[Udemy] Curso de Gulp

Seção: 1 Introdução ao Gulp - Tentativa e Erro.

1. Conhecendo um pouco de Gulp
2. Um pouco de Nodejs para o contexto
4. Criando um ambiente de trabalho
5. Testando o Node e Npm
6. Instalando Gulp globalmente
7. Criando o package.json
8. Instalando o Gulp localmente
9. Comentando um pouco sobre o package.json
10. Criando o gulpfile.js
11. Configurando o gulpfile.js

Seção: 2 Gulp na Prática - Workflow Básico.

13. Baixando o Sublime Text 3
15. Instalando o PackageControl
    - Para testar o PackageControl
16. Instalando o Seti_UI (Plugin Sublime)
17. Instalando Gulp (Plugin Sublime)
    - Executar o gulp diretamente do Sublime Text
18. Instalando vários plugins no npm
    - Sintaxe básica para instalar um plugin
    - Sintaxe básica para instalar vários plugins
19. Configurando o Auto-Save do Sublime Text 3
20. Plugins essenciais no Sublime
    - Plugins Essenciais
    - Instalação dos plug-ins
    - Listar os plugins instalados
22. Desenvolvimento e Produção
    - Separar desenvolvimento (src) de produção (dist)
23. Tarefas do Gulp
24. Minificando HTML
    - Definir precedência/dependência de execução de tarefas
    - Utilização da tarefa watch()
26. Compilando Scss
    - Definir precedência/dependência de execução de tarefas e Utilização da tarefa watch()
28. Notificando com gulp-notify
    - Pesquisar documentação gulp-notify
29. Documentação
    - Plugins com prefixo gulp-
    - Plugins com (ou sem) prefixo gulp-
30. browser-sync
    - Como criar um servidor estático e fazer o auto-reload da página em tempo real

Seção: 1 Introdução ao Gulp - Tentativa e Erro.

1. Conhecendo um pouco de Gulp

- Gulp é um conjunto de ferramentas que ajuda você a automatizar tarefas dolorosas, demoradas e repetitivas em seu fluxo de trabalho (workflow) de desenvolvimento
- Plataforma Agnóstica: As integrações são incorporadas em todos as principais IDEs e as pessoas estão usando gulp com PHP, .NET, Node.js, Java e outras plataformas.
- Forte Ecossistema: Use os módulos do "npm" para fazer o que quiser, com mais de 2000 plugins para transmissão de arquivos em fluxo contínuo
- Simples: Ao fornecer apenas uma superfície mínima da API, o gulp é fácil de aprender e simples de usar

2. Um pouco de Nodejs para o contexto

Antes de começar, um pouco Node para o contexto, ao contrário de qualquer outro programa de computador, você não poderá iniciar o Gulp a partir do seu dock ou barra de tarefas, você precisa de um servidor “Node” para rodar as tarefas do Gulp.

Você usará instruções em linha de comando através de uma ferramenta, como o Terminal ou o prompt de comando do Windows.

Para continuar, você deve instalar o Node.js e/ou npm em sua máquina.

- Emulador de Terminal (Linux) para Windows: http://cmder.net/

4. Criando um ambiente de trabalho

```
mkdir gulp
cd gulp
mkdir curso-gulp
cd curso-gulp
```

5. Testando o Node e Npm

- Verificar se o Node e NPM estão instalados

```
node -v
npm -v
```

6. Instalando Gulp globalmente

- Verificar se o Gulp está instalado

```
gulp -v
```

- Caso não esteja, instalar o Gulp globalmente

```
npm i gulp -g
```

7. Criando o package.json

a. Abrir o terminal no diretório de trabalho e executar:

```
npm init
```

b. Para cada opção exibida (e a possível resposta default), apertar <Enter>

```
"name": "curso-gulp",
"description": "Curso de Gulp - Tentativa e erro",
"keywords": [
    "gulp"
  ],
"author": "Fabio",
```

c. Após todas as perguntas, é exibido uma prévia de como ficará o arquivo package.json. Tecle <Enter> para criar o arquivo

d. Ao abrir o arquivo, temos um nome, uma versão, uma descrição, alguns scripts, um autor e uma licença

