escola britânica de artes criativas & tecnologia

Engenheiro Front-End

Automação de tarefas com Grunt

Neste módulo conhecemos o Grunt, uma ferramenta para automação de tarefas front-end.

Com ele podemos automatizar a compilação de SASS e LESS, comprimir arquivos, entre outras coisas.

Você pode consultar o código escrito durante o módulo clicando aqui.

Requisitos

O Grunt é executado a partir do NodeJS, portanto precisamos ter o Node e o NPM instalados.

Instalação

Primeiramente precisamos instalar o **Grunt CL**I de forma global:

```
npm install –g grunt-cli
```

Após isso, na pasta do projeto digitamos:

```
npm install ––save-dev grunt
```

Gruntfile

O arquivo de configuração do Grunt é o **Gruntfile.js**. Neste arquivo iremos configurar as tarefas e importar os plugins.

A configuração básica do Grunt é:

```
module.exports = function(grunt) {
    grunt.initConfig({
      pkg: grunt.file.readJSON("package.json");
});
}
```

No código acima fizemos a exportação do arquivo, que será acessado pelo Grunt, onde ele injetará o argumento "grunt".

Inicializamos a configuração com **grunt.initConfig**, onde, dentro da função, em **pkg**, importamos o arquivo **package.json**.

Tarefas

Para criar uma tarefa no Grunt devemos registrá-la. Uma tarefa no Grunt é basicamente uma função.

Escrevemos uma tarefa no Grunt assim:

```
// após grunt.initConfig({ ... })

grunt.registerTask("minhaTarefa", function() {
    // conteúdo da tarefas
    console.log("Olá Grunt");
});
```

Para executar a tarefa, no terminal digitamos: **grunt minhaTarefa**, no Windows é necessário adicionar o comando na seção de scripts, no arquivo **package.json:**

```
"scripts": {
    "grunt": "grunt"
}
```

Tarefas

Quando executamos apenas grunt, irá ocorrer um erro porque não possuímos uma tarefa default (padrão). Para resolver isso podemos registrar uma tarefa com o nome **"default"**.

Uma tarefa pode chamar outra, em **grunt.registerTask** onde inserimos o **callback**, também podemos inserir um **array** com os nomes das tarefas que iremos executar, por exemplo:

```
grunt.registerTask("default", ["minhaTarefa",
"minhaTarefa2"]);
```

Plugins

As automações no Grunt funcionam com o **uso de plugins.** Possuímos plugins para compilar SASS e LESS, para pré-processador código de estilos, minificar arquivos, limpar pastas, entre outros.

Os plugins também são tarefas, porém não utilizamos o grunt.registerTask para eles, mas sim a função **grunt.loadNpmTasks("nomeDoPacote"),** e dentro da **função initConfig**, após **"pkg",** configuramos o plugin.

Plugins

A configuração de um plugin pode ser dividida entre os ambientes, por exemplo: podemos compilar um código LESS para produção aplicando a minificação e compilar o mesmo arquivo, mas não minificando para o ambiente de desenvolvimento.

As configurações de plugins serão inseridas logo após a propriedade **"pkg"** dentro da função **grunt.initConfig.**

Plugins: exemplo de configuração

```
module.exports = function(grunt) {
    grunt.initConfig({
        pkg: grunt.file.readJSON('package.json'),
        less: {
          desenvolvimento: {
            files: {
              'final.css': 'origem.less'
            }
        }
    }
});

grunt.loadNpmTasks('grunt-contrib-less');
}
```

Plugins: exemplo de configuração

Na configuração, onde temos a palavra "desenvolvimento" podemos separar as configurações no que o Grunt chama de targets (alvos), então podemos ter um alvo para desenvolvimento e outro alvo para produção, enviando o arquivo final para outra pasta e comprimindo ele.

```
...
less: {
    desenvolvimento: { ... }
    producao: {
        options: { compress: true }
        files: { 'dist/final.css': 'origem.less' }
    }
}
```

Links úteis

- Documentação completa do Grunt