Basics of Javascript Objects

Relevant Links

- Flanagan's book, sections 4.2, 4.4, 6.1.1, 6.2.1
- MDN's guide on objects[1]
- MDN's reference on objects[2]

We will start with simple object literals.

Basics of Javascript Objects

Most typically objects are parts of classes and instantiated by calling the class constructor. We will discuss these objects shortly. For now we will learn some basic methods for working with objects, by looking at "objects as dictionaries".

- For the time being, we will be working with objects as simply key-value pairs. So think of them more like Python's *dictionaries*.
- An **object** is a *dynamic collection of key-value pairs*. The keys are usually called *properties*.
- There are no restrictions on what the values can be. The keys however are *strings* (or in modern Javascript also Symbols).
- Almost everything in Javascript is an object.
- Object literals are enclosed in curly braces: `let a = { foo: 123, "bar": "hello", "properties can be ar` The keys can be written without quotes around them if there is no ambiguity in doing so.
- Two ways to access a property:
  - "Dictionary" access, using a string: `a["foo"]`
  - "Object" access, using dot notation: `a.foo`
- You can also access a property if you have it as a variable value: `let b = "foo"; a[b]; // same as`
- Setting a property: `a["foo"] = 3`, `a.foo = 3`.
- You may delete properties, though this is rare and to be avoided: `delete a.foo`.
- You can chain accesses: `a.even.other`.
- You may use `hasOwnProperty` to determine if an object has a specific property: `a.hasOwnProperty("bar")`
- Two special values of importance: `null`, `undefined`. Both tend to indicate the absence of a value. The difference is that the former of those is an object: `typeof null; typeof undefined;`
- If you try to access a non-existent property, the result is `undefined`. This is very different behavior than in Python.
- You can also set the value of a property to equal `undefined`. This is different from not having that property: `let a = { foo: 5 }; a.bar = undefined; a.hasOwnProperty("bar"); // returns`

[1] https://developer.mozilla.org/en-US/docs/Web/JavaScript/Guide/Working_with_Objects
[2] https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/Object

Basic object/dictionary tasks

Here are some key tasks with dictionaries and how to carry them out:

Creating an object

```
let o = { key1: val1, key2: val2, ... };
```

Setting an object key

```
o[keyString] = newValue;
o.key = newValue;
```

Accessing a key value

```
o[keyString];
o.key;
```

Checking if a key exists

```
o.hasOwnProperty(keyString);
```

Traversing all keys-value pairs

```
// Approach 1
for (let key in o) {
    if (o.hasOwnProperty(key)) {       // <--- Must do this check!
      let value = o[key];              // Declaration should be earlier
      // Do things with key, value
    }
}
// Approach 2
let keys = Object.keys(o);      // <-- Returns array of keys
for (let i = 0; i < keys.length; i += 1) {
  let key = keys[i];            // Declaration should be earlier
  let value = o[key];           // Declaration should be earlier
  // Do things with key, value
}
```