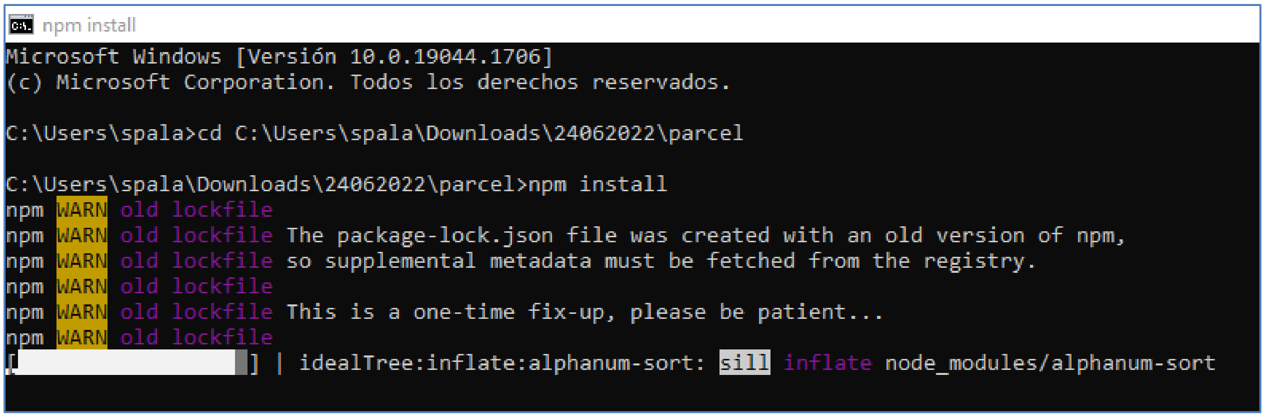
**WDD 330 Web Frontend Development II**
**Samuel Palacios**
**Week 09**
**June 13th, 2022**

