

Day 6.2

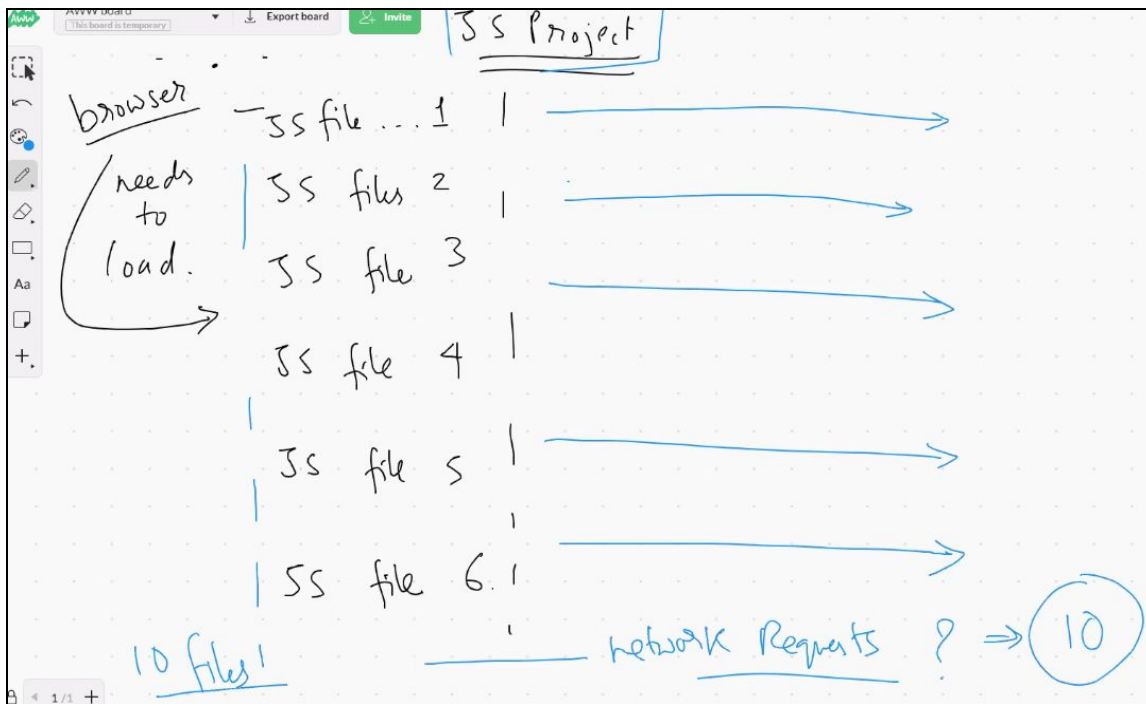
Title: Intro to REACT

Topics covered:

- Preprocessing Javascript Code
 - Minify
 - Uglification
 - Bundling
- NodeJS
 - JDK, JRE and NodeJS
 - JVM, V8 engine and SpiderMonkey
 - Maven/gradle and npm
- Running code of node-js using terminal
- Installing bootstrap
- Installing npm
- How to define a project
 - Importance of package.json

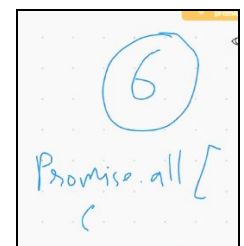
Building a JS project

- It will have multiple files.



- How many network request happen in parallel by a browser
- Ans is 6

- At a time only 6 network request is performed by the browser.
- Right now this number is standard thing it may change in future but now its is 6.



- If every network request takes x time to load. So if we have 10 files, 10 api request are needed.
- So minimum time required is x for 6 api request and $2x$ for 10api request.
- If 65 files it will be $11x$
- $65/6 + 1$
- Splitting js in multiple file, it helps the developer, But our clients will face slow loading of website.
- Solve it this way. Try to write it in less number of files. This a first solution
- Or Finding the middle ground
- Not hard for developer And not slow for client,
- Actual solution is the pre-processing the javascript code before using it.

Pre-proceess the Javascript code

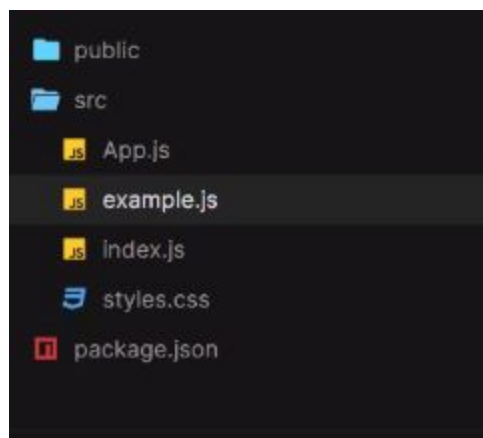
• Pre-Process
Javascript

First step is Minification of the code

① Minification
(remove all whitespace)

- Minification will remove all white space

An Example



- Clearted new file example.js
- Have simple function xyz and takes the input and prints abc

```
JS example.js x
1 function XYZ(abc) {
2   console.log(abc);
3 }
```

- Now if we minify ,All white space is removed

Minified code is like this

```
4 // MINIFY the JS code
5 function XYZ(abc){console.log(abc);}
```

number of lines are lesser

- Seeing an online minifier
- Tic-tac-toe problem removing its white space
- Total lines 166
- So many characters are reduced

We've been hosted on [Digital Ocean](#) since 2013 and highly recommend it for your hosting too. [Try it now!](#)

JavaScript Minifier

Online JavaScript Minifier/Compressor. Free! Provides an API. Simple Quick and Fast.

Written and hosted by [Andrew Chilton](#) (@andychilton) since 2012.

Input JavaScript

```
initializeGame();
}
}

let currentGame;

function initializeGame() {
  currentGame = new TicTacToe();
}

initializeGame();

function playTurn(that) {
  currentGame.playerClick(that.id);
}
```

Minified Output

```
class TicTacToe{constructor()
{this.playerTurn="X",this.turnDisplayElement=document.getElementsByClassName("turn")[0],this.boardState=[["","",""],["","",""],["","",""]],this.renderBoard()}renderBoard()
{this.showPlayerTurn(),this.updateBoard()}showPlayerTurn(){if("X"===this.playerTurn&&
(this.turnDisplayElement.innerText="Player 1's Turn"),"O"===this.playerTurn&&
(this.turnDisplayElement.innerText="Player 2's Turn"))updateBoard()}for(let
t=0;t<3;t++)for(let e=0;e<3;e++){const
r=this.convertIndexToId(t,e);document.getElementById(r).innerText=this.boardState[t][e]]}convertIndexToId(t,e){return 3*t+(e+1)}playerClick(t)
{const[e,r]=this.convertIdToIndex(t);""===this.boardState[e][r]&&(this.boardState[e][r]=this.playerTurn,this.changePlayerTurn())}convertIdToIndex(t){const
e=Number(t);return[Math.ceil(e/3)-1,(e-1)%3]}changePlayerTurn()
{this.playerTurn="X"===this.playerTurn?"O":"X",this.renderBoard(),this.checkGameEnd()}checkGameEnd(){let t=0;for(let e=0;e<3;e++){t=0;for(let
r=1;r<3;r++)if(this.boardState[e][r]===this.boardState[e][r-1])
```

[Minify](#)
[Download as File](#)
[RAW](#)
[Clear](#)

[Copy to Clipboard](#)
[Select All](#)

Limited time offer: Get 10 free Adobe Stock images.

