Loops

SHEro of the Month - Anita Borg

Anita Borg was a computer scientist and huge advocate for women in computer programming. After graduating college, she co-founded *Systers*, which was an online community for women to discuss their experiences and work. Programs like this are really important so that women involved can feel a sense of community.

She created the Institute for Women in Technology, which helped those already in computer science as well as those thinking of changing their careers to join the field. She also started a yearly celebration for Grace Hopper, called GHC, where women are celebrated in computer science. GHC stands for Grace Hopper Celebration.

Anita Borg Institute: https://www.youtube.com/watch?v=hHieLtOMdfU
Anita Borg Speech: https://www.youtube.com/watch?v=ApyTBucW4BA

Follow-up Questions:

- What organization did Anita co-found?
- What was the purpose of *Systers*?
- What does GHC stand for?
- What is something that you liked, or something that stood out to you, about Anita Borg?

Reinforcing Known Concepts

There are a couple topics that we've covered that are still confusing some of us, so these fun videos will help us review while also describing the info in a different way! We will need variables and conditionals to write our loops later.

Variables Video - https://www.youtube.com/watch?v=ghCbURMWBD8
Conditionals Video (first 7:25 minutes) - https://www.youtube.com/watch?v=HQ3dCWjfRZ4

(Allow questions, before we move on.)

Loops

Overview
What is a loop? Loops, in Javascript, will continue to do something with our code until they are stopped. There is always a conditional (like we learned about last month) that will tell the loop it's time to be done. When you write a loop in programming, it will often be called recursive, or recurring, because it keeps acting again and again.

There are lots of different kinds of loops (visual examples in the slides):

- While loop
- Foreach loop
- For loop
- A function that calls itself
- Infinite loop

We'll be learning about **while loops** today!

Reinforce the Material

Real Life Examples

Who knows how to hula-hoop?
A hula-hoop is a hoop, or loop, that you put around your waist and make go in circles over and over. The hoop will keep going in circles until you tell it to stop.

Can anyone think of a way that our Edison Robots were using loops?
There was an option to have your robot spin for a certain number of seconds, or even when the robot moved in a certain direction for so many seconds. The act of moving was actually a loop! The robot would keep going and going until our condition (the number of seconds) told the robot to stop.

A timer is a loop!
When you use a stopwatch, it keeps adding to the time until you click a button to tell it to stop. If you're using a timer that ticks down to 0, the condition that makes our loop stop is that we've run out of time.

While and Do While Loops Video: https://www.youtube.com/watch?v=v-K-4KuA8mQ

Let's write a loop!

Like last time, you will have a CodePen set up for you that will include working HTML and CSS. The only thing you will need to worry about today is your JavaScript tab.

Example #1 - Use a Loop to Count to 10

The code in front of you will have some JavaScript already set up! Look for this part of your code:

```
1  // Example 1 ----------------------------------
2  // We are going to count to the number 10!
3  // Setup
4  var countingBlock =
   document.getElementById("countingBlock");
5  var countingNumbers = 1;
6
7  // Start your code below this line.
```

You're going to be starting your code below line 7, as the comment shows above. What is already done for you?

We are setting up a variable called **countingBlock** that finds any element in our HTML by ID. This is where we're going to put the numbers that we're counting. We are also setting up a variable called **countingNumbers** that starts at 1. This is what we're going to use for counting.

Note: You will see other code in the JavaScript section of your CodePen. That is the setup for our other examples! Please start your code on line 8.

Add the following code below line 7:

```
while (countingNumbers <= 10) {
 countingBlock.innerHTML = countingBlock.innerHTML + "<p>" + countingNumbers + "</p>";
 countingNumbers = countingNumbers + 1;
}
```

What is our code doing? We are writing our first while loop! This loop will keep going until **countingNumbers** is 10. Remember that we need a way to STOP the loop. Inside our loop, we're adding 1 to **countingNumbers** each time the code runs. Once **countingNumbers** gets to 10, the loop will stop. Also in our loop, we are adding a

 with the value of **countingNumbers** to **countingBlock**. This will basically print the number that we've counted onto our website.

Bonus: Can you make the website count to 15? How about 20?

Example #2 - Basic Loop

This example is VERY similar to our last one! Instead of counting, we're going to print the same sentence on our website FIVE times using a loop. Find this section in your JavaScript:

```
// Example 2 ----------------------------------
// We are going to put 5 lines of text on the screen.
// Setup
var containerForText =
document.getElementById("basicPrintingBlock");
var numberOfLines = 1;

// Start your code below this line.
```

We are again setting up a variable that points to an HTML element on our website. **containerForText** is where we're going to be adding our text. The variable **numberOfLines** is what we're going to use to tell our loop when it's okay to stop.