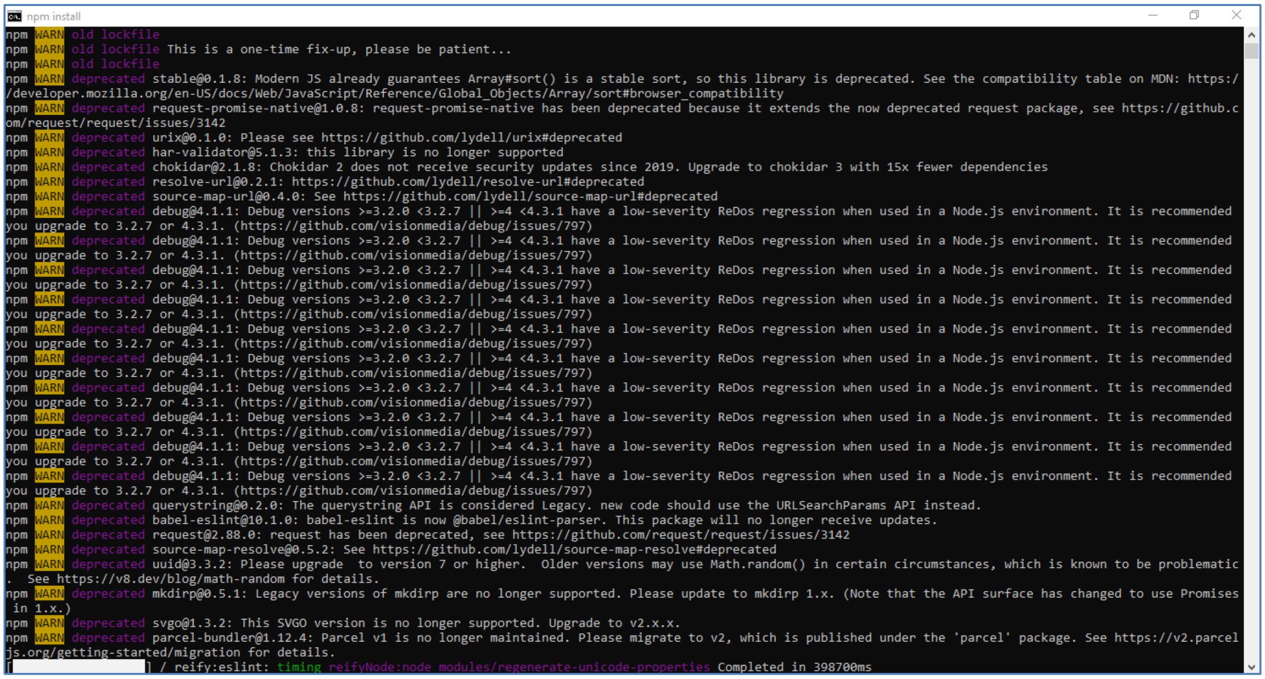
## Readings Assignment Notes

### 1. Managing the modern front-end workflow

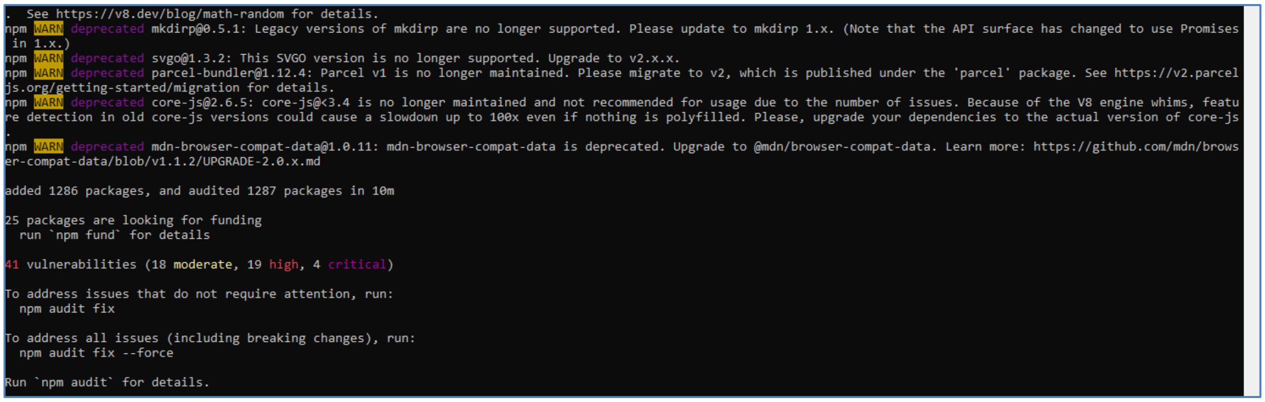
- ✓ npm install



```
Microsoft Windows [Versión 10.0.19044.1706]
(c) Microsoft Corporation. Todos los derechos reservados.

C:\Users\spala>cd C:\Users\spala\Downloads\24062022\parcel

C:\Users\spala\Downloads\24062022\parcel>npm install
npm WARN old lockfile
npm WARN old lockfile The package-lock.json file was created with an old version of npm,
npm WARN old lockfile so supplemental metadata must be fetched from the registry.
npm WARN old lockfile
npm WARN old lockfile This is a one-time fix-up, please be patient...
npm WARN old lockfile
[                  ] | idealTree:inflate:alphanum-sort: sill inflate node_modules/alphanum-sort
```