8. Instalando o Gulp localmente

```
npm i gulp --save-dev
```

- Verificar a versão global e local do Gulp

```
gulp -v
```

9. Comentando um pouco sobre o package.json

- Arquivo de registro do projeto, ou seja, todos os aplicativos que forem instalados, a partir de agora, devem estar registrado no arquivo package.json.

```
cat package.json
```

- Em um momento futuro, utilizar o comando abaixo para instalar todas as dependências/packages do projeto

```
npm install
```

10. Criando o gulpfile.js

Para o Gulp funcionar, é necessário o arquivo de configuração gulpfile.js.

- Criar o arquivo de configuração gulpfile.js

```
touch gulpfile.js
```

11. Configurando o gulpfile.js

```
vi gulpfile.js
```

a. Adicionar no arquivo gulpfile.js

```
var gulp = require("gulp");

gulp.task("default", function() {
  return
})
```

b. Verificar o conteúdo do gulpfile.js

```
cat gulpfile.js
```

c. Executar o comando

```
gulp
```

-- Resultado esperado (arquivo gulpfile.js configurado corretamente):

```
[15:43:29] Using gulpfile ~/git/gulp/curso-gulp/gulpfile.js
[15:43:29] Starting 'default'...
[15:43:29] Finished 'default' after 72 µs
```

Seção: 2 Gulp na Prática - Workflow Básico.

13. Baixando o Sublime Text 3

- https://www.sublimetext.com/

15. Instalando o PackageControl

a. Acessar https://packagecontrol.io/
b. Clicar em "Installation"
c. Escolher a versão do Sublime Text e copiar o código logo abaixo
d. Abrir o Sublime Text, clicar em "View" => "Show Console"
e. Colar no input text do console e clicar em <Enter>
f. Fechar e Abrir o Sublime Text

➢ **Para testar o PackageControl**

- Clicar em "Preferences" => "PackageControl"

OU

- Atalho: Ctrl + Shift + P => pesquisar por "PackageControl"

16. Instalando o Seti_UI (Plugin Sublime)

- Seti_UI: Tema que facilita a visualização dos arquivos do projeto (https://packagecontrol.io/packages/Seti_UI)

```
mkdir curso-gulp-II
cd curso-gulp-II/
npm init
```

```
"description": "Curso de gulp",
"keywords": [
  "curso",
  "gulp",
  "FabioEw"
],
"author": "Fabio Ewerton",
"license": "MIT"
```

a. No Sublime text, clicar em "Project" => "Add Folder to Project..." e adicionar o diretório recém-criado curso-gulp-II
b. Clicar em "Ctrl + Shift + P" => pesquisar por "install" => clicar em "Package Control: Install Package"
c. Pesquisar por "seti" e clicar em "Seti_UI"
d. Em "Package Control Messages", copiar o conteúdo entre chaves do tópico "## Example"
e. Clicar em "Preferences" => "Settings"
f. Colar acima de "ignored_packages":

--Resultado esperado/configurado

```
{
  "theme": "Seti.sublime-theme",

  "caret_extra_width": 2,      //to have a wider/thicker caret
  "caret_extra_bottom": 3,     //to make the caret = to the line height
                               //(the theme currently support 0,3,5)
  "caret_extra_top": 3,

  "overlay_scroll_bars": "enabled",    //to show scrollbars only when
                                       //scrolling
  "highlight_line": true,      //to highlight the current line

  "ignored_packages":
  [
    "Vintage"
  ],

  //"color_scheme": "Packages/Color Scheme - Default/Celeste.sublime-
//color-scheme", //optional

  "tab_size": 2,
  "translate_tabs_to_spaces": true,
  "word_wrap": "true",
  "font_size": 10
}
```

g. Ctrl + S para salvar

17. Instalando Gulp (Plugin Sublime)

a. Abrir o terminal no diretório de trabalho e executar:

```
npm init
gulp -v
npm i gulp --save-dev
touch gulpfile.js
```

