

AUTOMAÇÃO FRONT END

Babel

Babel

- ### Compilador

Transforma código novo em código antigo. Ex: `const nome = 'Andre';`
vira `var nome = 'Andre';`.

- ### Browser Suporte

Para que um browser possa suportar algo novo de JavaScript é preciso que ele esteja atualizado, mas nem todo usuário possui a última versão do browser instalada.

- ### Can I Use

O site <https://caniuse.com/> mostra em quais browsers a novidade está disponível ou não.

Polyfill vs Transpiler

- Polyfill

Cria métodos / funções com o mesmo nome das atuais, porém utilizando código antigo para permitir o uso em browsers que não possuem a API.

- Transpiler

Transforma código novo em código antigo. Ou seja, transforma `const` em `var`.

Instalar Babel

- <https://babeljs.io/docs/en/usage>

```
$ npm install --save-dev  
• @babel/core @babel/cli  
  @babel/preset-env
```

Instala o Babel, a CLI, e configurações pré definidas

- ```
$ npm install @babel/polyfill
```

Instala os polyfill's

## webpack.config.js

---

Precisamos configurar o webpack, para utilizarmos o @babel/polyfill.

```
const path = require('path');

module.exports = {
 entry: ['@babel/polyfill', './js/script.js'],
 output: {
 path: path.resolve(__dirname, './'),
 filename: 'main.js'
 }
};
```

```
"scripts": {
 "dev": "webpack --mode development --watch",
 "build": "webpack --mode production"
},
```

## Fetch Polyfill

---

Nem todo browser suporta a Fetch API e por padrão o Babel não possui um polyfill para o Fetch. Podemos resolver isso instalando um polyfill externo.

```
$ npm install whatwg-fetch
```

```
const path = require('path');

module.exports = {
 entry: ['@babel/polyfill', 'whatwg-fetch', './js/script.js'],
 output: {
 path: path.resolve(__dirname, './'),
 filename: 'main.js'
 }
};
```