Making things click together - real life project development

Today you are going to learn about:

- Javascript environments
- Project structure, clean code
- Tooling - npm, yarn, webpack, git and etc
- Code standards - linters, readable code
- Practical exercise
- What to do before development

Two environments - One Javascript

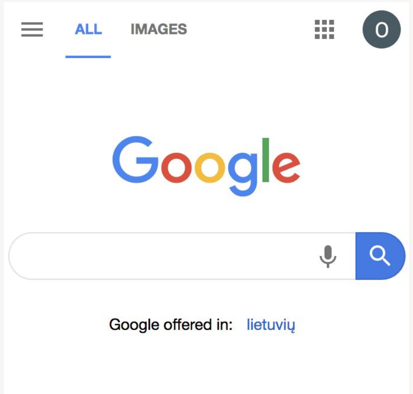
Browser environment

Node environment

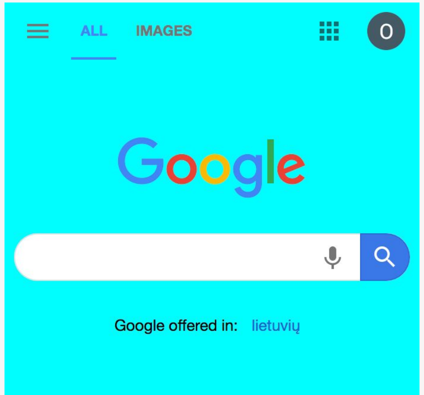
adapt

Browser environment

- Works in browser
- Javascript code are only executed from HTML page
- To execute the javascript code you need to:
  - Create HTML file that has internal (inline) or external script
  - Open HTML using browser
- Browser env javascript can manipulate DOM (or basically website visual content)
  - It can create, update, delete text, html elements, css styles, event listeners

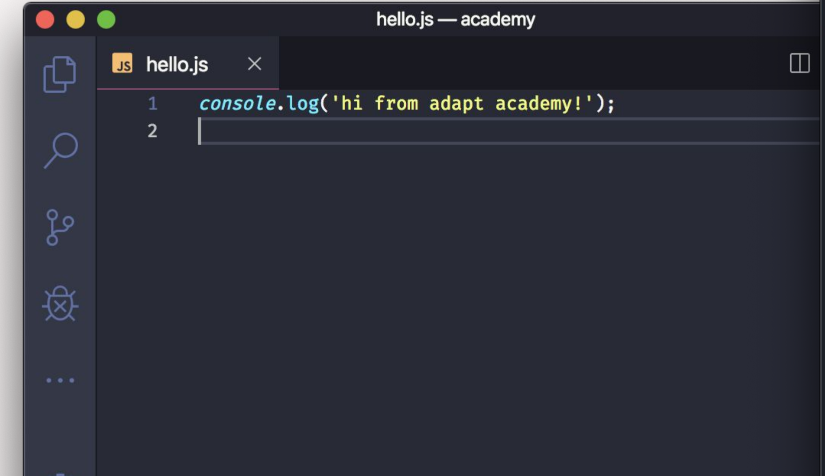


document.body.style.backgroundColor = "cyan"