Add the following of lines below the commented line:

```
while (numberOfLines <= 5) {
 numberOfLines = numberOfLines + 1;
 containerForText.innerHTML = containerForText.innerHTML + "<p>I love to code!</p>"
}
```

Each time we're in the loop, we need to add **1** to **numberOfLines**, so that the next time the code goes to loop it knows if it needs to stop (when **numberOfLines** is greater than 5). Our conditional is currently checking if **numberOfLines** is less than or equal to **5**.

Bonus: Can you print 10 lines on the screen? Can you change what text gets printed on the screen?

Example #3 - Stopwatch

This example is a little tougher! If you get stuck, please raise your hand for help.

Find this section in your JavaScript:

```
36  // Example 3 ----------------------------------
37  // We are going to build a stopwatch that starts at 0
    and shows a timer changing for 2 minutes.
38  // setTimeOut is a NEW function - this let's us do
    something every 1, 2, 3, etc. seconds. So that this
    looks and works like a real timer, we are going to use
    setTimeOut to make our code run every 1 second.
39  var isWatchStillGoing = false;
40  var startSeconds = 0;
41  var startMinutes = 0;
42
```

We're going to build a timer that starts at 0 and shows a timer changing every second for 2 minutes. This is a little tough to do in code. You could write a loop that counts "seconds" 60 times and makes a minute every time we get to 60, but the code will go so, so fast that you won't actually see the numbers change until we've counted to 120, or 2 "minutes". We're going to solve for this using **setTimeout**, which is a function we'll explain more later.

Above, we are setting up a few variables that you should already have in your code. The variable **isWatchStillGoing** will tell the code whether we have started or stopped the timer using these buttons:

The variable **startSeconds** will control how many seconds show on the timer. The variable **startMinutes** will control how many minutes show on the timer.

You will already have the functions we need in your CodePen, but you will be the one telling the function what it needs to do. Add the purple code to your **onStartWatch** and **onStopWatch** functions.

```
function onStartWatch() {
  isWatchStillGoing = true;
  increaseTime();
}

function onStopWatch() {
  isWatchStillGoing = false;
}
```

This code is telling our stopwatch what to do when we start or stop it. **onStartWatch** controls starting the watch - it sets **isWatchStillGoing** to **true** (starts the watch) and calls our **increaseTime()** function (responsible for increasing the time on the watch).

onStopWatch tells the watch it's time to stop! It sets **isWatchStillGoing** to **false**.

Remember when we talked about **setTimeout**? Don't worry, that code is already written for you. This function tells our code to execute after 1 second has passed. So, every second it runs the code. Notice that at the end of our code, we are calling **increaseTime** again. This is another way to loop! Because our code is in a **setTimeout** function, it is doing the following steps every time the function is called:

- Wait one second
- Is the watch going AND is it less than 2 minutes? Yes!
- Increase the seconds
    - If we have made it to 60 seconds, restart the seconds at 0 and increase the minutes
- Update our website to show the new time
- Call the function again
- Wait one second
- ...

It's not a while loop, but it's still looping because the function will keep calling itself over and over and over again until either the watch is stopped or we have made it to 2 minutes.

Add the purple code shown below to your **increaseTime** function:

```
function increaseTime() {
  setTimeout(function() {
    if (isWatchStillGoing && startMinutes < 2) {
      startSeconds = startSeconds + 1;
      if (startSeconds == 60) {
        startSeconds = 0;
        startMinutes = startMinutes + 1;
      }
      var secondsString = "";
      if (startSeconds < 10) {
        secondsString = "0" + startSeconds;
      } else {
        secondsString = secondsString + startSeconds
      }
      var minutesString = "";
      if (startMinutes < 10) {
        minutesString = "0" + startMinutes;
      } else {
        minutesString = minutesString + startMinutes;
      }

      document.getElementById("startMinutes").innerHTML = minutesString;
      document.getElementById("startSeconds").innerHTML = secondsString;
      increaseTime();
    }
  }, 1000)
}
```

If your code is working, you should be able to start and stop your stopwatch using the green and red buttons on your website. Once you hit 2 minutes, you will need to refresh your screen (save your changes first!) to play with it again.

Bonus: Can you change the colors of your buttons? Look for **.startButton** and **.stopButton** in your CSS.

Recap Quiz

- What is HTML?
- What is CSS?
- What is JavaScript?
- What is a variable?
- What is a loop?
- What does the tag do?
- What shows up on our website when we use <img></img>?
- What is a conditional?