

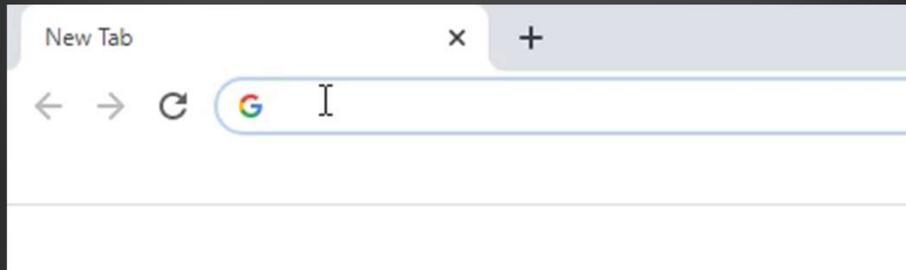


Today we'll be creating the elements for a game.



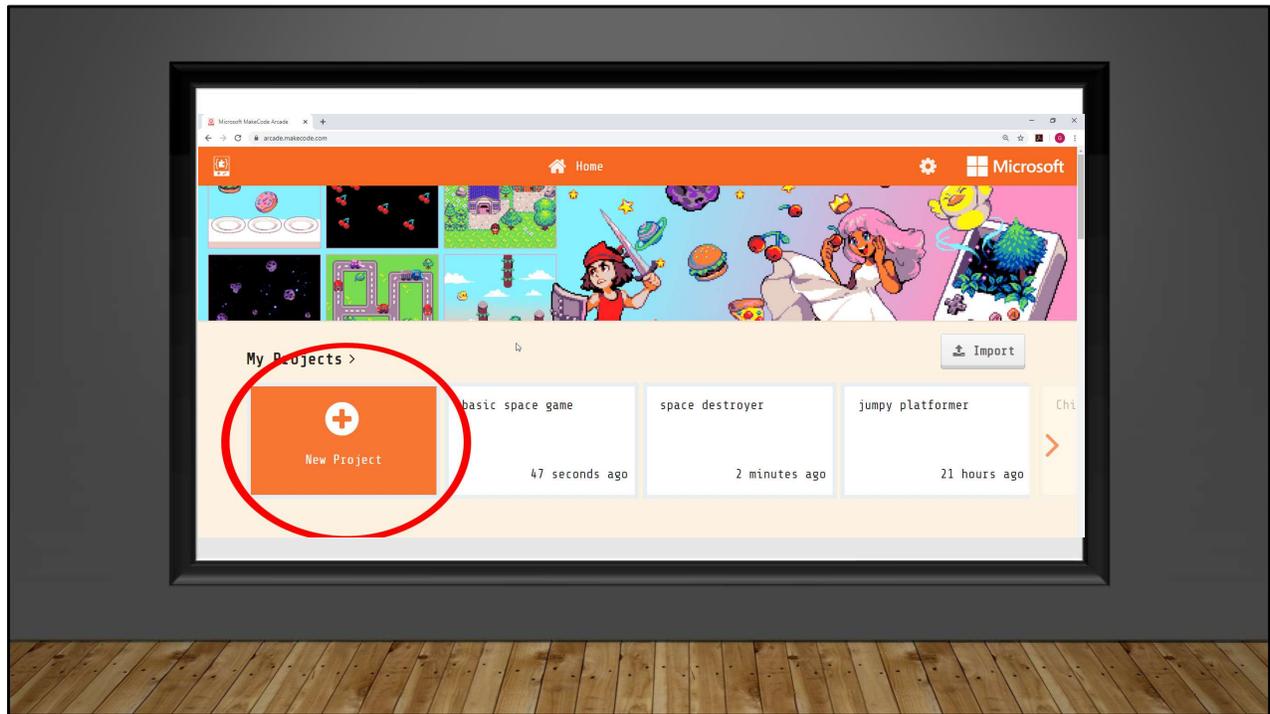
Let's create a Space Shooter game

Open a web browser and navigate to...

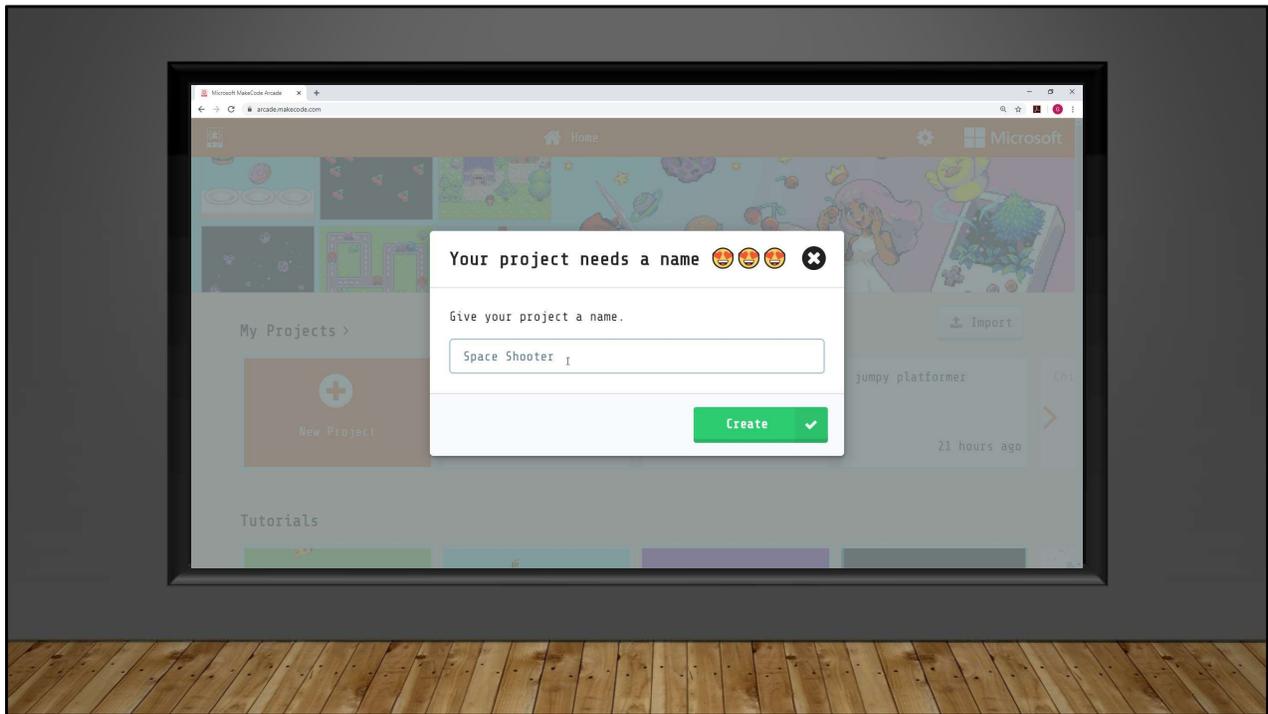


arcade.makecode.com

First open a browser window and navigate to arcade.makecode.com



Click on the 'New Project' Button, this will open a window to name our project and create a blank project for us to use.