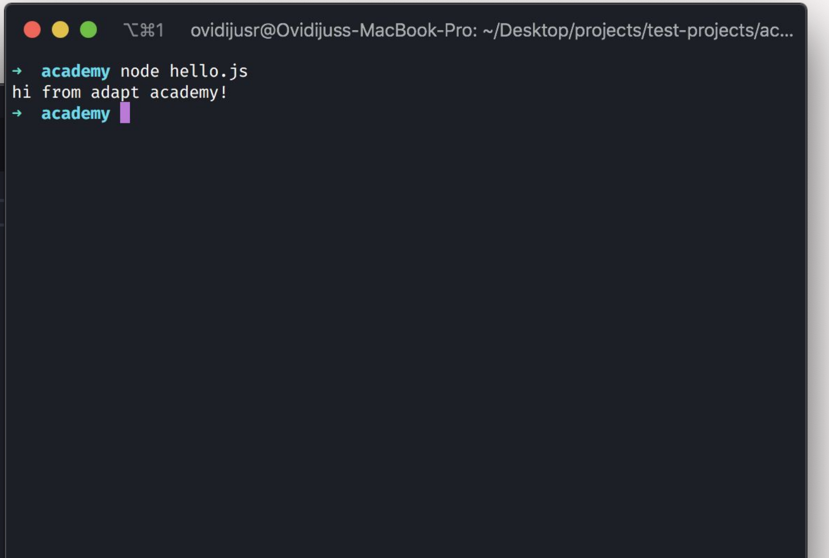


Node environment

- Javascript code can be executed directly by using `node mycodefile.js`
- Only works in terminal
- Can read / write files inside computer (just like c++, c# or python)
- Can work as a server (web server, email server, game server… you name it)



```
console.log('hi from adapt academy!');
```


⌥⌘1 ovidijusr@Ovidijuss-MacBook-Pro: ~/Desktop/projects/test-projects/ac...

```
→ academy node hello.js
hi from adapt academy!
→ academy
```

Which is best ? Both!

Libraries that use browser environment:

- React, Angular, Vue.JS
- React libraries
- Some of webpack (just a compiled part)

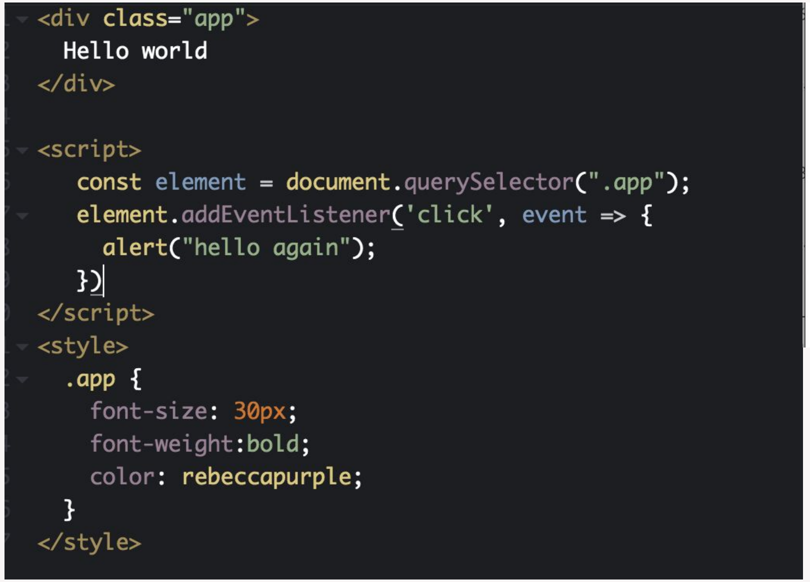
Libraries that use node environment:

- Webpack (and dev server)
- Babel
- NPM
- Linting tools (ESLint, Stylelint)
- Testing libraries (Mocha, Jest, Enzyme, Puppeteer)

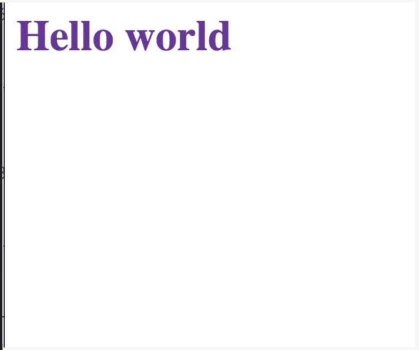
Most basic project structure

Project structure - basic structure

App.html



```
<div class="app">
  Hello world
</div>

<script>
   const element = document.querySelector(".app");
   element.addEventListener('click', event => {
     alert("hello again");
   })
</script>
<style>
  .app {
    font-size: 30px;
    font-weight:bold;
    color: rebeccapurple;
  }
</style>
```

