Getters & setters

Virtual property

In addition to the normal property that we have for objects, we can also set up virtual properties. They appears to the outside code as if they were the object's property but they are actually implemented as functions underneath.

```
let user = {
  name: "John",
    last: "Smith",

    get fullName() {
    return this.name + " " + this.last;
  }

   set fullName(value) {
     [this.name, this.last] = value.split(" ");
    }
};

console.log(user.fullName); // John Smith
user.fullName = "Ricky Lu"
console.log(user.fullName); // Ricky Lu
```

Like mentioned before to the outside, `fullName` looks like a property but underneath it is implemented as an method. There is getter that uses the `get` keyword before the function name to provide a virtual attribute that you can read from, and there is `set` keyword to provide a setter that you can assign the virtual attribute.

> For an object there is two type of property, an accessor property which is a virtual property that has either `get/set (or both)` method, or a data property that is just the normal property which we have been dealing with.

Property flags and descriptors

For each property that an object has besides the value it contains it also have some additional metadata. Namely, they are called property flags

- `writable`: If `true`, the value can be changed, otherwise it's read-only. If you attempt to assign to non-writable property then error will only show up in strict mode
- `enumerable`: If `true`, the value will be listed in loops, otherwise it is not listed
- `configurable`: If `true`, the property can be deleted and these attributes can be modified, otherwise, it cannot be deleted or modified anymore. If this flag is set the only flag changing operation that is permitted afterward is to turn `writable` to be `true -> false` to add another layer of protection
- `value`: Describes the value of the property

Object.getOwnPropertyDescriptor(obj, propertyName)

You can use this method to get the property descriptor of a specific property to see which property flag is set and which one isn't

Object.defineProperty(obj, propertyName, descriptor)

You can use this method to set a specific property's property flag, if it is defined in the object then it will just update the flag, otherwise, if the property doesn't exist then it will create the property with the given value and flags. If the flags aren't supplied it is assumed to be all `false`.

Object.defineProperties

You can define many properties at once instead of just one at a time

---

Revision #1
Created 26 December 2022 17:54:23 by Tamarine
Updated 26 December 2022 22:56:43 by Tamarine