Beginning Programming in JavaScript

Clarissa Littler

April 10, 2016

Dictionaries and Phonebooks

Dictionaries names → defintions

Dictionaries and Phonebooks

Dictionaries names → defintions

Phonebooks names → numbers

Dictionaries and Phonebooks

Dictionaries	names → defintions
Phonebooks	names → numbers
Directory	names → locations

Named Collections in JavaScript

- Objects

Named Collections in JavaScript

- Objects
  - `{name : val, name : val, ...}`

Named Collections in JavaScript

- Objects
  - `{name : val, name : val, ...}`
  - `obj.name`

Named Collections in JavaScript

- Objects
  - `{name : val, name : val, ...}`
  - `obj.name`
  - `obj.name = 10`

Named Collections in JavaScript

- Objects
  - `{name : val, name : val, ...}`
  - `obj.name`
  - `obj.name = 10`
- Sets of *properties* and *values*

Named Collections in JavaScript

- Objects
  - `{name : val, name : val, ...}`
  - `obj.name`
  - `obj.name = 10`
- Sets of *properties* and *values*

Named Collections in JavaScript

```
var obj = {prop1 : 0, prop2 : 1};
console.log(obj.prop1);
console.log(obj.prop2);
console.log(obj.prop3);
```

Making Objects

Quiz

Fill in the ellipses so that the two console.log statements print `true`

```
var obj = {...};
console.log(obj.name === "chicken");
console.log(obj.typeOfAnimal === "dog");
```

Making Objects

Quiz

Fill in the ellipses so that the two console.log statements print `true`

```
var obj = {...};
console.log(obj.name === "chicken");
console.log(obj.typeOfAnimal === "dog");
```

Answer

```
var obj = {name : "chicken", typeOfAnimal : "dog"};
console.log(obj.name === "chicken");
console.log(obj.typeOfAnimal === "dog");
```

Making Objects

Quiz

Fill in the ellipses so that the two console.log statements print `true`

```
var obj = {...};
console.log(obj.name === "chicken");
console.log(obj.typeOfAnimal === "dog");
```

Answer

```
var obj = {name : "chicken", typeOfAnimal : "dog"};
console.log(obj.name === "chicken");
console.log(obj.typeOfAnimal === "dog");
```

Are there any other answers?

Nesting Objects

Objects can contain other objects

Question

Fill in the ellipses so that the console.log statement prints `true`

```
var obj = {name : "Claire", possessions :
     {leftPocket : "A cell phone",
      rightPocket : "60 cents"}};

console.log(... === "60 cents");
```

Nesting Objects

Objects can contain other objects

Question

Fill in the ellipses so that the console.log statement prints `true`

```
var obj = {name : "Claire", possessions :
     {leftPocket : "A cell phone",
      rightPocket : "60 cents"}};

console.log(... === "60 cents");
```

Answer

```
var obj = {name : "Claire", possessions :
     {leftPocket : "A cell phone",
      rightPocket : "60 cents"}};

console.log(obj.posessions.rightPocket === "60 cents");
```

More on Objects

- Objects are ubiquitious in JavaScript

More on Objects

- Objects are ubiquitious in JavaScript
- `console.log`

More on Objects

- Objects are ubiquitious in JavaScript
- `console.log`
- `str.length`

More on Objects

- Objects are ubiquitious in JavaScript
- `console.log`
- `str.length`
- Everything is (basically) an object

Lists in real-life

- Grocery lists

Lists in real-life

- Grocery lists
- Directions

Lists in real-life

- Grocery lists
- Directions
- Bookshelves

Lists in real-life

- Grocery lists
- Directions
- Bookshelves
- Music collections

Arrays in JavaScript

- Lists in JavaScript are called arrays

Arrays in JavaScript

- Lists in JavaScript are called arrays
- `[1,2,3,4]`

Arrays in JavaScript

- Lists in JavaScript are called arrays
- `[1,2,3,4]`
- Is `arr[2]` the 2nd element or 3rd element?

