**Vanilla Javascript -> ES6 -> Object Orienteret**Gennemgang af værktøjer til opsætning og udvikling af HTML5/ES6/SASS.

**Der arbejdes med følgende opsætning:**

* Node.js
* npm
* Webpack
* Es6 (Javascript)
* Sass

Node.js skal være installeret på din computer. Du kan installere node.js og npm herfra:

<https://www.npmjs.com/get-npm>

Du kan tjekke din node version med cmd promt:

node -v

og du kan tjekke om du har npm installeret ved at køre denne i cmd promt

npm -v

**Der kan hentes en BoilerPlate på vidonline under jeres klassenavn eller på github (adresse).**

Når du har hentet en boilderplate ned, skal du åbne en cmd promt på din boilderplate mappe og skrive følgende:

npm i

Dette opretter en node\_modules mappe med alle de node pakker, som der er angivet i package.json. Kan ikke garantere at du skal bruge alle pakkerne 😊

Start dit projekt op ved at køre cmd på mappen

npm run start

Så åbner browseren og du er klar til at gå i gang.

**Mappe struktur:**

* Index.html
* dist -> bundle.js
* node\_modules
* assets
* css
* src
* app.js

Mappen “dist” og ”bundle.js” bliver oprettet første gang du kører cmd > npm run webpack på dit projekt. Hvis du åbner på ”webpack.config”, kan du på linie 79 se hvilken mappe den laver et output til.   
  
I index.html refereres der til ”dist/bundle.js”: <script src="dist/bundle.js"></script>

* Mappen src bruges til alle dine js filer, der kan sagtens laves undermapper i ”src” mappen.
* Mappen ”css” indeholder en SASS fil.
* Mappen ”assets” indeholder lige nu to undermapper ”images” og ”sound”.

**gennemgang af:**

* package.json:
* webpack.config
* prod.config

**package.json**: Henvisning til alle hentede npm pakker, som installeres i ”node\_modules” mappen.

**Webpack(.config):** Opsætning af regler for export af kode og bundle dem til development brug. Se: <https://webpack.js.org/>

Der er i ”package.json” lavet små scripts, som kan kaldes og gøre bruge af webpack filerne:

"scripts": {

    "webpack": "webpack --watch",

    "build": "webpack --config prod.config.js",

    "start": "http-server -p 8080 -o http://localhost:8080/index.html"

  }

**Link til opskrift på at sætte sin egen boilderplate op (der er mange af dem derude):**

<https://medium.com/javascript-in-plain-english/the-basics-of-setting-up-an-es6-development-environment-with-webpack-and-babel-153d1bc3b4a5Hvis>

Hvis alt kører som det skal skulle du gerne, når du kører en cmd:

npm run start

Skulle du gerne få en besked: ”alt ok”

Den console.log kommer fra js filen initialize.js filen.

Åbn projektet i Vs code:

I roden af projektet kan du åbne ”app.js” filen.

import Initialize from './Initialize';

(function () {

console.log('starting the app');

let init = new Initialize();

})();

Der importeres en klasse ”Initialize”, som kommer fra Initialize.js

Vi laver en instance af Initalize klassen og den bliver eksekveret.

Det vil sige at alt hvad vi laver i Initialize klassen bliver eksekveret ”igennem” App.js

Lad os prøve at erklære klassen Initialize, som bliver kaldt fra App.js

Opret en ny fil i src mappen og navngiv den ”Initilalize.js” og erklær en ny klasse:

export default class Initialize {

constructor() {

}

}

Som udgangspunkt, skal alt hvad der skal importeres , eksporteres.

Så en ES6 klasse skal have en export default. En klasse har som default en ”constructor”, som automatisk bliver kaldt, når en klasse initialiseres.

Klassen kan bruges på flere måder. Du kan også oprette din Initialize klasse på følgende måde:

class Initialize {

constructor() {

}

}

export default Initialize;

Det er helt op til dig selv, hvilken model der passer dig bedst. Lad os arbejde lidt videre med Initialize klasse.

Opret en console.log (”im alive”) i Initialize konstruktøren. For at se ændringerne skal du køre en cmd på dit projekt.

npm run webpack

Når du har kørt den en gang vil webpack selv sørge for at opdatere dit projekt, når du gemmer.