

Loops & Debugging

Week 1: Day 2

Learning Objectives

- Define **for** loop in JS
 - Implement **break** and **continue** in loops
- Analyze stack traces for errors
- Analyze code with Chrome Debugger
 - Use the **debugger** statement in chrome to inspect variable in for loop

What is a loop?

// Printing 'hello world'

```
console.log('hello world');
```

// Printing 'hello world' 3 times

// put the code block in a loop

Setting up a for loop - 4 steps

1. Initial Expression

`let i = 0;`

2. Condition Expression

`i < 3;`

3. Code block

`console.log('hello world');`

4. Increment Expression

`i = i + 1;`

Loop steps 2->3->4
As long as step 2 is true

```
1 for (let i=0; i < 3; i=i+1) {  
2   console.log('hello world');  
3 }
```

```
hello world  
hello world  
hello world
```

Chrome Debugger!

// Open CodePen

// Click 'Start Coding'

// Settings > Behavior > Auto-Updating Preview to 'Off'

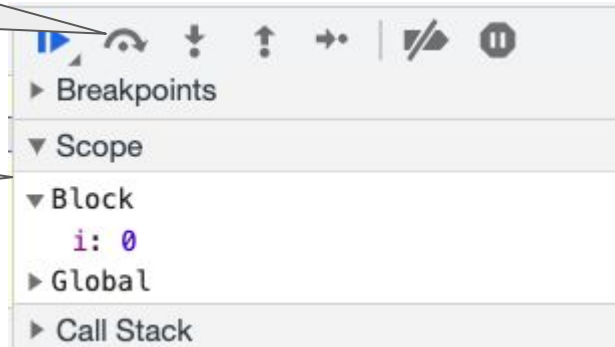
// Click 'View > Developer > Developer tools' ('alt + command + j')

// Write the for loop in 'js' panel, and hit 'Run'

// Add the magic term 'debugger' and hit 'Run' again

Click the 'Step over' button to execute the code one line at a time

Keep your eyes on the 'Scope' contents



For loops to try

// Print odd numbers from 1 to 11

// Intro to 'Remainder' (%)

// quick tip on i++ AND i += 2

// Count down from 10 to 0

// Infinite loop

// While and Do While are similar

Accessing characters in a string

// String variable representation

let letters = 'world';

0	1	2	3	4
w	o	r	l	d

```
console.log(letters[0])
>w
console.log(letters[4])
>d
console.log(letters[5])
>undefined
```

String properties & methods

```
let letters = 'world';
```

```
> letters.length //property
```

```
> 5
```

```
> letters.toUpperCase() //method
```

```
> WORLD
```

```
> letters[2].toUpperCase()
```

```
> R
```

```
> letters[5].toUpperCase()
```

```
> Uncaught TypeError
```


String in a for loop

```
1 let letters = 'world';  
2 for (let i = 0; i < letters.length; i++) {  
3   console.log(letters[i]);  
4 }
```

String in a for loop 2

// Create a string from another string

// by including only alternate characters

```
1 let letters = 'world';
2 let oddLetters = "";
3
4 for (let i = 0; i < letters.length; i++) {
5   if ( i%2 === 0) {
6     oddLetters += letters[i];
7     // oddLetters=oddLetters + letters[i];
8   }
9 }
10 console.log(oddLetters);
```

Continue in a for loop

// the continue keyword will cause the loop to skip to the next iteration

```
1 let letters = 'world';  
2 for (let i = 0; i < letters.length; i++) {  
3   if(letters[i] === 'r') {  
4     continue;  
5   }  
6   console.log(letters[i]);  
7 }
```

Break in a for loop

// the break keyword breaks out of the loop permanently

```
1 let letters = 'world';  
2 for (let i = 0; i < letters.length; i++) {  
3   if(letters[i] === 'r') {  
4     break;  
5   }  
6   console.log(letters[i]);  
7 }
```

Codepen workshop before solving

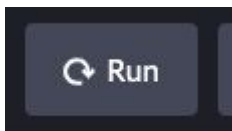
// 1. Read the Readme tab

// 2. Look at the Specs tab

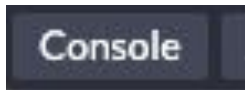
Codepen workshop reading Specs



// each **x** or **●** signals a failed or passed test



// click 'Run' button on top bar to rerun code



// click 'Console' on bottom left to see console output

Codepen workshop Solving


- // 1. Function syntax with name and argument(s)
- // 2. Initialize return variable with default value
- // 3. Return the variable from step 2
- // 4. Write your solution

Debugging - error messages

/* Let's start by considering bugs that come from writing invalid JavaScript code. */

/* The testem page in your browser passes helpful error messages to you if it couldn't run your code as written */

/* This ReferenceError means the the code tried to reference a variable called sum that was never defined */

 **Jasmine** 2.4.1 Options

● × × ×

4 specs, 3 failures finished in 0.013s

Spec List | Failures

onlyOdds returns a number

ReferenceError: sum is not defined


ReferenceError: sum is not defined
at onlyOdds (http://localhost:7357/only-odds.js:11:7)
at Object.it (http://localhost:7357/only-odds.spec.js:8:
at attemptSync (https://cdnjs.cloudflare.com/ajax/libs/

Debugging - error messages

/* Note the stack trace below the error */

/* The first at... line gives the location where the error occurred in '01 Only Odds': it looks like the error happened on line 11. */

/* This line number may not always be accurate, but its often a good place to start */
/* Google unfamiliar errors */

 **Jasmine** 2.4.1 Options

● × × ×

4 specs, 3 failures finished in 0.013s

Spec List | Failures

onlyOdds returns a number

ReferenceError: sum is not defined

ReferenceError: sum is not defined
at onlyOdds (http://localhost:7357/only-odds.js:11:7)
at Object.it (http://localhost:7357/only-odds.spec.js:8:
at attemptSync (https://cdnjs.cloudflare.com/ajax/libs/

Debugging - failing tests

/ When your test is failing, you'll get an output that compares the value your function returned against the expected value. */*

 **Jasmine** 3.0.0



4 specs, 1 failure

Spec List

onlyOdds > returns the sum of all odd nums between the provided argument and 0

Expected 0 to equal 25.

Error: Expected 0 to equal 25.

at <Jasmine>

at UserContext.<anonymous> (<https://cdpn.io/cp/internal/boomboom/pen.js?key=>

Debugging - failing tests

// It can also help to look directly at the code that defines how the test is supposed to work.
// You can see that the test is passing in the number 10 to your function. This can help you debug!

// All of the code inside of the tests, besides the line that starts with expect, is plain-old JavaScript

// Using 'fit'

```
it('returns the sum of all odd nums between the provided  
argument and 0', () => {  
  let returnedValue = onlyOdds(10);  
  expect(returnedValue).toEqual(9 + 7 + 5 + 3 + 1);  
});
```

Recap

- Define **for** loop in JS
 - Implement **break** and **continue** in loops
- Analyze stack traces for errors
- Analyze code with Chrome Debugger
 - Use the **debugger** statement in chrome to inspect variable in for loop