ADS VIA CARBON

Everything is minified in single line, Amount of space is less

```
JS example.js
1 class TicTacToe{constructor(){this.playerTurn="X",this.turnDisplayElement=document.getElementsByClassName("turn")[0],this.boardState=[["","",""],["","",""],["","",""]],this.renderBoard()}renderBoard(){this.showPlayerTurn(),this.updateBoard(),showPlayerTurn()}showPlayerTurn(){this.playerTurn&&(this.turnDisplayElement.innerHTML="Player 1's Turn"),this.playerTurn&&(this.turnDisplayElement.innerHTML="Player 2's Turn")}updateBoard(){for(let t=0;t<3;t++)for(let e=0;e<3;e++){const r=this.convertIndexToId(t,e),document.getElementById(r).innerHTML=this.boardState[t][e]}convertIndexToId(t,e){return 3*t+e+1}playerClick(t){const[e,r]=this.convertIdToIndex(t),this.boardState[e][r]&&(this.boardState[e][r]=this.playerTurn,this.changePlayerTurn(),convertIdToIndex(t),const e=Number(t),return Math.ceil(e/3)-1,(e-1)%3)}changePlayerTurn(){this.playerTurn="X"==this.playerTurn?"O":"X",this.renderBoard(),this.checkGameEnd()}checkGameEnd(){let t=0;for(let e=0;e<3;e++){t=0;for(let r=0;r<3;r++){if(this.boardState[e][r]!=this.boardState[e][r-1]){t=1;break}}if(t&&"!"==this.boardState[e][0])return this.alertWin(this.boardState[e][0])}for(let e=0;e<3;e++){t=0;for(let r=0;r<3;r++){if(this.boardState[r][e]!=this.boardState[r-1][e]){t=1;break}}if(t&&"!"==this.boardState[0][e])return this.alertWin(this.boardState[0][e])}for(let e=0;e<3;e++){if(this.boardState[e][0]==this.boardState[e][1]||this.boardState[e][1]==this.boardState[e][2])return this.alertWin(this.boardState[e][0])}for(let e=0;e<3;e++){if(this.boardState[e][2-e]==this.boardState[e-1][2-e-1])return this.alertWin(this.boardState[e][2-e])}if(t&&"!"==this.boardState[2][0])return this.alertWin(this.boardState[2][0])}let e=0;for(let t=0;t<3;t++){for(let r=0;r<3;r++){if("!"==this.boardState[t][r])e=1;break}if(0==e)break}return e?this.alertDraw():void 0}alertWin(t){alert("Congratulations! Player1 wins"),alert("Congratulations! Player2 wins"),initializeGame(),alertDraw(),alert("Draw!"),initializeGame()}let currentGame,function initializeGame(){currentGame=new TicTacToe,function playTurn(t){currentGame.playerClick(t.id)}initializeGame()}}
```

- As the characters are less now. 100kb file is reduced to 20kb. This processing should be done after completing your code.
- Keep copy of original source code and minified code.
- Never modified original code

Now each file is taking less time to download



Next step is

② Uglification / Obfuscation

- It reduces the size of file by renaming the variables like class tic tac toe is just renamed to T.

```
class T {
  constructor() {
    this.p = "X";
    this.a = document.getElementsByClassName("turn")[0];
    this.b = [
      ["", "", ""],
      ["", "", ""],
      ["", "", ""]
    ];
    this.r();
  }
}
```

- Next it can change normal for loop to foreach loop
- Minification is reversible
- Uglification is irreversible
- Uglifier will also reduce the size of source code by removing comments

Bundling is the next step

③ Bundling :

- Bundling reduces no of files and combines all 65 files to 1 file or 2 files.

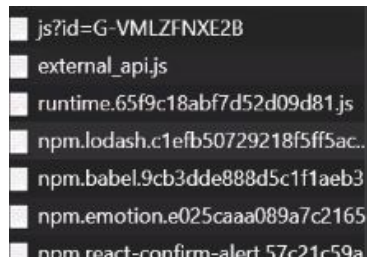
combine multiple JS files
into a smaller number.



```
1 (window.webpackJsonp = window.webpackJsonp ||
  "00945206146f7a17044b": function(e, t, n)
    var c = n("5a7ccd6273ec77fdeabc")
    , r = Object.prototype.hasOwnProperty
    e.exports = function(e) {
      var t = this.__data__;
      if (c) {
        return __webpack_require__
      }
      return r.call(t, e) ? t[e] : void
    }
  },
  "01d665d42ac0fb526292": function(e, t, n)
    var c = n("f8446e843ad1613417d0")
    , r = n("e99524bb668f88c6c42e")
    , f = n("f2ef6f6e54aad97aa22")
    , a = n("f72c7c366a646d5e453e")
    , o = n("84696c4e387dcb8648dc")
    , b = n("22fd2f70e6f18dac8668");
    e.exports = function(e, t, n) {
      for (var u = -1, i = (t = c(t, e))
        var s = b(t[u]);
        if (!(d = null != e && n(e, s)
          break;
        e = e[s]
      )
    }
```

preview of uglify notice the variables

- After the name of the file their is a huge hash that is available



```
js?id=G-VMLZFNXE2B
external_api.js
runtime.65f9c18abf7d52d09d81.js
npm.lodash.c1efb50729218f5ff5ac.
npm.babel.9cb3dde888d5c1f1aeb3
npm.emotion.e025caaa089a7c2165
npm.react-confirm-alert.57c21c59a
```

`bundle.[hash].js`

- Converting data to hash value
- Whenever your data changes hash value also changes
- Helps the browser to use cached file (old file visited that's saved) or the new one. If the hash value is changed the browser uses/loads the new file, if hash value is not changed then the browser uses the old saved file.

`bundle.[hash].js` → browser does not cache the file.

- Uglification is also good for protecting the source code. Reverse engineering the source code is hard.
- Every modern website does this (even irctc does this).

Where will preprocessing take place on the browser or on the developer machine

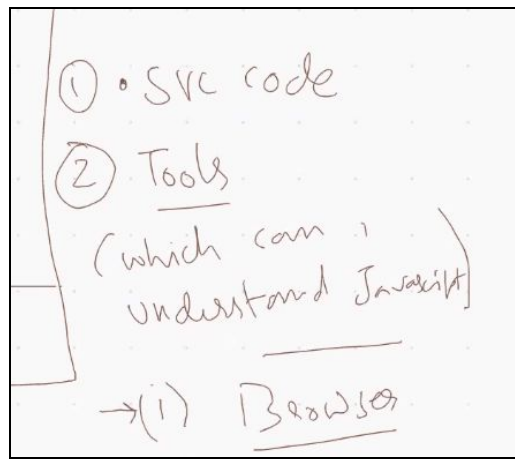
`Pre-Process Javascript` → browser?
→ developer machine?

Ans is developer machine

- Obviously as we want to optimise the loading speed of browser, why we will give new job of preprocessing to a browser.

List of things required at developer's machine to perform the preprocessing

- Source code (obviously developer created it, he/she will have it)
- Tools (That understands JS)
 - Browser is one tool



- One such tool is browser

Instead

We need some other tool to do this

That is the node js

That actually understand the js

- Be ready with nodejs For the next class
- Install the lts version above then 12. Prefer 14LTS(long term support)

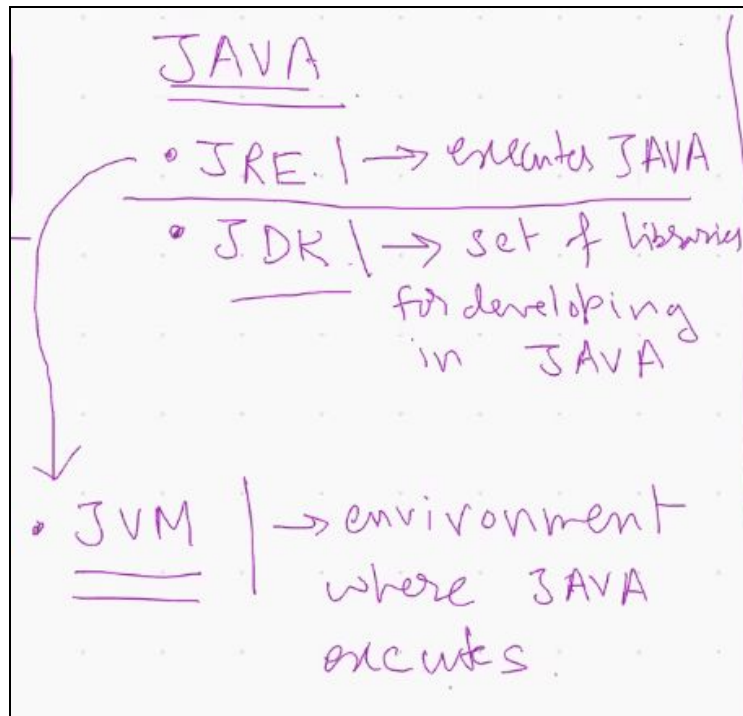


- Anything above version 12 is ok

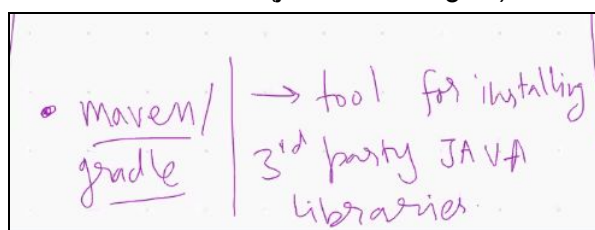
What is nodejs?