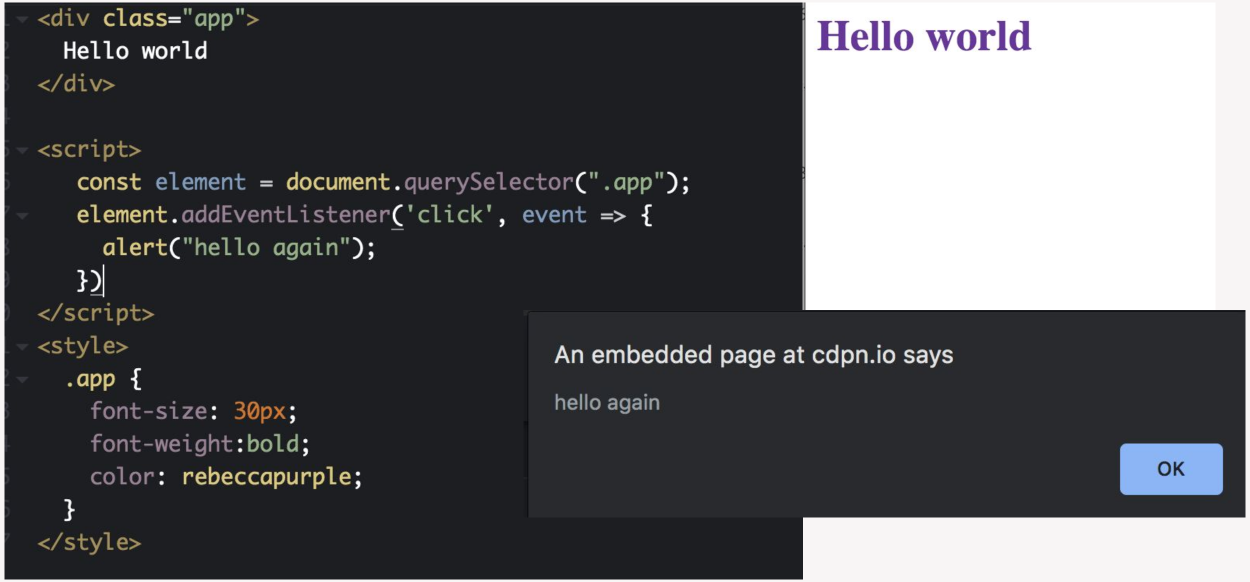


Hello world

Project structure - basic structure

HTML

App.html



```
<div class="app">
  Hello world
</div>

<script>
   const element = document.querySelector(".app");
   element.addEventListener('click', event => {
     alert("hello again");
   })
</script>
<style>
  .app {
    font-size: 30px;
    font-weight:bold;
    color: rebeccapurple;
  }
</style>
```

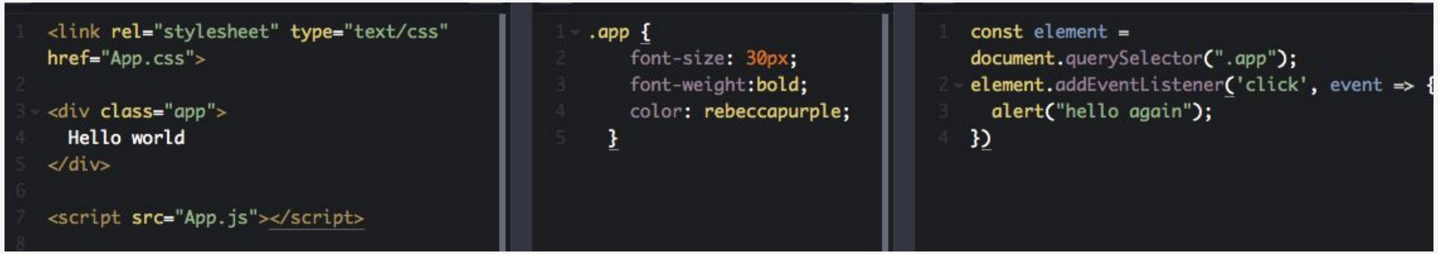
Hello world

An embedded page at cdpn.io says

hello again

adapt

Project structure - basic splitted structure



Index.html

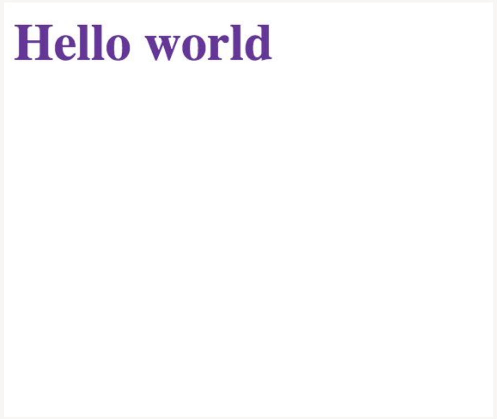
```html
<link rel="stylesheet" type="text/css" href="App.css">

<div class="app">
  Hello world
</div>

<script src="App.js"></script>
```