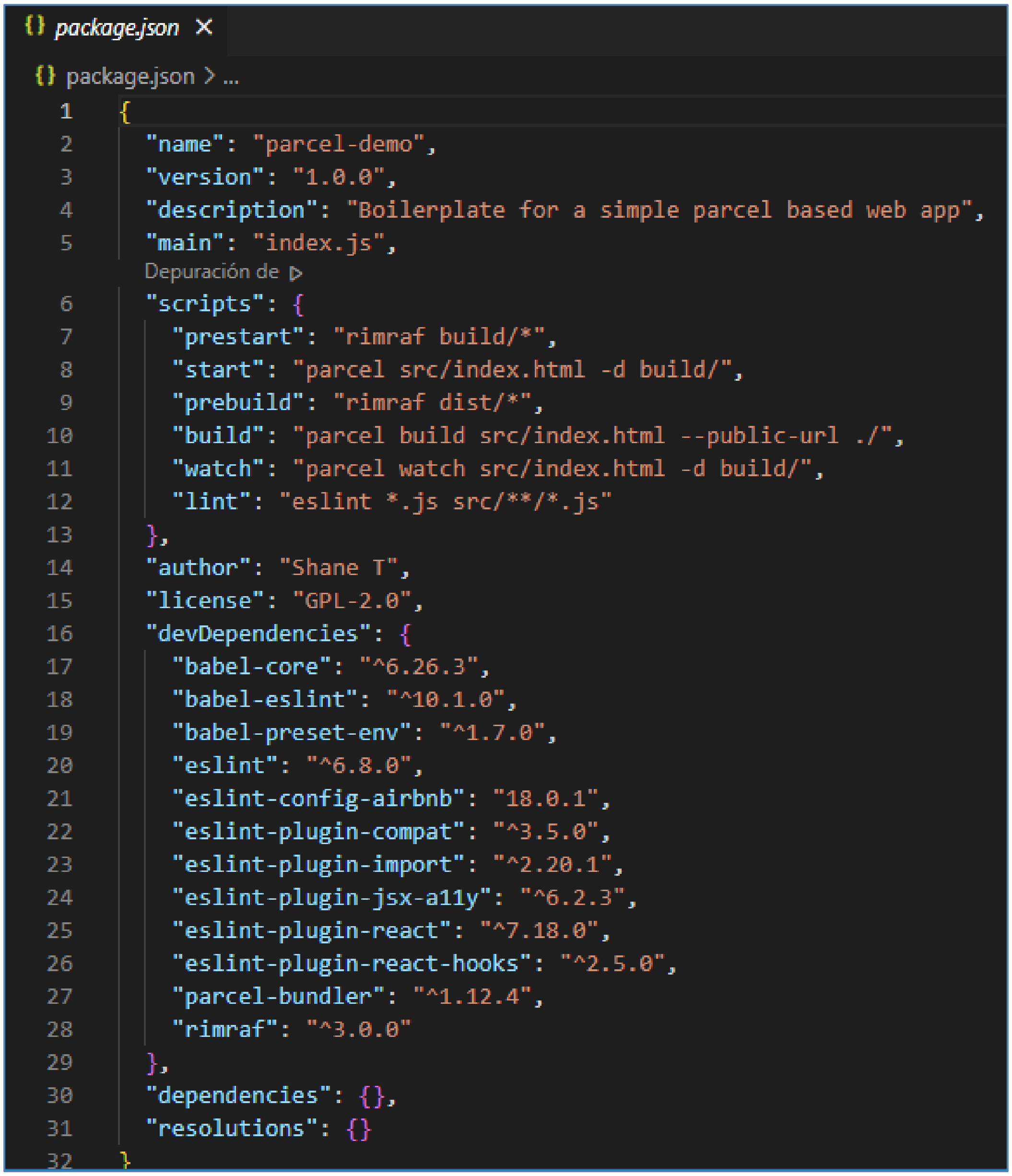


```
npm WARN old lockfile
npm WARN old lockfile This is a one-time fix-up, please be patient...
npm WARN old lockfile
npm WARN deprecated stable@0.1.8: Modern JS already guarantees Array#sort() is a stable sort, so this library is deprecated. See the compatibility table on MDN: https:/
/developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/Array/sort#browser_compatibility
npm WARN deprecated request-promise-native@1.0.8: request-promise-native has been deprecated because it extends the now deprecated request package, see https://github.c
om/request/request/issues/3142
npm WARN deprecated urix@0.1.0: Please see https://github.com/lydell/urix#deprecated
npm WARN deprecated har-validator@5.1.3: this library is no longer supported
npm WARN deprecated chokidar@2.1.8: Chokidar 2 does not receive security updates since 2019. Upgrade to chokidar 3 with 15x fewer dependencies
npm WARN deprecated resolve-url@0.2.1: https://github.com/lydell/resolve-url#deprecated
npm WARN deprecated source-map-url@0.4.0: See https://github.com/lydell/source-map-url#deprecated
npm WARN deprecated debug@4.1.1: Debug versions >=3.2.0 <3.2.7 || >=4 <4.3.1 have a low-severity ReDos regression when used in a Node.js environment. It is recommended
you upgrade to 3.2.7 or 4.3.1. (https://github.com/visionmedia/debug/issues/797)
npm WARN deprecated debug@4.1.1: Debug versions >=3.2.0 <3.2.7 || >=4 <4.3.1 have a low-severity ReDos regression when used in a Node.js environment. It is recommended
you upgrade to 3.2.7 or 4.3.1. (https://github.com/visionmedia/debug/issues/797)
npm WARN deprecated debug@4.1.1: Debug versions >=3.2.0 <3.2.7 || >=4 <4.3.1 have a low-severity ReDos regression when used in a Node.js environment. It is recommended
you upgrade to 3.2.7 or 4.3.1. (https://github.com/visionmedia/debug/issues/797)
npm WARN deprecated debug@4.1.1: Debug versions >=3.2.0 <3.2.7 || >=4 <4.3.1 have a low-severity ReDos regression when used in a Node.js environment. It is recommended
you upgrade to 3.2.7 or 4.3.1. (https://github.com/visionmedia/debug/issues/797)
npm WARN deprecated debug@4.1.1: Debug versions >=3.2.0 <3.2.7 || >=4 <4.3.1 have a low-severity ReDos regression when used in a Node.js environment. It is recommended
you upgrade to 3.2.7 or 4.3.1. (https://github.com/visionmedia/debug/issues/797)
npm WARN deprecated debug@4.1.1: Debug versions >=3.2.0 <3.2.7 || >=4 <4.3.1 have a low-severity ReDos regression when used in a Node.js environment. It is recommended
you upgrade to 3.2.7 or 4.3.1. (https://github.com/visionmedia/debug/issues/797)
npm WARN deprecated debug@4.1.1: Debug versions >=3.2.0 <3.2.7 || >=4 <4.3.1 have a low-severity ReDos regression when used in a Node.js environment. It is recommended
you upgrade to 3.2.7 or 4.3.1. (https://github.com/visionmedia/debug/issues/797)
npm WARN deprecated debug@4.1.1: Debug versions >=3.2.0 <3.2.7 || >=4 <4.3.1 have a low-severity ReDos regression when used in a Node.js environment. It is recommended
you upgrade to 3.2.7 or 4.3.1. (https://github.com/visionmedia/debug/issues/797)
npm WARN deprecated debug@4.1.1: Debug versions >=3.2.0 <3.2.7 || >=4 <4.3.1 have a low-severity ReDos regression when used in a Node.js environment. It is recommended
you upgrade to 3.2.7 or 4.3.1. (https://github.com/visionmedia/debug/issues/797)
npm WARN deprecated debug@4.1.1: Debug versions >=3.2.0 <3.2.7 || >=4 <4.3.1 have a low-severity ReDos regression when used in a Node.js environment. It is recommended
you upgrade to 3.2.7 or 4.3.1. (https://github.com/visionmedia/debug/issues/797)
npm WARN deprecated querystring@0.2.0: The querystring API is considered Legacy. new code should use the URLSearchParams API instead.
npm WARN deprecated babel-eslint@10.1.0: babel-eslint is now @babel/eslint-parser. This package will no longer receive updates.
npm WARN deprecated request@2.88.0: request has been deprecated, see https://github.com/request/request/issues/3142
npm WARN deprecated source-map-resolve@0.5.2: See https://github.com/lydell/source-map-resolve#deprecated
npm WARN deprecated uuid@3.3.2: Please upgrade  to version 7 or higher.  Older versions may use Math.random() in certain circumstances, which is known to be problematic
.  See https://v8.dev/blog/math-random for details.
npm WARN deprecated mkdirp@0.5.1: Legacy versions of mkdirp are no longer supported. Please update to mkdirp 1.x. (Note that the API surface has changed to use Promises
 in 1.x.)
npm WARN deprecated svgo@1.3.2: This SVGO version is no longer supported. Upgrade to v2.x.x.
npm WARN deprecated parcel-bundler@1.12.4: Parcel v1 is no longer maintained. Please migrate to v2, which is published under the 'parcel' package. See https://v2.parcel
js.org/getting-started/migration for details.
[              ] / reify:eslint: timing reifyNode:node_modules/regenerate-unicode-properties Completed in 398700ms
```



