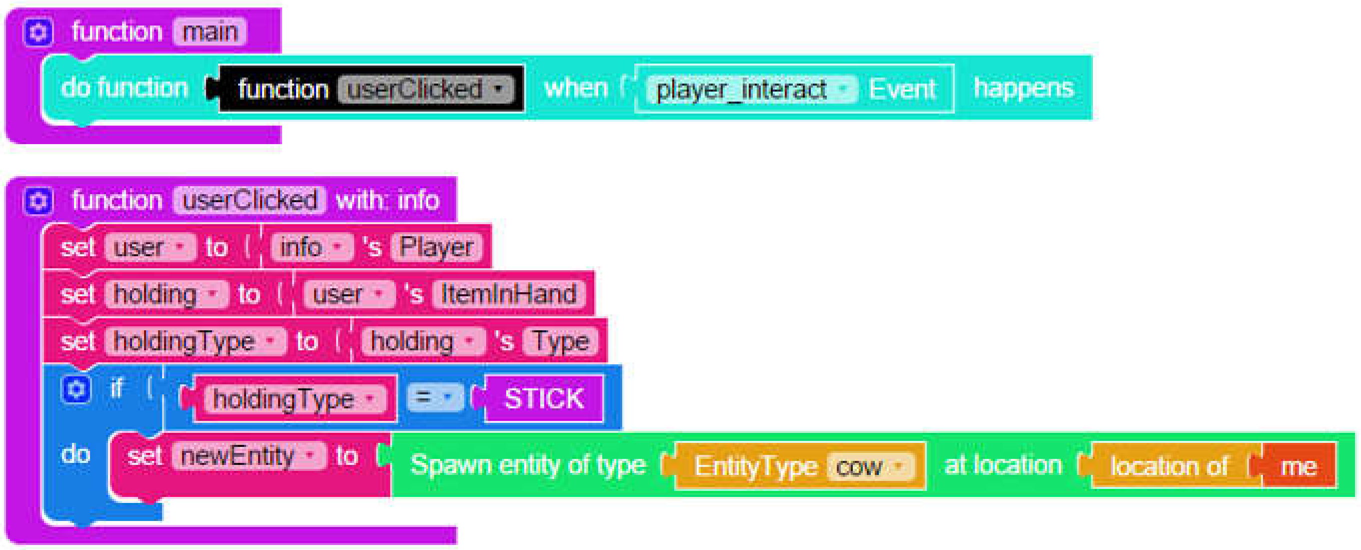
First add an event inside of the main function. You need to add a function reference (to

userClicked) and a "**player_interact Event**". The function that you reference (**userClicked**) needs to have a parameter (in this case **info**), this will allow the code to retrieve information from the event.

Inside of this new function you will need three variables: one for the player that clicked, one for the item the player is holding, and another one for the type of item (**user**, **holding**, and **holdingType**). You need to set user to "**info's Player**", holding to "**user's ItemInHand**", and holdingType to "**holding's Type**".

Now that you have the necessary information, by using an if statement you can compare that to the desired item/block, and make something happen accordingly.

In this mod a cow will spawn whenever the player clicks while holding a stick:



```
function main
    do function  function userClicked   when  player_interact Event  happens

function userClicked with: info
    set user to  info 's Player
    set holding to  user 's ItemInHand
    set holdingType to  holding 's Type
    if  holdingType = STICK
    do  set newEntity to  Spawn entity of type  EntityType cow  at location  location of  me
```

Links to similar badges:

https://mod.learntomod.com/badge_builders/2808?sequence=9049&position=1

https://mod.learntomod.com/badge_builders/2801?sequence=9018&position=11

https://mod.learntomod.com/badge_builders/3188?sequence=9084&position=8

///How to change the item/block drops of the block I break?

Mr_Sonny

First add an event inside of the main function. You need to add a function reference (to **somethingBroke**) and a "**block_break Event**". The function that you reference (**somethingBroke**) needs to have a parameter (in this case **info**), this will allow the code to retrieve information from the event.

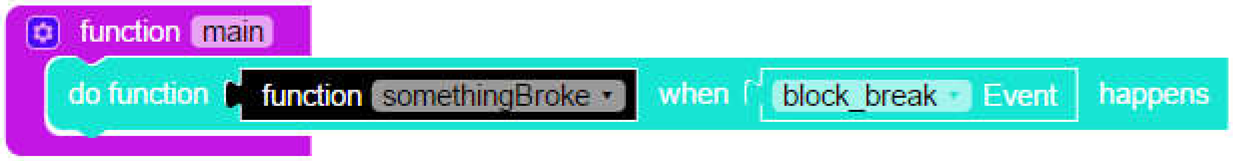
Inside of this new function you will need four variables: one to get the block that was broken, one to get the location, one to get the world, and the last one to hold the new itemstack (**brokenBlock**, **blockLoc, blockWorld**, and **newDrop**).

You need to set brokenBlock to “**info’s Block**”, blockLoc to “**brokenBlock’s Location**”, blockWorld to “**brokenBlock’s World**”, and newDrop to a new item stack containing 1 diamond.

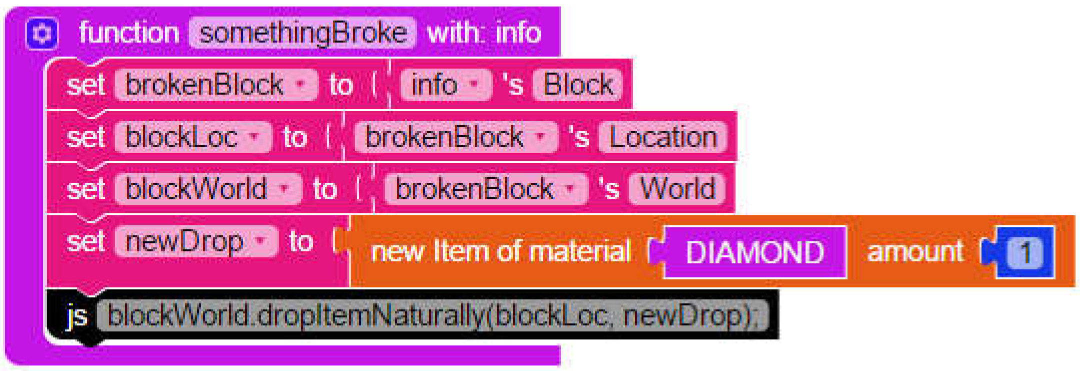
Now that you have the necessary infomartion you can add the following command in a js block:

blockWorld.dropItemNaturally(blockLoc, newDrop);

In this mod a diamond will drop whenever you break a block:

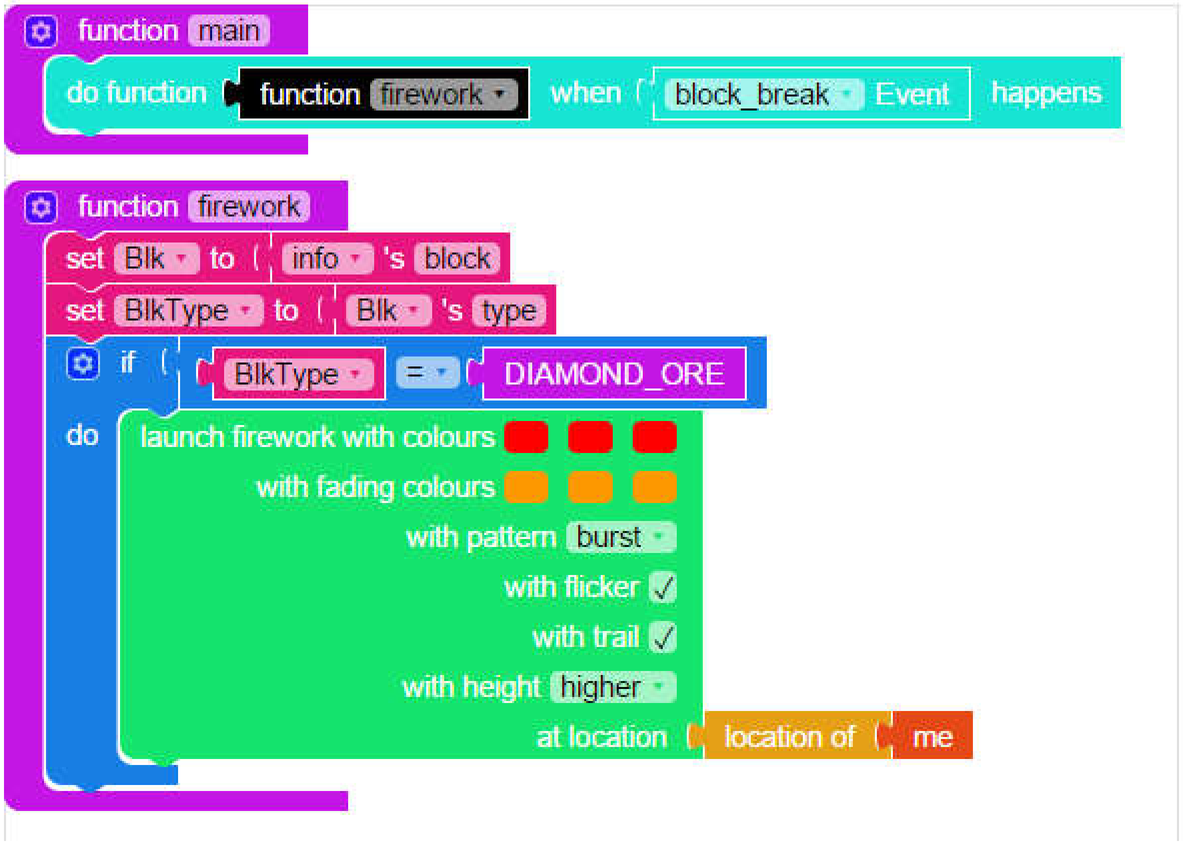


```
function main
    do function ( function somethingBroke ) when ( block_break Event ) happens
```



```
function somethingBroke with: info
    set brokenBlock to ( info 's Block )
    set blockLoc to ( brokenBlock 's Location )
    set blockWorld to ( brokenBlock 's World )
    set newDrop to ( new Item of material DIAMOND amount 1 )
    js blockWorld.dropItemNaturally(blockLoc, newDrop);
```

To incorporate checking block types, use an if statement:



```
function main
  do function ( function firework ) when ( block_break Event ) happens

function firework
  set Blk to ( info 's block
  set BlkType to ( Blk 's type
  if ( BlkType = DIAMOND_ORE
  do launch firework with colours 🟥 🟥 🟥
             with fading colours 🟧 🟧 🟧
                    with pattern burst
                      with flicker ✓
                        with trail ✓
                    with height higher
                     at location ( location of ( me
```

Link to similar badge: https://mod.learntomod.com/badge_builders/2830?sequence=9084&position=5

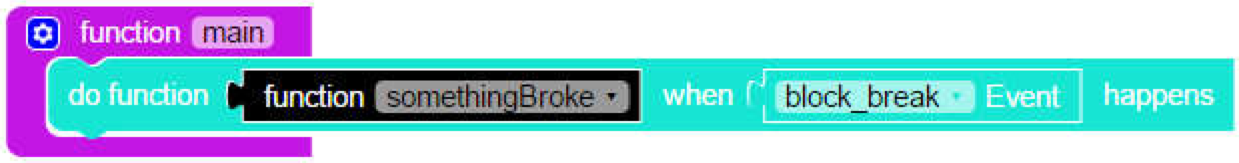
///How to make a block "unbreakable"?

