Basics of Javascript Objects (Dictionaries)

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 for now.

Basics of Javascript Objects

- 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*.
- Almost everything in Javascript is an object.
- Object literals are enclosed in curly braces:

```
var a = {
    foo: 123,
    "bar": "hello",
    "properties can be any string": "values can be anything",
    even: { other: "objects" }
};
```

  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:

```
var b = "foo";
a[b];               // same as a["foo"]
```

- Setting a property: a["foo"] = 3, a.foo = 3.
- You may delete properties, though this is rare and to be avoided: delete a.foo.

[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

- 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:

```
var a = { foo: 5 };
a.bar = undefined;
a.hasOwnProperty("bar");     // returns true
```

Basic object/dictionary tasks

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

Creating an object `var o = { key1: val1, key2: val2, ... };`

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

Setting an object key **Accessing a key value** `o[keyString];`

```
o.key;

o.hasOwnProperty(keyString);
```

Checking if a key exists **Traversing all keys-value pairs**

```
// Approach 1
for (var key in o) {
    if (o.hasOwnProperty(key)) {       // <--- Must do this check!
      var value = o[key];              // Declaration should be earlier
      // Do things with key, value
    }
}
// Approach 2
var keys = Object.keys(o);      // <-- Returns array of keys
for (var i = 0; i < keys.length; i += 1) {
  var key = keys[i];            // Declaration should be earlier
  var value = o[key];           // Declaration should be earlier
  // Do things with key, value
}
// Approach 3
Object.keys.forEach(function(key) {
  var value = o[key];           // Declaration is OK, inside a function
  // do things with key, value
});
```