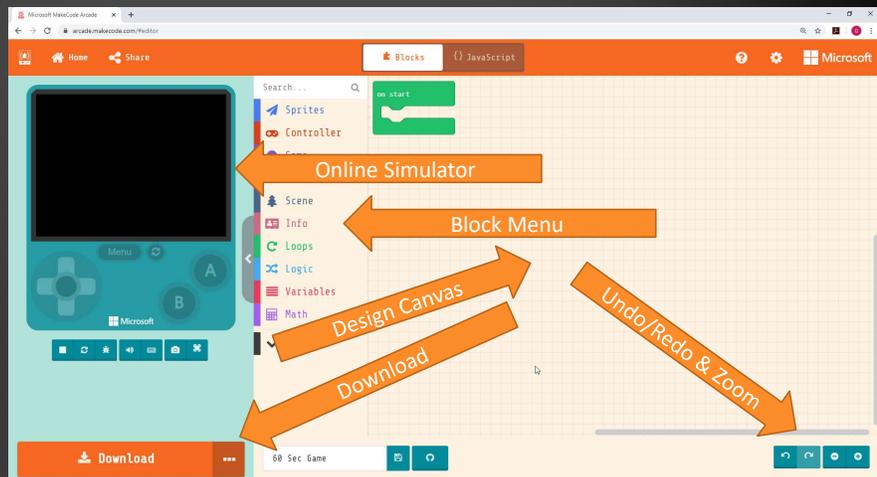


Now let's name the new project we're about to create. We'll call it 'Space Shooter'

DESIGN SURFACE

This is where the game you create will come to life!

- Online Simulator
- Block Menu
- Design Canvas
- Download (for later)
- Undo/Redo & Zoom



To change how a sprite looks, they will have to click on the white sprite box and then create their own sprite or use one from the gallery.



Computer Term:

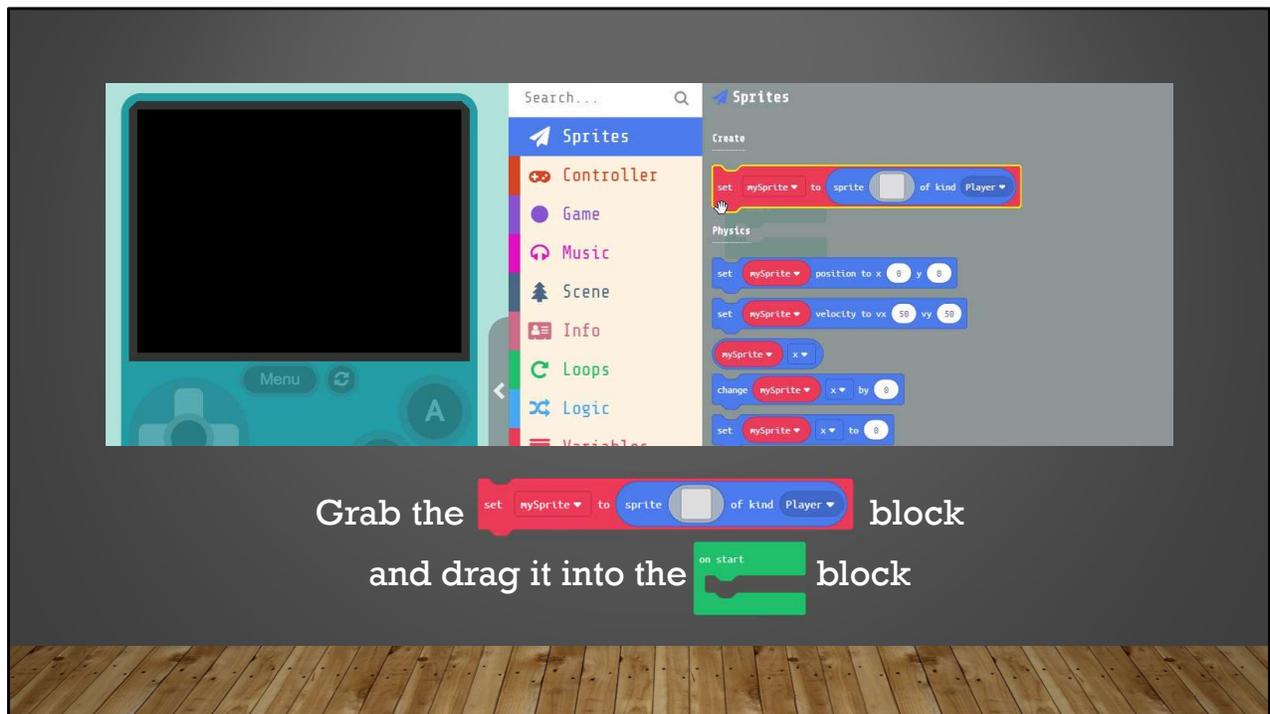
SPRITE

A sprite is a two-dimensional image that is integrated into a larger scene, most often in a 2D video game.

EXAMPLE:



SPRITE....a sprite is a two-dimensional image that is integrated into a larger scene, most often in a 2D video game. A video game has many sprites. The player we move on screen is a sprite. Monsters or enemies that chase the player are sprites. Gold coins we might collect in a game are also sprites.



Let's create our first sprite. First we need to add a block to put our player 'Sprite' in. Under 'Sprites' in the menu, grab the 'set mySprite to' block and drag it into the 'on start' block that is already in our work area.



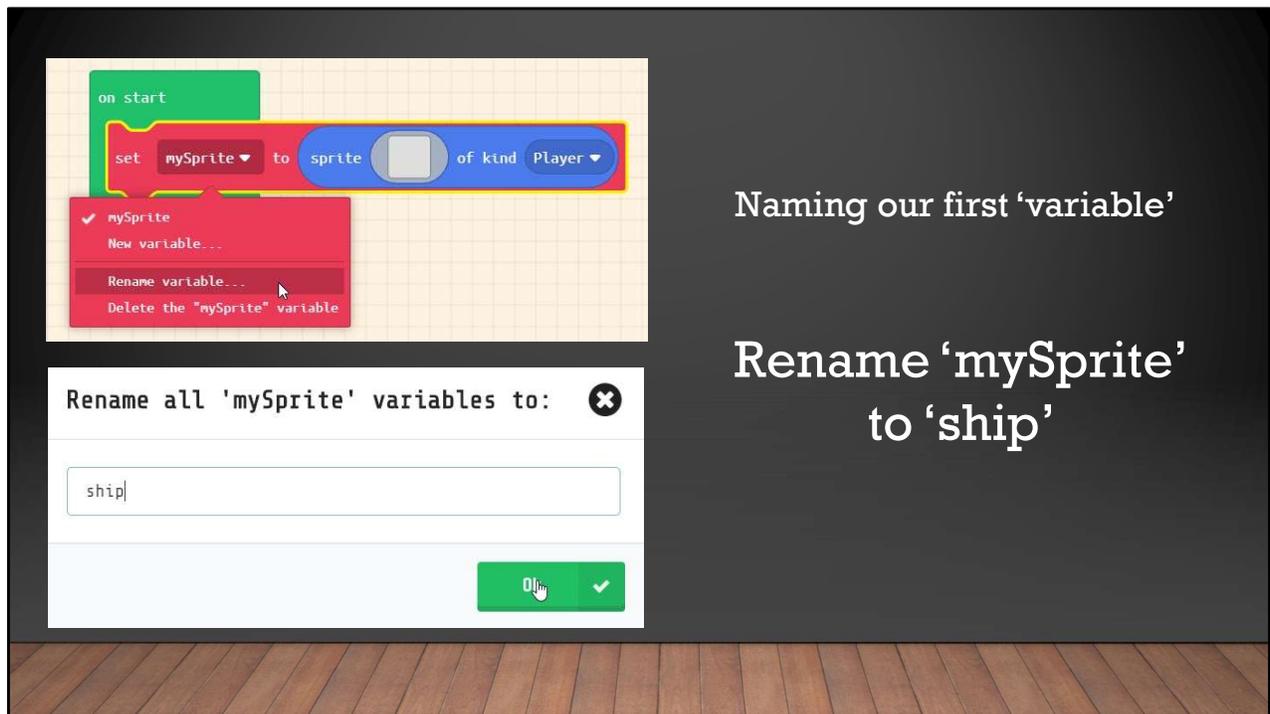
Computer Term:

VARIABLE

A variable is a letter or word, such as “x” or “score” that represents a changing value. Variable can be named anything but should be meaningful. To make code easier to read.

