Scratch Space Invaders

Created by: Bob Freitas for TriValleyCoderDojo
Modified from original version presented at CoderDojoSV by BreakoutMentors
https://docs.google.com/file/d/0ByCWJ6l0aPOMUF83dmYwYlE4Qkk/

In this exercise we will do something a little different. You will be provided a partially completed project, and enough information to figure out how to complete the game. Sounds like fun right!

Okay, the game is the basic Space Invaders game, a popular arcade game in the 1980's. An armada of alien ships start at the top of the page and move back and forth across the screen as they gradually move down the screen getting closer and closer to you. You are the lone defender who must shoot the invaders before they can get to you, and of course they are shooting back at you. You are only allowed to move from left or right, and are limited to stay on the screen. There is no cover. You win if you can destroy all the alien ships, and you lose if the aliens shoot you or reach the bottom.

The starter project that you will begin the coding adventure with is provided in the URL below. This project will one have the code for one of the Sprites, and you will need to complete the code for all of the other Sprites.

http://scratch.mit.edu/projects/18378340/

You may either remix this project, or download it and work offline, which ever you prefer. Either way you need to have a working copy of this project that you can edit. Please leave the original as is, so it can be used for future sessions.

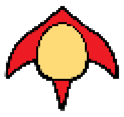
Figuring Out What Needs to be Done

The first thing you will notice is that you have three Sprites: ship_glider, spaceship1 and laser in the project. The Sprint spaceship1. is the only one that has code in it. Let's walk thru what the spaceship code is doing with some pseudo-code.

Notice that with pseudo-code, we just make a brief statement of what needs to be done, and do not worry about the details at this point. Some of those brief statement will expand into a large number of lines of code, but that is okay. We just want to get an idea of what needs to be done. This is called *Top Down Design*. You move to finer and finer levels of detail, as you work through each of the levels of detail.

```
when the program starts
make it so no one can see me
go to the upper left hand corner
create three rows of six copies of me


when my copy starts
make it so the copy can be seen
loop
     move right across the screen to the end
     move down the screen, just a little bit
     move left across the screen to the other end
     move down the screen, just a little bit
endloop


when my copy starts
loop
     if the laser shot me
          delete me from the page
     endif
endloop
```

So these three script blocks give us the basic functionality of our invading aliens. The first script block creates the army of invaders at the top of the page. The second script block moves the aliens across and down the screen. The third block allows for individual aliens to be killed when shot. However, I need to warn you, the shooting stuff is not quite done yet, but I promise we will get back to that later.