b. Clicar em "Ctrl + Shift + P" => pesquisar por "install" => clicar em "Package Control: Install Package"
c. Pesquisar por "gulp", clicar em "Gulp" e aguardar a conclusão da instalação
d. No arquivo gulpfile.js, inserir o trecho abaixo:

```
var gulp = require("gulp");

gulp.task("default", function() {
  return
})
```

c. Executar o comando no terminal

```
gulp
```

-- Resultado esperado (arquivo gulpfile.js configurado corretamente):

```
[10:03:57] Using gulpfile ~/git/gulp/curso-gulp-II/gulpfile.js
[10:03:57] Starting 'default'...
[10:03:57] Finished 'default' after 69 µs
```

➢ **Executar o gulp diretamente do Sublime Text**

- https://packagecontrol.io/packages/Gulp

a. Clicar em "Preferences" => "Package Settings" => "Gulp" => "Settings - Default"
b. Incluir em "exec_args", o caminho onde está instalado o node, conforme trecho abaixo:

```
// Override your environment PATH
"exec_args": {
  "path": "/home/wjuniori/.nvm/versions/node/v8.11.3/bin/"
},
```

c. Clicar em "Ctrl + Shift + P" => pesquisar por "gulp" => clicar em "Gulp"
d. Escolher o arquivo gulpfile.js (quando houver mais de um projeto no Sublime Text)
e. Na primeira vez, é criado um arquivo de cache (.sublime-gulp.cache)
f. Escolher a tarefa 'default' do arquivo gulpfile.js para executar
g. Verificar o status da execução no console

```
Running 'default'...
[11:23:00] Using gulpfile ~/git/gulp/curso-gulp-II/gulpfile.js
[11:23:00] Starting 'default'...
[11:23:00] Finished 'default' after 90 µs
```

18. Instalando vários plugins no npm

➢ **Sintaxe básica para instalar um plugin**

```
npm i gulp --save-dev
```

➢ **Sintaxe básica para instalar vários plugins**

```
npm i gulp-sass gulp-htmlmin gulp-notify del --save-dev
```

- Referências/Documentação

https://www.npmjs.com/package/gulp-htmlmin
https://www.npmjs.com/package/gulp-sass
https://www.npmjs.com/package/gulp-notify
https://www.npmjs.com/package/del

19. Configurando o Auto-Save do Sublime Text 3

a. Clicar em "Ctrl + Shift + P" => pesquisar por "settings" => clicar em "Preferences:Settings"
b. Em "Preferences.sublime-settings - Default", pesquisar por "save"
c. Copiar "save_on_focus_lost:" false
d. Em "Preferences.sublime-settings - User", colar e substituir false por true
e. Salvar

20. Plugins essenciais no Sublime

- Write Less, Do More (Escreva Menos, Faça Mais)

➢ **Plugins Essenciais**

Package Control	Gerenciador de pacotes
Seti_UI	Tema
HTML5	Snippets (atalhos/abreviações) para HTML5
SCSS	Sintaxe para SCSS
Emmet	Atalhos para HTML e CSS
AutoFileName	Auto complete de caminhos e extensões de arquivos
Gulp	Abreviações do Gulp

➢ **Instalação dos plug-ins**

a. Clicar em "Ctrl + Shift + P" => pesquisar por "install" => clicar em "Package Control: Install Package"
b. Pesquisar por "html", clicar em "HTML5" e aguardar a conclusão da instalação

c. Clicar em "Ctrl + Shift + P" => pesquisar por "install" => clicar em "Package Control: Install Package"
d. Pesquisar por "SCSS", clicar em "SCSS" e aguardar a conclusão da instalação

e. Clicar em "Ctrl + Shift + P" => pesquisar por "install" => clicar em "Package Control: Install Package"
f. Pesquisar por "autofile", clicar em "AutoFileName" e aguardar a conclusão da instalação

g. Clicar em "Ctrl + Shift + P" => pesquisar por "install" => clicar em "Package Control: Install Package"
h. Pesquisar por "emmet", clicar em "Emmet" e aguardar a conclusão da instalação

i. Reiniciar o Sublime Text

➢ **Listar os plugins instalados**

- Clicar em "Ctrl + Shift + P" => pesquisar por "list packa" => clicar em "Package Control: List Packages"