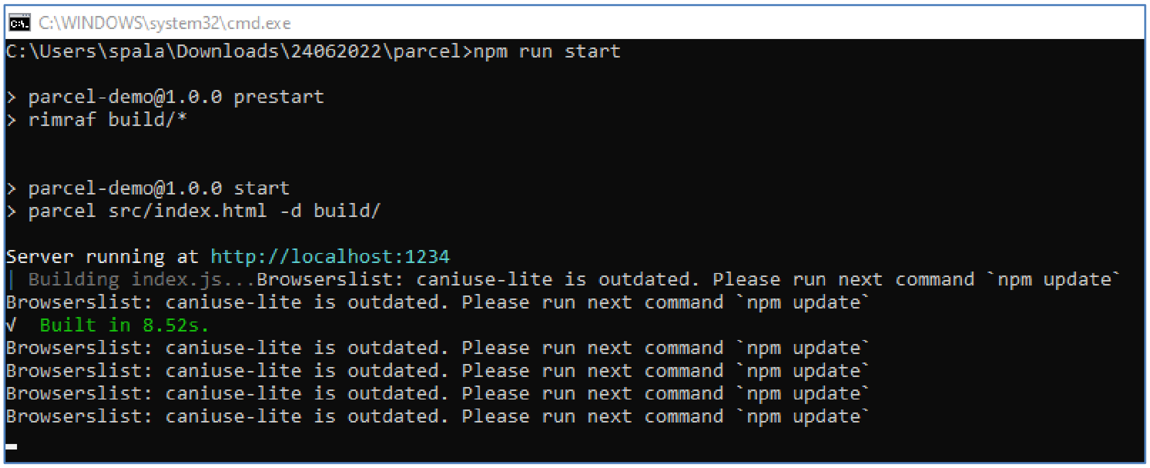
```
.  See https://v8.dev/blog/math-random for details.
npm WARN deprecated mkdirp@0.5.1: Legacy versions of mkdirp are no longer supported. Please update to mkdirp 1.x. (Note that the API surface has changed to use Promises
 in 1.x.)
npm WARN deprecated svgo@1.3.2: This SVGO version is no longer supported. Upgrade to v2.x.x.
npm WARN deprecated parcel-bundler@1.12.4: Parcel v1 is no longer maintained. Please migrate to v2, which is published under the 'parcel' package. See https://v2.parcel
js.org/getting-started/migration for details.
npm WARN deprecated core-js@2.6.5: core-js@<3.4 is no longer maintained and not recommended for usage due to the number of issues. Because of the V8 engine whims, featu
re detection in old core-js versions could cause a slowdown up to 100x even if nothing is polyfilled. Please, upgrade your dependencies to the actual version of core-js
.
npm WARN deprecated mdn-browser-compat-data@1.0.11: mdn-browser-compat-data is deprecated. Upgrade to @mdn/browser-compat-data. Learn more: https://github.com/mdn/brows
er-compat-data/blob/v1.1.2/UPGRADE-2.0.x.md

added 1286 packages, and audited 1287 packages in 10m

25 packages are looking for funding
  run `npm fund` for details

41 vulnerabilities (18 moderate, 19 high, 4 critical)

To address issues that do not require attention, run:
  npm audit fix

To address all issues (including breaking changes), run:
  npm audit fix --force

Run `npm audit` for details.
```