Mr_Sonny

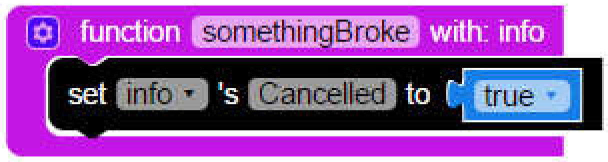
First add an event inside of the main function. You need to add a function reference (to **somethingBroke**) and a "**block_break Event**". The function that you reference (**somethingBroke**) needs to have a parameter (in this case **info**), this will allow the code to retrieve information from the event.

Inside of this new function you need pass a value to a variable. You need to simply set "**info's Cancelled**" to "**true**". This will cancel the event and replace the block with itself as soon as it breaks.

In this mod the block that you break will be replaced as soon as it destroyed, making it "unbreakable":



```
function main
  do function ( function somethingBroke ) when ( block_break Event ) happens
```



```
function somethingBroke with: info
  set info 's Cancelled to ( true )
```

///How to get the health percentage of an entity?

Mr_Sonny

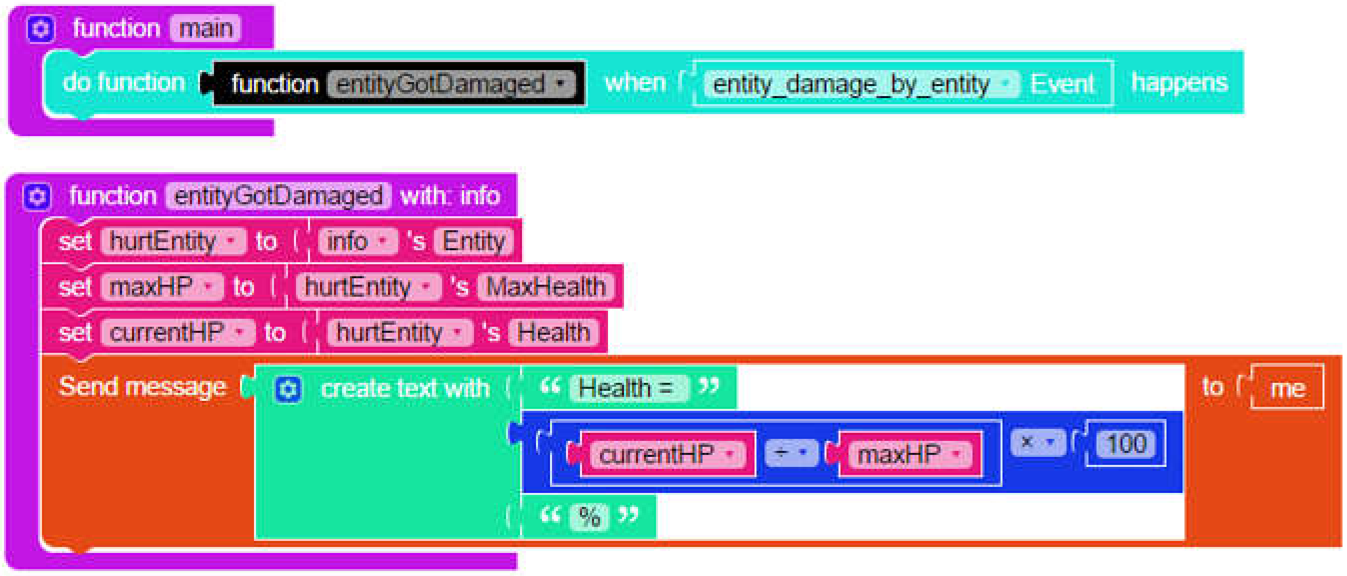
First add an event inside of the main function. You need to add a function reference (to **entityGotDamaged**) and a "**entity_damage_by_entity Event**". The function that you reference (**entityGotDamaged**) needs to have a parameter (in this case **info**), this will allow the code to retrieve information from the event.

Inside of this new function you will need three variables: one to get the entity that is being hurt, one to get the entity's maximum health, and another to get the entity's current health (**hurtEntity**, **maxHP**, and **currentHP**). You need to set hurtEntity to "**info's Entity**", maxHP to "**hurtEntity's MaxHealth**", and currentHP to "**hurtEntity's Health**".

To get the percentage first divide (currentHP / maxHP), then multiply that result by 100.

Now that you have the necessary information, you can simply send a message with the health percentage.

In this mod whenever you hurt an entity it will print out a message saying: "Health = ##%".
NOTE: (## will correspond to the result of the operation):



```
function main
  do function [function entityGotDamaged] when [entity_damage_by_entity Event] happens

function entityGotDamaged with: info
  set hurtEntity to info 's Entity
  set maxHP to hurtEntity 's MaxHealth
  set currentHP to hurtEntity 's Health
  Send message create text with "Health = "
                                ((currentHP ÷ maxHP) × 100)
                                "%"
    to me
```

Link to similar badge:

https://mod.learntomod.com/badge_builders/2791?sequence=9018&position=1

///How to apply texture packs to everybody in the server?

Mr_Sonny

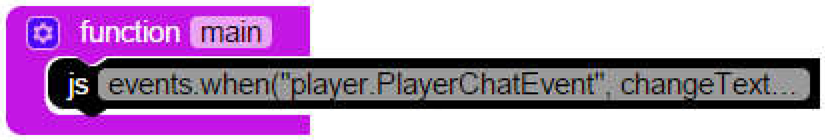
First create a main function, and add a player chat event, in this mod you want the event to get triggered whenever anybody on the server chats -- not just you -- so we need to do something different. Grab a js block and type in the following:

events.when("player.PlayerChatEvent", changeTexture, me, false);

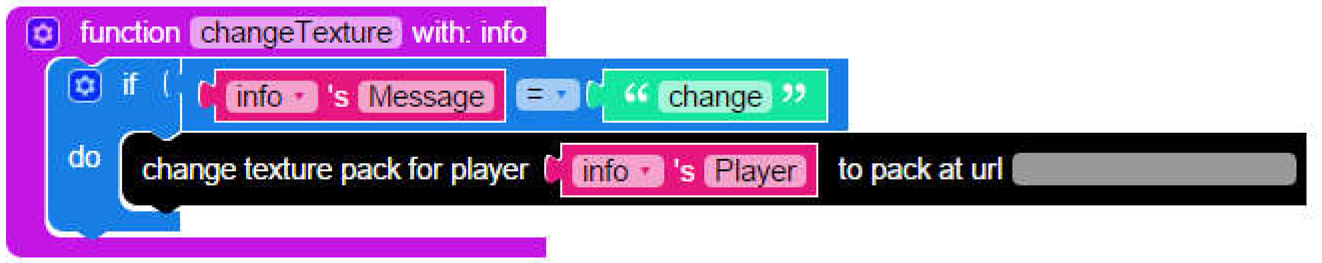
NOTE: "changeTexture" is the name of the function in this mod, you can change this according to your code, but the rest should remain the same.

Now you have to create the function (**changeTexture)** that's going to get triggered, whenever a player chats; make sure to add the info parameter to be able to retrieve information from the event. Then add the "change texture" block, instead of using the "me" block to change the texture add "**info's Player**", and then the "url" of the texture pack.

In this mod, whenever the players chat "**change"** their texture pack will change:



```
function main
  js events.when("player.PlayerChatEvent", changeText...
```



```
function changeTexture with: info
  if ( info 's Message = " change "
  do change texture pack for player info 's Player to pack at url
```

NOTE from IronGiant: If you want the texture to run immediately when the mod runs, without the player chat event

 put the "Change Texture Pack" block at the very beginning of your main function

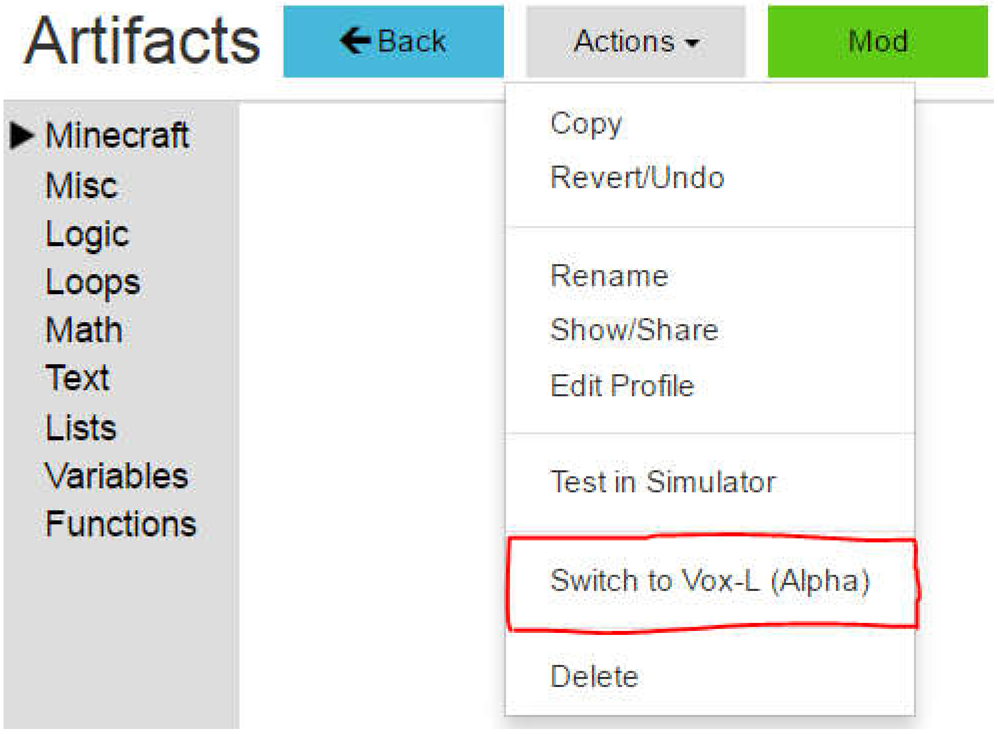
from coehlrich:

if you want to apply texture packs to everyone and the first player on the server runs the mod before anyone else joins the server world, then swap the player.PlayerChatEvent to player.PlayerJoinEvent and put the texture pack in the main function, too