Arrays in JavaScript

- Lists in JavaScript are called arrays
- `[1,2,3,4]`
- Is `arr[2]` the 2nd element or 3rd element?
- `arr.length`

Arrays in JavaScript

- Lists in JavaScript are called arrays
- `[1,2,3,4]`
- Is `arr[2]` the 2nd element or 3rd element?
- `arr.length`
- `arr.slice(1,3)`

Array Exercises

Question

Fill in the ??? so that the following code prints "[2,3]"

```
var arr1 = [1,2,3,4];
console.log(arr1.slice(???,???));
```

Array Exercises

Question

Fill in the ??? so that the following code prints "[2,3]"

```
var arr1 = [1,2,3,4];
console.log(arr1.slice(???,???));
```

Answer

```
var arr1 = [1,2,3,4];
console.log(arr1.slice(1,3);
```

Iteration

- Peel six potatoes

Iteration

- Peel six potatoes
- Take the next three lefts

Iteration

- Peel six potatoes
- Take the next three lefts
- Run for twenty minutes

Iteration

- Peel six potatoes
- Take the next three lefts
- Run for twenty minutes
- Performing an action a number of times

For-loops

For-loops/for-statements

For-loops

For-loops/for-statements

```
for(var i=0;i < 10; i = i + 1){
    console.log(i);
}
```

For-loops

```
for(initial expression; condition for ending; next step){
    body
}
```

For-loops

```
for(initial expression; condition for ending; next step){
    body
}
```

- Setup

For-loops

```
for(initial expression; condition for ending; next step){
    body
}
```

- Setup
- How you know when you're done

For-loops

```
for(initial expression; condition for ending; next step){
    body
}
```

- Setup
- How you know when you're done
- The next step to take

For-loop Exercises

Question

Fill in the question marks so the following code only prints even numbers

```
for(var i=0;i < 11; ???){
    console.log(i);
}
```

For-loop Exercises

Question

Fill in the question marks so the following code only prints even numbers

```
for(var i=0;i < 11; ???){
    console.log(i);
}
```

Answer

```
for(var i=0;i < 11; i = i + 2){
    console.log(i);
}
```

For-loop Exercises

Question

Fill in the question marks so the following code prints the numbers from 3-20

```
for(var i=???;i < ???; i = i + 1){
    console.log(i);
}
```

For-loop Exercises

Question

Fill in the question marks so the following code prints the numbers from 3-20

```
for(var i=???;i < ???; i = i + 1){
    console.log(i);
}
```

Answer

```
for(var i=3;i < 21; i = i + 1){
    console.log(i);
}
```

For-loop Exercises

Question

Fill in the question marks so the following code prints the contents of the array an element at a time

```
var myArray = [0,1,2,3,4];
for(var i = 0; i < ???; i = i +1){
    console.log(???);
}
```

For-loop Exercises

Question

Fill in the question marks so the following code prints the contents of the array an element at a time

```
var myArray = [0,1,2,3,4];
for(var i = 0; i < ???; i = i +1){
    console.log(???);
}
```

Answer

```
var myArray = [0,1,2,3,4];
for(var i = 0; i < myArray.length; i = i +1){
    console.log(myArray[i]);
}
```

Iteration

- *Do* this, *until* that

Iteration

- *Do* this, *until* that
- *While* that, *do* this

Iteration

- *Do* this, *until* that
- *While* that, *do* this
- While it's raining, use an umbrella

Iteration

- *Do* this, *until* that
- *While* that, *do* this
- While it's raining, use an umbrella
- Until you reach 750 words, keep typing

While-loops

While-loop/while-statement

While-loops

While-loop/while-statement

```
while (condition) {
body
}
```

While-loops

- For is for a set number of times

While-loops

- For is for a set number of times
    - Or set number of items

While-loops

- For is for a set number of times
  - Or set number of items
- While is for general "loops"

While-example

```
var sum0 = 0;
var sum1 = 1;
while (sum1 < 1000) {
    console.log(sum0);
    var temp = sum1;
    sum1 = sum1 + sum0;
    sum0 = temp;
}
```

While-loop Exercises

Question

Fill in the following code so that it prints all the powers of two less than 1500

```
var sum = 1;
while (???) {
    console.log(sum);
    sum = 2*sum;
}
```

While-loop Exercises

Question

Fill in the following code so that it prints all the powers of two less than 1500

```
var sum = 1;
while (???) {
    console.log(sum);
    sum = 2*sum;
}
```