App.css

```css
.app {
    font-size: 30px;
    font-weight:bold;
    color: rebeccapurple;
  }
```

App.js

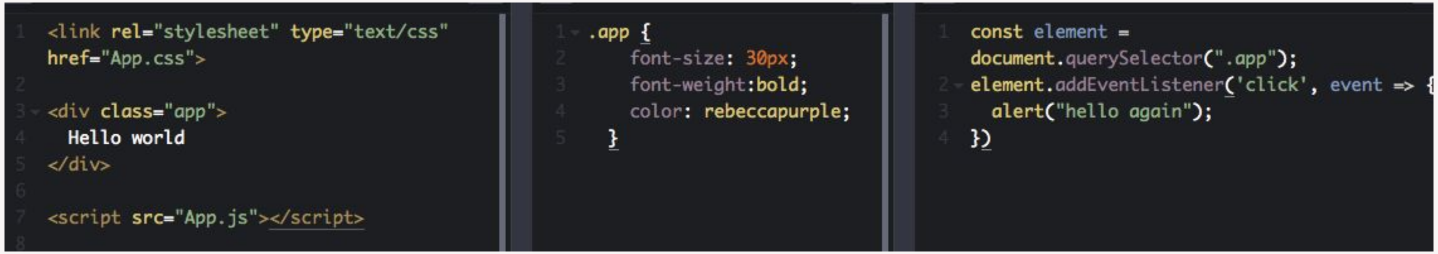
```javascript
const element = document.querySelector(".app");
element.addEventListener('click', event => {
  alert("hello again");
})
```



Hello world

Project structure - basic splitted structure

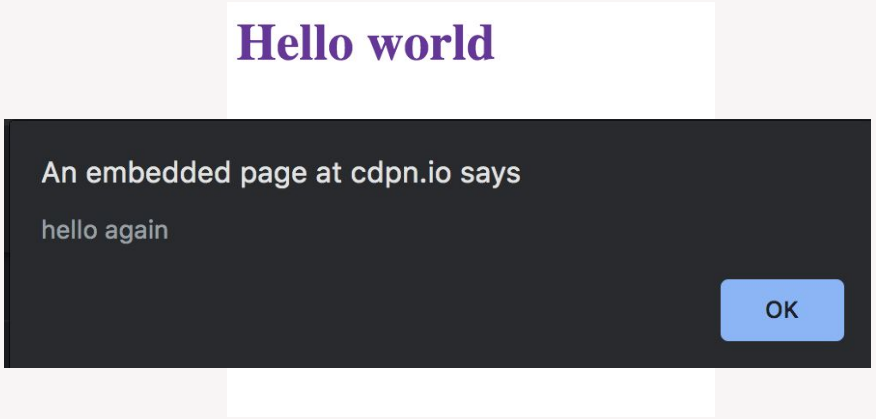
Index.html **App.css** **App.js**



```html
<link rel="stylesheet" type="text/css"
href="App.css">

<div class="app">
  Hello world
</div>

<script src="App.js"></script>
```

```css
.app {
    font-size: 30px;
    font-weight:bold;
    color: rebeccapurple;
  }
```

```js
const element =
document.querySelector(".app");
element.addEventListener('click', event => {
  alert("hello again");
})
```



Hello world

An embedded page at cdpn.io says

hello again

Actual project structure

Package.json - Your project’s information file

Most basic required information:

```
{
  "name": "my-awesome-package",
  "version": "1.0.0"
}
```