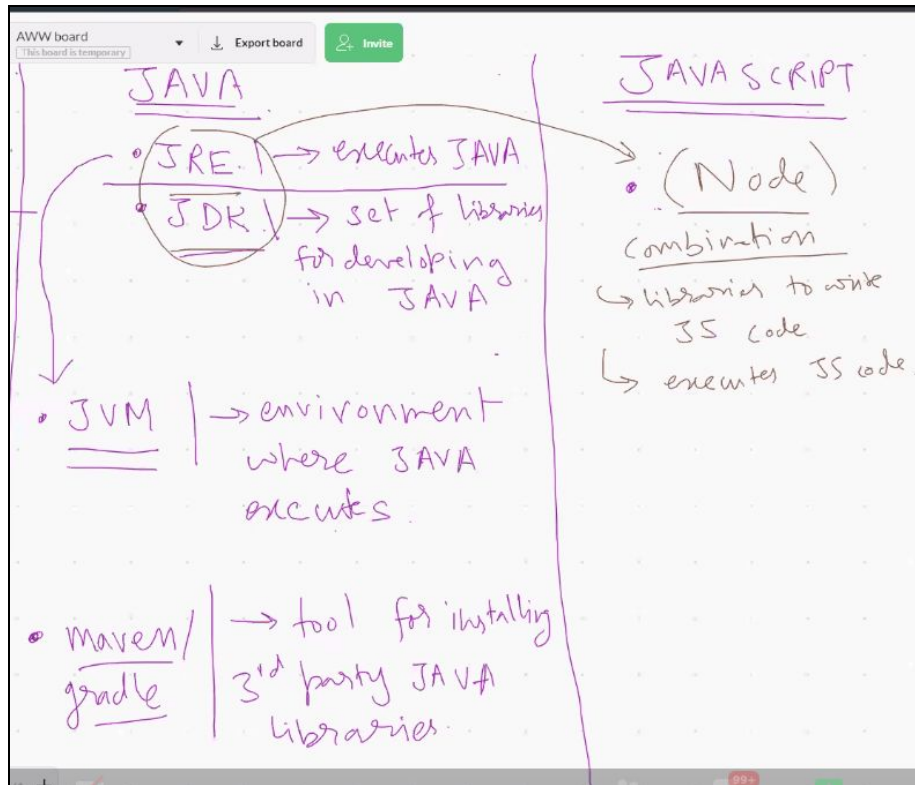
Understand by comparing with java

- We have two tools one is jre and one is jdk
- Java development kit, Jdk is set of libraries for developing in java
- Jre is runtime environment One that executes java
- With jre you can just run the java code
- Jdk is used to develop code in java
- Jvm is the environment on which jre executes java

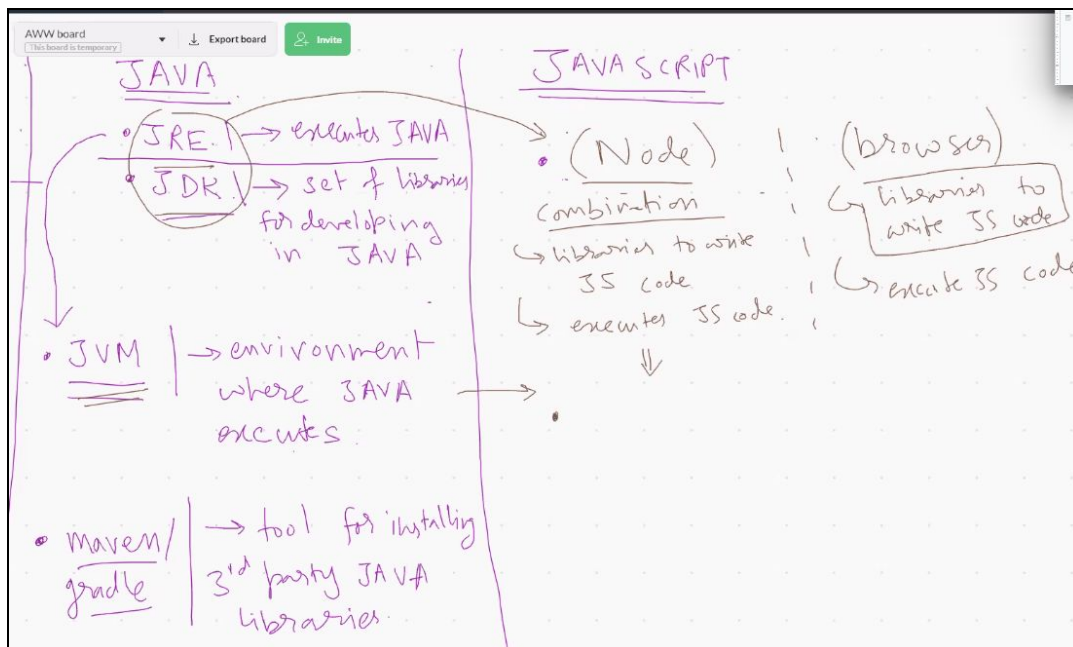


- Jre executes the java code inside jvm
- Tools for installing third party java library
 - Maven and gradle
 - These are called package management tools
 - Maven allows you to install external libraries that someone else has developed for you
- NodeJS is like a combination of JRE and JDK (just for analogies)





- Browser can execute js code and it also has certain library
 - Examples like alert
 - Window.confirm this are available only in browser but not in node
- NodeJS and Browser are similar
- But libraries are different
- Node is for developer and browser is for client who will load the code