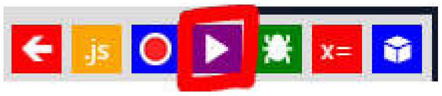
///How to create a Vox-L mod

carlosiscool

In order to create Vox-L mod, start by creating a **Blockly Multiplayer** mod. Once you have created a new mod, click on the **Actions** drop down menu and select the "**Switch to Vox-L (Alpha)**" option. This will load your current Vox-L world, and give you a new blockly menu with blocks that apply to Vox-L.



To run your mod, click on the purple play button on the bottom menu then press the "m" key, like in the old simulator.



///How to trigger a mod using a chat message in Vox-L?

Mr_Sonny

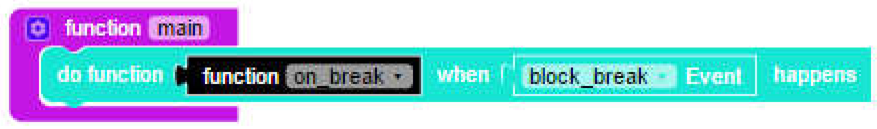
It is now possible to trigger a mod using a player chat event in Vox-L. It works similar like our server-side (for Minecraft) blocks. The only thing that changes is how to retrieve the information. Instead of using the "get block" (info's message), you will need to use a JS block and type in "info.message" and then compare that to your keywords.

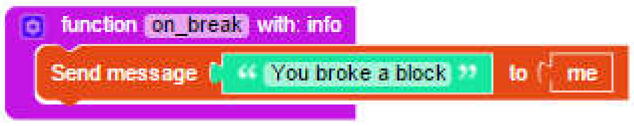
///How to make an event work for all the players in the server?

Mr_Sonny

If you're trying to make a mod that will work for anyone playing in your sever, you will need to have functions that are triggered by events (block break, player chat, player interact, etc.).

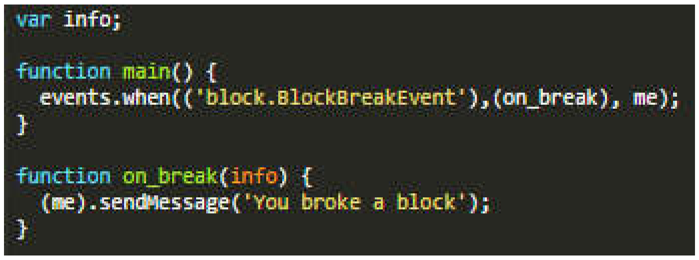
The most common way to use an event trigger is like this:


function main
do function function on_break when block_break Event happens


function on_break with: info
Send message “ You broke a block ” to me

This code will simply send a "You broke a block" message whenever you break a block. But if other players in the server break a block they won't receive it.

First we have to look at the JS by clicking the JS button:

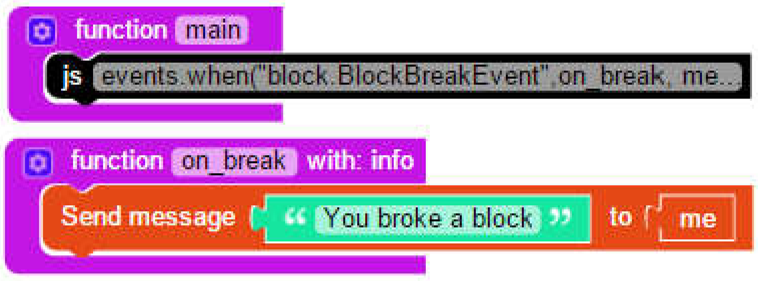


```
var info;

function main() {
  events.when(('block.BlockBreakEvent'),(on_break), me);
}

function on_break(info) {
  (me).sendMessage('You broke a block');
}
```

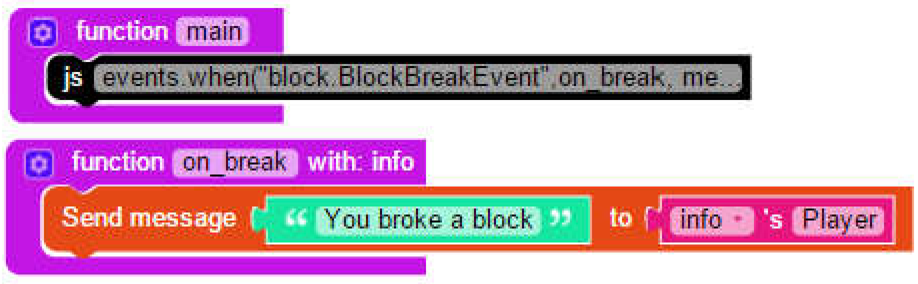
In order to make the event work for everyone we need to add an extra parameter at the end of the event trigger, it would look something like this:

events.when("block.BlockBreakEvent",on_break, me, false);

The false at the end indicates, that anyone in the server can trigger it. This line of code should be placed in a JS block, and it would replace the event block:


function main
js events.when("block.BlockBreakEvent",on_break, me...
function on_break with: info
Send message “ You broke a block ” to me

Now, we need to change one more thing. If we leave the code like that, every time a player breaks a block you (or the person who ran the mod) will receive the message; not the player who destroyed the block. So, instead of sending the message to "me" we have to send it to the "info's player"; meaning the person who destroyed the block:



```
function main
  js events.when("block.BlockBreakEvent",on_break, me...
function on_break with: info
  Send message "You broke a block" to info 's Player
```

To summarize: when you run the mod, anytime a player in the server breaks a block, they will get a message saying "You broke a block".

///How to pick up items from a larger radius

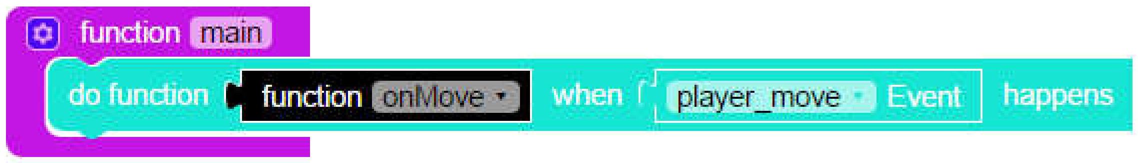
carlosiscool

Create a **player_move event** in the main function. This is to detect the items nearby anytime the player moves.

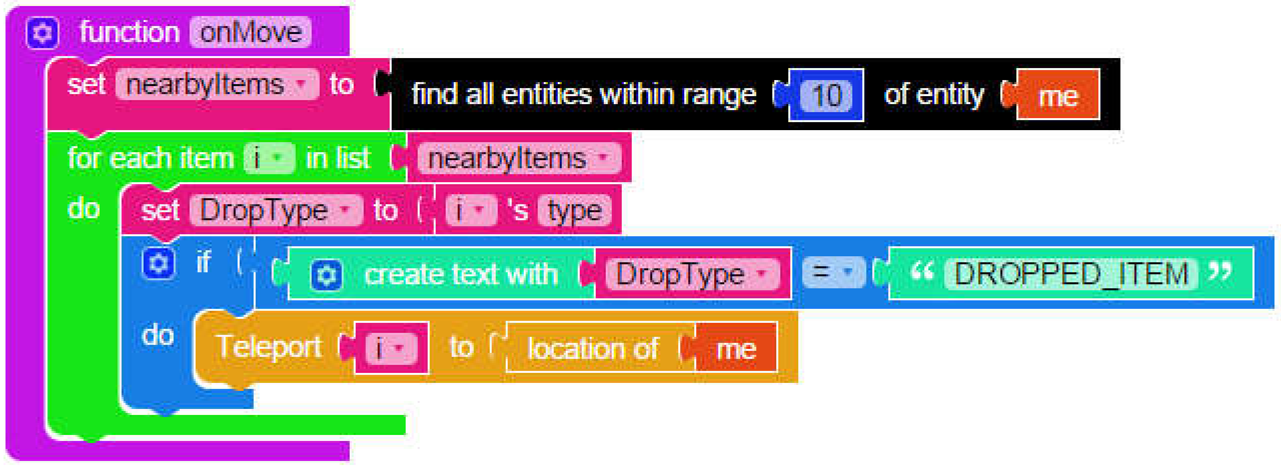
In the function being called by the player move event (**onMove**) create a variable (**nearbyEntities**) find all entities within range (**10**) of entity (**me**). This variable will create a list of the nearby entities within 10 blocks of the player each time the player moves. This list will include mobs, projectiles and dropped items.

Next you will need to iterate through each item in the list **nearbyEntities** using the "for each item i in list___" block. For each item in the list, you want to get its type by creating a variable inside the loop (**DropType**). After you get the entities type, create a condition that checks if the text created with **DropType** is the equal to "**DROPPED_ITEM**". This is done so only dropped items are picked up in the specified radius. Lastly, in the "do" part of the condition, teleport "i" to the location of the player.