Can be generated by using: `npm init` in terminal

Project structure - how to implement code by others in your project

npm

OR

yarn

1. Install NODE.JS (which includes npm itself)
2. Initiate NPM in your project folder using `npm init`
3. Install your needed library `npm install lodash`

adapt

Project structure - how to implement code by others in your project

npm

OR

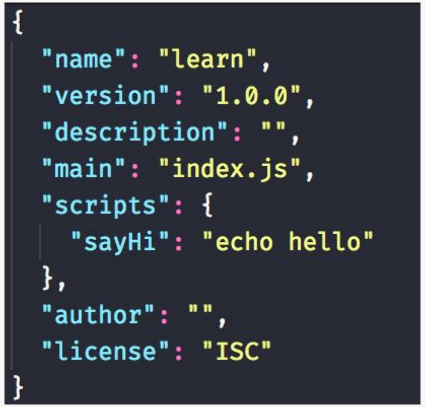
yarn

1. Install NODE.JS (which includes npm itself)
2. Initiate NPM in your project folder using `npm init`
3. Install your needed library `npm install lodash`

But how to connect libraries to your code?

adapt

Package.json - Your project's information file

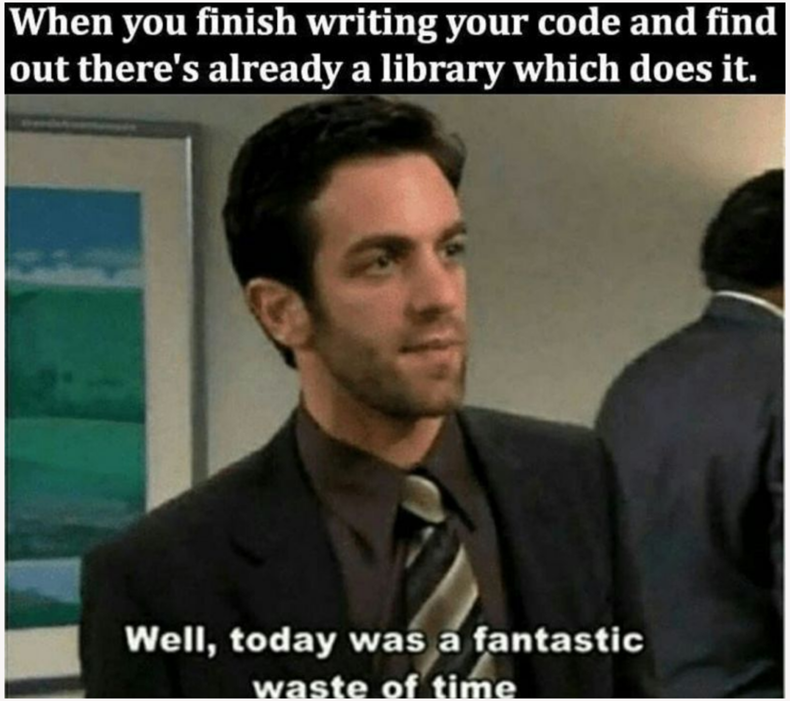


```json
{
  "name": "learn",
  "version": "1.0.0",
  "description": "",
  "main": "index.js",
  "scripts": {
    "sayHi": "echo hello"
  },
  "author": "",
  "license": "ISC"
}
```