EXAMPLE: `score=score + 1`

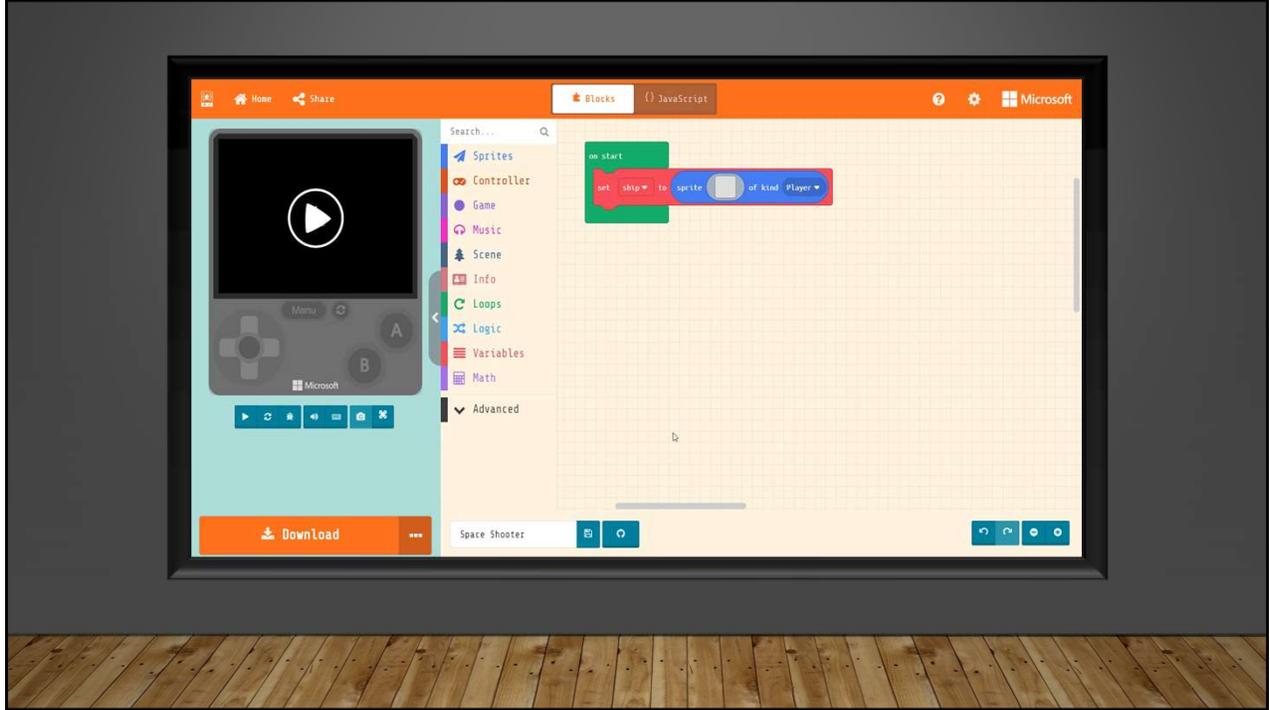
Variable....a variable is a letter or word, such as ‘x’ or ‘score’ that represents a changing value. Variables within a program will always be changing. We can name variables what ever we want, but they should use meaningful names, so that when reading our code we better understand what’s happening inside of it. An example of a good variable name would be ‘score’. What do you think the variable ‘score’ represents? What do you think the variable ‘score’ would equal when we first start our game?



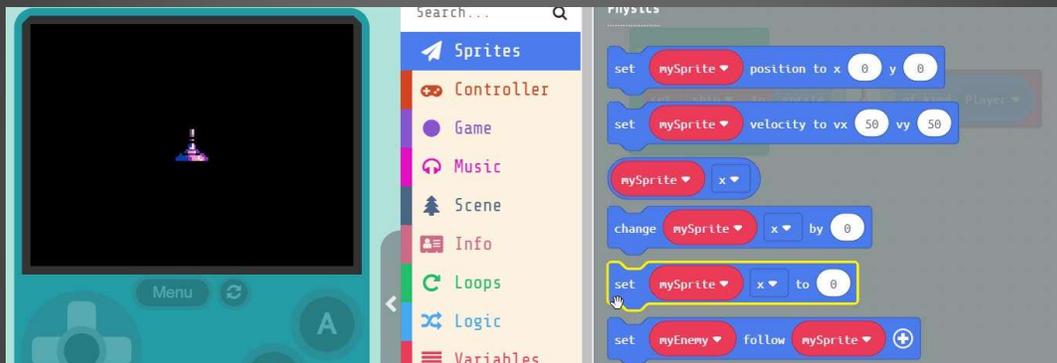
Naming our first 'variable'

Rename 'mySprite'
to 'ship'

Now let's rename 'mySprite' to something more meaningful. Since we're making a space shooter game, let's name the player 'ship'. We do this by clicking on the DOWN arrow next to the word 'mySprite'. Variables should always have meaningful names and not the 'default' names. What do you think the new variable we just created is going to represent in our program?



Now let's add our Spaceship sprite to the block. To do this click on the blank gray box inside the 'set ship' block. This will bring up the Sprite editor. This is where Sprites are created or selected. We will grab one of the pre-created Sprites from the 'Gallery'. Click on gallery and scroll down until you see one of the 'Spaceships' select it and press 'DONE'



Grab the  Block
and drag it into the  block just below our 'ship'

By default any Sprite we create is placed in the middle of the screen. We want to position the 'ship' near the bottom of the screen. To do this we need to select the 'set mySprite x to 0' block. Drag this to the 'on Start' block just below our 'ship' Sprite.