Browser means you can see and browse website but using node you cannot

V8 engine is like jvm



Its similar to jvm but this analogy is not correct

But for now you can understand it this way

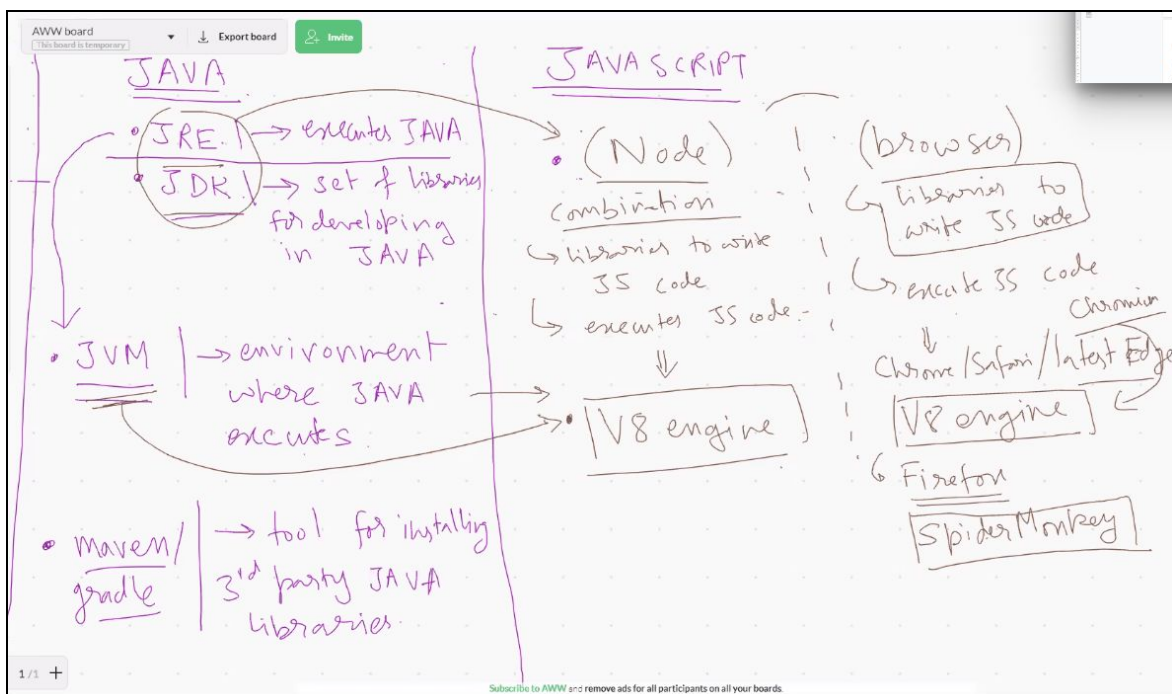
chrome safari/latest edge uses v8 engine

Firefox is uses the engine called spiderMonkey

V8 engine is developed by google

Companies can develop their own engine for performance optimisations.

But this (V8 engine) is where the js is actually executed



- Brave also uses v8

Analogy to maven and gradle is npm

npm → node packages manager.
→ install external libraries

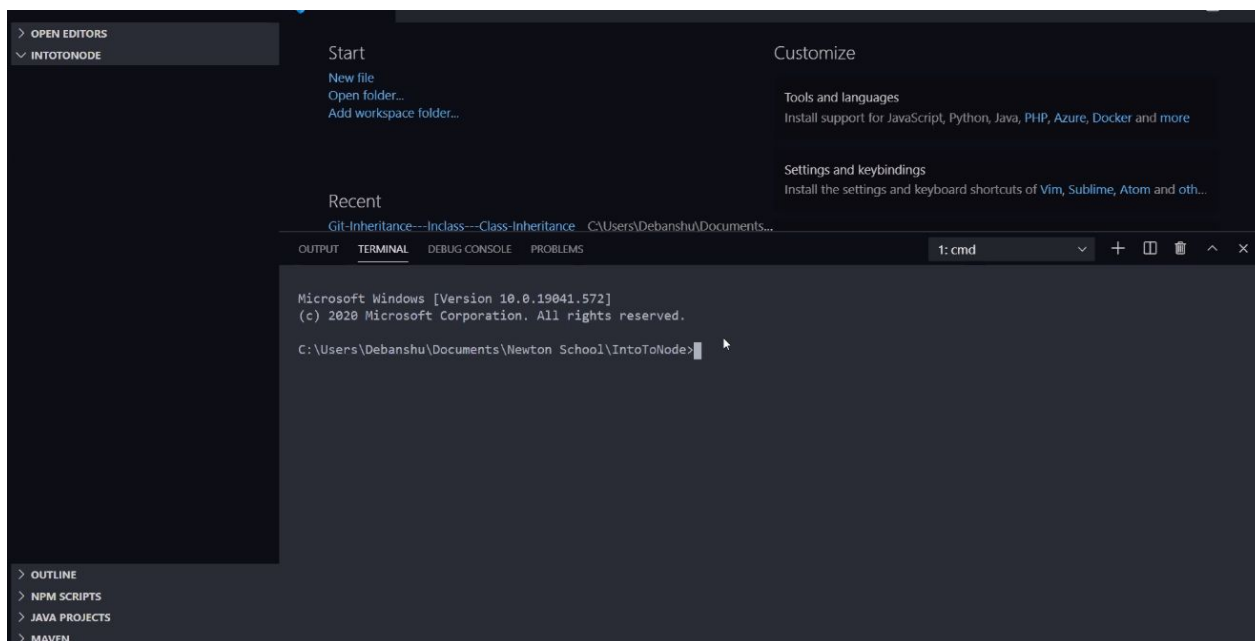
- All js and css libraries can be installed using npm
- Npm is used to install external libraries

Node.js® is a JavaScript runtime built on Chrome's V8 JavaScript engine.

Once you install node js, You can see it in terminal

In vs code we can also use terminal

Create folder and open it in vscode



```
Microsoft Windows [Version 10.0.19041.572]
(c) 2020 Microsoft Corporation. All rights reserved.

C:\Users\Debashu\Documents\Newton School\IntoToNode>node
Welcome to Node.js v12.16.1.
Type ".help" for more information.
> 
```

Terminal at top of the window of VSCODE

- Type node hit enter, Node shell opens up

Console.log is different here, it prints the value and also returns something, here it returned undefined.

```
> console.log("Test")
Test
undefined
```

Creating file

Writing simple function in it

```
JS example.js > ...
1 function XYZ(abc) {
2   |   console.log("Ran XYZ with ", abc);
3 }
4
5 XYZ("someArgument");
```

How to execute a file

Run this command to execute the file example.js -> "node example.js"

```
C:\Users\Debanshu\Documents\Newton School\IntoToNode>node example.js
```

```
C:\Users\Debanshu\Documents\Newton School\IntoToNode>node example.js
Ran XYZ with  someArgument
```

Ran the example.js

Npm version do not matters that much. Node version should be 12 or Above 12

Code to check version of node and npm

```
C:\Users\Debanshu\Documents\Newton School\IntoToNode>node
Welcome to Node.js v12.16.1.
Type ".help" for more information.
> .exit

C:\Users\Debanshu\Documents\Newton School\IntoToNode>node example.js
Ran XYZ with  someArgument

C:\Users\Debanshu\Documents\Newton School\IntoToNode>node -v
v12.16.1

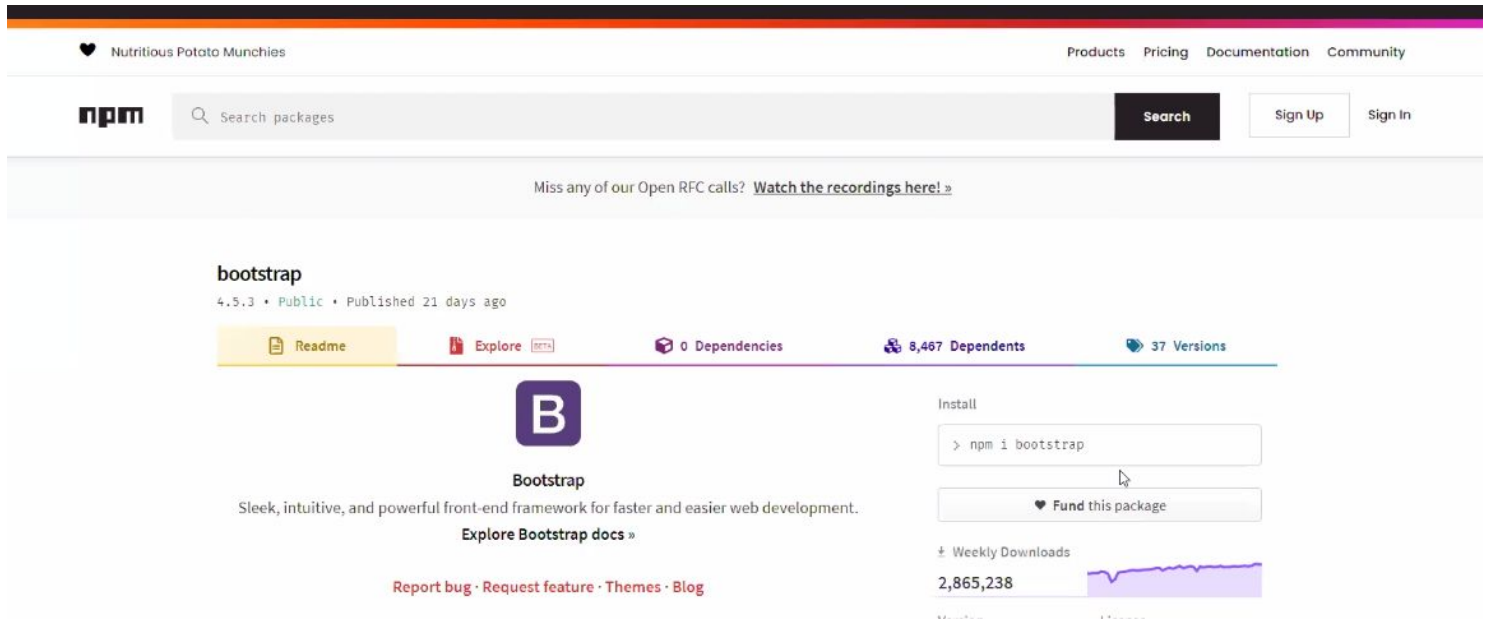
C:\Users\Debanshu\Documents\Newton School\IntoToNode>npm -v
6.13.4

C:\Users\Debanshu\Documents\Newton School\IntoToNode>
C:\Users\Debanshu\Documents\Newton School\IntoToNode>
```