Scripts object inside package.json is used for creating shortcuts commands

In this example by running `npm run sayHi` we would get response hello

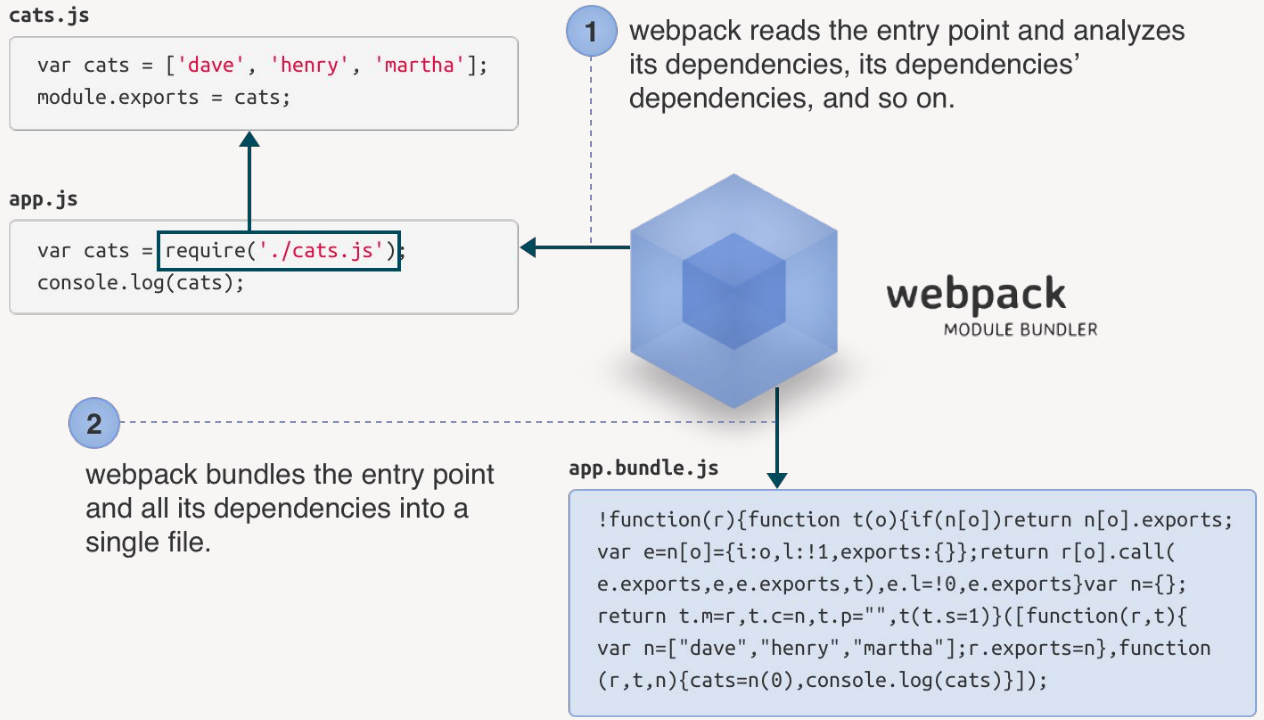
Project structure - code by others


When you finish writing your code and find out there's already a library which does it.
Well, today was a fantastic
waste of time

adapt

Project structure - how to implement code by others in your project

Project structure - how to implement code by others in your project



cats.js

```
var cats = ['dave', 'henry', 'martha'];
module.exports = cats;
```

app.js

```
var cats = require('./cats.js');
console.log(cats);
```

1 webpack reads the entry point and analyzes its dependencies, its dependencies' dependencies, and so on.

2 webpack bundles the entry point and all its dependencies into a single file.

app.bundle.js

```
!function(r){function t(o){if(n[o])return n[o].exports;
var e=n[o]={i:o,l:!1,exports:{}};return r[o].call(
e.exports,e,e.exports,t),e.l=!0,e.exports}var n={};
return t.m=r,t.c=n,t.p="",t(t.s=1)}([function(r,t){
var n=["dave","henry","martha"];r.exports=n},function
(r,t,n){cats=n(0),console.log(cats)}]);
```

adapt

Project structure - how to implement code by others in your project

```
import myLodashLibrary from 'lodash';
```

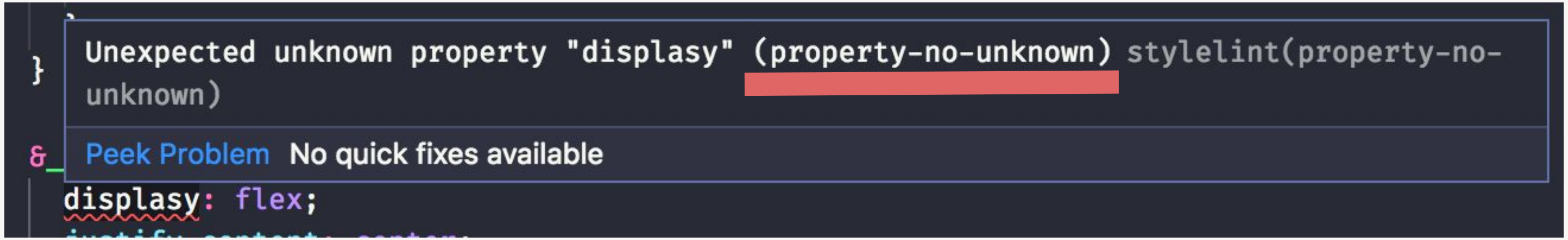
1. Write the import
2. Use the installed library (lodash)
3. Compile it with webpack

Linting - what?

