Take 2

JavaScript 104

Objects & DOM Part 1

Describing the real world with a program

Often times we want to describe something with a program.

```
let name1 = "Alice";
let age1 = 25;
let job1 = "Teacher";

let name2 = "Bob";
let age2 = 26;
let job2 = "Sparky";
```

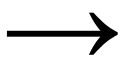
What do these six variables describe?

- A person named Alice, 25 years old, she's a teacher.
- A person named Bob, 26 years old, he's a sparky.

We can use an `Object` to group related information.

How many variables are declared on the left vs on the right?

```
let name = "Alice";
let age = 25;
let job = "Teacher";
```



```
let person = {
  name: 'Alice',
  age: 25,
  job: 'Teacher'
}
```

Three variables

One variable

Objects

- `{ }` were used before to open and close a block of code
- They can also be used to make an `Object` in JavaScript
- Objects can have 0 or more properties in the form `<key>: <value>`

```
let <identifier> = { <properties> }
let person = { name: "John", age: 100 };
```

Example Objects

```
let person = { name: "John" };

let car = { model: "Kia Rio", year: 1995, isNice: false };

// Nicely formatted but same meaning:
let car = {
  model: "Kia Rio",
  year: 1995,
  isNice: false
};
```

```
let transportOption1 = {
  name: 'Bus',
  cost: 2.60
};
let transportOption2 = {
  name: 'Uber',
  cost: 10.45
}
```

The value type is `object` (as opposed to `string`, `number`, `boolean`, etc.)

- `identifier`: a name for the variable which stores the object
- `key`: a name for a variable *inside* the object
- `value`: a value for a variable *inside* the object

Accessing objects

```
let name = "Alice";
name = "Bob";
console.log(name); // Prints 'Bob'

let person = {
  name: "Bob"
};
person.name = "Carol";
console.log(person.name); // Prints 'Carol'
person["name"] = "David"
console.log(person["name"]); // Prints 'David'
```

Let’s write some objects together

Objects can have functions stored in their keys

- `this` refers to the object itself while we're inside it

```
let person = {
  name: "Alice",
  greet: function() {
    console.log("Hi, my name is " + this.name);
  }
}
```

```
console.log(person.name); // Prints 'Alice'
console.log(person.greet); // Prints 'function'
console.log(person.greet()); // Prints 'Hi, ...'
```

Built-in objects and their properties & functions

```
let name = "Alice";

console.log(name); // 'Alice'
console.log(typeof name); // 'string'
```

Strings are also objects

```
console.log(name.length); // 5
console.log(name.toUpperCase()); // 'ALICE'
console.log(name.startsWith("A")); // true
console.log(name.startsWith("B")); // false
console.log(name.toLowerCase().startsWith("A")); // false
```

Numbers are also objects

```
let someYear = 1999;

console.log(someYear); // Prints 1999
console.log(someYear + 10); // Prints 2009
console.log(someYear.toString()); // Prints '1999'
console.log(someYear.toString() + 10); // Prints '199910'
```

Can we use an object to describe a HTML element?

- People have name, age, job, ...
- Which properties do HTML elements have ?

```
<div class="profile" style="color: red">A</div>
```

```
let div = {
  class: 'profile',
  style: 'color: red',
  innerText: 'A',
  id: null,
  ...
}
```

The Document Object Model (DOM)

- The website our browser is showing is called the `document`.
- We can access all the elements on this website using JavaScript.
- Because JS gives us access to every HTML element by means of a JS object, this is called the `Document Object Model` or `DOM`.
- `document` is a JavaScript `object` that we can use to change the elements on the website

Find and print a `<div>` from JS

```
<div id="header" class="profile" style="color: red">
  HELLO
</div>
<div id="footer" class="profile" style="color: red">
  BYE
</div>
```

```
let theDiv = document.getElementById('header');
console.log(theDiv.innerText); // Prints 'HELLO'
console.dir(theDiv); // Print all properties
```

Changing an HTML element's content

```
<div id="header" class="profile" style="color: red">
  HELLO
</div>
```

```
let theDiv = document.getElementById('header');
theDiv.innerText = "GOODBYE";
```

```
<div id="header" class="profile" style="color: red">
  GOODBYE
</div>
```

Getting an `input` field's value

```
<input type="text" id="name" placeholder="Enter your name"/>
```

```
let name = document.getElementById('name').value;
console.log(name); // Prints the name from the input field
```

Event handling

Often times we want to run JavaScript based on an event, like

- "when the user clicks this button, then..."
- "when the user enters their name here, then..."
- "when the user hovers over this `div`, then..."

The solutions for these are called 'event handlers'.

onClick handlers

```
<script>
  function updateDiv() {
    document.getElementById('test').innerText = 'After';
  }
</script>
<div id="test">Before</div>
<button onClick="updateDiv()">Update</button>
```

onClick handlers

```
<div id="test">Before</div>
<button id="my-button">Update</button>
<script>
  function updateDiv() {
    document.getElementById('test').innerText = 'After';
  }
  document.getElementById('my-button')
    .addEventListener('click', updateDiv);
</script>
```

onClick handlers

```
<div id="test">Before</div>
<button id="my-button">Update</button>
<script>
  document.getElementById('my-button')
    .addEventListener('click', function() {
      document.getElementById('test').innerText = 'After';
  });
</script>
```

Recap