Now the player should have a much larger item pick up radius.



```
function main
  do function  function onMove  when  player_move Event  happens
```



```
function onMove
  set nearbyItems to  find all entities within range 10 of entity me
  for each item i in list nearbyItems
  do  set DropType to  i 's type
      if  create text with DropType  =  " DROPPED_ITEM "
      do  Teleport i to  location of me
```

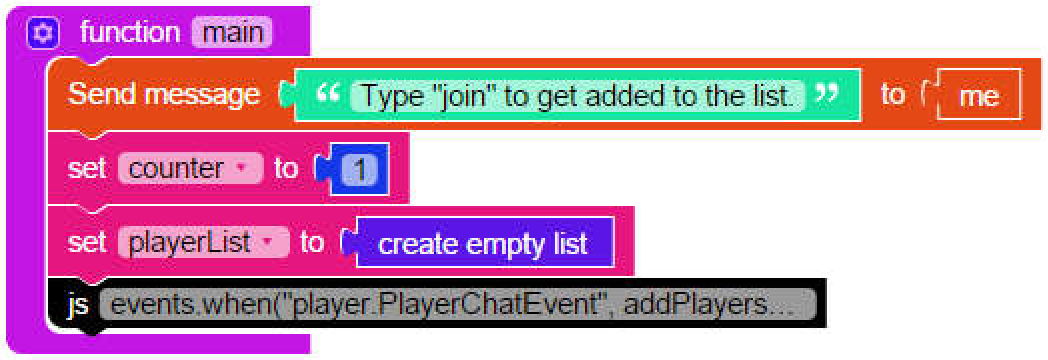
///How to create and add users to a player list?

Mr_Sonny

First create a main function, and then set two new variables: **counter**, will keep track of how many players have been added to the list , and **playerList** will be the list itself. After that you need to add a player chat event, in this mod you want the event to get triggered whenever anybody on the server chats -- not just you -- so we need to do something different. Grab a js block and type in the following:

events.when("player.PlayerChatEvent", addPlayersToList, me, false);

NOTE: "addPlayersToList" is the name of the function in this mod, you can change this according to your code, but the rest should remain the same

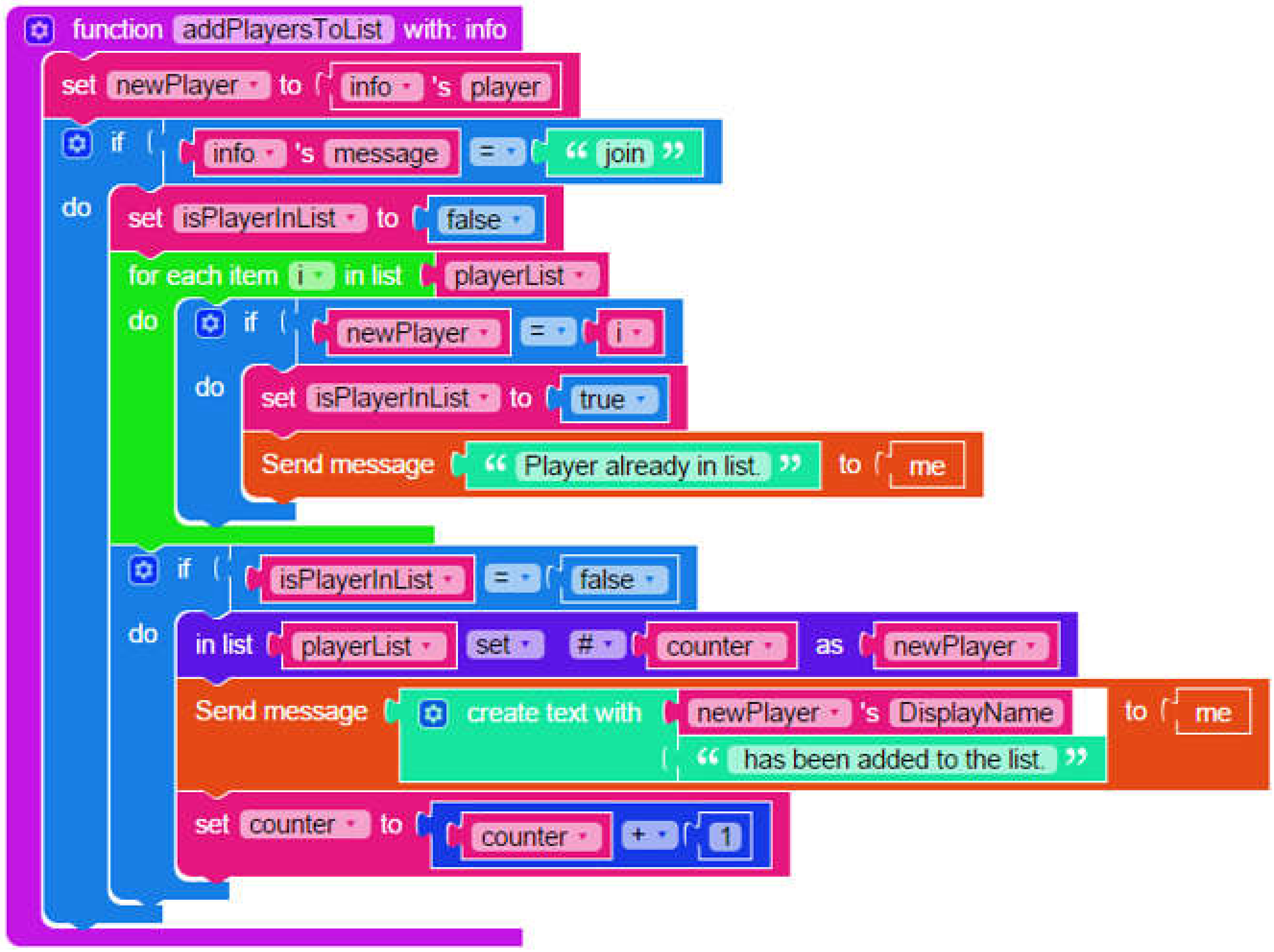


```
function main
  Send message  " Type "join" to get added to the list. "  to  me
  set counter to 1
  set playerList to  create empty list
  js  events.when("player.PlayerChatEvent", addPlayers...
```

Now you have to create the function (**addPlayersToList**) that's going to get triggered, whenever a player chats; make sure to add the **info** parameter to be able to retrieve information from the event. Inside of this function create a variable (**newPlayer**) that will hold the player that chatted.

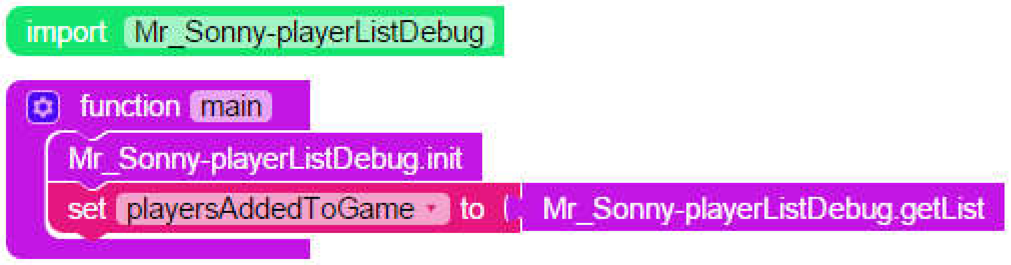
Then in an if statement check if **info's message** is equal to **join**; this "codeword" can change to whatever you want the players to say to get added to the list.

Next, create a boolean variable (**isPlayerInList**) that will change its value to true if it finds the player. To check that, loop through the player list and try to find the player that chatted. If the player is not in the list, then we can add that player to the list, and increment the number of players.



```
function addPlayersToList with: info
    set newPlayer to info 's player
    if info 's message = "join"
    do
        set isPlayerInList to false
        for each item i in list playerList
        do
            if newPlayer = i
            do
                set isPlayerInList to true
                Send message "Player already in list." to me
        if isPlayerInList = false
        do
            in list playerList set # counter as newPlayer
            Send message create text with newPlayer 's DisplayName
                                          " has been added to the list." to me
            set counter to counter + 1
```

If you want, you can add this exact code by importing the library **Mr_Sonny-playerListDebug**. You can also save the list into a variable by attaching the **getList** function:



```
import Mr_Sonny-playerListDebug

function main
    Mr_Sonny-playerListDebug.init
    set playersAddedToGame to Mr_Sonny-playerListDebug.getList
```

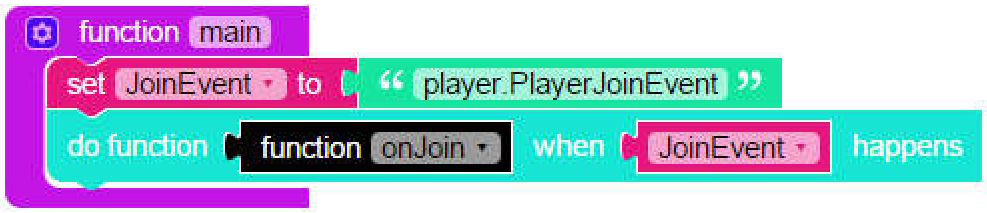
///How to check if a player joins your server

carlosiscool

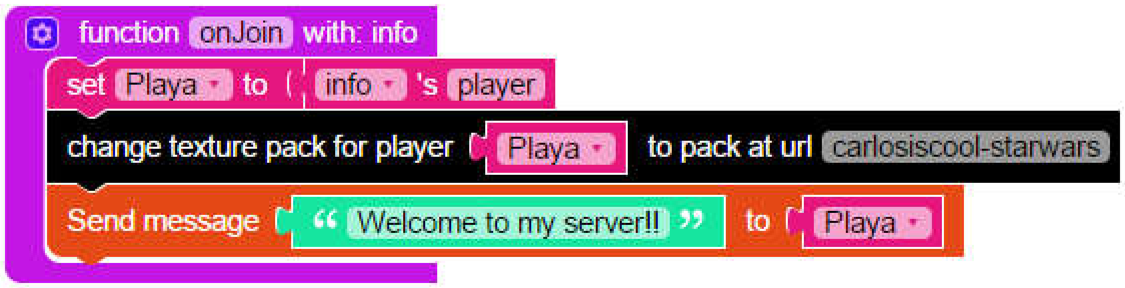
Create a new variable with the name of your event (JoinEvent) and set it to "player.PlayerJoinEvent" in a text block. Since this event doesn't exist in the event dropdown menu, a variable can be created with whichever new event you wish to add.

