01

Gerando arquivos para distribuir

Transcrição

Começando deste ponto? Você pode fazer o DOWNLOAD (https://s3.amazonaws.com/caelum-online-public/vue/stages/13-alurapic.zip) completo do projeto do capítulo anterior e continuar seus estudos a partir deste capítulo. Será necessário abrir seu terminal, entrar dentro da pasta alurapic e executar o comando npm install para baixar novamente todas as dependências da aplicação.

Terminamos nossa aplicação e você deve estar se perguntando qual são os arquivos que precisamos copiar para realizar seu *deploy* em nosso serviço de hospedagem favorito?

Durante nosso curso subimos a aplicação com o o comando `npm run dev` . Este modo de desenvolvimento disponibiliza os arquivos do projeto em memória para que tenhamos maior rapidez em sua construção. Para gerarmos os arquivos para produção, precisamos executar outro comando,

Vamos até o terminal e com CLI parado, dentro da pasta `alurapic` vamos executar a instrução:

```
npm run build
```

COPIAR CÓDIGO

Uma série de passos será realizado, como a concatenação e minificação de scripts e toda lógica de criação de bundle do Webpack será aplicada. No final

teremos a pasta *alurapic/dist* criada com o arquivo *build.js* e *build.map*. Por incrível que pareça, todos os nossos componentes, serviços e diretivas foram adicionados dentro do arquivo `build.js` . O arquivo `build.map` é apenas para ajudar a depurar o código. Sendo assim, basta enviarmos para o servidor o arquivo `alurapic/index.html` e a pasta `dist` com seu conteúdo.

No entanto, lembre-se que seu servidor, independente da linguagem utilizada, deve sempre retornar `index.html` para todas as requisições que forem feitas para ele. Por padrão, um servidor não adota esse comportamento. Aliás, você deve responder com `index.html` inclusive para páginas não encontradas.

Para que possamos fixar esse processo, faremos o deploy local da nossa aplicação.