22. Desenvolvimento e Produção

➢ **Separar desenvolvimento (src) de produção (dist)**

```
mkdir src dist
```

23. Tarefas do Gulp

task()	Define tarefas no Gulp; função de execução das tarefas dentro do Gulp

src()	Arquivos que entrarão no fluxo de tarefas para serem tratados ou manipulados
dest()	Destino dos arquivos que já passaram pelo fluxo de tarefa (geralmente, o diretório dist)
watch()	Assite/observa os arquivos e faz alguma tarefa quando estes são alterados
pipe()	Concatena tarefas no gulp

24. Minificando HTML

a. No arquivo gulpfile.js, adicionar

```
var html = require('gulp-htmlmin');

//html
gulp.task('html', function() {
  return gulp.src('./src/index.html')
    .pipe(html({collapseWhitespace:true}))
    .pipe(gulp.dest('./dist/'));
});
```

b. Clicar em "Ctrl + Shift + P" => pesquisar por "gulp" => clicar em "Gulp"
c. Escolher o arquivo gulpfile.js (quando houver mais de um projeto no Sublime Text)
d. Na primeira vez, é criado um arquivo de cache (.sublime-gulp.cache)
e. Escolher a tarefa 'html' do arquivo gulpfile.js para executar
f. Verificar o status da execução no console e o arquivo minificado em dist

```
Running 'html'...
[13:59:38] Using gulpfile ~/git/www/curso-gulp-II/gulpfile.js
[13:59:38] Starting 'html'...
[13:59:38] Finished 'html' after 34 ms
```

➢ **Definir precedência/dependência de execução de tarefas**

- Neste caso, a tarefa 'html' será executada antes da execução da tarefa default.

```
gulp.task('default',['html'], function() {
  return
});
```

-- Resultado esperado

```
Running 'default'...
[14:08:28] Using gulpfile ~/git/gulp/curso-gulp-II/gulpfile.js
[14:08:28] Starting 'html'...
[14:08:28] Finished 'html' after 36 ms
[14:08:28] Starting 'default'...
[14:08:28] Finished 'default' after 28 µs
```

➢ **Utilização da tarefa watch()**

- Neste caso, a tarefa 'default' executará o 'html', depois o 'default' e, em seguida, ficará aguardando a alteração do arquivo index.html, através da função watch();

```
gulp.task('default',['html'], function() {
```

```
  gulp.watch('./src/index.html', ['html']);
});
```

26. Compilando Scss

a. No arquivo gulpfile.js, adicionar

```
var sass = require('gulp-sass');

//sass
gulp.task('sass', function() {
    return gulp.src('./src/scss/style.scss')
      .pipe(sass({outputStyle:"compressed"}))
      .pipe(gulp.dest('./dist/css/'));
});
```

b. Clicar em "Ctrl + Shift + P" => pesquisar por "gulp" => clicar em "Gulp"
c. Escolher o arquivo gulpfile.js (quando houver mais de um projeto no Sublime Text)
d. Na primeira vez, é criado um arquivo de cache (.sublime-gulp.cache)
e. Escolher a tarefa 'sass' do arquivo gulpfile.js para executar
f. Verificar o status da execução no console e o arquivo compilado em /dist/css/

➢ **Definir precedência/dependência de execução de tarefas e Utilização da tarefa watch()**

- Neste caso, as tarefas 'html' e 'sass', nesta ordem, serão executadas antes da execução da tarefa 'default'. E, em seguida, ficará aguardando a alteração do arquivo index.html ou style.scss, através da função watch();

```
gulp.task('default',['html', 'sass'], function() {
  gulp.watch('./src/index.html', ['html']);
  gulp.watch('./src/scss/style.scss', ['sass']);
});
```

-- Resultado esperado

```
Running 'default'...
[14:45:28] Using gulpfile ~/git/gulp/curso-gulp-II/gulpfile.js
[14:45:28] Starting 'html'...
[14:45:28] Starting 'sass'...
[14:45:28] Finished 'html' after 39 ms
[14:45:28] Finished 'sass' after 31 ms
[14:45:28] Starting 'default'...
[14:45:28] Finished 'default' after 7.95 ms
```

28. Notificando com gulp-notify

- Notificação de erros (como tratar erros no Gulp): mostrar erros, por exemplo, ao realizar um save errado.