Create a function with the info parameter. In this function, which I named "onJoin create a variable (Playa) that gets the player from info. You can use this variable to make sure the player either sees a certain texture pack when they enter your server, send a "Welcome" message to them, or give certain items/effects. The variable Playa can be used in any occasion where the red "me" block would normally be used. The mod will continue to run until the events are cancelled even when all the players leave the server. If you leave your server and come back the next day you be sure that the Player Join event will be active.

In the mod bellow, I am loading a Star Wars texture pack for every person that comes into my server and sending them a "Welcome to my server" message



```
function main
  set JoinEvent to "player.PlayerJoinEvent"
  do function function onJoin when JoinEvent happens
```



```
function onJoin with: info
  set Playa to info 's player
  change texture pack for player Playa to pack at url carlosiscool-starwars
  Send message "Welcome to my server!!" to Playa
```

///How to debug entity damage by entity event?

Mr_Sonny

The following mod debugs the entity damage by entity event, meaning that it will print out useful information to help you fix any problems you might have in a mod.

entityHurt = The entity that was damaged

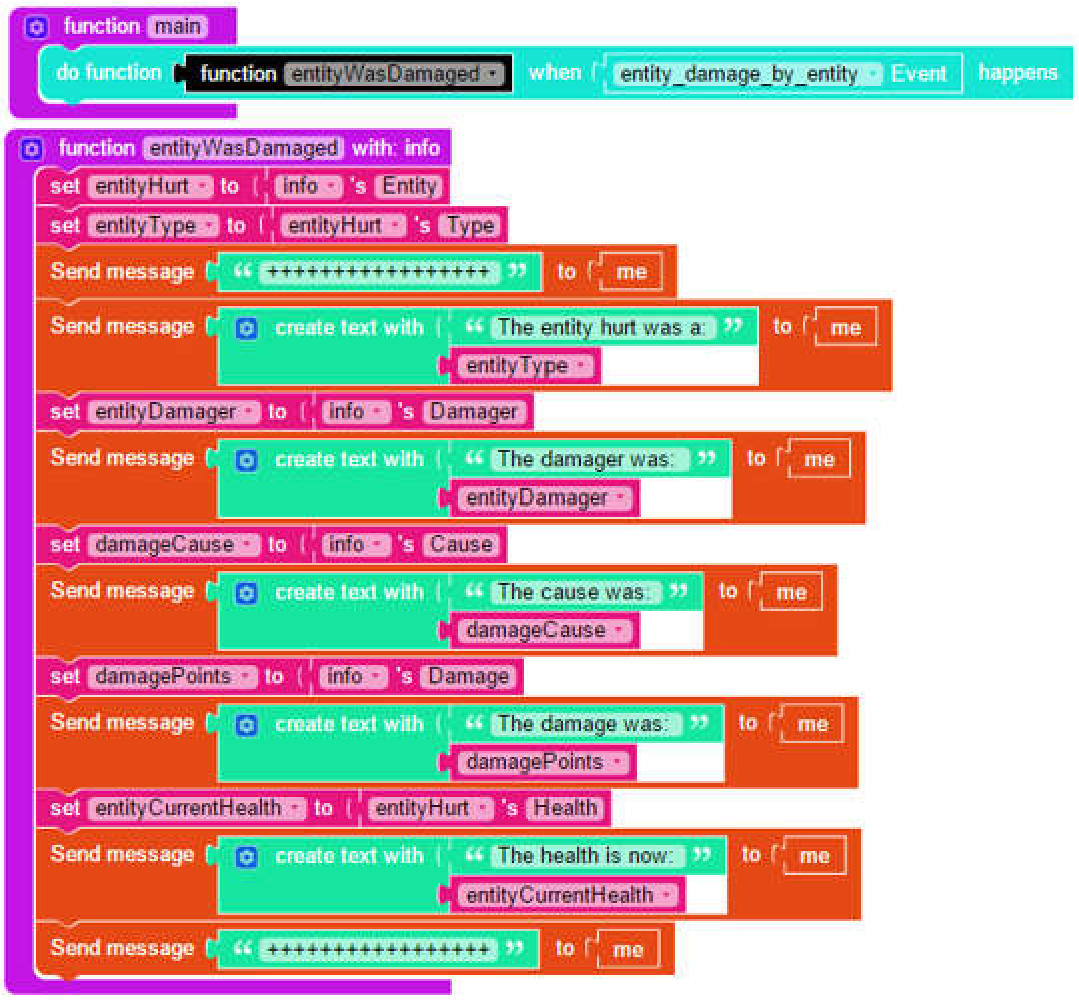
entityType = The type of entity damaged

entityDamager = The entity/item that caused the damaged

damageCause = The reason for the damage

damagePoints = The number of HP inflicted to the entity

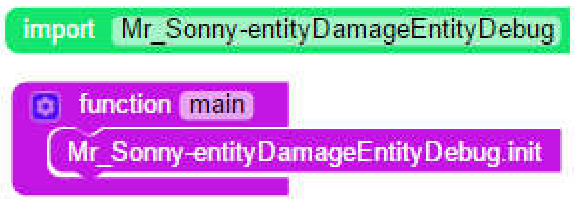
entityCurrentHealth = The current HP of the entity



```
function main
  do function ( function entityWasDamaged ) when ( entity_damage_by_entity Event ) happens

function entityWasDamaged with: info
  set entityHurt to ( info 's Entity )
  set entityType to ( entityHurt 's Type )
  Send message ( " +++++++++++++++++ " ) to ( me )
  Send message ( create text with ( " The entity hurt was a: " ) ( entityType ) ) to ( me )
  set entityDamager to ( info 's Damager )
  Send message ( create text with ( " The damager was: " ) ( entityDamager ) ) to ( me )
  set damageCause to ( info 's Cause )
  Send message ( create text with ( " The cause was: " ) ( damageCause ) ) to ( me )
  set damagePoints to ( info 's Damage )
  Send message ( create text with ( " The damage was: " ) ( damagePoints ) ) to ( me )
  set entityCurrentHealth to ( entityHurt 's Health )
  Send message ( create text with ( " The health is now: " ) ( entityCurrentHealth ) ) to ( me )
  Send message ( " +++++++++++++++++ " ) to ( me )
```

NOTE: There is more information you can retrieve from this event, these are just the most commonly used.

If you want to test it in your mod, simply import "Mr_Sonny-entityDamageEntityDebug" and add the call in main:



```
import Mr_Sonny-entityDamageEntityDebug

function main
  Mr_Sonny-entityDamageEntityDebug.init
```

Check out the mod here: http://mod.learntomod.com/program_profiles/1081105

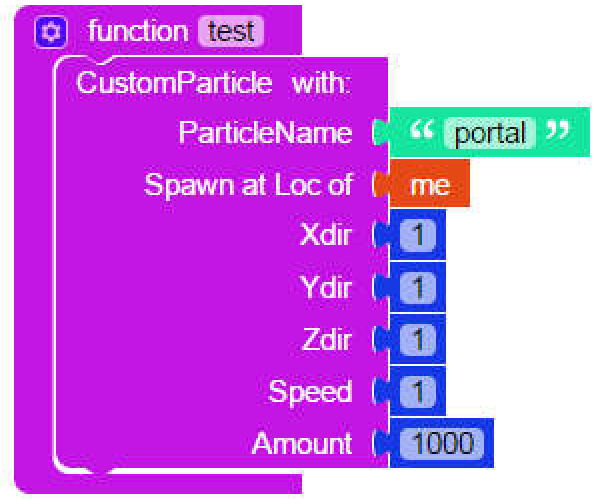
///How to turn commands into an easy to use library?

carlosiscool

How to turn this:

/particle ~ ~ ~ 1 1 1 1 1000

into this:



Commands can be very useful, but often times we forget what inputs it takes and where those are located. By turning a command into an exportable function, the editable commands are labeled and easy to work with.

Find the command you wish to turn into a library. I will be using the **Particle Command**. You can find different commands in the following site: https://www.digminecraft.com/game_commands/index.php2

The particle command breaks down as such:

/particle [name] [x] [y] [z] [xd] [yd] [zd] [speed] [count] [mode]

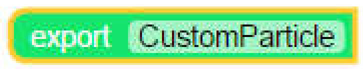
name - The name of the particle. All particle names can be found here

x, y and z - Where the particle will spawn

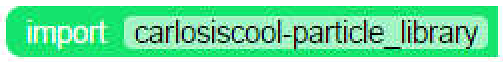
xd, yd and zd - The x, y and z radius in which you want the particles to be limited

speed - How fast the particles will move within the specified radius. This can be left as 0 for motionless particles.

mode - This can either be normal or force. Force will allow all player to see the particles when they are spawned.

