

Babel, instalação e build-step

Transcrição

Quando usamos Babel, estamos adicionando em nosso projeto um `build step`, ou seja, um passo de construção em nossa aplicação, isso significa que ela não pode ser consumida diretamente antes de passar por esse processo de construção. Sendo assim, o primeiro passo é instalar o Babel em nosso projeto para então configurá-lo.

O Babel depende que o Node.js esteja instalado e a boa notícia é que a infraestrutura do nosso curso já solicita a instalação desta plataforma para que o servidor web disponibilizado no curso pudesse funcionar.

Abra seu terminal favorito dentro da pasta `aluraframe/client`. Será dentro dela que criaremos o arquivo `package.json`, no qual os módulos do Node.js (Babel é um módulo) serão listados. Sua criação é simples, basta executar o comando `npm init` e teclar `ENTER` para todas as perguntas:

```
npm init
```

No final, será criado arquivo `aluraframe/client/package.json` com a seguinte estrutura:

```
{
  "name": "client",
  "version": "1.0.0",
  "description": "",
  "main": "index.js",
  "scripts": {
    "test": "echo \"Error: no test specified\" && exit 1"
  },
  "author": "",
  "license": "ISC"
}
```

Agora, precisaremos instalar o `babel-cli`, o módulo do Babel que facilita bastante seu uso:

```
npm install babel-cli@6.10.1 --save-dev
```

É importante que você use a mesma versão que usei durante o treinamento, porque ela foi testada e homologada por mim. Outras versões podem introduzir bugs e você ficará horas e horas no fórum procurando ajuda achando que é um problema no seu código quando não é. No final do treinamento, depois de ter certeza de que tudo está funcionando, atualize para a versão mais nova se assim desejar.

Instalar o `babel-cli` não é suficiente, precisaremos instalar o suporte ao ES2015. Para isso, vamos instalar o módulo `babel-preset-es2015`:

```
npm install babel-preset-es2015@6.9.0 --save-dev
```

Agora que temos os dois módulos instalados, precisamos indicar para o Babel que ele deve usar o módulo `babel-preset-es2015`. Para isso, vamos criar o arquivo `.babelrc` dentro de `aluraframe/client` com a seguinte configuração:

```
{  
  "presets": ["es2015"]  
}
```

Agora, vem uma parte **muito** importante. Vamos renomear a pasta `aluraframe/client/app` para `aluraframe/client/app-es6`. O motivo dessa alteração é que ela deixará clara que tudo que estiver dentro dela são scripts usando ES6. Babel irá analisar essa pasta e em seguida, criará `aluraframe/client/app`. É esta pasta que terá nosso código em ES5. Inclusive veja que nem precisaremos alterar as importações dos scripts de `aluraframe/client/index.html`.

Renomeie o `aluraframe/client/app` para `aluraframe/client/app-es6`