Computer Term:

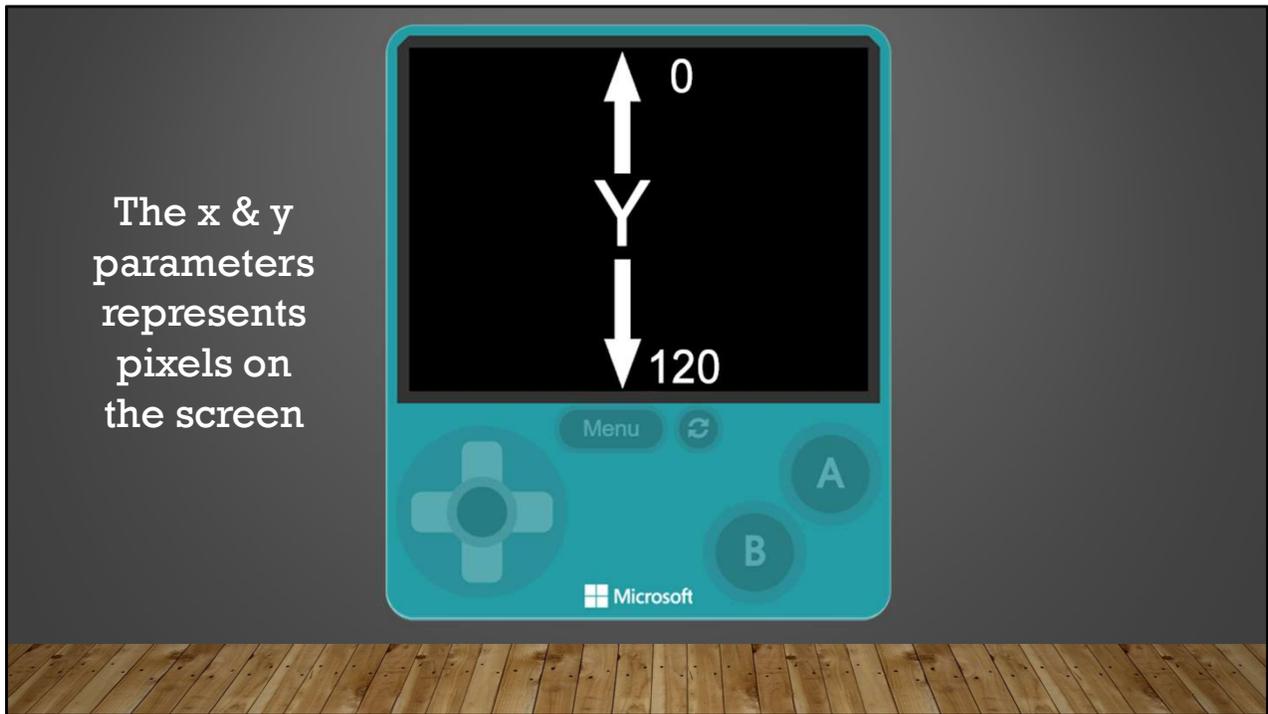
PARAMETER

A parameter is a value that we add inside a block. This number is passed into the block. In the example block below '0' would be the parameter. Parameters can also be referred to as 'ARGUMENTS'

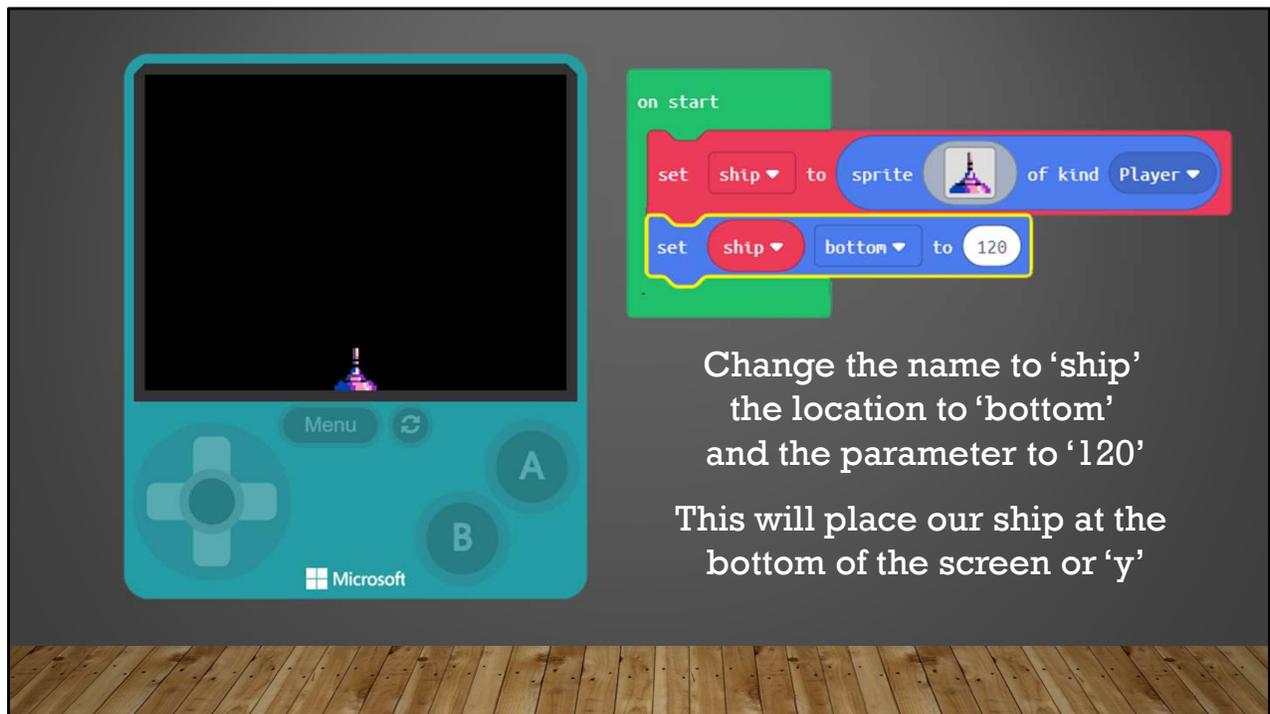
EXAMPLE:



Parameter, a parameter is a value we add inside a block. Parameters are also sometimes called 'Arguments' . This number is passed into the block. In the example here '0' is the parameter. Some 'block' can contain multiple parameters. The one in this example only contains 1.



When we talk about the 'x & y' parameter? What does the x and y mean? These are pixels on the screen. The display is 160x120, which means our display consists of 19200 individual pixels. We use the x and y to define exact pixels on the screen. The x parameter starts on the LEFT at ZERO and goes all the way to the right at 160. y reads top to bottom, with the top pixel of y = 0 and the bottom pixel of y= 120. These parameters help us place and move our Sprites around the screen.