- ✓ package.json



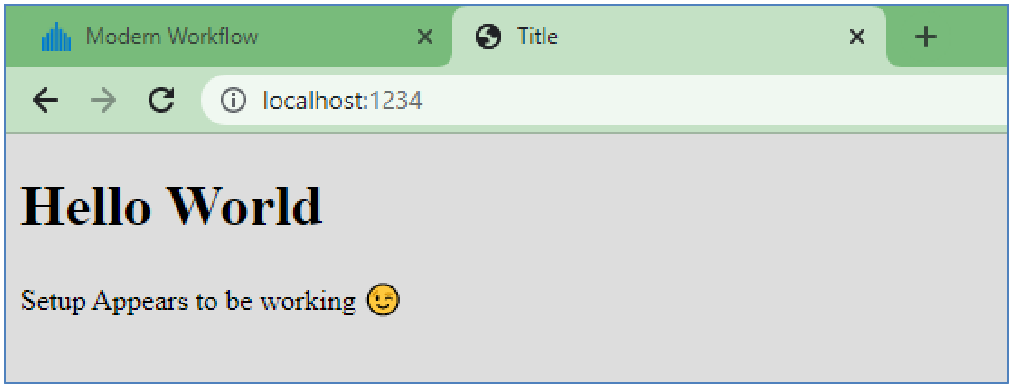
```
{
  "name": "parcel-demo",
  "version": "1.0.0",
  "description": "Boilerplate for a simple parcel based web app",
  "main": "index.js",
  Depuración de ▷
  "scripts": {
    "prestart": "rimraf build/*",
    "start": "parcel src/index.html -d build/",
    "prebuild": "rimraf dist/*",
    "build": "parcel build src/index.html --public-url ./",
    "watch": "parcel watch src/index.html -d build/",
    "lint": "eslint *.js src/**/*.js"
  },
  "author": "Shane T",
  "license": "GPL-2.0",
  "devDependencies": {
    "babel-core": "^6.26.3",
    "babel-eslint": "^10.1.0",
    "babel-preset-env": "^1.7.0",
    "eslint": "^6.8.0",
    "eslint-config-airbnb": "18.0.1",
    "eslint-plugin-compat": "^3.5.0",
    "eslint-plugin-import": "^2.20.1",
    "eslint-plugin-jsx-a11y": "^6.2.3",
    "eslint-plugin-react": "^7.18.0",
    "eslint-plugin-react-hooks": "^2.5.0",
    "parcel-bundler": "^1.12.4",
    "rimraf": "^3.0.0"
  },
  "dependencies": {},
  "resolutions": {}
}
```

- ✓ npm run start



C:\WINDOWS\system32\cmd.exe

```
C:\Users\spala\Downloads\24062022\parcel>npm run start

> parcel-demo@1.0.0 prestart
> rimraf build/*


> parcel-demo@1.0.0 start
> parcel src/index.html -d build/

Server running at http://localhost:1234
| Building index.js...Browserslist: caniuse-lite is outdated. Please run next command `npm update`
Browserslist: caniuse-lite is outdated. Please run next command `npm update`
√  Built in 8.52s.
Browserslist: caniuse-lite is outdated. Please run next command `npm update`
Browserslist: caniuse-lite is outdated. Please run next command `npm update`
Browserslist: caniuse-lite is outdated. Please run next command `npm update`
Browserslist: caniuse-lite is outdated. Please run next command `npm update`
```



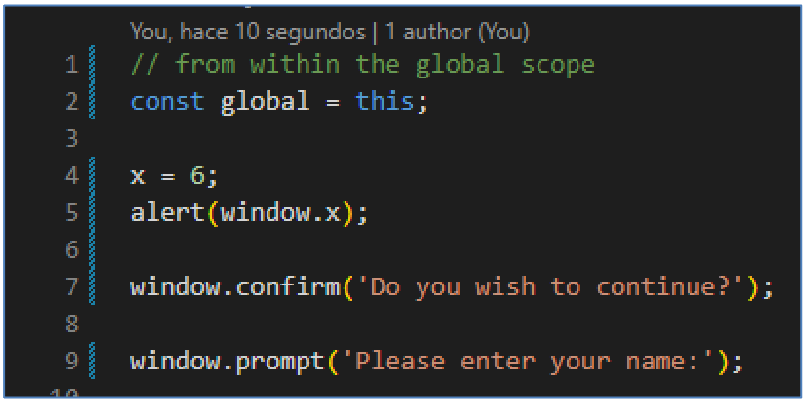
Hello World

Setup Appears to be working 😉

I will try to do it again later because I need to update my “npm” installation.

## 2. Chapter 9: The Window Object

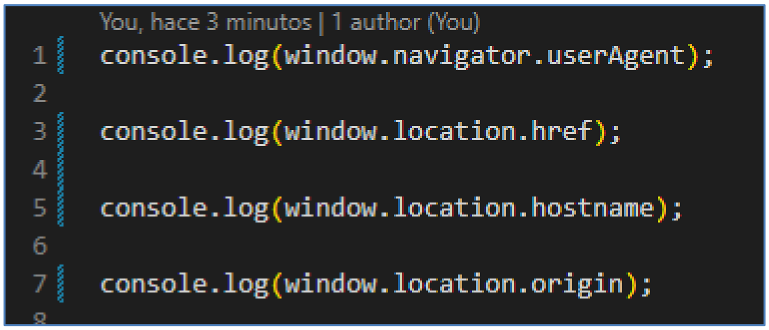
- ✓ The **window object** is quite interesting, it is practically the browser window, and it is important to know the implications of manipulating it, what we can do and what not. We can say that this object is a global object that we can do references anytime and anywhere. It can contain references to variables, dialogs, etc.



