the shortcut

Fullstack Web Development Tutorial Lesson 5

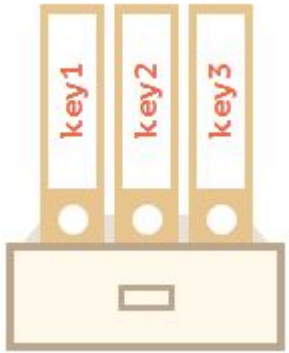
Today's lesson will cover

- Objects
- Git commit and setup Github pages (If we have the time)

JavaScript fundamentals

Objects

- Primitive data types contain only single thing be it string or number or whatever
- In contract, objects store keyed collections of various data
- Object can be created with figure brackets `{…}` with an optional list of *properties*
    - Property is a “key: value” pair, where `key` is a string (also called a “property name”), and `value` can be anything
- Empty object (“empty cabinet”) can be created using one of two syntaxes:
    - `let user = new Object(); // "object constructor" syntax`
    - `let user = {}; // "object literal" syntax`



Imagine an object as a cabinet with signed files

Objects: Literals and properties

- Property has a key (also known as “name” or “identifier”) before the colon `":"` and a value to the right of it.
- From our example, in the `user` object, there are two properties:
  - The first property has the name `"name"` and the value `"John"`.
  - The second one has the name `"age"` and the value `30`.
  - The resulting `user` object can be imagined as a cabinet with two signed files labeled “name” and “age
- We can add, remove and read files from it any time.
- Property values are accessible using the dot notation
- To remove a property, we can use `delete` operator
- We can also use multiword property names, but then they must be quoted
  - For multiword properties, the dot access doesn’t work. You have to use **square bracket notation**
- The last property in the list may end with a comma
- That is called a “trailing” or “hanging” comma. Makes it easier to add/remove/move around properties, because all lines become alike

Exercise: Objects Literals and properties

1. Create an empty object `user`.
2. Add the property `name` with the value `John`.
3. Add the property `surname` with the value `Smith`.
4. Change the value of the `name` to `Pete`.
5. Remove (Delete) the property `name` from the object.

Objects: Computed properties

- Computed Property Names is an ES6 feature which allows the names of object properties in JavaScript object literal notation to be determined dynamically, i.e. computed
- Use square brackets in an object literal, when creating an object
- Square brackets are much more powerful than the dot notation but cumbersome to write

Objects: Miscellaneous characteristics of properties

- Property value shorthand: Property values may derive values from variables. In such case, using shorthand can be handy by assigning only property names without value
  - Instead of `name: name`, can simply write `name`
- Property names limitations: Javascript language reserved terms such as `for, let or return` aren't allowed as variable names but for object property, there's no such restrictions. In short, there are no limitations on property names. They can be any strings or any other type of identifiers but they are automatically converted to strings
- Property existence test, "in" operator: In JavaScript, compared to many other languages, it's possible to access any property even if it doesn't exist. There will be no error if the property doesn't exist!
  - Reading a non-existing property just returns `undefined`
  - Most of the time the comparison with `undefined` works fine. But there's a special case when it fails, but `"in"` works correctly. It's when an object property exists, but stores `undefined`

Objects: The “for...in” loop

- To walk over all keys of an object, there exists a special form of the loop: `for..in`. This is a completely different thing from the `for(;;)` construct that we studied before
- The syntax:
  - ```
    for (key in object) {
     // executes the body for each key among object properties
    }
    ```
- Loop output order: “ordered in a special fashion” - **integer** properties are sorted, others appear in creation order
    ```

Exercise: Objects “for...in” loop

1. We have an object storing salaries of our team:
   - ```
     let salaries = {
     John: 100,
     Ann: 160,
     Pete: 130
     }
     ```
2. Write the code to sum all salaries and store in the variable `sum`. Should be `390` in the example above.
3. If `salaries` is empty, then the result must be `0`.
     ```

Exercise: Manipulate properties with function

1. Create a function `multiplyNumber(obj)` that multiplies all numeric properties of `obj` by `2`.
2. As a result, the original object will have the property values changed the following way:
   - ```
     // before the call
     let navigation = {
     width: 200,
     height: 300,
     title: "My navigation"
     };
     ```
   - ```
     multiplyNumber(navigation);
     // after the call
     navigation = {
     width: 400,
     height: 600,
     title: "My navigation"
     };
     ```
3. Please note that `multiplyNumber` does not need to return anything. It should modify the object in-place.
4. Hint: Use loop to go through properties. Use `typeof` to check for a number here and if property is a number, multiply by 2.
5. console.log the whole object to check if the object has been updated.

Summary

- They store properties (key-value pairs), where:
  - Property keys must be strings or symbols (usually strings).
  - Values can be of any type
- To access a property, we can use:
  - The dot notation: `obj.property`.
  - Square brackets notation `obj["property"]`. Square brackets allow to take the key from a variable, like `obj[varWithKey]`.
- Additional operators:
  - To delete a property: `delete obj.prop`.
  - To check if a property with the given key exists: `"key" in obj`.
  - To iterate over an object: `for (let key in obj)` loop.
- What we've studied today is called a "plain object", or just `Object`.
- There are many other kinds of objects in JavaScript:
  - `Array` to store ordered data collections
  - `Date` to store the information about the date and time
  - `Error` to store the information about an error and so on

the shortcut

Self Study Assignments

To Dos

- Continue freecodecamp Javascript. Ideally finish before we resume after summer.
- Continue with FCC HTML, CSS lessons. Ideally finish all the lessons by end of this month.
- Work on the HTML, CSS assignments to make the projects as complete as you desire and push latest version on Git repository, and use Github pages to have it as a live page
- Please let me know if you would prefer a 1-2-1 session this Friday 12 June, 2020 during training hours