the shortcut

Fullstack Web Development Tutorial Lesson 14

Today's lesson will cover

- Classes

JavaScript fundamentals

Class basics

- In JavaScript, a class is a kind of function not an entirely new language-level entity
- Basic syntax
  - ```
    class MyClass {
     // class methods
     constructor() { ... }
     method1() { ... }
     method2() { ... }
     method3() { ... }
     ...
    }
    ```
- What `class User {...}` construct really does is:
  - Creates a function named `User`, that becomes the result of the class declaration. The function code is taken from the `constructor` method (assumed empty if we don't write such method).
  - Stores class methods, such as `sayHi`, in `User.prototype`.
  - After `new User` object is created, when we call its method, it's taken from the prototype, just as described in the chapter F.prototype. So the object has access to class methods.
- Not just a syntactic sugar even though the same can be declared without `class` keyword
    ```

Class basics *(Contd.)*

- **Class expressions:** Just like functions, classes can be defined inside another expression, passed around, returned, assigned, etc.
- **Getters/Setters:** Classes may include getters/setters, computed properties etc.
- **Computed names:** Computed method name can be used using brackets `[...]`.
- **Class fields:** “Class fields” is a syntax that allows to add any properties.

Exercise

- The `Clock` class is written in functional style. Rewrite it using the “class” syntax.

```
function Clock({ template }) {

  let timer;

  function render() {
    let date = new Date();

    let hours = date.getHours();
    if (hours < 10) hours = '0' + hours;

    let mins = date.getMinutes();
    if (mins < 10) mins = '0' + mins;

    let secs = date.getSeconds();
    if (secs < 10) secs = '0' + secs;

    let output = template
      .replace('h', hours)
      .replace('m', mins)
      .replace('s', secs);

    console.log(output);
  }

  this.stop = function() {
    clearInterval(timer);
  };

  this.start = function() {
    render();
    timer = setInterval(render, 1000);
  };

}

let clock = new Clock({template: 'h:m:s'});
clock.start();
```

Class inheritance

- Class inheritance is a way for one class to extend another class.
- The “extends” keyword: Class syntax allows to specify not just a class, but any expression after `extends`.
- Overriding a method: Classes provide `"super"` keyword for that.
  - `super.method(...)` to call a parent method.
  - `super(...)` to call a parent constructor (inside our constructor only).
- Overriding constructor: If a class extends another class and has no `constructor`, then the following “empty” `constructor` is generated: Derived constructor’s internal property label affects its behavior with `new`.
  - When a regular function is executed with `new`, it creates an empty object and assigns it to `this`.
  - But when a derived constructor runs, it doesn’t do this. It expects the parent constructor to do this job.
  - So a derived constructor must call `super` in order to execute its parent (non-derived) constructor, otherwise the object for `this` won’t be created. And we’ll get an error.

Static properties and methods

- We can also assign a method to the class function itself, not to its `"prototype"`. Such methods are called *static*, prepended by `static` keyword.
- Static properties are also possible, they look like regular class properties, but prepended by `static`.
- Static properties and methods are inherited.

Exercise

- Here's the code with `Rabbit` extending `Animal`.
- Unfortunately, `Rabbit` objects can't be created. What's wrong? Fix it.

```
class Animal {
      constructor(name) {
      this.name = name;
      }

}

class Rabbit extends Animal {
      constructor(name) {
      this.name = name;
      this.created = Date.now();
      }
}

let rabbit = new Rabbit("White Rabbit");
console.log(rabbit.name);
```

the shortcut

Self Study Assignments

To Dos

- Continue freecodecamp (FCC) Javascript. Ideally finish before we resume after summer.
- Continue with FCC HTML, CSS lessons. Ideally finish all the lessons by end of this month.
- If you believe FCC exercises aren’t the best for you if you are quite advanced already, please start working on your own project and reach out to mentors for help if needed.