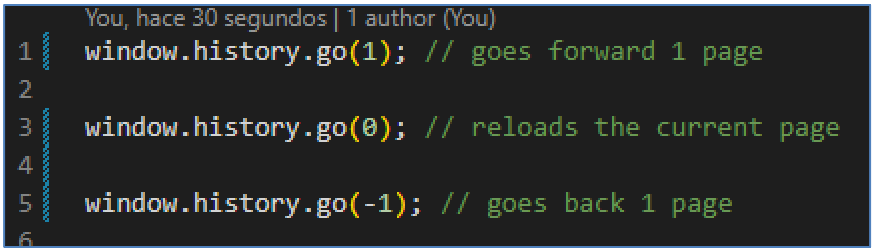
You, hace 10 segundos | 1 author (You)

```
// from within the global scope
const global = this;

x = 6;
alert(window.x);

window.confirm('Do you wish to continue?');

window.prompt('Please enter your name:');
```

- ✓ This **window object** has many other interesting properties, practically any characteristic of the client or browser used by our web application can be known, for example user agent, hostname, origin, etc. It also allows us to manipulate/query data such as **cookies** and **history**.

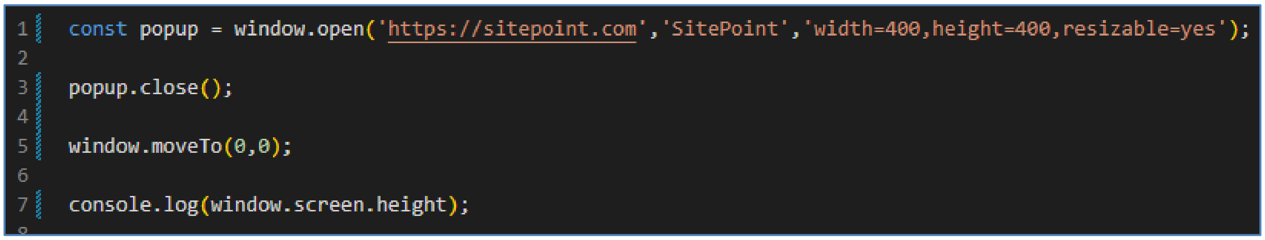


```
You, hace 3 minutos | 1 author (You)
console.log(window.navigator.userAgent);

console.log(window.location.href);

console.log(window.location.hostname);

console.log(window.location.origin);
```



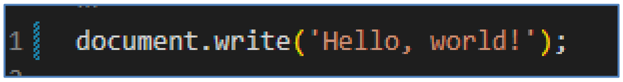
```
You, hace 30 segundos | 1 author (You)
window.history.go(1); // goes forward 1 page

window.history.go(0); // reloads the current page

window.history.go(-1); // goes back 1 page
```

- ✓ This object has a wide use, you can also build/move other child windows or **popups** that allow users to request or display certain information.



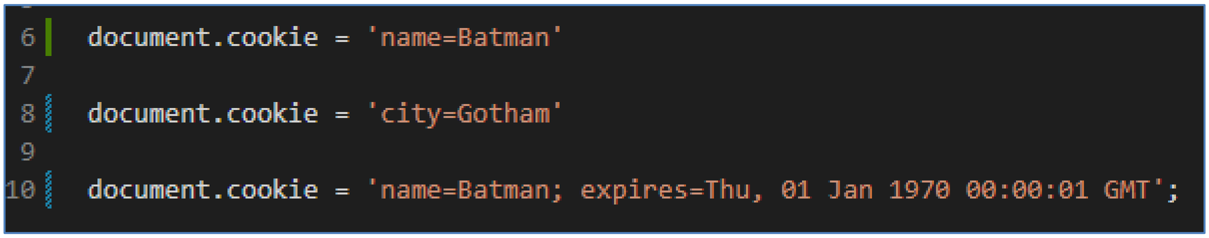
```
const popup = window.open('https://sitepoint.com','SitePoint','width=400,height=400,resizable=yes');

popup.close();

window.moveTo(0,0);

console.log(window.screen.height);
```

- ✓ We can also use the **write function** of this object, which allows you to write directly to the body object of the window. This function, depending on when it is done, overwrites any input or object that may be before, so it is mostly used for testing or debugging.