➢ **Pesquisar documentação gulp-notify**

a. Acessar https://gulpjs.com/
b. Clicar em "Plugins"

c. Pesquisar por "gulp-notify"
d. Clicar em "gulp-notify", sendo redirecionado para o seu repositório npm
e. Clicar em "notify.onError()" no "Table of Contents"
f. Em "Or simply:", copiar a linha ".on("error", notify.onError("Error: <%= error.message %>"))"
g. No arquivo gulpfile.js, colar embaixo da linha do html e do css, conforme trecho abaixo:

```
var notify = require('gulp-notify');

//sass
gulp.task('sass', function() {
    return gulp.src('./src/scss/style.scss')
      .pipe(sass({outputStyle:"compressed"}))
      .on("error", notify.onError("Error: <%= error.message %>"))
      .pipe(gulp.dest('./dist/css/'))
});
```

```
//html
gulp.task('html', function() {
  return gulp.src('./src/index.html')
    .pipe(html({collapseWhitespace:true}))
    .on("error", notify.onError("Error: <%= error.message %>"))
    .pipe(gulp.dest('./dist/'))
});
```

h. Executar o gulp
i. Caso haja erro nos arquivos index.html ou style.scss, será informado pelo "gulp-notify: [Error running Gulp] Error:"

29. Documentação

➢ **Plugins com prefixo gulp-**

a. Acessar https://gulpjs.com/
b. Clicar em "Plugins"
c. Pesquisar por "gulp-<name>"
d. Clicar no plugin desejado ("gulp-<name>"), sendo redirecionado para o seu repositório npm
e. Ler a documentação

➢ **Plugins com (ou sem) prefixo gulp-**

a. Acessar https://www.npmjs.com/
b. Em "Search packages", pesquisar por "gulp-<name>" ou "<plugin-name>"
c. Clicar no plugin desejado ("gulp-<name>" ou "<plugin-name>")
d. Ler a documentação

30. browser-sync

- Dinamizar os saves ao longo do desenvolvimento, ou seja, ao realizar um save no arquivo html, automaticamente o resultado é exibido no browser

a. Acessar https://browsersync.io/
b. Clicar em "Documentation"
c. Em "Sections", clicar em "+ Gulp"

➢ **Como criar um servidor estático e fazer o auto-reload da página em tempo real**

a. Instalar o browser-sync

```
npm i browser-sync --save-dev
```

b. Escrever a tarefa do browser-sync (conforme documentação em "Install") no arquivo gulpfile.js

```
var browserSync = require('browser-sync').create();

//browser-sync
gulp.task('BS', function() {
    browserSync.init({
      server:{
        baseDir: './dist/'
      }
    })
});
```

c. Na documentação, em "SASS + CSS Injecting", chamar o .stream() depois do gulp.dest (o browser-sync só se preocupa com o seu HTML e CSS quando ele é compilado)

```
//sass
gulp.task('sass', function() {
  [...]
  .pipe(gulp.dest('./dist/css/'))
  .pipe(browserSync.stream())
});

//html
gulp.task('html', function() {
  [...]
  .pipe(gulp.dest('./dist/'))
  .pipe(browserSync.stream())
});
```

d. Transpor as tarefas watch de 'default' para 'BS' assim como a precedência/dependência 'html' e 'sass';
e. Na tarefa 'default', adicionar a precedência/dependência 'BS'
f. Executar no terminal

```
gulp
```

Será configurado um servidor local HTTP (http://localhost:3000) com os arquivos do diretório ./dist/ e exibido no navegador/browser padrão.

g. Efetuar modificações no arquivo HTML e SCSS e verificar, em tempo real, o resultado no navegador