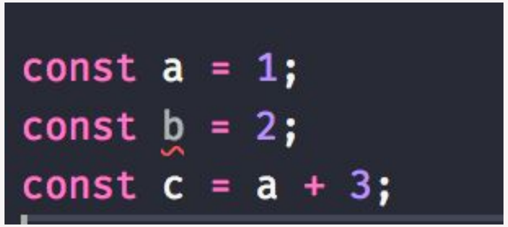
Linting is the process of running a program that will analyse code for potential errors.

- Some guy from stackoverflow

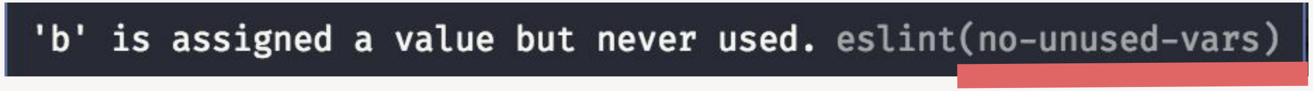
Linting - what?



```
}
    Unexpected unknown property "displasy" (property-no-unknown) stylelint(property-no-
    unknown)
    Peek Problem  No quick fixes available
&_
  displasy: flex;
```



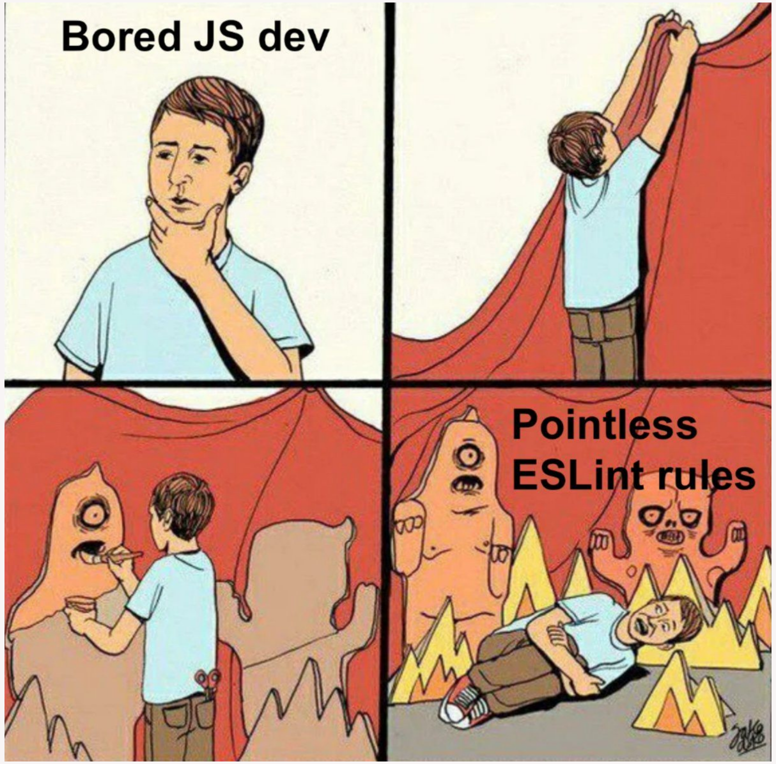
```
const a = 1;
const b = 2;
const c = a + 3;
```



```
'b' is assigned a value but never used. eslint(no-unused-vars)
```

adapt

Linting - why?



Linting - why?