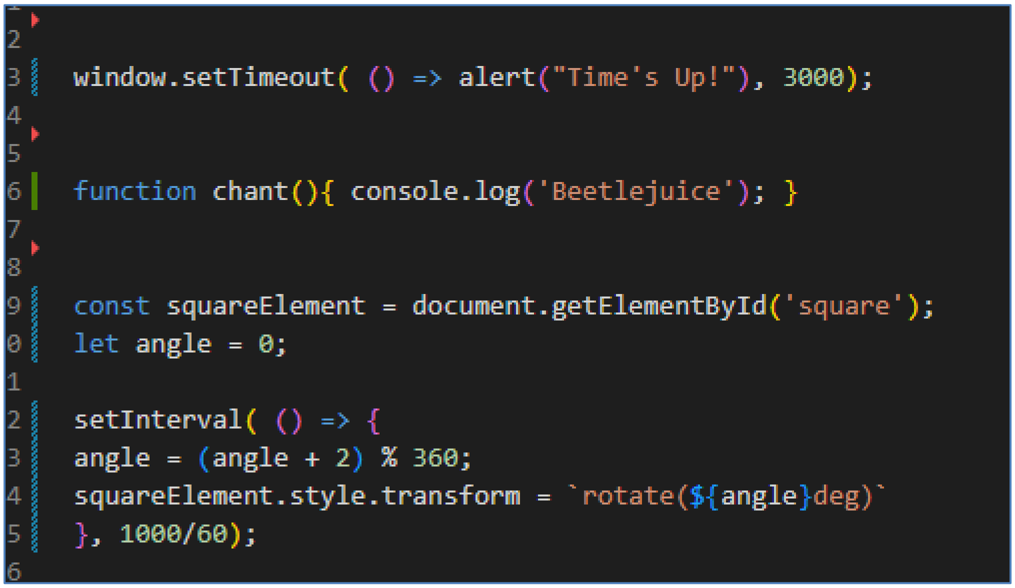
```
document.write('Hello, world!');
```

- ✓ "**Cookies"** is another big topic. These elements are small files or pieces of data that allow you to set a certain condition of a session in a web client, for example when we log into a site... the ID of our session is stored in a cookie called "ID_SESSION". This issue is very important, and depending on the data that the cookie has, security issues must also be validated.



```
document.cookie = 'name=Batman'

document.cookie = 'city=Gotham'

document.cookie = 'name=Batman; expires=Thu, 01 Jan 1970 00:00:01 GMT';
```

- ✓ We can also make **delays** or manage **timers.** The latter allows you to have the famous "call backs", and with these functions you can do all kinds of implementation, including **animations**.



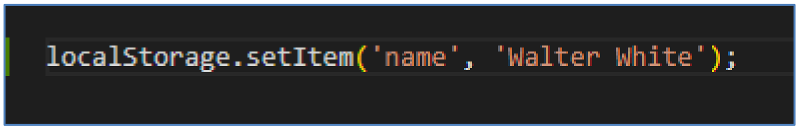
```

window.setTimeout( () => alert("Time's Up!"), 3000);


function chant(){ console.log('Beetlejuice'); }


const squareElement = document.getElementById('square');
let angle = 0;

setInterval( () => {
angle = (angle + 2) % 360;
squareElement.style.transform = `rotate(${angle}deg)`
}, 1000/60);

```

## 3. Chapter 14: HTML5 APIs

- ✓ What is an **API**? An API is a programming interface that abstracts some complexity and simplifies it so that any programmer can simply use it, without worrying about how things happen. It is basic for all types of development to work with API's.

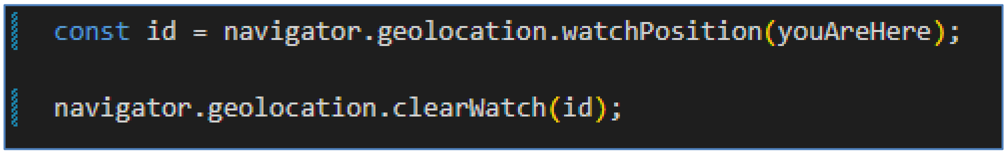
- ✓ **HTML5** is not the exception and allows defining API's that fulfill this purpose. The "data" attribute in this version of HTML is crucial and allows you to embed data of all kinds in a web page.

- ✓ One of the most important APIs in HTML is the **Web Storage** API, which allows using local storage in a simple and transparent way. It allows us to storage both text and binary data on the client browser.



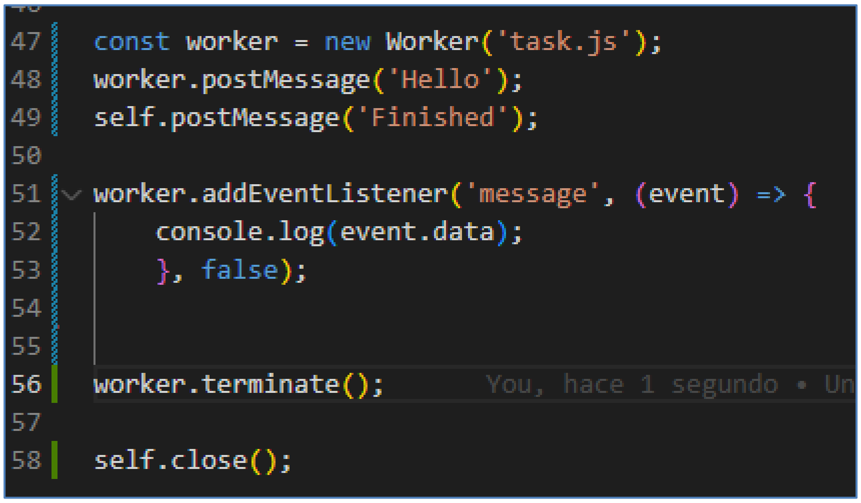
```
localStorage.setItem('name', 'Walter White');
```

- ✓ **Geolocation**, allows you to establish the client's position and even GPS issues that were previously only accessible for native mobile applications, allows a wide variety of applications and options.