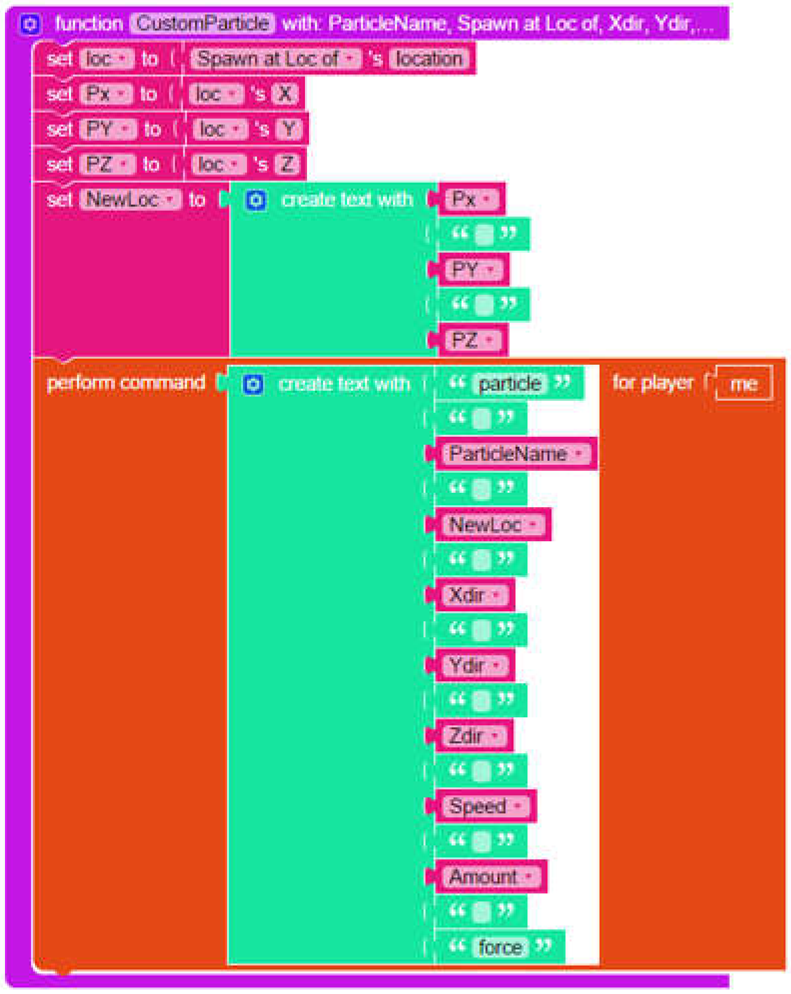


In your whichever mod you wish to use your command library, simply import the function with an "Import" block with the LearnToMod nickname and the name of the mod. You should get a new function in your functions tab that runs your command with whatever inputs you give it!



```
import carlosiscool-particle_library
```

This is how my Custom Particle command function looks when finished:



```
function CustomParticle with: ParticleName, Spawn at Loc of, Xdir, Ydir,...
  set loc to Spawn at Loc of 's location
  set Px to loc 's X
  set PY to loc 's Y
  set PZ to loc 's Z
  set NewLoc to create text with Px
                                 " "
                                 PY
                                 " "
                                 PZ
  perform command create text with "particle"
                                   " "
                                   ParticleName
                                   " "
                                   NewLoc
                                   " "
                                   Xdir
                                   " "
                                   Ydir
                                   " "
                                   Zdir
                                   " "
                                   Speed
                                   " "
                                   Amount
                                   " "
                                   "force"
                  for player me
```

Turn each input into a parameter for the function that will be exported. In this case, I named my function "Custom Particle". In the same function I can define each parameter using variables if needed. In this case, the Parameter "Spawn at location of" was created to allow the user to change the location from which the x, y and z coordinates are acquired from.

Inside a "Perform Command" block place all of the variables for command inputs in a "create text with" block. Be sure to separate each variable by a text block with a space inside of it, otherwise the text generated will be one long spaceless string that Minecraft won't understand. After the command is complete, export your function using the "export" block

///How to debug player interact event?

Mr_Sonny

The following mod **debugs** the **player interact event**, meaning that it will print out useful information to help you fix any problems you might have in a mod.

itemInteracted = The item that the player interacted with

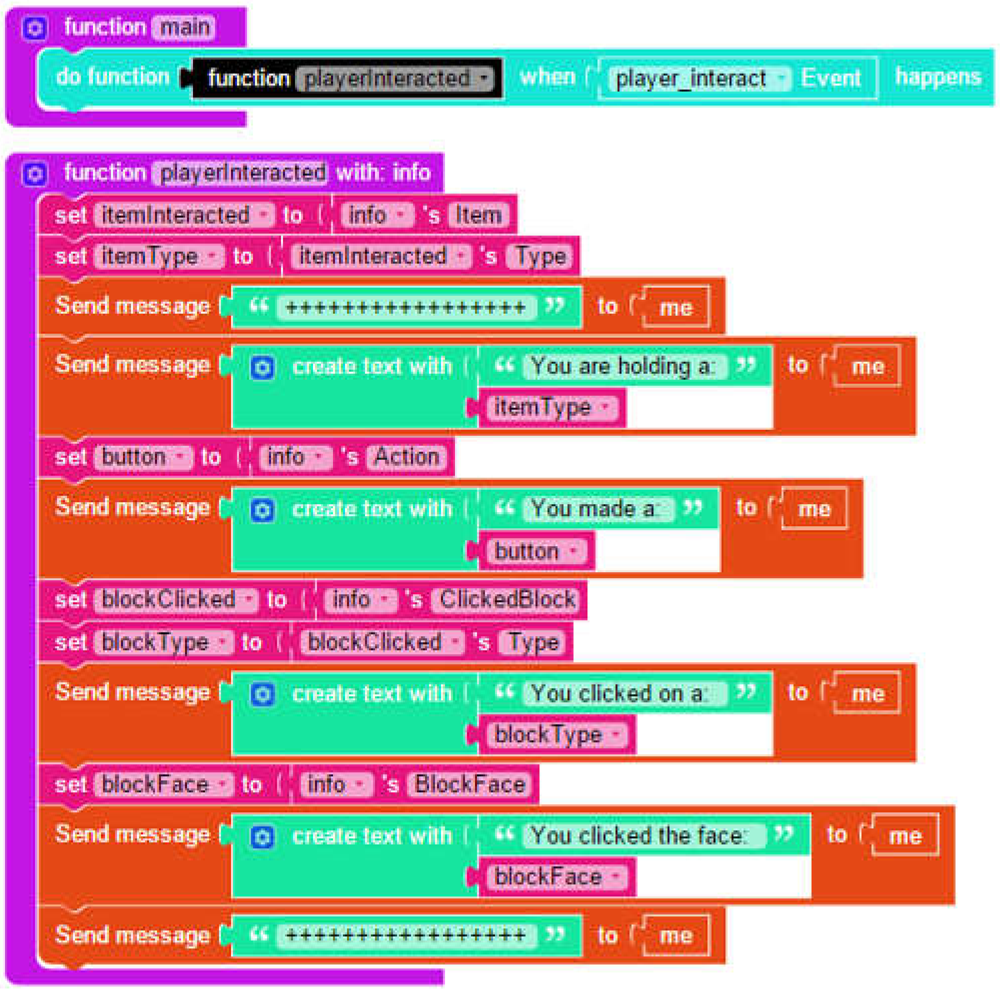
itemType = The type of block/item in the player's hand

button = The button and type of block the player clicked (left or right, block or air)

blockClicked = The block the player clicked on

blockType = The type of block the player clicked on (if it was in the air, null will be returned)

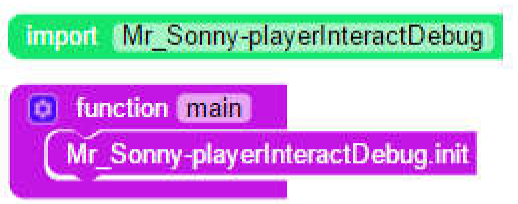
blockFace = The face of the block the player clicked on (if it was air, nothing will print)



```
function main
    do function ( function playerInteracted ) when ( player_interact Event ) happens

function playerInteracted with: info
    set itemInteracted to ( info 's Item )
    set itemType to ( itemInteracted 's Type )
    Send message ( " +++++++++++++++++ " ) to ( me )
    Send message ( create text with ( " You are holding a: " , itemType ) ) to ( me )
    set button to ( info 's Action )
    Send message ( create text with ( " You made a: " , button ) ) to ( me )
    set blockClicked to ( info 's ClickedBlock )
    set blockType to ( blockClicked 's Type )
    Send message ( create text with ( " You clicked on a: " , blockType ) ) to ( me )
    set blockFace to ( info 's BlockFace )
    Send message ( create text with ( " You clicked the face: " , blockFace ) ) to ( me )
    Send message ( " +++++++++++++++++ " ) to ( me )
```

NOTE: There is more information you can retrieve from this event, these are just the most commonly used.

If you want to test it in your mod, simply import "Mr_Sonny-playerInteractDebug" and add the call in main:



```
import Mr_Sonny-playerInteractDebug

function main
    Mr_Sonny-playerInteractDebug.init
```

Check out the mod here: https://mod.learntomod.com/program_profiles/1080974

///How to debug block break event?

Mr_Sonny

The following mod **debugs** the **block break event**, meaning that it will print out useful information to help you fix any problems you might have in a mod.

blockBroken = The block that was broken

blockType = The type of block that was broken

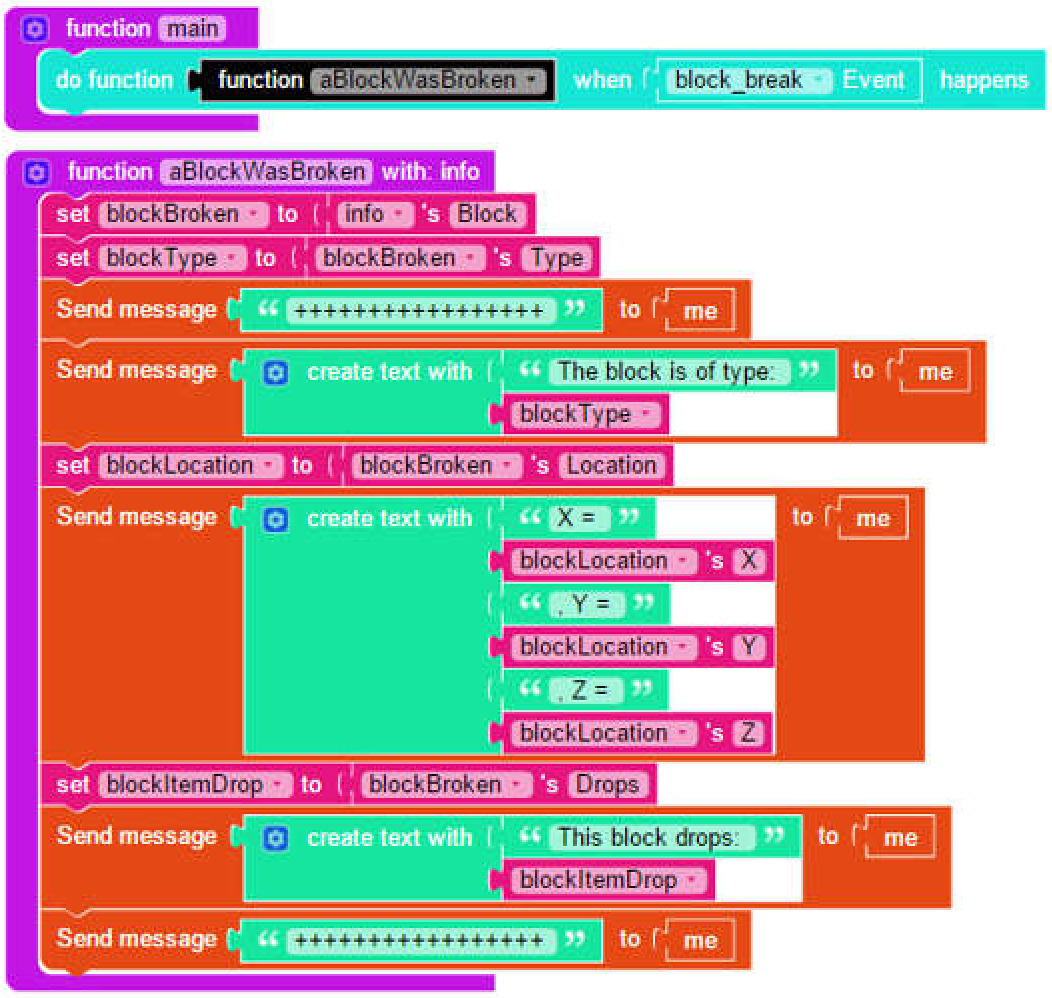
blockLocation = The X, Y, and Z coordinates of the broken block