Answer

```
var sum = 1;
while (sum < 1500) {
    console.log(sum);
    sum = 2*sum;
}
```

Defining Functions

- Function expression

Defining Functions

- Function expression
  - `function (arg1, arg2, ..) { body }`

Defining Functions

- Function expression
  - `function (arg1, arg2, ..)  { body }`
  - `return` says *stop*, *exit*, give back a value

Defining Functions

- Function expression
    - `function (arg1, arg2, ..) { body }`
    - `return` says *stop*, *exit*, give back a value
    - with no `return`, function gives back `undefined`

Defining Functions

- Function expression
  - `function (arg1, arg2, ..)  { body }`
  - `return` says *stop*, *exit*, give back a value
  - with no `return`, function gives back `undefined`
    - like `console.log`

Defining Functions

- Function expression
  - `function (arg1, arg2, ..) { body }`
  - `return` says *stop*, *exit*, give back a value
  - with no `return`, function gives back `undefined`
    - like `console.log`
  - Assign to variables and properties

Defining Functions

```
var adding = function (x,y) {return x + y;};
console.log(adding(1,2));
var myObj = {};
myObj.adder = adding;
console.log(myObj.adder(1,2));
```

Function Exercises

Will the following code print 1 or 2?

```
var myVar = 0;
var fun = function () {
    myVar = myVar + 1;
    return;
    myVar = myVar + 1;
};
fun();
console.log(myVar);
```

Function Exercises

Will the following code print 1 or 2?

```
var myVar = 0;
var fun = function () {
    myVar = myVar + 1;
    return;
    myVar = myVar + 1;
};
fun();
console.log(myVar);
```

It prints 1. Why?

Function Exercise

Question

Fill in the following code to make a function that will return 0 if the argument is less than 0, and 1 if the argument is greater than or equal to 0.

```
var compare = function (x) {
    if (???) {
       ???
    }
    else {
       ???
    }
}
```

Function Exercise

Answer

```
var compare = function (x) {
    if (x < 0) {
return 0;
    }
    else {
return 1;
    }
}
```

What Happens When You Visit a Webpage

- You enter a URL

What Happens When You Visit a Webpage

- You enter a URL
- The DNS system finds the server

What Happens When You Visit a Webpage

- You enter a URL
- The DNS system finds the server
- Your browser makes an HTTP request

What Happens When You Visit a Webpage

- You enter a URL
- The DNS system finds the server
- Your browser makes an HTTP request
- The server processes the request

What Happens When You Visit a Webpage

- You enter a URL
- The DNS system finds the server
- Your browser makes an HTTP request
- The server processes the request
- The server sends a response

What Happens When You Visit a Webpage

- You enter a URL
- The DNS system finds the server
- Your browser makes an HTTP request
- The server processes the request
- The server sends a response
- Your browser receives the response

HTTP

- GET

HTTP

- `GET`
- `POST`

HTTP

- `GET`
- `POST`
- `PUT`

HTTP

- GET
- POST
- PUT
- DELETE

The Client

- Sends the request

The Client

- Sends the request
- Receives

The Client

- Sends the request
- Receives
    - HTML

The Client

- Sends the request
- Receives
  - HTML
  - CSS

The Client

- Sends the request
- Receives
  - HTML
  - CSS
  - JavaScript

The Client

- Sends the request
- Receives
  - HTML
  - CSS
  - JavaScript
- Renders the webpage

Where Javascript fits in

- View source

Where Javascript fits in

- View source
- The JavaScript code changes the page

Where Javascript fits in

- View source
- The JavaScript code changes the page
- Listens for events:

Where Javascript fits in

- View source
- The JavaScript code changes the page
- Listens for events:
    - keyboard input

Where Javascript fits in

- View source
- The JavaScript code changes the page
- Listens for events:
  - keyboard input
  - mouse movement

Where Javascript fits in

- View source
- The JavaScript code changes the page
- Listens for events:
  - keyboard input
  - mouse movement
  - mouse clicks

Where Javascript fits in

- View source
- The JavaScript code changes the page
- Listens for events:
  - keyboard input
  - mouse movement
  - mouse clicks
- `https://www.tinkercad.com/`

Code Academy

- More time for practice

Code Academy

- More time for practice
- I'll answer any questions you have