- Comma also giving space

Open vs code only after you have installed node

Installing bootstrap in our project



Link -> <https://www.npmjs.com/package/bootstrap>

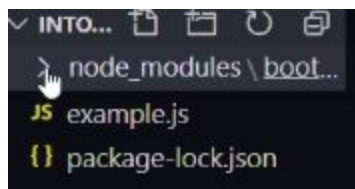
Npm install name of the package

```
C:\Users\Debanshu\Documents\Newton School\IntoToNode>npm install bootstrap
```

If you want to install some version

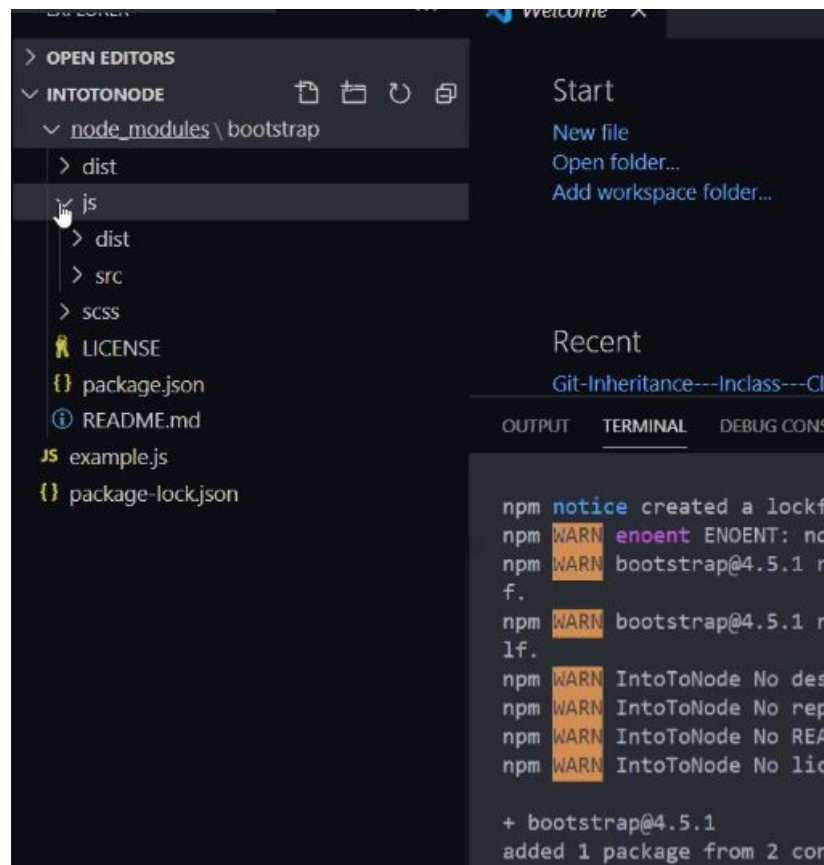
```
C:\Users\Debanshu\Documents\Newton School\IntoToNode>npm install bootstrap@4.5.1
```

after installing you will find special directory node_modules



Source code from bootstrap is downloaded and installed

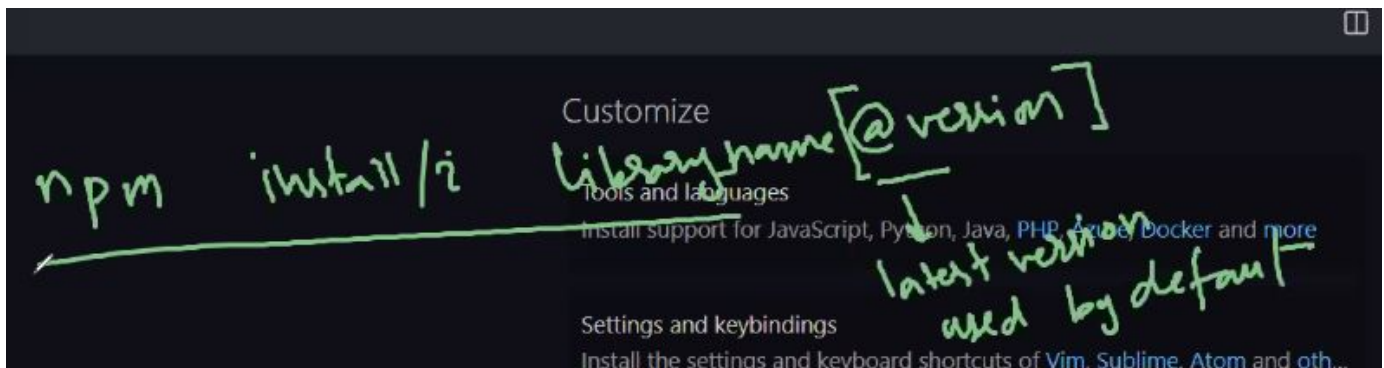
Now if you don't give the version it will download the latest version and override the same folder if the version is different.



It will rewrite the older version

If you do it again it will happen (again override) in the node_module

“npm i bootstrap” <- short command for installing



Square brace thing is optional

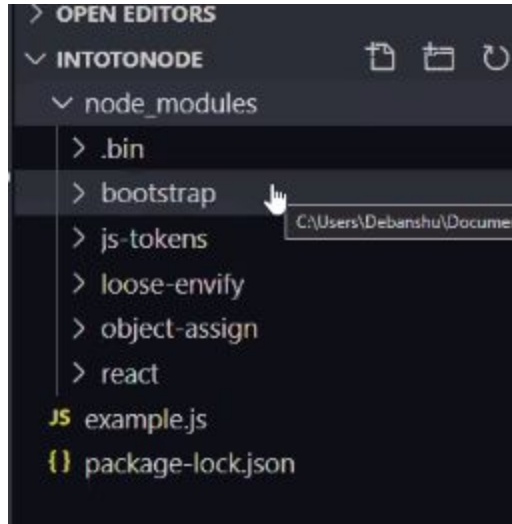
It will download a library and put it inside the a directory called node_modules

Every time you install any library that will be installed in node_module directory

Installing React

“npm i react”

React is installed but other things also installed, Its because the react depend on some other dependencies. So those dependencies are also downloaded.



