06

Agilizando o desenvolvimento

Transcrição

Podemos agilizar o processo de codificação e teste da nossa aplicação.

Para toda e qualquer alteração feita no nosso código, precisamos salvar o arquivo, parar o servidor, executá-lo novamente com `node server.js` e então testá-lo no navegador.

Seria mais interessante se tivéssemos um mecanismo que automaticamente atualizasse a instância do servidor a cada salvamento do projeto, permitindo que testássemos as alterações diretamente no navegador.

Existem várias ferramentas que nos possibilitam esse tipo de procedimento. No mundo Node, uma das mais conhecidas é o módulo `nodemon`.

Para instalarmos esse módulo, digitaremos o comando `npm install nodemon@1.18.4 --save-dev -save-exact`. Dessa forma, estaremos fixando a versão do módulo desejado e salvando-o como uma **dependência de desenvolvimento** da nossa aplicação - ou seja, não precisaremos dele para executar a aplicação, somente para desenvolvê-la.

Finalizado o processo de instalação, o Node irá inserir no `package.json` uma seção chamada `devDependencies`, listando somente o `nodemon` na versão especificada.

De volta ao terminal, executaremos novamente o `npm`, mas de maneira diferente:

```
npm install -g nodemon@1.18.4 --save-dev --save-exact
```

Dessa vez, estamos indicando que queremos instalar o módulo `nodemon` de maneira global - ou seja, para todo o sistema operacional -, de modo que possamos utilizá-lo a partir do Prompt de Comando. Terminada a instalação, estaremos aptos a rodar nossa aplicação por meio desse módulo, utilizando o comando `nodemon server.js`.

Agora, toda vez que salvarmos uma alteração qualquer no nosso código, o próprio `nodemon` se encarregará de reiniciar a aplicação, atualizando-a.

No arquivo `package.json`, temos uma seção `scripts`, que indica como o Node pode executar nosso projeto:

```
"scripts": {
  "test": "echo \"Error: no test specified\" && exit 1",
  "start": "node server.js"
},
```

Repare que, se executarmos o comando `npm test`, será devolvido um erro informando que não é um teste especificado para nossa aplicação. Já se usarmos `npm start`, o comando `node server.js` será executado. Vamos alterar esse campo para que `nodemon server.js` seja executado.

Feito isso, no terminal, poderemos utilizar os comandos especificados, desde que estejamos acessando a pasta do projeto. Se executarmos `npm start`, *o script* definido (nesse caso, `nodemon server.js`) rodará.

Essa é uma prática muito comum em projetos Node, agilizando bastante a execução e os testes. A seguir, começaremos a criar as telas do nosso sistema, a começar pela listagem de livros.

Vale ressaltar que feita a alteração no arquivo package.json não é necessária a instalação do nodemon como global.