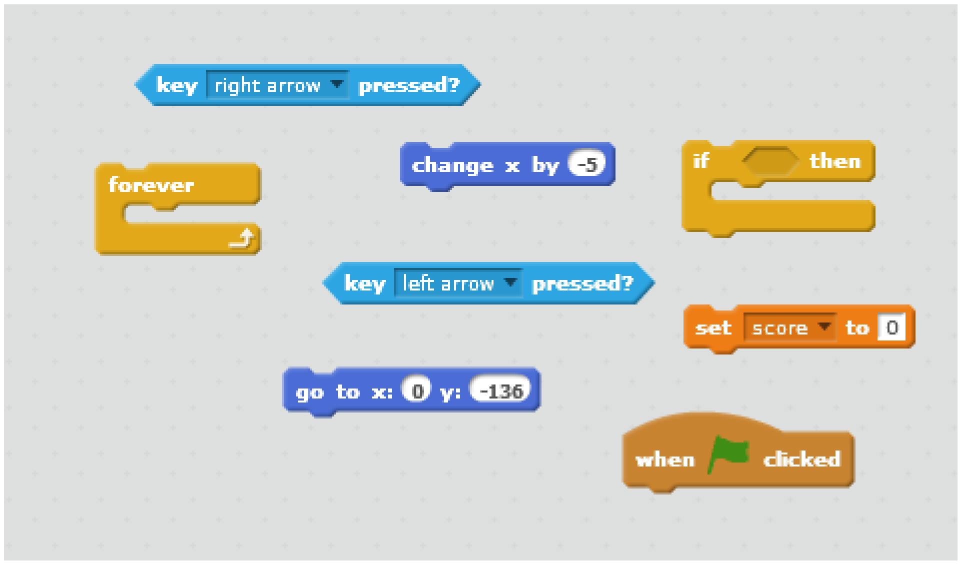
Moving Your Defender

The first things you are going to want is to move the defending spaceship (ship_glider) and of course do some setup from any previous games, stuff like reposition to the starting point and reset the score, etc... The original Space Invaders game used a joystick to move the defender. Clearly, we don't have a joystick, so we will use the left and right arrow keys.

Let's think through the steps that need to happen here:

```
when the program starts
put the defender back in the middle
reset the score to 0
loop forever
     if want to move left
          move the defender to the left
     endif
     if want to move right
          move the defender to the right
     endif
endloop
```

Some of the important blocks to use are shown below, but you will need to use them and a few more to do what needs to be done in the above pseudo-code. Think about what you will need to do to make that happen.



The observant reader will have noticed this score thing being set to 0. I am pretty sure you can figure out what this is for and how to create it.

Shooting Your Laser

Okay great, so now we can think about shooting our laser. The original Space Invaders game used a button to shoot. Since we don't have a button, we will use the space bar instead. The project came with a laser Sprite, which gives us a really good hint on where to start, but the question is how do we use the laser Sprite? Let's start by asking some questions.

Why does it need to be a separate Sprite?

Where should the laser start shooting from?

How do we not show the laser when it is not shooting?

What needs to happen when an alien gets shot?

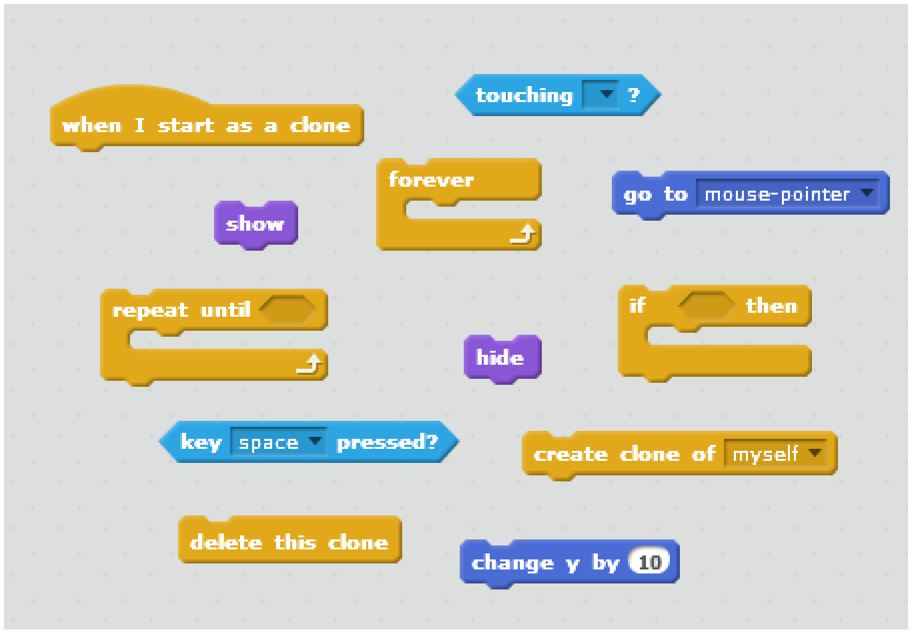
Well, we can answer some of these questions right! The laser needs to appear as if it is coming from our defender's ship. Clearly, we are going to need to use the hide and show blocks to only show the laser after someone presses the space bar. As to what needs to happen when an alien gets shot, well best to go to the pseudo-code for that.

```
when the program starts
hide the laser
loop forever
     if space gets pressed
          go to the defender, but keep me still hidden
          create a clone of myself
          wait a short amount of time to let the clone move
     endif
endloop
```

```
when a copy of me starts
show the clone
while not touching the edge
     move the laser straight up a little bit
     if touching an alien
          wait just a little bit
          delete me
     endif
end while
delete me
```

There a couple of important things to notice and consider. The first is that we are going to be working with clones here. What this means is that we start with an original object, this case the laser Sprite, and then we make a copy of it, and then it's the copy that goes and does something. We need to have separate code blocks for the original laser, which starts in the same place as the defender and creates a clone. Next we need a second block of code for the clone that actually goes up the screen and hopefully hits an alien. Another thing to notice is that we need to add short wait times which will allow the clone and the rest of the game to catch up with each other. Finally, notice that we need to make sure that we clean up each of the clones to make sure they get deleted.

Some of the important blocks to use are shown below, but you will need to use them and a few more to do what needs to be done in the above pseudo-code. Think about what you will need to do to make that happen.



This is the place where you need to think about what needs to happen back in the alien when it gets hit. Here we are deleting the laser on impact, but something else needs to happen in the alien. I will give you a hint. It has something to do with that strange score thing in your defender.

Make the Aliens Shoot Back

Okay, now it only makes sense that the aliens be able to shoot at you too. That would only be fair right? Besides what kind of self-respecting alien armada comes to invade a hapless planet without weapons.

How are we going to make this happen exactly? Well, we already know how to make our defender shoot, so wouldn't it be similar to that, only more of the same? Let's review what we did to make our defender shoot.

1) Use a Sprite to model the laser blast
2) Keep the laser blast Sprite hidden until we need it
3) When it's time to shoot, move the hidden Sprite to where we are going to shoot from
4) Create a clone of the hidden Sprite to become the shooter Sprite
5) Unhide the shooter sprite and move it in the direction of the shot

Huh, well, when you break it down that way, it's really not that hard is it! It's just a sequence of steps that need to be repeated over and over again for each of the alien spaceships. Now of course, we need to make sure that all the alien spaceships don't shoot at the same time, because that would just be a wall of laser blasts, and our defender would have no chance--not really all that much fun, huh! We will need to address this in our pseudo-code.

The first thing we are going to need is a new laser Sprite for the aliens to use. It seems pretty reasonable to start with the laser Sprite that we already have, right. Go ahead and make copy of that Sprite, give it a new name and delete all of the script blocks, because you will need to do something different. You may also want to change the color of the alien laser blast, just to make it different from the defender.

Now let's think about what we need here. The alien laser blast really needs to do two things: 1) it needs to stay hidden until we need it, and 2) it needs to move down the screen. Well we already know how to both of these right. We did this for the defender's laser blast. We are going to need one script block for the original alien laser and another for each of the clones.

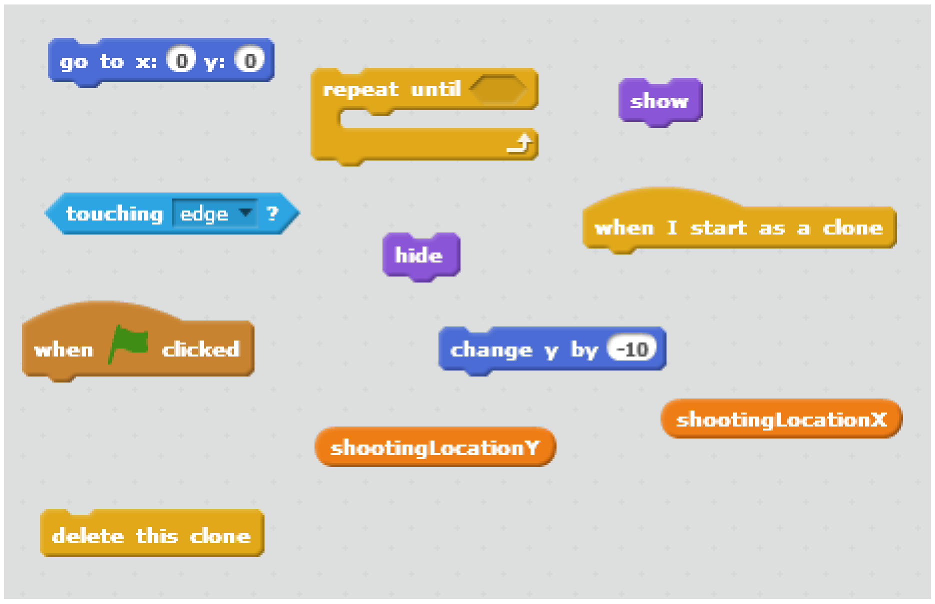
```
when the program starts
hide the laser

when a copy of me starts
go to the shooting location
show the clone
while not touching the edge
     move the laser straight down a little bit
end while
delete me
```

The shooting location might seem to be a little bit tricky, but it really isn't once you think it through. We have a total of 18 alien spaceships and each are at a different location at any moment. This means that we are going to have 18 different shooting locations. But each of those shooting locations are exactly where each of those alien ships are.

Well, hey we know what that is. It's the X & Y coordinates of each alien ship. Okay, this is really not that big of a deal. We just need to save the X,Y coordinate of alien ship someplace where we can get to it, and then use that for our shooting location. You can do this with a pair of variables that are available to all Sprites. Let's call them, shootingLocationX and shootingLocationY.

Notice, the use of mixed lower and uppercase letters in the variable names. This is calledcamelcase, and it is commonly used for variable names. Scratch allows you to include spaces in variable names, but other more complicated programming languages will not. It is a good habit to start using camelcase now.

The blocks that we are going to need are:

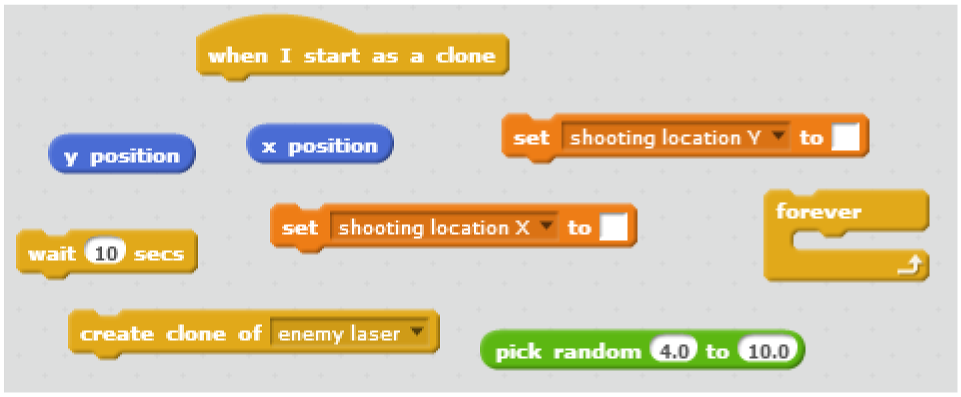


Okay, things are going well. We have our alien laser blast, and we have figured out part of what we need to do. Now, we still need to think about creating the clone and making sure that all the aliens don't shoot at once. We already know it has to be in the alien spaceship, since we need to get the X & Y coordinates, and that is the only place we can get that information. For the not shooting all at once, we should be able to get away with adding some kind of random wait time to each of the alien spaceships. That should work. Let's go to the pseudo-code.

```
when a copy of me starts
loop
      wait some random amount of time
      save the current X coordinate
      save the current Y coordinate
      create a clone of the alien laser
```

```
end loop
```

The blocks that we are going to need are:



Stop Your Defender From Moving Off the Screen

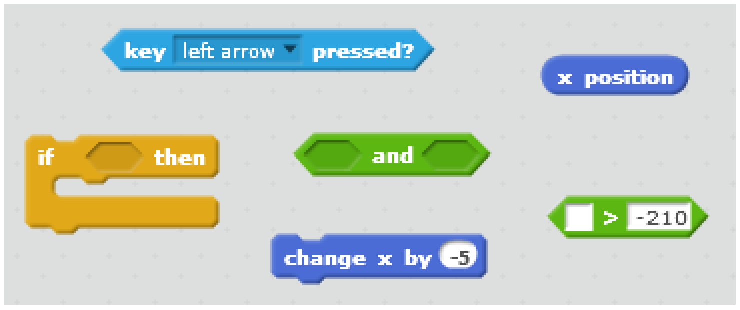
One final touch to the movement of the defender is to make sure it stays on the screen. Really all we need to do for this is back in the code for the defender is to not move if we are already on the edge of the screen. We just need to change the conditionals (if statement) to also consider the current position. \

Here is what we know. The screen goes from -240 to 240 pixels. The defender is about 50 pixels wide. So if we want to make sure that the defender does not move off the screen, then it can't go beyond -240 on the left and 240 on the right. But we also need to consider how big the defender is, because otherwise it would still stick out. So we need to measure from the middle of the defender, which is 50/2 = 25, and subtract that from the edge. So we get 240 - 25 = 215, but we should also give it just a little bit more to be safe. So let's take off an additional 5 pixels. After all this pretty easy math, we know that we don't want our defender to go beyond -210 on the left and 210 on the right. That should do it!

So back in the defender, we need to extend the logic to do something like this:

```
if want to move left and not already on left edge
      move the defender to the left
endif
if want to move right and not already on right edge
      move the defender to the right
endif
```

The blocks we will need are:



Losing the Game

WOW! This game is really coming together isn't it! We need to think about how to end the game. Let's start with losing the game first. If your spaceshipgets shot or runs into one of the enemy spaceships, you lose. Let's take a step back and think about what we are going to need to lose the game:

1) A new view to let you know you lost
2) A way to detect that we have lost the game
3) A way to let the other Sprites know that the game is over

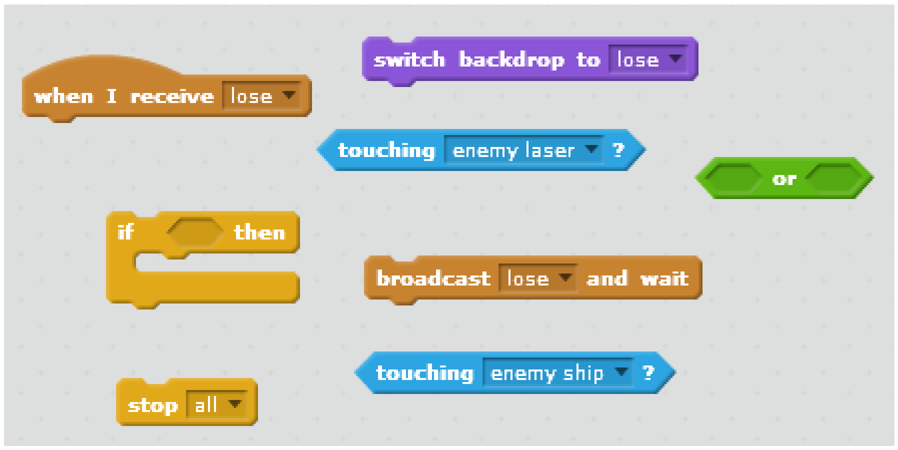
The first one is pretty easy, and you should be familiar with the general approach. First, create a new backdrop in the Stagefor the you lose view. Since you want it to be similar to the existing backdrop (backdrops1), it makes good sense to just create a duplicate of that one and some text to say something really original like ‘You Lost, Try Again’, and give it a new name, something like 'lose'.

Next, we detect that we have lost, and let the other Sprites know the game is over. Well, since the action is happening in the defender it makes sense to put it there. We are going to need to check for two things. First, if one the alien laser has touched the defender, and second if one of the alien spaceships have touched the defender. Well, hey we already know how to do this. Let's go to the pseudo-code.

```
if touching enemy laser or touching enemy ship
      change the backdrop to lose
      stop the game
endif
```

I think we need to drill down into the 'change the backdrop' line. What does this really mean? The backdrop we need to change is one of the backdrops in the Stage, but we need to change it in the defender. Okay, well we know how to do that. All we need to do is use a broadcast message. We create and send a broadcast message from the defender, and then in the Stage we catch that do what we need.

Here are the blocks we will need in both the Stage and the defender:



However, now we have a problem. When we added the new you lose backdrop, it creates the chance that we could have wrong backdrop showing. For example, we might be starting the game with the you lose backdrop. Not really such a great thing, huh!So what we want to do is make sure that every time the game starts it show the first backdrop, backdrop1. You guys are Coder Ninjas, and I have faith you can figure this out on your own.

Winning the game

Okay, we are coming into the home stretch here. All your hard work and coding expertise is paying off, you are almost done!

So then how do we know when we have won the game? Well, we can do that by keeping track of the alien spaceship that we shoot down. We are starting the game with 18 aliens, so we know we have won after we have shot down __ alien ships.

Ah ha, you may say "I remember something about this strange score thingy". Perhaps we can use that, and yes that would be correct. We are going to need a score variable to keep track of the number of aliens, and then as each alien gets kill we increment the score by one, and finally in our defender when the score reaches 18 we know we have won.

Here we are going to need a few things to make this all happen:

1) A new you win view

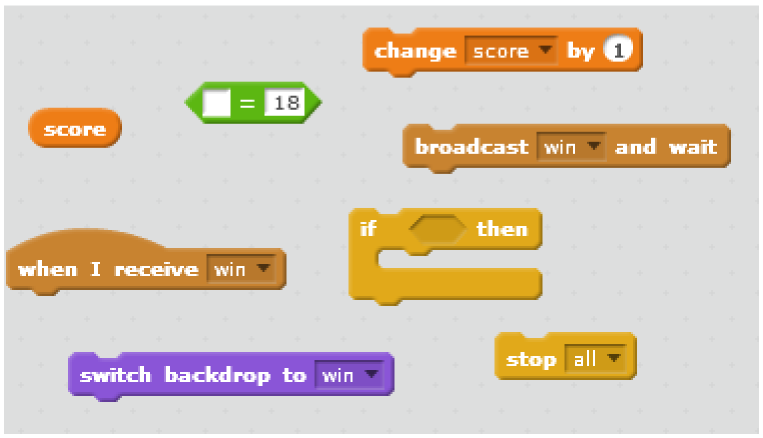
2) Keeping track of the each alien that gets killed
3) Checking for all aliens to be dead

For the new you win view, that is actually pretty easy. You just did that for the you lose view. Not a problem for a Coder Ninja like you!

The keeping track of the aliens that go killed is going to need to do that in the alien spaceship, if you have not already done it. Finally, the checking for all alien spaceships will of course be done in the defender and will need to look something like:

```
if score is equal to 18
      change the backdrop to win
      stop the game
endif
```

Really! It just can't that easy! But it is. Here are the blocks we will need in the Stage, alien spaceship and defender:



Additional Ideas and Enhancements

Adding Sounds

One way to make games more fun is to add sounds. Here are some ideas for sounds in your game:

- Laser shooting
- Hitting enemies
- You won
- You lost

Allow the Defender to Move Vertically

How about making it so you defender could move up and down. It might be nice to make sure your early shots really count. What would you need to do that?

Make the Game Have More Than One Level

Right now the game only has one level. You either win or lose. How about creating additional levels that get harder and harder. Maybe reduce the random wait time of the defenders shots. Maybe increase the speed that they descend towards you. There are load possibilities here.

Give the Defender a Shield

Wouldn't it be cool to create a special function to give the defender a shield to protectitself. Something like be able to press the S key and get an invincible shield. Of course, it would only make sense to limit it to only 2 or 3 usages. Otherwise, it would be just cheating...