- Maven is for java
- Pip is for python
- These things allow you to install external libraries
- Anything that you want to publish and distribute is library
- Framework is a special type of library

We will talk about framework and library afterwards

How will you define Project

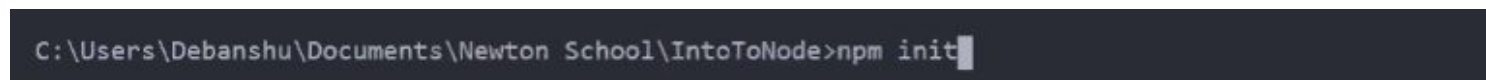
- My project needs bootstrap and react to work properly.
- If I want to give my project to someone, it would be a bad idea to also give the node_module folder.
- Node_module directory can become very large, so project size will be even larger
- They should be able to download all the dependency by themselves.
- We are not giving node module, they will download it, Npm helps in that also

Package.json file you can write this dependency whatever(dependency) your project needed.

Deleted everything

Command

“npm init”



This allows you to write your own project,

First thing is to add some instructions.

If you distribute this package.json it will help them to download all dependencies required in your project.

```
C:\Users\Debanshu\Documents\Newton School\IntoToNode>npm init
This utility will walk you through creating a package.json file.
It only covers the most common items, and tries to guess sensible defaults.

See `npm help json` for definitive documentation on these fields
and exactly what they do.

Use `npm install <pkg>` afterwards to install a package and
save it as a dependency in the package.json file.

Press ^C at any time to quit.
package name: (intotonode) █
```

Name of the package (Name of the project)

Give name to your Project

```
Press ^C at any time to quit.
package name: (intotonode) intro-to-node █
```

What is the version of your package

```
Press ^C at any time to quit.
package name: (intotonode) intro-to-node
version: (1.0.0) █
```

Description

```
description: First NPM package for August 2020 NS students
entry point: (index.js) █
```

Entry point Not modify this for now will be discussed @ backend development

```
description: First NPM package for August 2020 NS students
entry point: (index.js)
test command: █
```

Test command

Ignore it

For connecting online github, To give any link

```
entry point: (index.js)
test command:
git repository: █
```

Keywords is for helping the public to you when people are searching for your package

```
test command:
git repository:
keywords:
author: Debanshu Sinha
license: (ISC) █
```

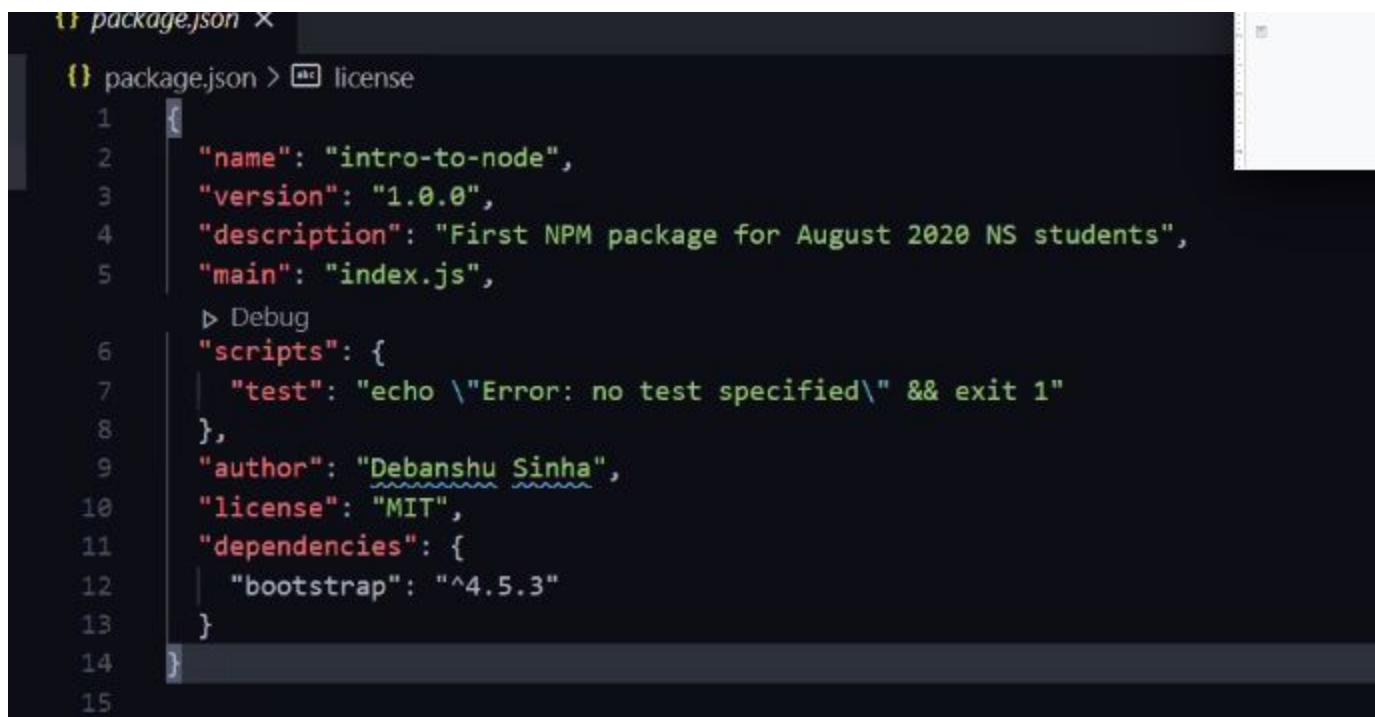
```
About to write to C:\Users\Debanshu\Documents\Newton School\IntoToNode\package.json:
```

```
{
  "name": "intro-to-node",
  "version": "1.0.0",
  "description": "First NPM package for August 2020 NS students",
  "main": "index.js",
  "scripts": {
    "test": "echo \"Error: no test specified\" && exit 1"
  },
  "author": "Debanshu Sinha",
  "license": "MIT"
}
```