blockLocation's X = The X coordinate

blockLocation's Y = The Y coordinate

blockLocation's Z = The Z coordinate

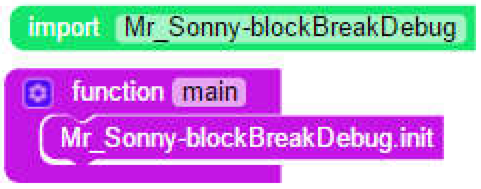
blockItemDrop = The item(s) the block drops when broken, if any



```
function main
  do function  function aBlockWasBroken  when  block_break Event  happens

function aBlockWasBroken with: info
  set blockBroken to  info 's Block
  set blockType to  blockBroken 's Type
  Send message  " +++++++++++++++++ "  to  me
  Send message  create text with  " The block is of type: "
                                   blockType
                to  me
  set blockLocation to  blockBroken 's Location
  Send message  create text with  " X = "
                                   blockLocation 's X
                                   " , Y = "
                                   blockLocation 's Y
                                   " , Z = "
                                   blockLocation 's Z
                to  me
  set blockItemDrop to  blockBroken 's Drops
  Send message  create text with  " This block drops: "
                                   blockItemDrop
                to  me
  Send message  " +++++++++++++++++ "  to  me
```

NOTE: There is more information you can retrieve from this event, these are just the most commonly used.

If you want to test it in your mod, simply import "Mr_Sonny-blockBreakDebug" and add the call in main:



```
import  Mr_Sonny-blockBreakDebug

function main
  Mr_Sonny-blockBreakDebug.init
```

Check out the mod here: https://mod.learntomod.com/program_profiles/1080971

///How to check the type of item/block the player picks up?

Mr_Sonny

First you need to create a variable (**newEvent**) and set it to "**player.PlayerPickupItemEvent**". You need to do this because in the dropdown menu there is not an "player_pickup_item Event" option. There are hundreds of different events that are not in the dropdown list, and this is how you would add them to your code.

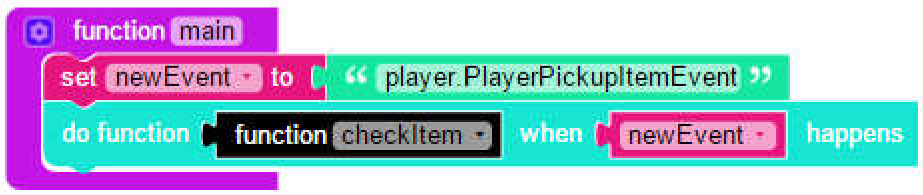
Then add an event inside of the main function. You need to add a function reference (to checkItem) and add the variables “newEvent” to the second slot. The function that you reference (checkItem) needs to have a parameter (in this case info), this will allow the code to retrieve information from the event.

Inside of this new function you will need three variables: one for getting the item that was picked up, one to get the itemstack of that item/block, and another one to get the type of item/block that the player picked up (pickedItem, pickedItemStack, and pickedType).

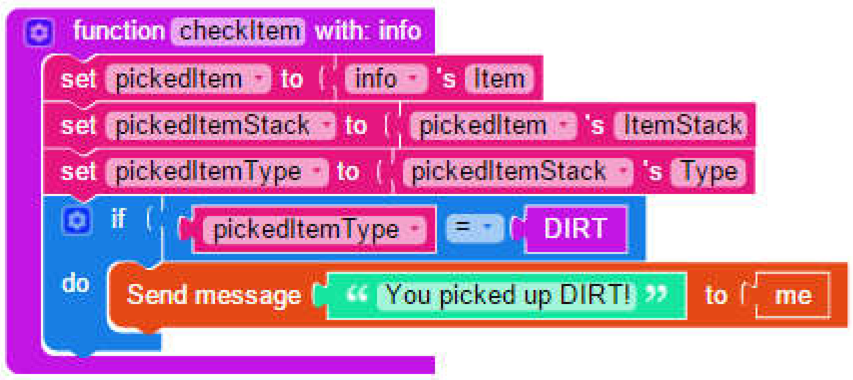
You have to set pickedItem to “info’s Item", then set pickedItemStack to "pickedItem's ItemStack", and finally set pickedType to "pickedItemStack's Type".

Now that you have the necessary information, by using an if statement you can compare that to the desired item/block, and if this is true make something happen.

In this mod whenever the player picks up a Dirt block they will get a message that says "You picked up DIRT!".



```
function main
  set newEvent to “ player.PlayerPickupItemEvent ”
  do function  function checkItem  when  newEvent  happens
```



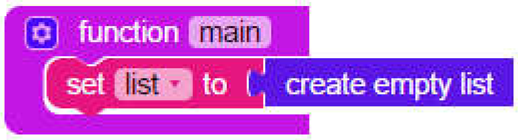
```
function checkItem with: info
  set pickedItem to  info 's Item
  set pickedItemStack to  pickedItem 's ItemStack
  set pickedItemType to  pickedItemStack 's Type
  if  pickedItemType = DIRT
  do  Send message “ You picked up DIRT! ” to  me
```

link to similar badge: http://mods.learntomod.com/badge_builders/2798?sequence=9018&position=8

///How to add to a list?

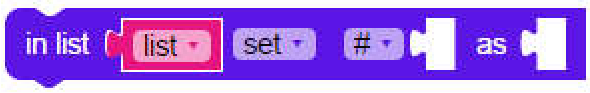
IronGiant

First you need to grab a new variable block from the Variables tab and name it whatever you want. Then head over to the Lists tab and grab the first Create Empty List block you see. Plug it into your variable.



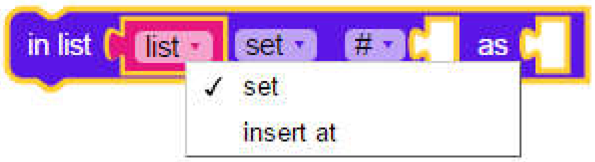
Congratulations! You've made a list. It's an empty list, of course, so it doesn't do anything at the moment. Let's add something to it!

There are several ways to add to a list once you've created it. The block that you'll need to do this is in the Lists tab and looks like this:

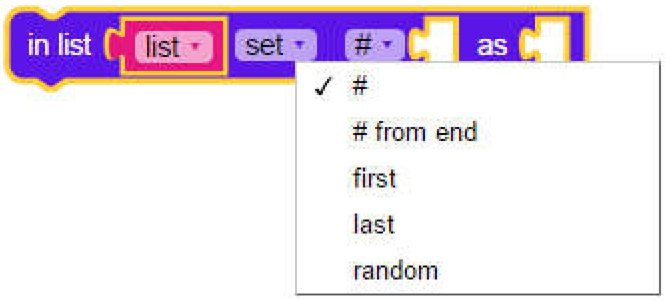


Inside the block, set the "list" variable to whatever you named your variable at the beginning. This makes sure you're adding whatever you want to add to the right list.

Now let's go through the different options you have with this block. You can choose to either "set" or "insert at":



"#", "# from last", "first", "last", or "random":



"set" means that you'll replace anything that's already in that slot in the list. For example, if I had a list that had the sentence "This is a sentence!" in the first slot and I then set the first slot to the number 101, then "This is a sentence!" would disappear and be replaced with 101.

"insert at" means that you'll push back everything in the list and add whatever you're inserting before that. For example, if I had the variable "var" at the 2nd slot of a list and decided to insert the word "Word" there, "Word" would be at the 2nd slot of the list and "var" would now be at the 3rd.

"#" is what slot you want to add at. If your list is empty, you always want to start at 1.

"# from last" is as many slots backwards from the end as you specify. For example, if I had a list that had 5 slots and I set a text block at 2 slots from the last, it would replace the 4th slot.

"first" is just the first entry in the list.

"last" is the last entry in the list.

"random" is, well, a random entry.

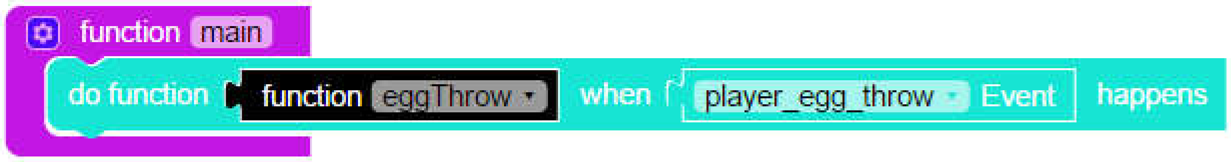
And that's all you need to know. Good luck listing!

///How to make the egg that I throw always hatch?