- Helps you find stupid mistakes
- Let’s you have a consistent code rules between your team
- Can make your code cleaner / faster

Linting - how? - linting principles

- Linting application

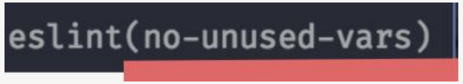
Linting - how? - linting principles

- Linting application

ESLint

stylelint

- Linting rule



```
eslint(no-unused-vars)
```

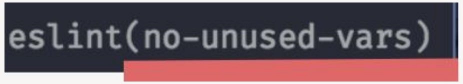
adapt

Linting - how? - linting principles

- Linting application

ESLint stylelint

- Linting rule



```
eslint(no-unused-vars)
```

- Your javascript / css file

JS

adapt

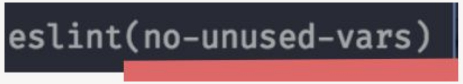
Linting - how? - linting principles

- Linting application

ESLint

stylelint

- Linting rule

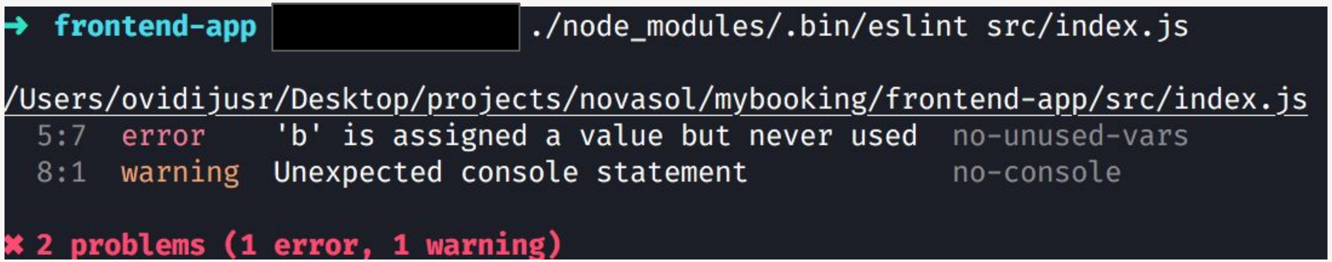


```
eslint(no-unused-vars)
```

- Your javascript / css file

JS

- Test outcome



```
➜  frontend-app [redacted] ./node_modules/.bin/eslint src/index.js

/Users/ovidijusr/Desktop/projects/novasol/mybooking/frontend-app/src/index.js
  5:7  error    'b' is assigned a value but never used  no-unused-vars
  8:1  warning  Unexpected console statement            no-console

✖ 2 problems (1 error, 1 warning)
```

adapt

Things to do before starting development on new project

Before starting development on new project (0)

Communication paths

- Knowing skills of team members
- Who handles what
- How to gain access to external tools
- Information storage and accessibility

Before starting development on new project (1)

Defining project goal

- **Short term** - identify project's critical parts and estimate their technical difficulties and possible challenges
- **Long term** - gain understanding of what waits ahead

Before starting development on new project (2)

Choosing framework based on:

- Project scope
- Existing solutions
- Team experiences
- Internal research
- Accessibility

Before starting development on new project (3)

Analyzing project design/wireframes

- Defining what is section/block/element
- Determine breakpoints
- Identifying anomalies

Define base style (CSS)

- Basic style for basic HTML elements
- Defining what is block/element/section
- Extensions (SASS/SCSS/PostCSS/...)
- Methodology (BEM/OOCSS/SMACSS/...)

Before starting development on new project (4)

Style guide is a set of standards for the writing and design of documents, either for general use or for a specific publication, organization, or field. (It is often called a style sheet, though that term has other meanings.) A style guide establishes and enforces style to improve communication.

- Do we need it?
- Advantages?
- Disadvantages?

Before starting development on new project (5)

Code coverage: can we do it and should we do it?

Functional testing types:

- Unit testing
- Integration testing
- System testing
- Sanity testing
- Smoke testing
- Interface testing
- Regression testing

Before starting development on new project (6)

Data structure

- What data do we need?
- How it should be structured?
- What can we get it?
- Should we normalize data and if so - where?
- Data overhead problem

Proof of concept

Proof of concept (**PoC**) is a realization of a certain method or idea in order to demonstrate its feasibility, or a demonstration in principle with the aim of verifying that some concept or theory has practical potential. A proof of concept is usually small and may or may not be complete.

Tasks

Task

- Setup git (git init)
- Setup npm (npm init)
- Create folder structure (dist src and public)
- Install webpack
- Setup npm launch command (script in package.json)
- Setup initial file and run it through webpack
- Make your function module and import it to the main function
- Install webpack dev server
- Setup empty html page
- Setup eslint with rule for no console logs