Hit enter

Package .json has been created

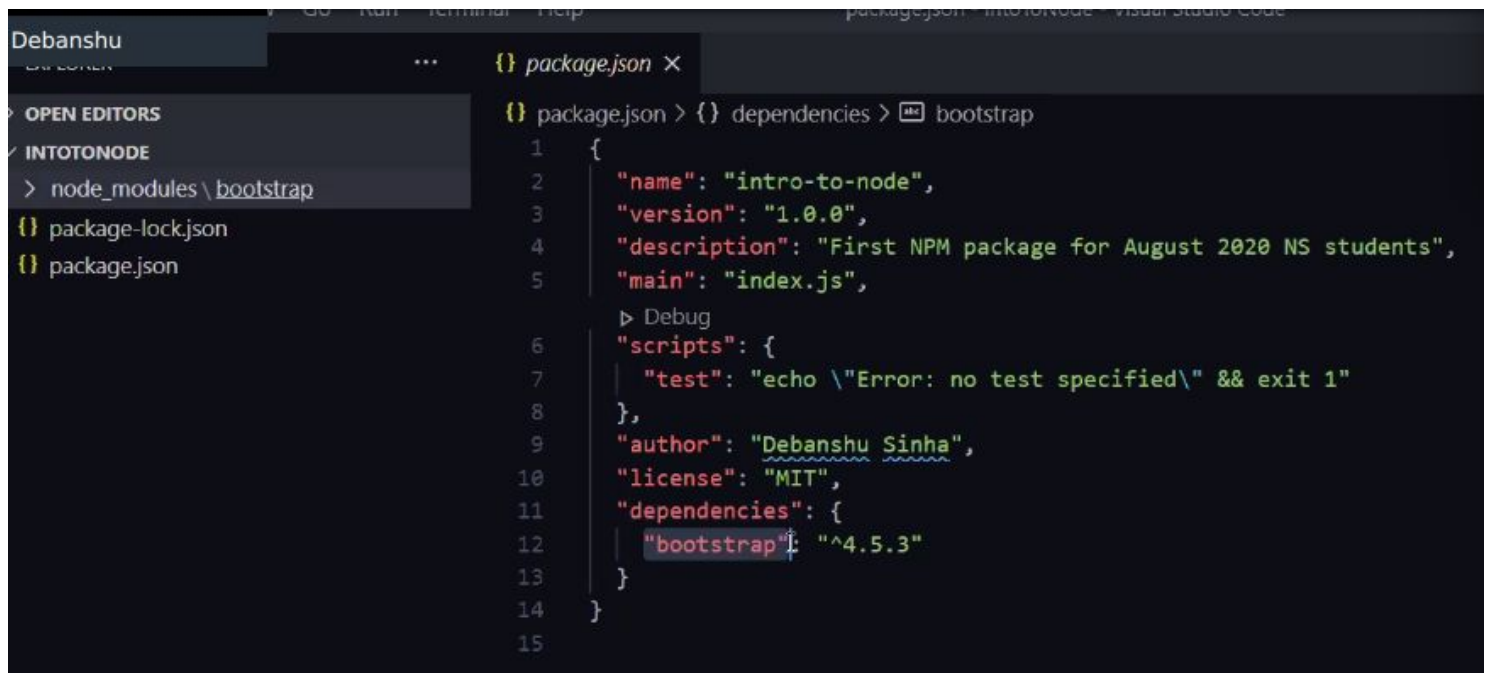
No dependencies listed in this now

A screenshot of a code editor window titled 'package.json'. The editor shows the contents of the package.json file, which now includes a 'dependencies' section. The text is as follows:

```
{
  "name": "intro-to-node",
  "version": "1.0.0",
  "description": "First NPM package for August 2020 NS students",
  "main": "index.js",
  "scripts": {
    "test": "echo \"Error: no test specified\" && exit 1"
  },
  "author": "Debanshu Sinha",
  "license": "MIT",
  "dependencies": {
    "bootstrap": "^4.5.3"
  }
}
```

The editor has line numbers on the left side, ranging from 1 to 15. The 'dependencies' section was added between lines 11 and 13.

Now do `npm i bootstrap`



The screenshot shows the Visual Studio Code editor with the `package.json` file open. The file content is as follows:

```
1 {
2   "name": "intro-to-node",
3   "version": "1.0.0",
4   "description": "First NPM package for August 2020 NS students",
5   "main": "index.js",
6   "scripts": {
7     "test": "echo \"Error: no test specified\" && exit 1"
8   },
9   "author": "Debanshu Sinha",
10  "license": "MIT",
11  "dependencies": {
12    "bootstrap": "^4.5.3"
13  }
14 }
```

The `bootstrap` dependency is highlighted in the `dependencies` object. The Explorer sidebar on the left shows the file structure with `package-lock.json` and `package.json` in the `node_modules` directory.

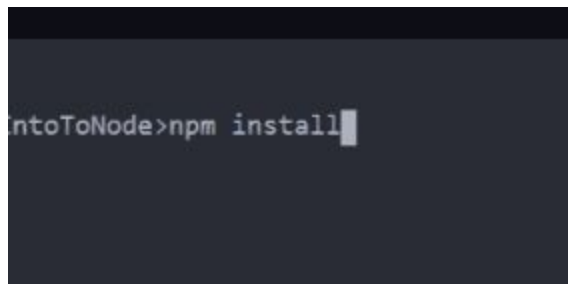
Node_modules is downloaded and added bootstrap in package.json

Package.json is keeping the track of your dependencies

Not sub-dependencies
Only root dependencies

“Missed something”

npm install



The terminal shows the command `npm install` being executed in a shell. The prompt is `intoToNode>`.

If you just do this, Automatically parse package.json add or download only those dependencies whose names are present in the package.json

“node seen”
“npm seen”

Which install libraries

Node can understand JS

We started from preprocessing

We have to do Preprocessing
We got tool to understand js
That is node

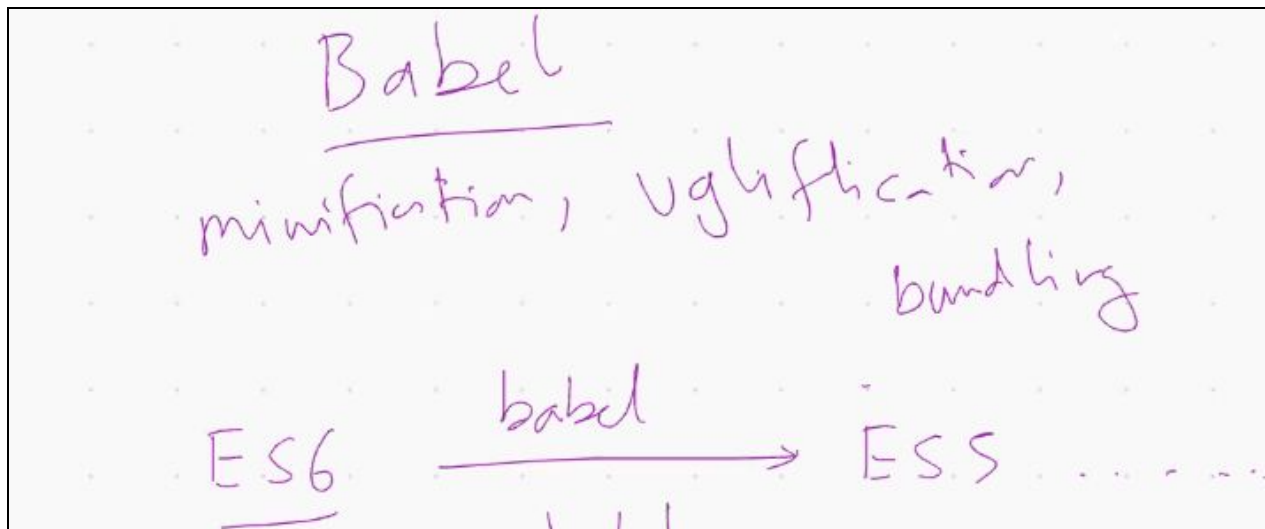
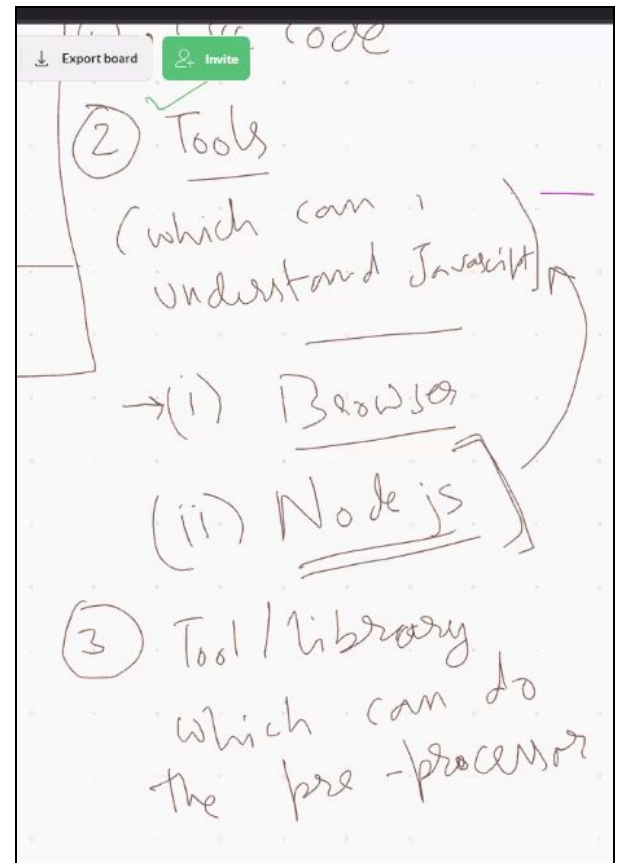
We have source code

