JavaScript II

INFO 253A: Front End Web Architecture

Kay Ashaolu

More JavaScript!

- We're now going to go to more loops, data structures, and control flow
- The goal is to provide a foundation of understanding how to express yourself in JavaScript
- This will take time dependent on your experience. That's okay

Conditionals

- Computers execute commands line by line
- But what if you don't want to execute every line
- Have the computer make a decision?

```
function getGenerationalCohort(yearBorn) {

        let generationalCohort = ""

        if (yearBorn > 1900 && yearBorn <= 1926) {
                generationCohort = "GI Generation";
        }
        else if (yearBorn > 1926 && yearBorn <= 1945) {
                generationalCohort = "Silent Generation";
        }
        else if (yearBorn > 1945 && yearBorn <= 1964) {
                generationalCohort = "Baby Boomers";
        }
        else if(yearBorn > 1964 && yearBorn <= 1980) {
                generationalCohort = "Generation X";
        }
        else if(yearBorn > 1980 && yearBorn <= 2001) {
                generationalCohort = "Millennium";
        }

        else if(yearBorn > 2001 && yearBorn < 2020) {
                generationalCohort = "Generation Z";
        }
        else {
                generationalCohort = "Outside of our named generations";
        }

        return generationalCohort;
}

let year = parseInt(prompt("Enter your year of birth"));
let cohort = getGenerationalCohort(year);
alert("The generational cohort of someone born in " +
        year + " is: " + cohort + ".");
```

What did that do?

- We ask the user their year of birth
- We take that value and check to see which generation cohort the user is
- We print it to the screen

For Loops

What are For Loops?

- What computers are good at are doing the same thing over and over again *very fast*.
- With a for loop, we can define how many times we want something to happen over and over again

What are For Loops?

- The key to the for loop is the expression that evaluates to a Boolean (true or false)
- While that Boolean is true, the for loop keeps going
- The moment when that Boolean is false, the loop terminates

Let's use a For Loop!

```
1  function countToX(x) {
2
3          let message = "";
4          for(let i = 0; i <= x; i = i+1) {
5                  message = message + i + " ";
6          }
7
8          return message;
9  }
10
11 let limit = parseInt(prompt("Enter a number"));
12 let output = countToX(limit);
13 alert(output);
```

What did that do?

- We created the variable *message* with an empty string
- We created a loop that will start at 0, and end while it is *less than or equal to* x
- For each iteration we will add 1 to i

What did that do (2)

- So for the first iteration, i = 0, for the next one, i = 1, the next i = 2, and so on
- For each iteration, the expression i <= x is evaluated
- First it figures out if 1 <= 3, and the boolean that results from that (True) tells the loop to keep going

What did that do? (3)

- Within that loop, at each iteration we then added the number i, and then a space to message
- Note that message is getting longer each iteration. Why do think that is?

What did that do? (4)

- Once i becomes less than or equal to x, the for loop terminates
- The function returns the message
- We then ask the user for a number, and pass it to the countToX function, and then print out the output to the console

While Loops

What are While Loops?

- For loops are good at repeating an action over and over again a set amount of times
- But what if we don't know when to stop repeating an action?
- This is a key opportunity to use while loops in

What are While Loops

- A while loop executes as long as a condition is true
- The statements inside the loop should (eventually) make that condition false to end the loop
- Let's start with an action

Let's use a While Loop!

```
let answerQuestion = function() {
        let answer = prompt("What is 4 + 4");

        if (answer == "8") {
                return true;
        }
        else {
                return false;
        }
}

let answer = false;

while (answer != true) {
        answer = answerQuestion();
}

alert("Correct Answer!");
```

What did that do?

- The answerQuestion function asks the user what is the answer to the math question
- If it's correct, return true, if not, return false
- We then execute a while loop that continues until answerQuestion returns true

Arrays and Objects

Arrays and Objects

- Up to now we have been using single variables
- Sometimes you want to store a list of variables
- Perhaps you want to represent something more complicated in code
- You can use Arrays and Objects for this purpose

Let's start with a compound example

```
let student = {
        name: "Kay Ashaolu",
        id: 232324,
        lab_grades: [1, 1, 1],
        assignment_grades: [87, 98, 82]
};

alert(student);
alert(student.name);
alert(student["lab_grades"]);
alert(student.assignment_grades[2]);
```

What did that do?

- We created an object and stored it in the variable "student"
- We then attempted to display the object
- Then we printed the student's name
- Then the student's lab grades
- Then the student's third assignment grade
  - Array indexes start from 0

One more example

```
let student = {};

student.name = prompt("Enter your name");
student.attempts = [];

let answer = false
while(answer != true) {
        value = prompt("What is 8+8?");
        student.attempts.push(value);

        if(value == "16") {
                answer = true;
        }
        else {
                answer = false;
        }
}

alert(student.name + " answers: " + student.attempts);
```

What did that do?

- We created an object (student) where you can store her name and her attempts to the question "What is 8+8?"
- In a while loop, we store the results in an array
- We then print the previous attempts that she made

Questions?