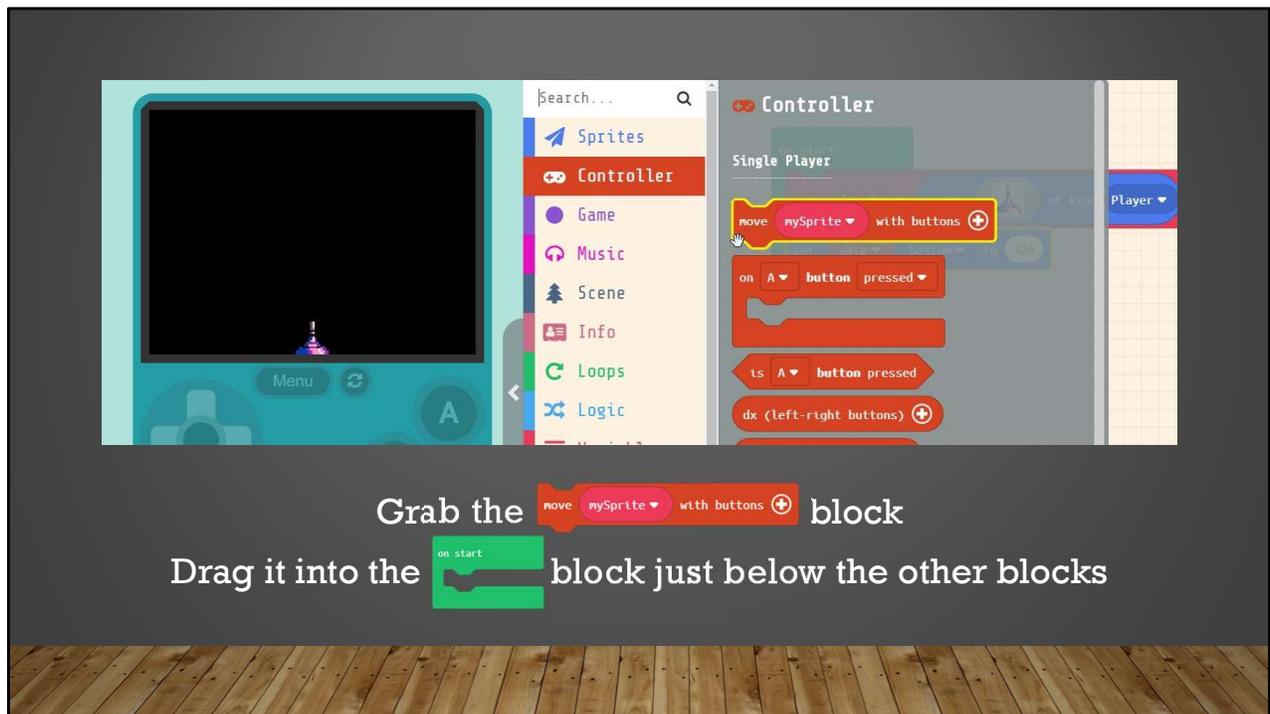


The image shows a game development interface. On the left, a teal-colored game controller is overlaid on a black screen. A small, colorful ship sprite is positioned at the bottom center of the screen. Below the screen are buttons for 'Menu', 'A', and 'B', and the Microsoft logo. On the right, there is a Scratch script area with a green 'on start' block. It contains two 'set' blocks. The first block is 'set ship to sprite of kind Player', where 'ship' is highlighted in red. The second block is 'set ship bottom to 120', where 'ship' is highlighted in red, 'bottom' is highlighted in blue, and '120' is highlighted in yellow.

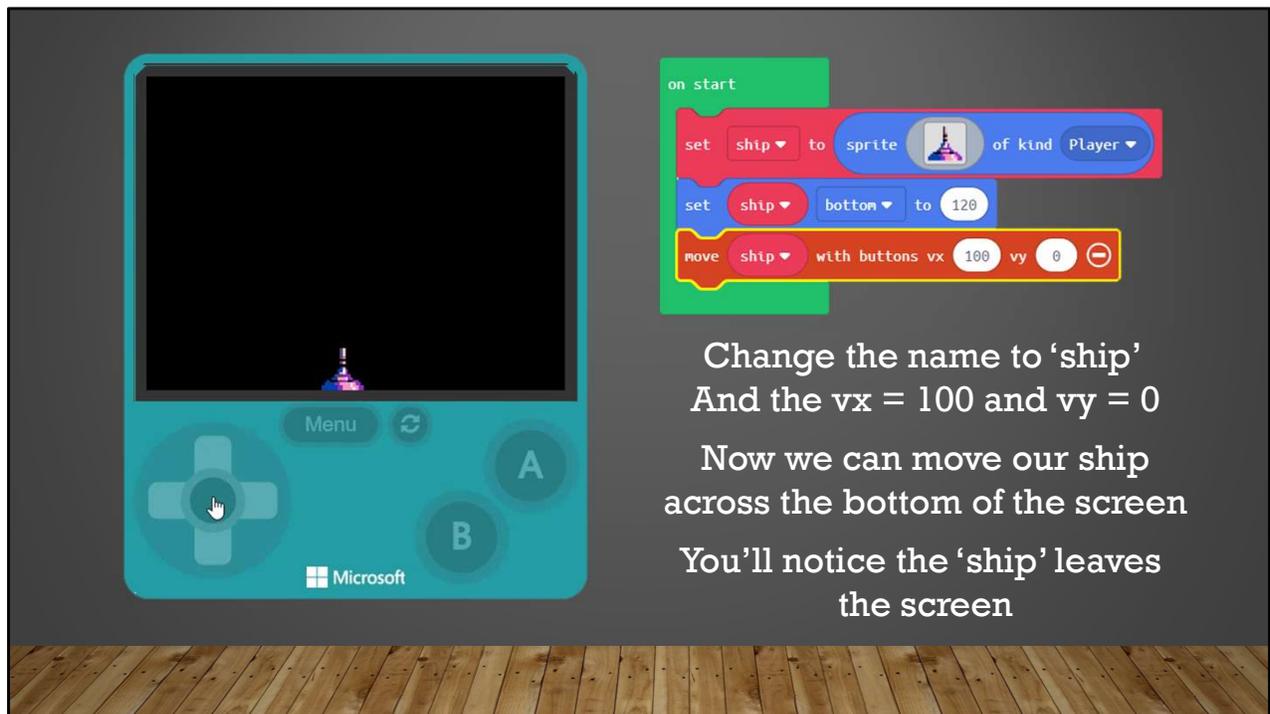
Change the name to 'ship'
the location to 'bottom'
and the parameter to '120'

This will place our ship at the
bottom of the screen or 'y'

We need to set the variable name to match our Sprite name which is 'ship', we also need to click on the down-arrow in the block and set our location to 'bottom' and finally the value to '120'. This will put our 'ship' at the bottom where we need it. We could also tell it to use the Y pixel, because as we noted 120 is the bottom Y pixel.



Now that our 'ship' Sprite is located at the bottom, we'll want to move it side to side using the LEFT and RIGHT buttons. Under 'Controller' in the menu select the '**move mySprite with buttons**' block drag it into the '**on start**' block just below the other blocks.



The image shows a game simulator on the left and its corresponding Scratch code on the right. The simulator is a teal-colored device with a black screen, a directional pad, and buttons labeled 'Menu', 'A', and 'B'. A small ship sprite is visible at the bottom center of the screen. The Scratch code is as follows:

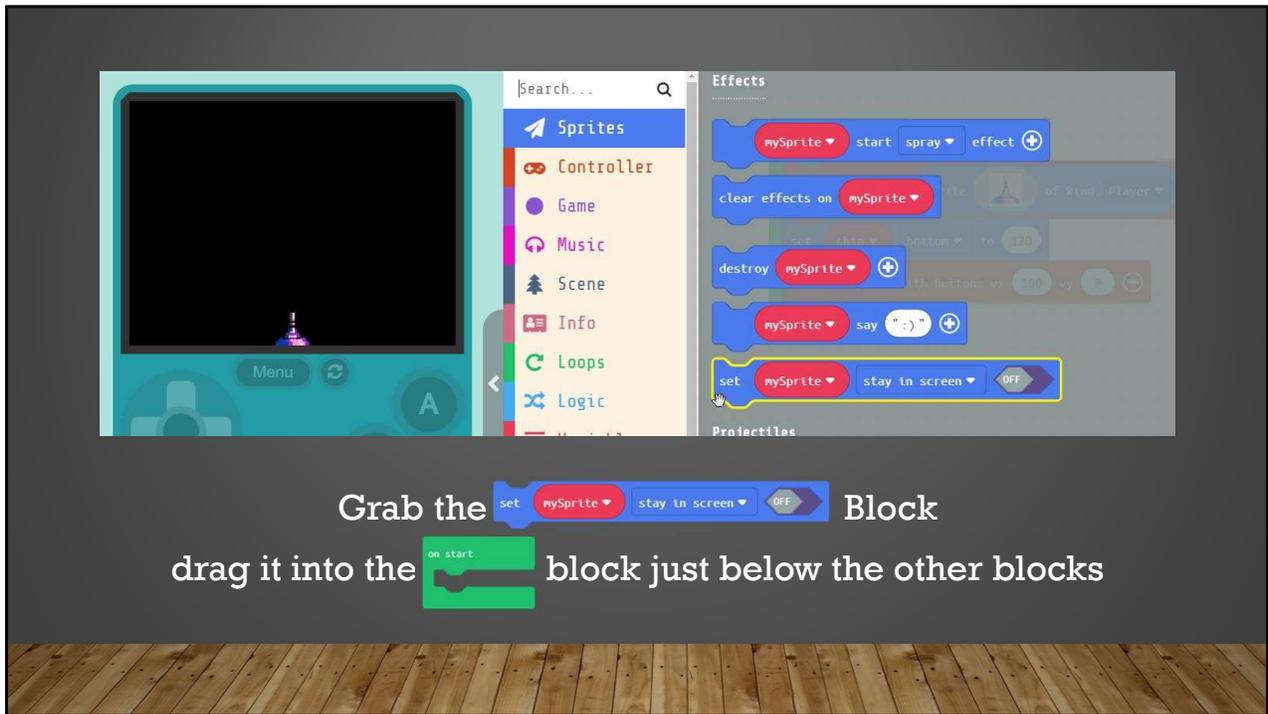
```
on start
  set ship to sprite of kind Player
  set ship bottom to 120
  move ship with buttons vx 100 vy 0
```

Change the name to 'ship'
And the vx = 100 and vy = 0

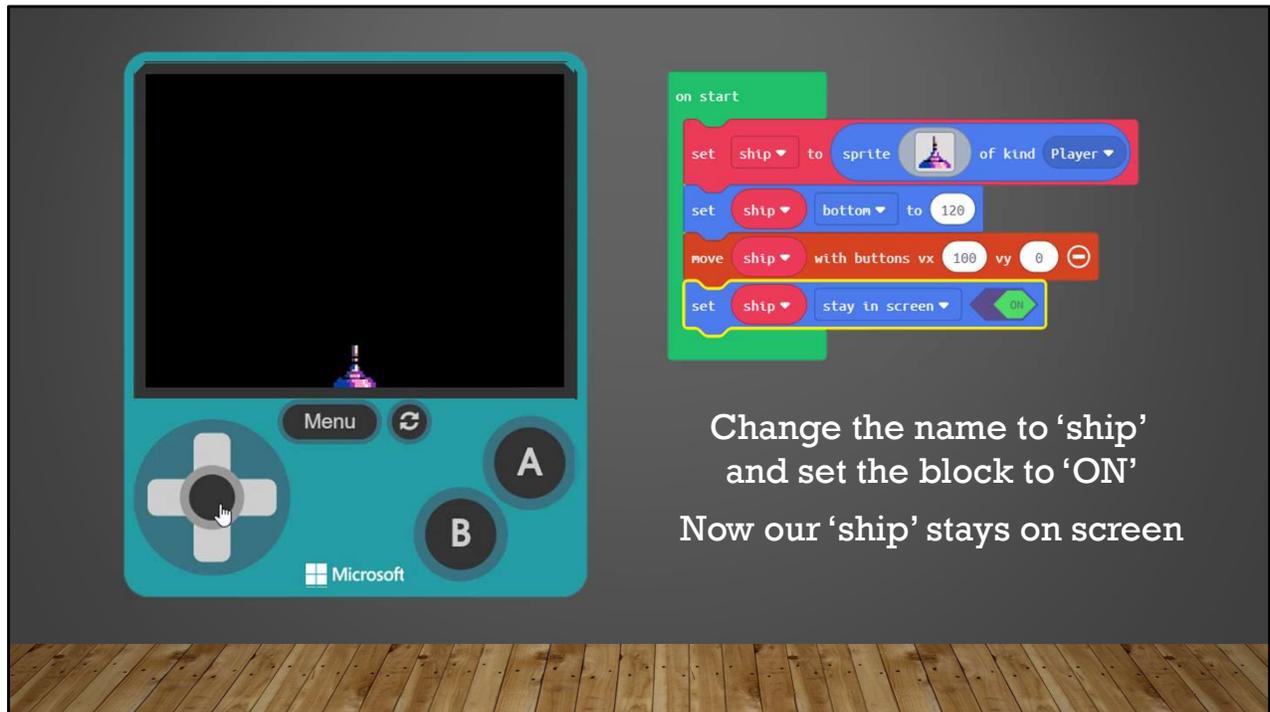
Now we can move our ship
across the bottom of the screen

You'll notice the 'ship' leaves
the screen

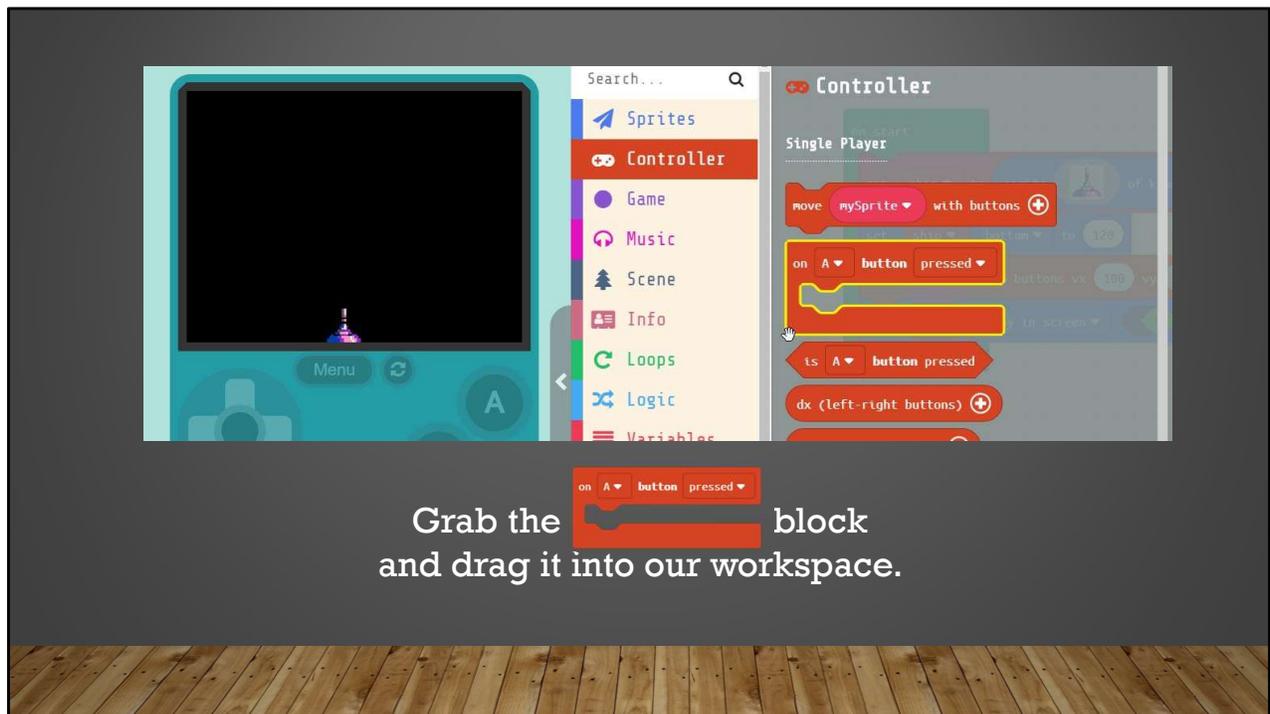
We need to change some of the values in this block to make it work. First, we need to set it to control 'ship' by changing the name from 'mySprite' to 'ship'. Next click on the '+' inside the block to reveal additional parameters we can change. Since we only need the 'ship' to move across the bottom we can set vx to '100 and vy to '0'. This will prevent the ship from moving up and down and only allow side to side. Try it in the simulator. You may notice that the ship can go right off the screen. We can fix that.