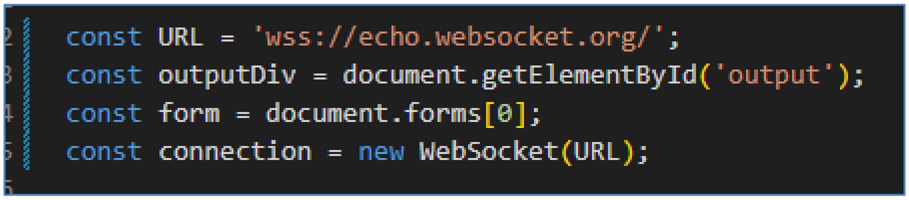
```
const id = navigator.geolocation.watchPosition(youAreHere);

navigator.geolocation.clearWatch(id);
```

- ✓ **Web Workers**, this API has quite a bit of power, it allows you to execute JavaScript functions asynchronously in thread mode. Previously we worked with the "async" directive and with callbacks, but this API allows us to have a more comprehensive and better structured approach. It is like a threads manager. Also, this API allows you to distribute and publish functions through Shared Web Workers and Service Workers, that is, as services.



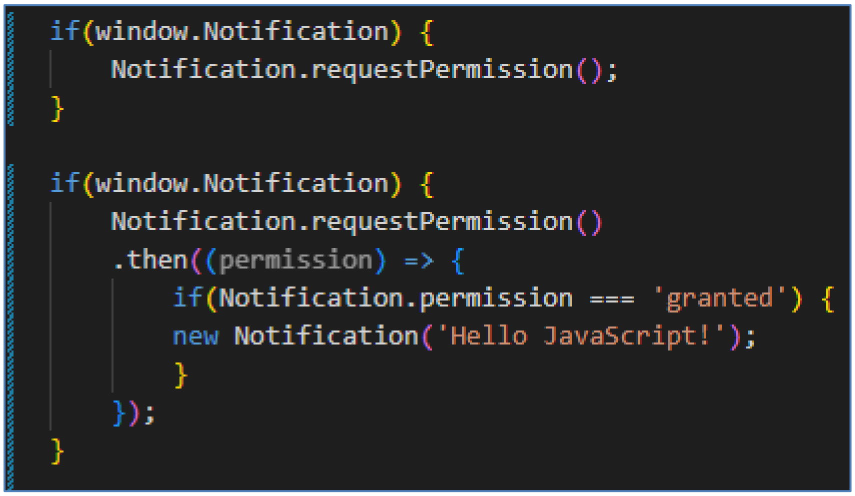
```
const worker = new Worker('task.js');
worker.postMessage('Hello');
self.postMessage('Finished');

worker.addEventListener('message', (event) => {
    console.log(event.data);
    }, false);


worker.terminate();

self.close();
```

- ✓ **Web Sockets**, another great API. This allows establishing point-to-point communication via HTTP, with which dynamic and real-time applications such as chat can be made. Unlike a client-server application where everything happens in ports, here it is done at the level of dedicated web channels that are managed by a server but allow point-to-point or client-to-client communication.



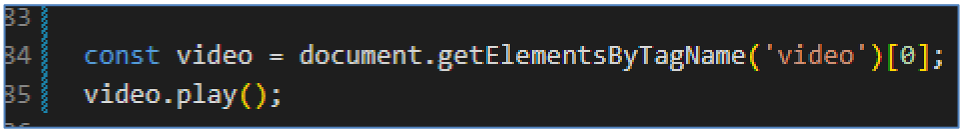
```
const URL = 'wss://echo.websocket.org/';
const outputDiv = document.getElementById('output');
const form = document.forms[0];
const connection = new WebSocket(URL);
```

- ✓ **Notifications**, another great API that has allowed a web app to become a progressive web app. This allows you to manage notifications through the web browser, it is a widely used API that allows users to notify users of news on a website, regardless of what type of device they have, computer or mobile.



```
if(window.Notification) {
    Notification.requestPermission();
}

if(window.Notification) {
    Notification.requestPermission()
    .then((permission) => {
        if(Notification.permission === 'granted') {
        new Notification('Hello JavaScript!');
        }
    });
}
```

✓ **Multimedia** is another great advance of HTML5. Before HTML5 it was necessary to have a third-party API based on flash or JS to have multimedia (audio, sound, music, video) in the browser, today with this API it is extremely easy to present sound, music and videos, and you need nothing more than a browser that is not so outdated.



```

const video = document.getElementsByTagName('video')[0];
video.play();
```