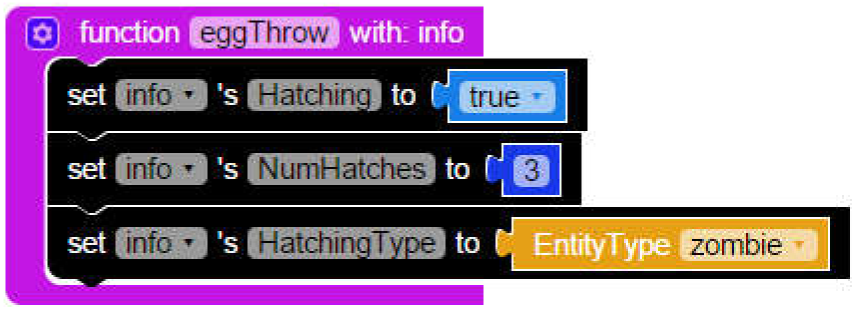
First add an event inside of the main function. You need to add a function reference (to **eggThrow**) and a "**player_egg_throw Event**". The function that you reference (**eggThrow**) needs to have a parameter (in this case **info**), this will allow the code to retrieve information from the event.

Inside of this new function you have to set three variables. First set "**info's Hatching**" to "**true**", then set "**info's NumHatches**" to "**3**", and finally set "**info's HatchingType**" to "**EntityType zombie**".

In this mod whenever you throw an egg it will always hatch three zombies instead of a chicken:



```
function main
    do function ( function eggThrow ) when ( player_egg_throw Event ) happens
```



```
function eggThrow with: info
    set info 's Hatching to true
    set info 's NumHatches to 3
    set info 's HatchingType to EntityType zombie
```

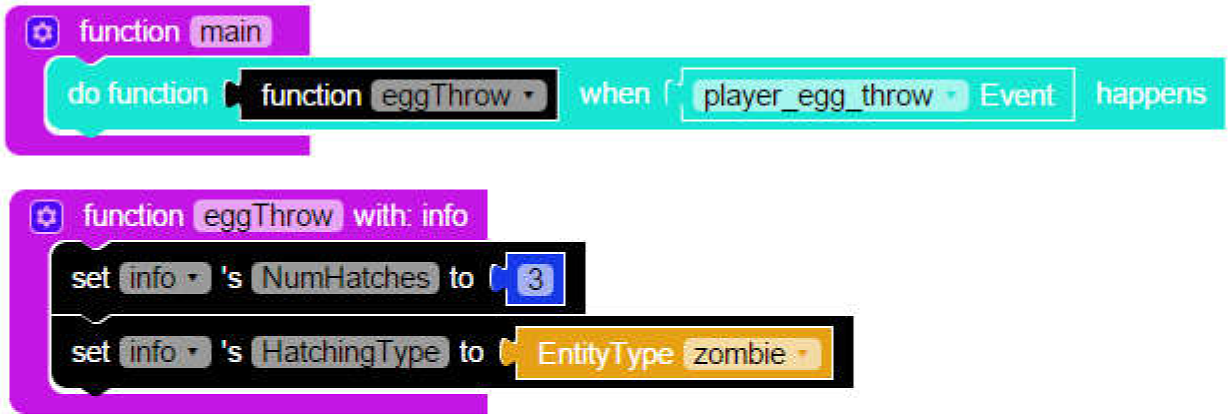
///How to hatch more than one entity from an egg that I throw?

Mr_Sonny

First add an event inside of the main function. You need to add a function reference (to eggThrow) and a "player_egg_throw Event". The function that you reference (eggThrow) needs to have a parameter (in this case info), this will allow the code to retrieve information from the event.

Inside of this new function you have to set two variables, one will set the number of entities that should hatch and the other one will change the entity that will hatch. First set “info’s NumHatches” to “3” and then set “info’s HatchingType” to “EntityType zombie”.

In this mod whenever you throw an egg three zombies will hatch instead of a chicken (there is still a chance the egg doesn’t hatch anything):



```
function main
    do function  function eggThrow  when  player_egg_throw Event  happens

function eggThrow with: info
    set info 's NumHatches to 3
    set info 's HatchingType to EntityType zombie
```

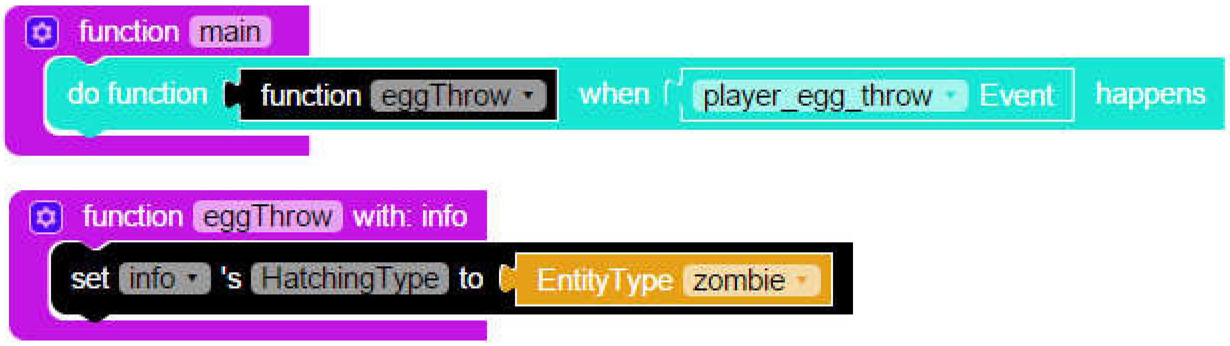
///How to hatch a different entity from an egg that I throw?

Mr_Sonny

First add an event inside of the main function. You need to add a function reference (to eggThrow) and a “player_egg_throw Event”. The function that you reference (eggThrow) needs to have a parameter (in this case info), this will allow the code to retrieve information from the event.

Inside of this new function you simply have to set “info’s HatchingType” to “EntityType zombie”.

In this mod whenever you throw an egg the hatching entity will be a zombie instead of a chicken (there is still a chance the egg doesn’t hatch anything):



```
function main
    do function  function eggThrow  when  player_egg_throw Event  happens

function eggThrow with: info
    set info 's HatchingType to EntityType zombie
```

///How to make something happen at the location of the block I break?

Mr_Sonny

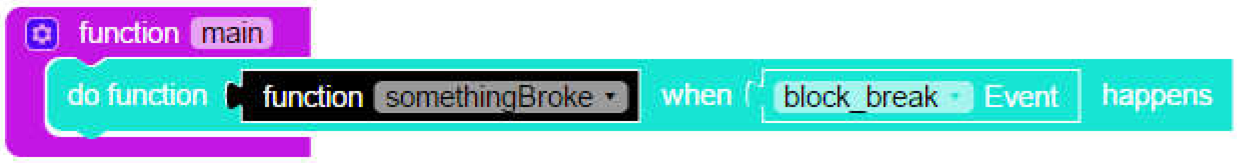
First add an event inside of the main function. You need to add a function reference (to

somethingBroke) and a “**block_break Event**”. The function that you reference (**somethingBroke**) needs to have a parameter (in this case **info**), this will allow the code to retrieve information from the event.

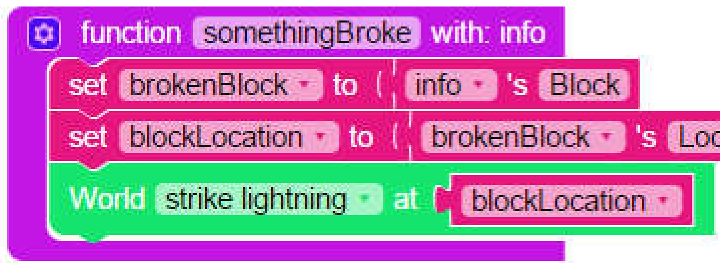
Inside of this new function you need two variables: one to get the block that was broken, and another one to get the location of the block (**brokenBlock**, **blockLocation**). You need to set brokenBlock to “**info’s Block**”, and blockLocation to “**brokenBlock’s Location**”.

Now that you have the block’s location, you can use it to spawn entities.

In this mod lightning strikes at the location of a broken block:



```
function main
  do function ( function somethingBroke ) when ( block_break Event ) happens
```



```
function somethingBroke with: info
  set brokenBlock to ( info 's Block )
  set blockLocation to ( brokenBlock 's Location )
  World strike lightning at ( blockLocation )
```

///Loading texture packs

With Learn to mod's texture packs you can't download to your laptop in all of one chunk (at least not that I know of) you have to download each image one by one and then you need a copy of minecraft's texture folder get rid of the specific images you made replace them with the Learn to mod's image. Then, place that copy in the .minecraft folder resource-pack folder then go to minecraft (In snapshot 17w15a) go into resource packs there the resource-pack will be on the opposite side of the Active resource packs. click the resource pack it will move then click done it will load and boom, you will have it. (This also works with regular non Learn to mod resource-packs only without the whole download each image and get a copy of minecraft's texture folder and getting rid of specific images and so on till placing the resource-pack into the .minecraft resource pack folder.)

Also the image names have to rename the same - you can't rename them.

///How to get the item's display name?

The type and display names are two different data in an item and can never be compared. The display name is stored in another big data called "ItemMeta" while the type and this "ItemMeta" are stored directly under the item stack.

Set "player" to "info" 's player

Set "item" to "player" 's ItemInHand

Set "type" to "item" 's type

Set "ItemMeta" to "item" 's ItemMeta

Set "Name" to "ItemMeta" 's DisplayName

///I want to create an enchanted item using the .addEnchantment() function of the ItemStack class, not by using the /give command.

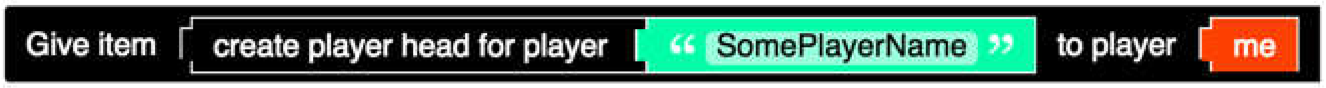
axe.addEnchantment(org.bukkit.enchantments.Enchantment.DAMAGE_ALL, 5);

Check for air mod: https://mod.learntomod.com/program_profiles/1700095

/// How to make your own player head

Make a multiplayer mod that gives you a player skull.

Just put this line of code (You can find the blocks in the Items, Player, and Text categories) into your main function, and run the mod.



Give item create player head for player " SomePlayerName " to player me