Under 'Sprites' in the menu, scroll down to the 'set mySprite stay in screen' block and drag it into the bottom of the 'on start' block.



Again, we'll need to change the name to 'ship' in order to attach the blocks action to our specific Sprite. We also have to turn the block 'ON'. Now when you move to the edge our 'ship' stops and won't go off the screen.



Now that we've got our ship moving let's make our 'ship' shoot. To do this we need to first add a button event to our project. Under 'Controller' select the '**on A button pressed**' block and drag it to our workspace.

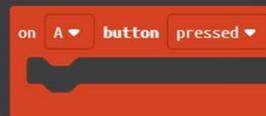


Computer Term:

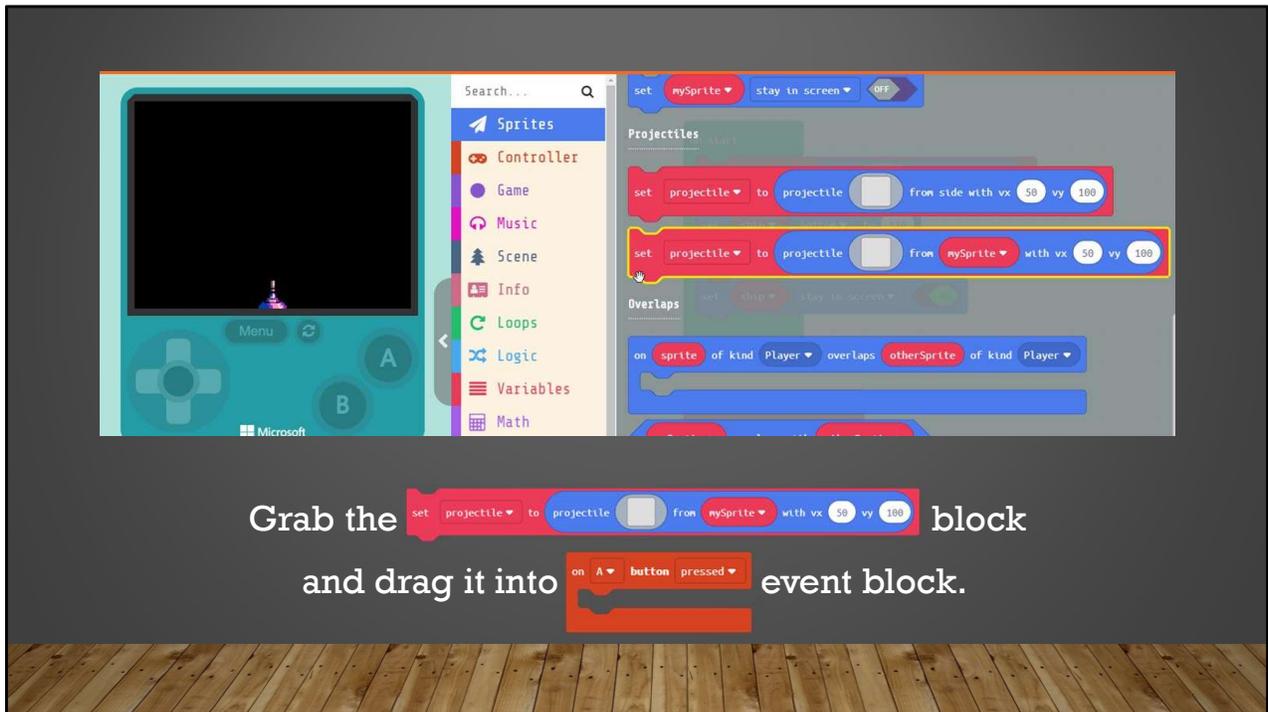
EVENT

An event is an action or occurrence detected by a program. Events can be a user action like clicking a button.

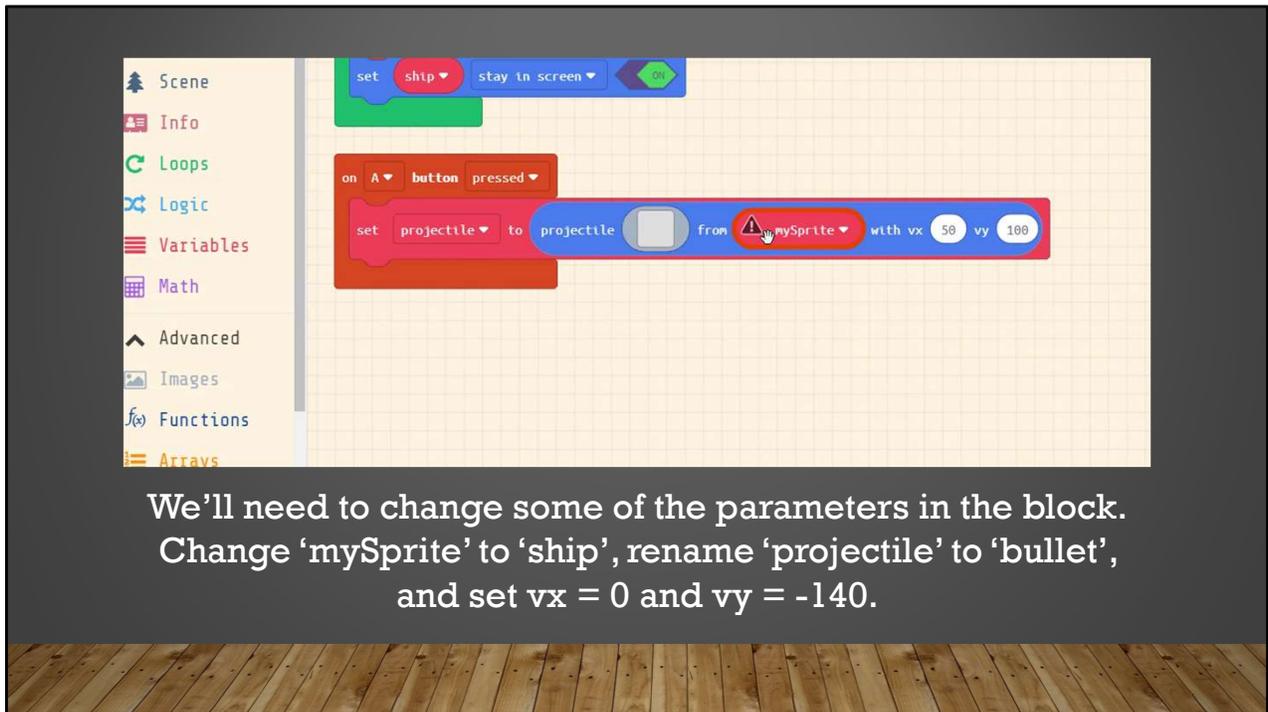
EXAMPLES:



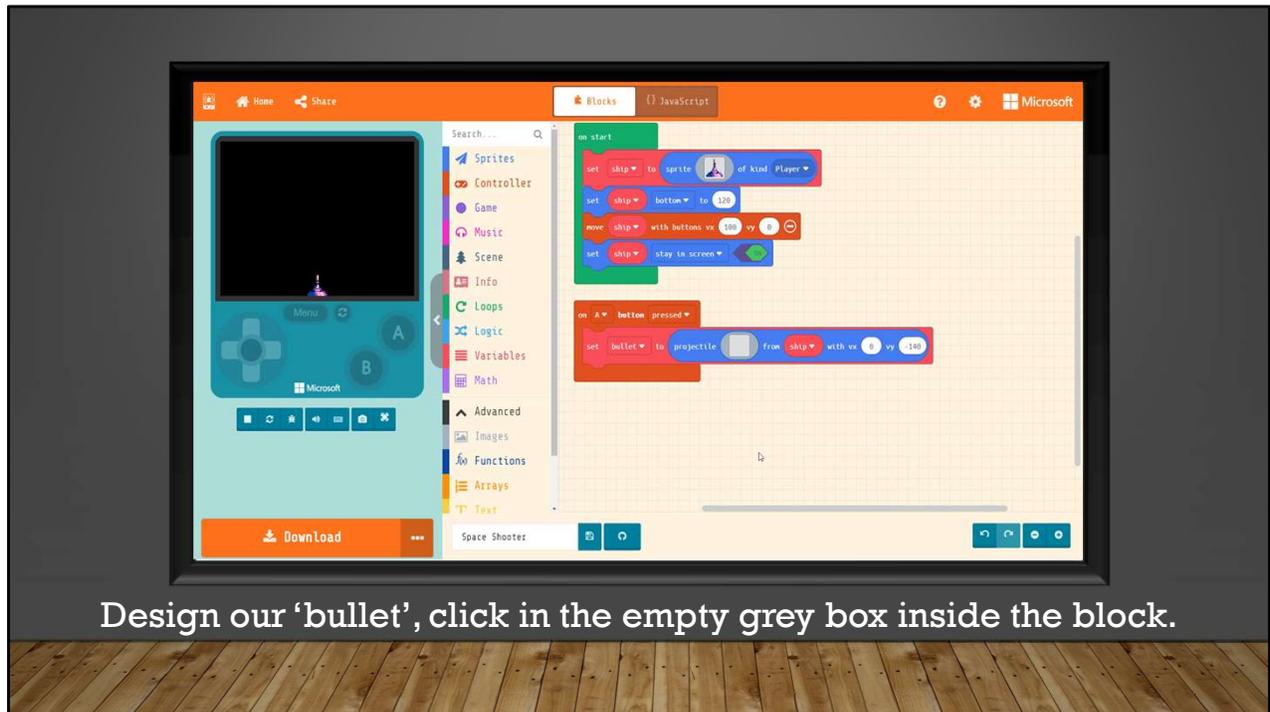
Now you may have noticed that **'on A button pressed'** block looks different than the other blocks. It looks more like the **'on start'** block, like a square. This is because it's what is known as an **'EVENT'** block. An **'EVENT'** block runs the code we place inside of it when an **'EVENT'** occurs. Like a button press. **'on start'** is an **'EVENT'** too. The first time we run our program, everything inside of it runs.



Now that we've got our ship moving let's make our 'ship' shoot. Under 'Sprites' in the menu grab the 'set projectile to from mySprite' block and drag it into the 'on A button pressed' event block.

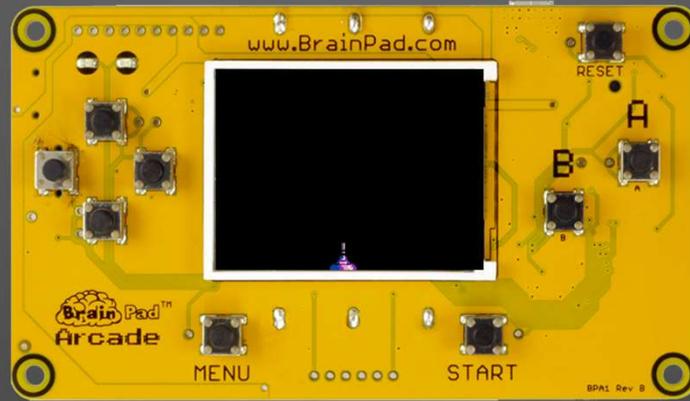


Now we'll need to change some of the parameters of the block. First let's change 'mySprite' to 'ship' this will ensure that the projectile we create will fire from our 'ship'. Next, we need to rename 'projectile' to 'bullet'. Finally let's set the speed. Change the vx to '0' and the vy to '-140'. This will make our 'bullet' shoot upwards along the Y-axis from the top of our 'ship' Sprite.



Now let's create the actual 'bullet' sprite, click in the empty grey box inside the block. This will open the Sprite editor. Here we can grab the pencil and draw our 'bullet' once created hit done. Then try it in the simulator by pressing the 'A' button.

Hardware Break



Now let's load what we have on to the BrainPad.

Now let's load what we have so far on the BrainPad.

EXTRA CREDIT

During the EXTRA CREDIT section students are shown a snippet of JavaScript Code. They must decide which block in their program the snippet represents.

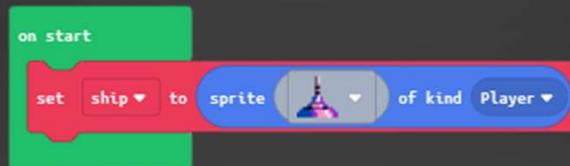


Code to Blocks:

JavaScript:

```
let ship: Sprite = null  
ship = sprites.create(sprites.space.spacePinkShip, SpriteKind.Player)
```

Block:





Code to Blocks:

JavaScript:

```
let ship: Sprite = null  
ship = sprites.create(sprites.space.spacePinkShip, SpriteKind.Player)  
ship.bottom = 120
```

Block:





Code to Blocks:

JavaScript:

```
let ship: Sprite = null
ship = sprites.create(sprites.space.spacePinkShip, SpriteKind.Player)
ship.bottom = 120
controller.moveSprite(ship, 100, 0)
```

Block:





Code to Blocks:

JavaScript:

```
controller.A.onEvent(ControllerButtonEvent.Pressed, function () {  
  bullet = sprites.createProjectileFromSprite(img`...  
    `, ship, 0, -140)  
})
```

Block:

