WOMEN IN ENGINEERING DAY
VIRTUAL SESSIONS Oct. 21 & 28

University of Idaho
College of Engineering

Activity Introduction

Over the next week you will be working on virtual activities to learn more about engineering – specifically computer science! There are three activities in this packet to help you explore computer science:

Complete activities 1 and 2 to explore how programming works and submit evidence of your completion to www.uidaho.edu/wie-challenge in order to get credit for attending this event. Want more of a challenge? Take a look at activity 3 and see how many points you can get!

If you get stuck, have a question, or get curious, our team is here to help! Email engr-ambassadors@uidaho.edu with your question and you'll be connected with an engineer who can help you!

Submit your completed work on the WIE Day website: www.uidaho.edu/wie-challenge by 10:00 p.m. PDT on Tuesday, October 27, 2020.

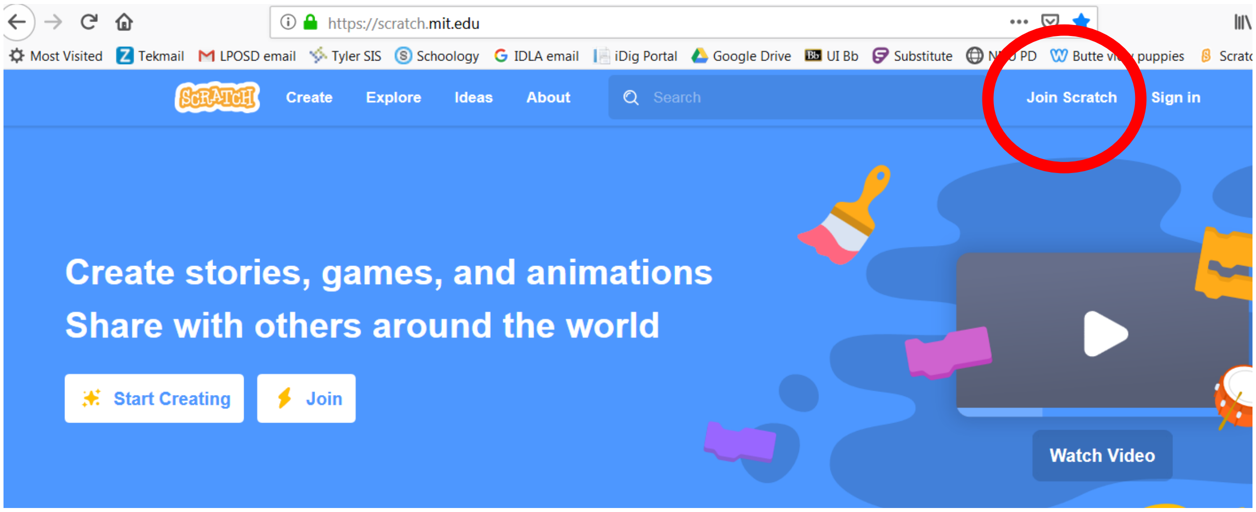
(this can include screenshots or videos of your scratch program, the link to view your shared scratch program, or a screenshot of your score in the cybersecurity game)

WOMEN IN ENGINEERING DAY
VIRTUAL SESSIONS Oct. 21 & 28

University of Idaho
College of Engineering

Creating a Scratch Account

1) Go to www.scratch.mit.edu
2) Click on “Join Scratch”



Create stories, games, and animations
Share with others around the world

3) Enter the requested information: Age, Gender, and Country. MIT uses this information to find out whose enjoying Scratch. See *Privacy Policy* at scratch.mit.edu/privacy_policy
4) Choose a Scratch username (not your name). *Write it at the bottom of this page.* Choose a password. *Write it at the bottom of this page.*
Take this page with you so you can log onto Scratch later.
5) Enter your or your parent’s email address.
6) You don’t need to check the email now. But if you ever want to share projects you will need to confirm the email from Scratch.

That’s it! You now have your own scratch account that you can use from any computer connected to the internet.

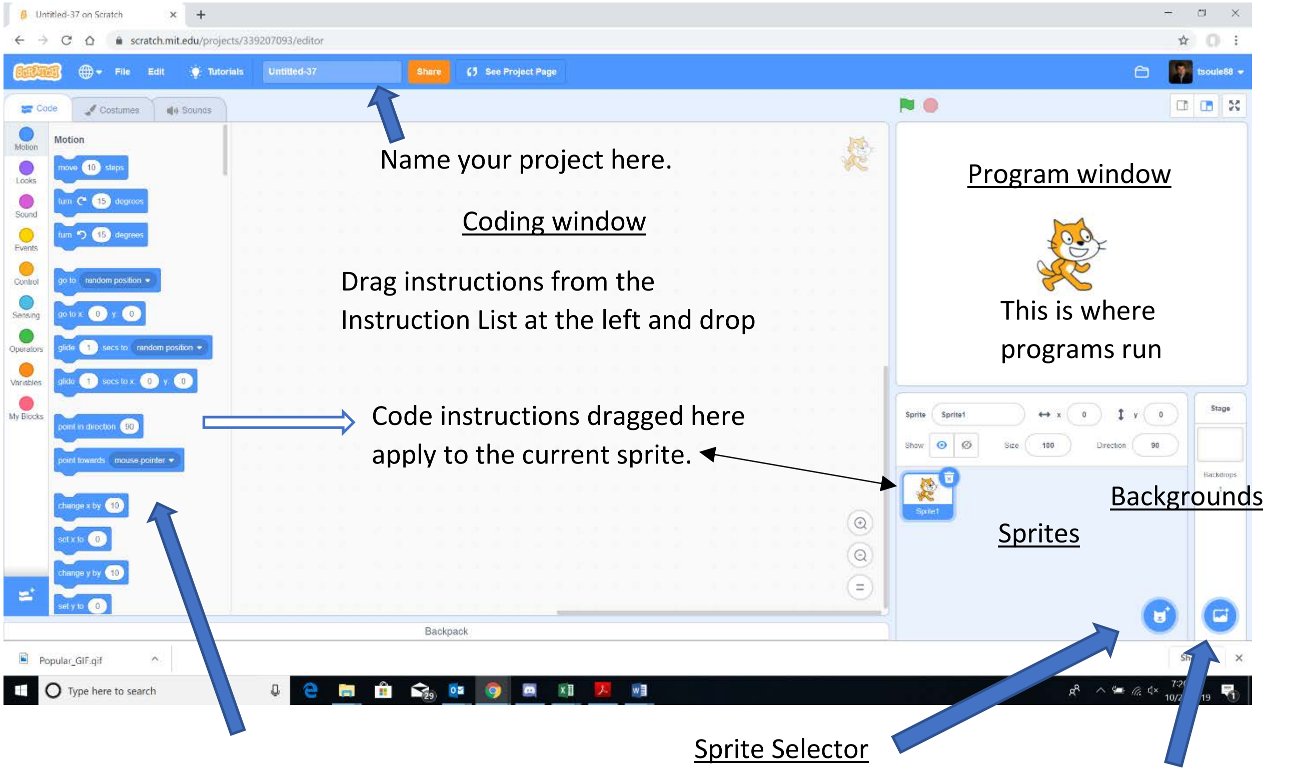
WOMEN IN ENGINEERING DAY VIRTUAL SESSIONS Oct. 21 & 28

University of Idaho College of Engineering

The Scratch Programming Window

Start a new project by clicking the folder icon in the upper right corner and then selecting new project: + New Project

You will see a Scratch programming page that looks like:



Name your project here.

Coding window

Drag instructions from the Instruction List at the left and drop

Code instructions dragged here apply to the current sprite.

Program window

This is where programs run

Backgrounds

Sprites

Sprite Selector

Pick another Sprite

Background Selector

Pick a different background

Instruction List

Divided by category (Motion, Looks, Sound, etc.)

The general idea with Scratch is to select one or more Sprites to include in the scene and then attach one or more code blocks to each Sprite to make them move and interact. Feel free to experiment with selecting Sprites, trying out instructions, etc. Nothing you do in Scratch will do any damage and you can always start over with a new project. All of your projects are saved on the web, so you can continue working on them from any computer that's connected to the Web.

Scratch Introduction: Basic Pong Game

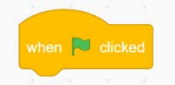
This activity will create a simple game where the player controls a paddle and bounces a ball.

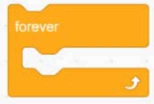
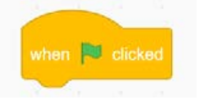
The Ball – Part A

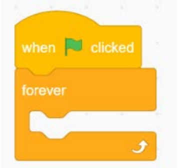
For this part of the project we will be creating a ball that bounces around the screen.

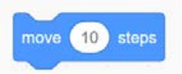
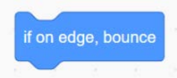
Step 1: Go to the Sprite Selector and pick a ball-like sprite . After you do this you may want to remover the Scratch sprite (the cat) by clicking on the trashcan icon in the sprites window.

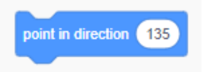
Step 2: Now we need to program the ball to bounce around.

a. We want the ball to start moving when we press the green flag so drag the  when 🏳 clicked block from the **Events** group into the Coding Window.

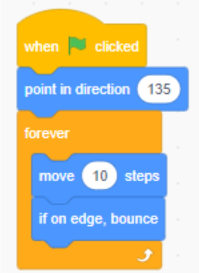
b. Next we want the ball to keep bouncing forever (or at least until we click the stop sign) so drag the  forever block from the **Control** group into the Coding Window, and connect it to the  when 🏳 clicked block. The result should look like:


when 🏳 clicked
forever

c. Next we need to make the ball move and bounce. Take two blocks from the **Motion** group:  move 10 steps and  if on edge, bounce, drag them inside the forever block.

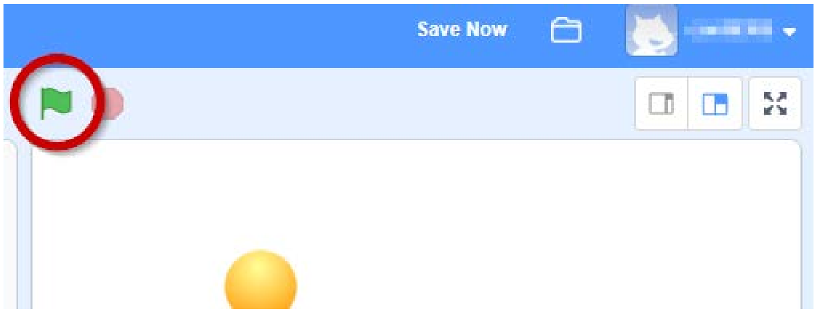
d. We need the ball to point in a reasonable direction. Drag the  point in direction 135 block from the Motion group and place it *right after* the when the green flag is clicked command. Click on the oval and drag the arrow to point in the direction you want the ball to move in.

e. When you are done you should have a block of code that looks like this:



```
when 🏁 clicked
point in direction 135
forever
    move 10 steps
    if on edge, bounce
```

Try pressing the green flag. Does the ball bounce around? Press the stop sign to get it to stop. Try changing the number (the 10) inside the move block. Can you make the ball move faster? Slower?

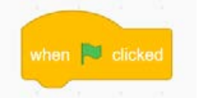


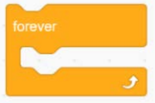
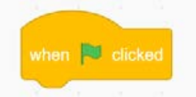
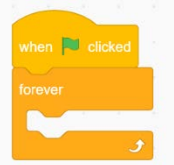
The Paddle

Now we are going to add a paddle that the player can control with the mouse.

Step 3: First we need a paddle sprite. Go to the Sprite Selector and pick a long, low sprite like the paddle or the broom .

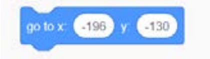
Step 4: We need code so the paddle (or the broom) follows the mouse. Note that if you click on the ball you see the block of code that we wrote for the ball, if you click on the paddle the coding window is empty because we haven't written any code for the paddle yet.

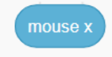
a. Just like the ball we want the paddle to start moving when we press the green flag, so drag the  `when 🏁 clicked` block from the **Events** group of commands into the Coding Window.

b. Next we want the paddle to follow the mouse pointer forever (or at least until we click the stop sign) so drag the  `forever` block from the **Control** group of commands into the Coding Window, and connect it to the  `when 🏁 clicked` block. The result should look like: 

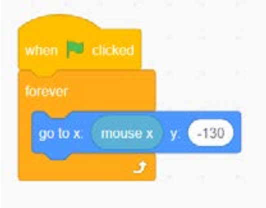
```
when 🏁 clicked
forever
```

Look familiar?

c. Now the paddle should *go to* where the mouse pointer is. So from the **Motion** group of commands drag the  `go to x: -196 y: -130` block and put it inside the forever loop. This block will have two numbers inside it. The x: number is where the paddle goes in a left-right direction and the y: number is where the paddle goes in an up-down direction.

d. To get the paddle to follow the mouse in the left-right direction we need to go to the **Sensing** group of commands replace the x: number with the  `mouse x` command. Note that this command is in an ellipse, so it fits in the x: number opening. For the y: number pick something like -130.

f. When you are done you should have a block of code that looks like this:



```
when 🏁 clicked
forever
    go to x: mouse x y: -130
```

Try pressing the green flag. Does the paddle follow the mouse across the bottom of the window? Press the stop sign to get it to stop. Try changing the number inside the y: oval. Can you set the paddle higher or lower? Pick a height you like.

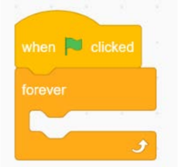
Does the ball bounce off the paddle? We'll fix that next.

The Ball – Part B

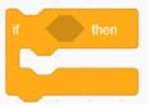
Now we need to make the ball bounce off the paddle.

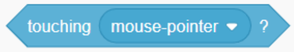
Step 5: we need to add another block of code to the ball's code. Click on the ball in the Sprite Window. You should see the ball's code, with the 'if on edge bounce' block in it.

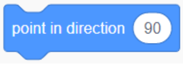
a. Once again we need a forever loop that starts when the green flag is clicked, so create another block like:

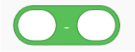


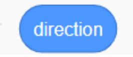
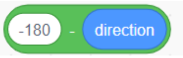
```
when 🏁 clicked
forever
```

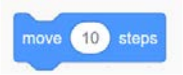
b. For this block of code we want that *if* the ball is touching the paddle *then* it bounces. So from the **Control** block of commands drag the  `if < > then` command inside the forever loop.

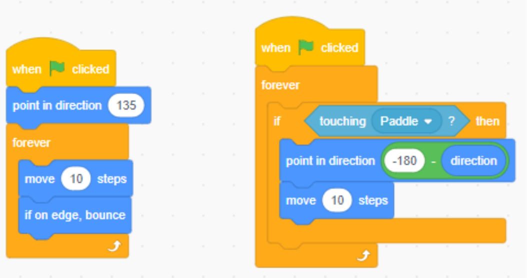
c. Notice the open hexagon shape in this command, we need to fill it with a condition. When the condition is true the commands inside the block are run. In this case the condition is if the ball is touching the paddle. So, from the **Sensing** group of commands drag the  `touching mouse-pointer ▾ ?` command into the hexagon in the if-then block. Now click the triangle and select the name of the sprite you are using for a paddle.

d. To make the ball bounce off the paddle we need to change its direction. From the Motion group of commands drag the  `point in direction (90)` in to the if-then block.

e. To get the ball to bounce in the correct direction off the paddle is a little tricky. From the **Operators** group of commands drag the  `( ) - ( )` command (this is subtraction) into the oval in the 'point in direction' command.

f. Put the value -180 in the left oval and from the *bottom* of the **Motion** group of commands put  `direction` in the right oval to get the combination  `(-180) - (direction)`

g. Finally, put a  `move (10) steps` after the 'point in direction' command. The ball should now have two blocks of code that look like:



```
when 🏁 clicked
point in direction (135)
forever
    move (10) steps
    if on edge, bounce
```

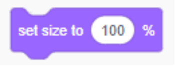
```
when 🏁 clicked
forever
    if <touching Paddle ▾ ?> then
        point in direction ((-180) - (direction))
        move (10) steps
```

Defense!

Now the player can bounce the ball, but there’s not much reason to – this isn’t much of a game yet. Let’s add some things that the player has to keep the ball from hitting.

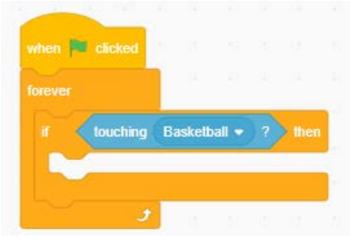
Step 6: First we need pick some things for the player to defend. Go to the Sprite Selector and pick any sprite you would like the player to defend from the ball. Like a happy little hedgehog .

Step 7: Now that we have a sprite let’s add its code.

a. First the Hedgehog (or the sprite you picked) may be a little too big. Go to the **Looks** group and drag the  `set size to 100 %` command into the Coding Window. Change the number to 50% and click on the command block – your sprite should shrink. Experiment with different sizes to find the right one.

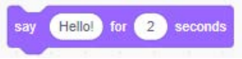
b. Drag your sprite to the bottom of the Program Window where the paddle can protect it.

c. We want the new sprite to respond whenever it is hit by the ball. So, we need a forever loop with an if touching ball inside it, like this:

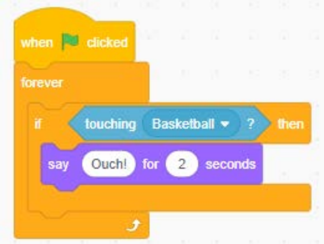


```
when 🏁 clicked
forever
    if touching Basketball ? then
```

(Your ball sprite might have a different name than Basketball.) Note that this is very similar to the code block we used for when the ball hits the paddle.

d. We need to decide what happens when the ball sprite hits the hedgehog sprite. For now we’ll have the Hedgehog say “Ouch”. From the **Looks** group drag the  `say Hello! for 2 seconds` command into the if block. Change the “Hello!” to “Ouch!”.

e. Your final code block should look like:



```
when ⚑ clicked
forever
    if touching Basketball ? then
        say Ouch! for 2 seconds
```

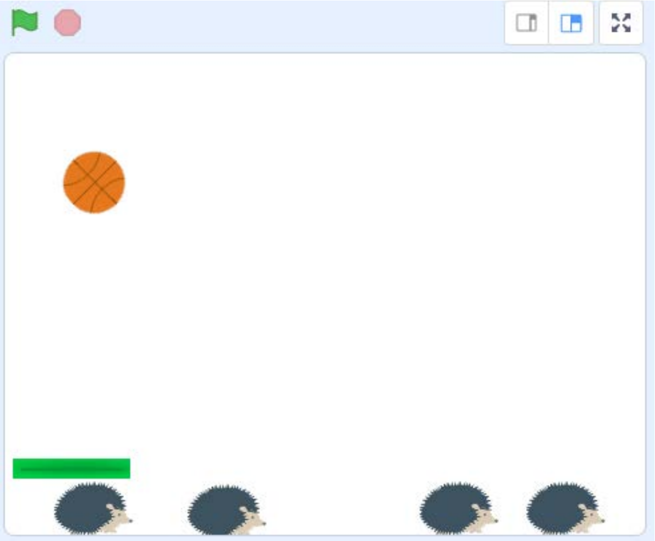
f. Try running the program. Can you defend your sprite? Is it too easy?

g. Let’s add some more things to defend. Right click on your sprite (mine’s a hedgehog) and select duplicate. Create several sprites to protect and spread them across the bottom of the screen. Now the game should be a bit harder.

And More

Step 8: Lots more things to try.

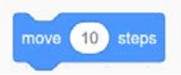
Right now my program looks like:

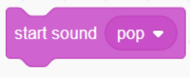


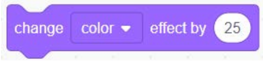
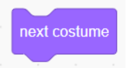
Yours might look a little different. It’s a simple game, but kind of fun.

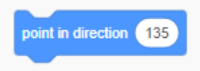
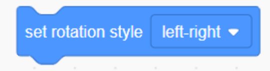
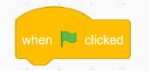
However, there’s lots that we can add to it. Here’s some things to try:

1) Go to the background selector (bottom right corner) and pick a background image for your game.

2) Change the speed of the ball – by changing the number inside the  move 10 steps command in the ball's block of code.

3) Go to the **Sound** group and add a pop sound whenever the ball touches the paddle (or a hedgehog). Place the  start sound pop command inside the 'if touching paddle' block on the ball. If you click on the sound tab at the top of the screen you can pick lots of other sounds to add. Note that you will have to make this change to all of the Hedgehogs' code blocks.

4) Go to the **Looks** group and add the  change color effect by 25 command above the Say command in the Hedgehog's block of code. Now they change colors when the ball hits them. Or use the  next costume command to change how they look.

5) Make the Hedgehogs move. You can do this by copying the code that makes the ball move. Just leave out the  point in direction 135 command – we don't want the hedgehogs bouncing around the whole screen. We probably don't want the hedgehogs flipping upside down so add the  set rotation style left-right command right after the  when clicked command in their code block.

6) There's lots more that can be done – explore the other Scratch commands and see what they do. Or go to the Scratch page and check out the projects other Scratchers have created.

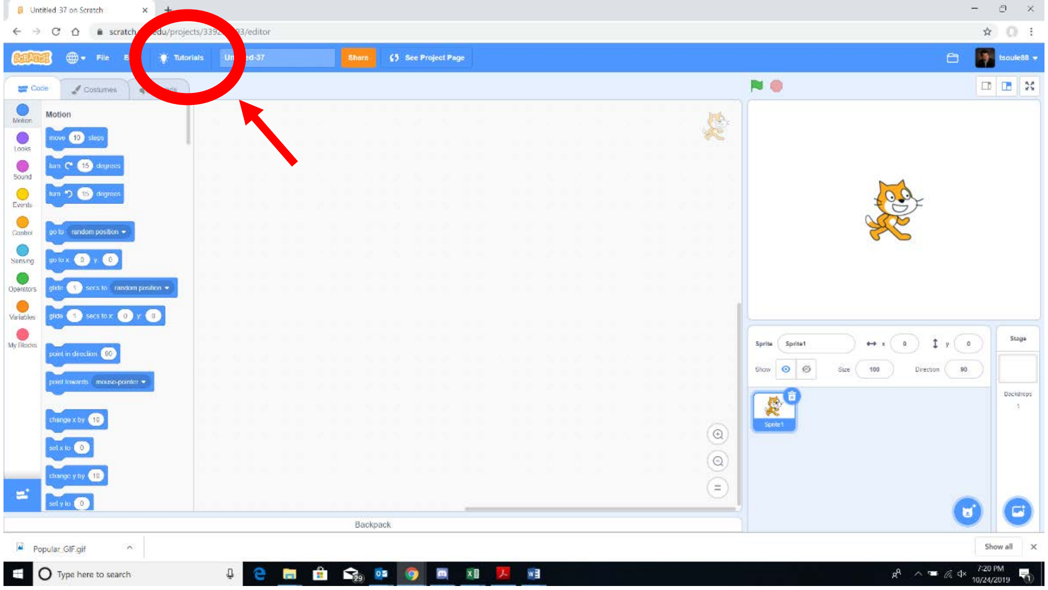
Scratch Exploratory Challenge

Now that you have some experience coding in scratch, it's time to explore the program more and challenge yourself! Your objective is to create your own program using scratch – and be creative!

As a challenge, your code must accomplish the following goals:

1. Your program should include a backdrop and at least 1 sprite.
2. Incorporate code that you learned from **at least two** of the following tutorials (note that not all of the tutorials are listed here – be sure to check out these ones):
    a. Animate a name
    b. Make music
    c. Create a story
    d. Make a chase game
    e. Animate a character or Animate a sprite
    f. Make a clicker game
    g. Make it fly
    h. Create animations that talk
    i. Use arrow keys
    j. Add effects

   You are welcome to use more than just two of the tutorials and explore what interests you the most! These tutorials outline instructions on some simple programs that you can make using scratch and provide a great way to get started!

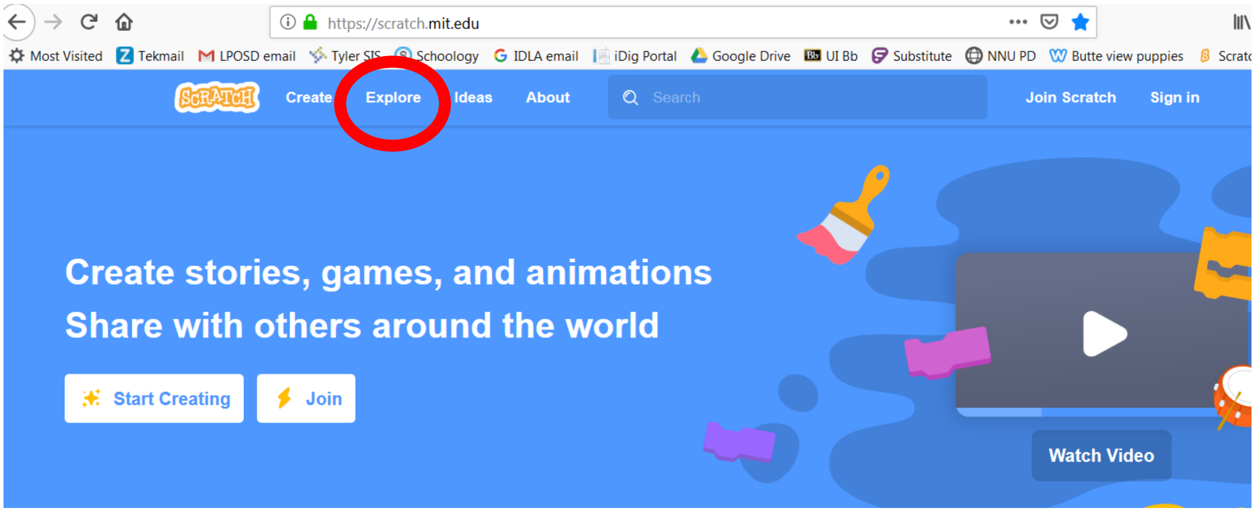


3. Your project must be about something that you are passionate about. Whether this means that you want to create the best virtual instrument, or you want to tell us a story about yourself – we want to see a project that is unique to you.
4. Your project must be appropriate.

5. Make something that you are proud of and ready to share with your other team members on October 28th! It can be fun, challenging, visually appealing, or just an exploration of the program.
6. <u>Bonus:</u> For a bonus, incorporate Women in Engineering into your project or some aspect of engineering! What kind of engineering are you passionate about?

The goals of this activity are to have fun, be creative, and experience engineering. We want you to learn how to approach the challenges that you find and engineer solutions to them.

For more project ideas if you're looking for inspiration, try looking at some of the existing projects in the explore tab.



Create stories, games, and animations
Share with others around the world

Cybersecurity Trivia Game (optional)

Go to cybersecurity game: https://ctf.cybercorestem.org/

Play the game and try to answer as many questions as you can!

Tips:

- Focus on the easy problems. you'll rack up points faster by rapidly solving easy problems then struggling with the hard ones.
- The server has rate limiting on answer submissions. So, if you get an error saying you have to wait to submit work on something harder in the meantime.
- This isn't a test so if you don't know the answer search for it. Once you've finished the problems you already know that is.
- Most importantly HAVE FUN!

Here are some informational videos on the various topics of this game:

- How Computers Work (code.org videos) - youtube.com/watch?v=OAx_6-wdslM&list=PLzdnOPI1iJNcsRwJhvksEo1tJqjIqWbN-
- Introduction to Linux - youtube.com/watch?v=navuBR4aJSs
- Linux for beginners tutorial - youtube.com/watch?v=HjuHHI60s44&list=PL6gx4Cwl9DGCkg2uj3PxUWhMDuTw3VKjM
- Introduction to Networking | Network Fundamentals - youtube.com/watch?v=cNwEVYkx2Kk
- Intro to Networking tutorial - youtube.com/playlist?list=PL6gx4Cwl9DGBpuvPW0aHa7mKdn_k9SPKO
- The Coding Train - youtube.com/c/TheCodingTrain/playlists?view=50&sort=dd&shelf_id=10
- Introduction to circuits and Ohm's law - youtube.com/watch?v=F_vLWkkOETI

Contact Information

Event Questions/Comments: Marie Wagner, mariew@uidaho.edu

Challenge Activity Questions/Comments: **engr-ambassadors@uidaho.edu**