There must be a tool or library which can do this preprocessing for you

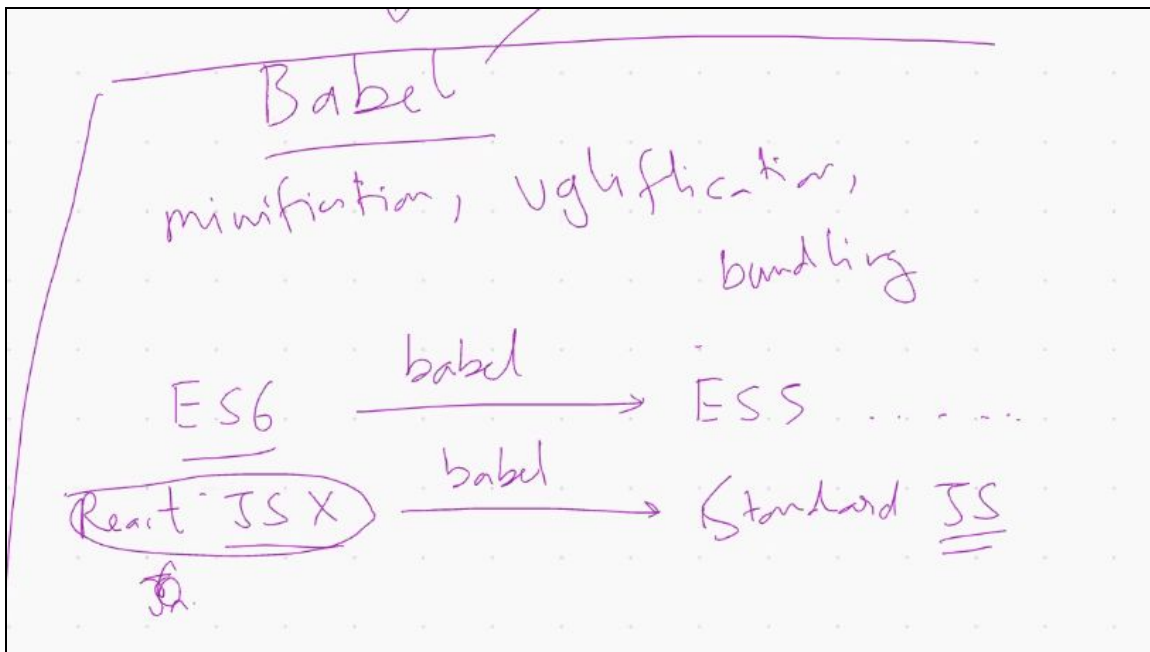
Babel library for processing js
Babel can do stuff like minification, uglification and bundling

Extra things

Babel



Babel can convert reactjs to standard js



will discuss tomorrow

Babel plugins

Myth

Tower of babel

In babylon

It had the power to understand every language and could give birth to new languages

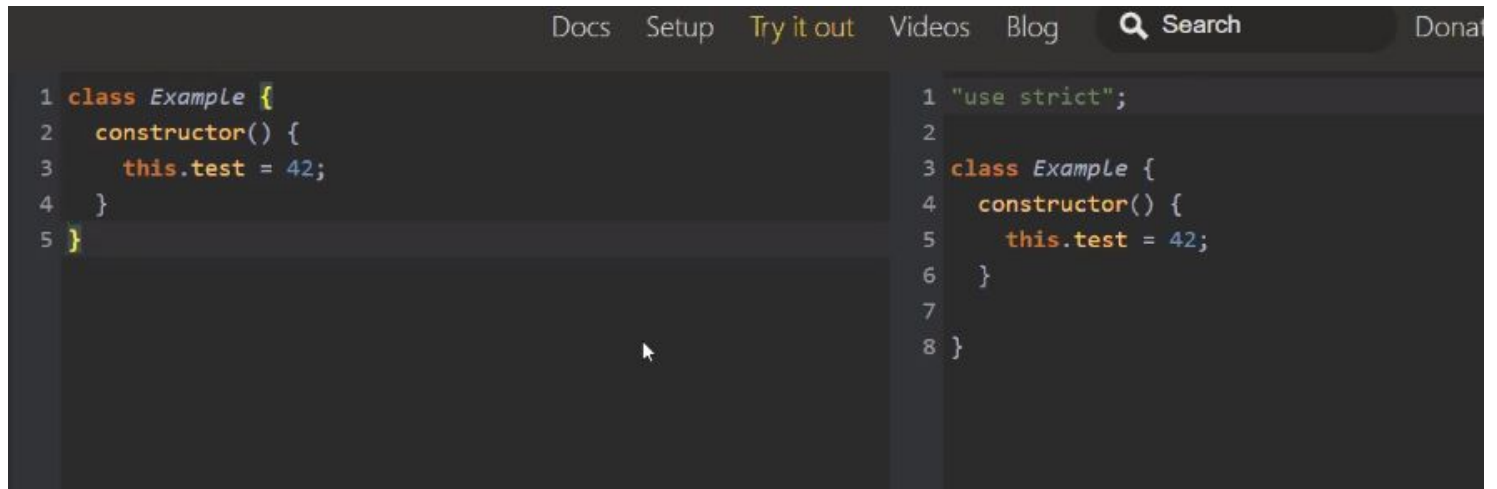
From their this term came

Tryout babel



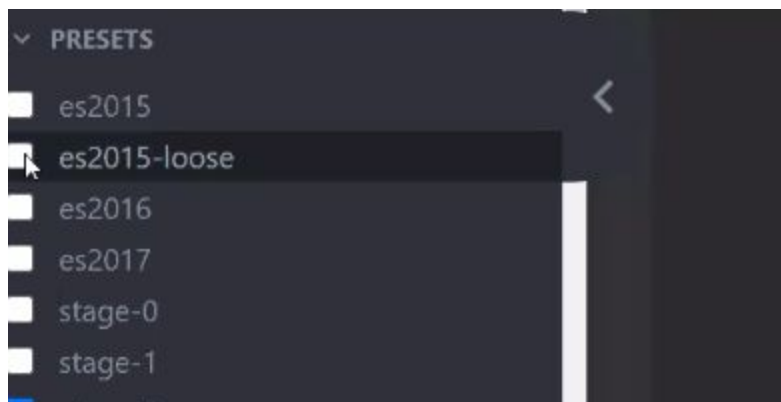
<https://babeljs.io/en/repl>

Try out at home

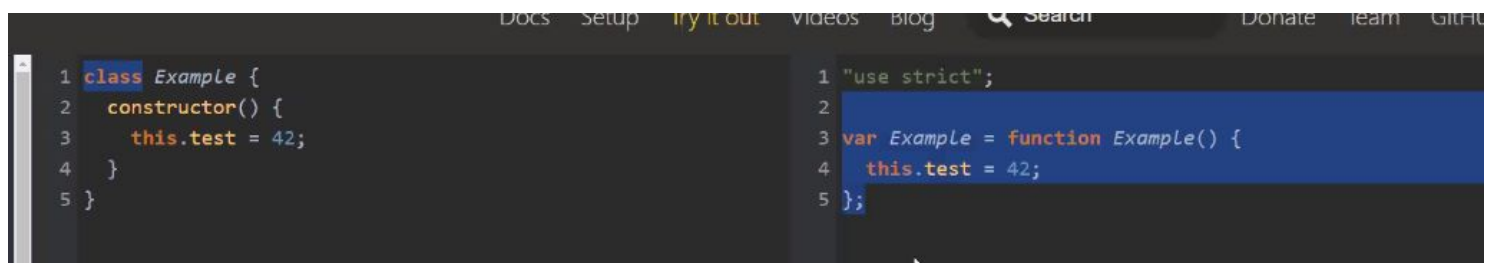


The screenshot shows a code editor with two panels. The left panel contains ES6 class syntax, and the right panel shows the equivalent ES5 function-based syntax.

```
1 class Example {  
2   constructor() {  
3     this.test = 42;  
4   }  
5 }  
  
1 "use strict";  
2  
3 class Example {  
4   constructor() {  
5     this.test = 42;  
6   }  
7  
8 }
```



converted to es 2015



The screenshot shows the code editor after conversion to ES5. The left panel shows the original ES6 class syntax, and the right panel shows the converted ES5 code.

```
1 class Example {  
2   constructor() {  
3     this.test = 42;  
4   }  
5 }  
  
1 "use strict";  
2  
3 var Example = function Example() {  
4   this.test = 42;  
5 };
```