
node
JS
TM

Preparação do Ambiente

Seguir as orientações destes slides para preparar o ambiente para as próximas aulas (a partir de 28/08).

Instalando o Visual Studio Code

- Iremos utilizar como IDE de programação o software Visual Studio Code: https://code.visualstudio.com/download
- Basta seguir as instruções do site.

Instalando o node.js

- Para rodar o node, você precisa primeiro instalá-lo: http://nodejs.org
- Basta seguir as instruções do site.
- Ao concluir a instalação, você consegue verificar o resultado utilizando o seguinte comando no *prompt:*

```
node -v
```

Testando: Hello World

- Vamos testar se tudo está ok com um Hello World.
- No *prompt* digite `node` e pressione <enter> para acessar o REPL (*Read-Eval-Print-Loop*) que permite executar código Javascript diretamente no terminal.
- Digite `console.log('Hello World');`
- Pressione <enter>
- O texto Hello World será exibido na tela!

Package Express

- O Express é um *middleware* web.
- Uma camada que fica entre o HTTP server criado usando o módulo http do Node.js e a sua aplicação web, interceptando cada uma das requisições, aplicando regras, carregando telas, servindo arquivos estáticos, etc.

Instalando *Package* Express

- Vamos instalar o nosso primeiro pacote node utilizando o npm: **Express**.
- Este pacote permite que você gere de forma muito simples uma aplicação web em node com os requisitos mínimos necessários, incluindo a criação do seu servidor web.
- Para instalar usamos:

```
npm install -g express
```

```
npm install -g express-generator@4
```

Package Express

- Para criar o seu projeto node, usamos:

```
express projeto1 -e --git
```

- Este comando criará uma pasta chamada `projeto1` com todos os arquivos e configurações iniciais para rodar o seu app node.
- A diretiva `-e` permite usar a view-engine ejs (Embedded JavaScript).
- A diretiva `--git` deixa seu projeto preparado para versionamento com git.

package.json

- Arquivo essencial para um projeto Node.js.
- Um package.json mal escrito pode causar bugs ou impedir o funcionamento correto do seu módulo, pois ele possui alguns atributos chaves que são compreendidos pelo Node.js e NPM.

package.json

```
{
  "name": "projeto1",
  "version": "0.0.0",
  "private": true,
  "scripts": {
    "start": "node ./bin/www"
  },
  "dependencies": {
    "cookie-parser": "~1.4.3",
    "debug": "~2.6.9",
    "ejs": "~2.5.7",
    "express": "~4.16.0",
    "http-errors": "~1.6.2",
    "morgan": "~1.9.0"
  }
}
```

package.json

```
{
  "name": "projeto1",
  "version": "0.0.0",
  "private": true,
  "scripts": {
    "start": "node ./bin/www"
  },
  "dependencies": {
    "cookie-parser": "~1.4.3",
    "debug": "~2.6.9",
    "ejs": "~2.5.7",
    "express": "~4.16.0",
    "http-errors": "~1.6.2",
    "morgan": "~1.9.0"
  }
}
```

Na seção scripts, você pode definir “comandos” para serem executados pelo npm.

Em `./bin/www` estão todas as configurações do seu servidor (normalmente na porta 3000).

O express já cria o comando `start` para que você possa inicializar o seu servidor e sua aplicação web.

Executando sua aplicação

- Para executar sua aplicação, acesse a pasta `projeto1` criada.
- Digite: `npm install`
- Todas as dependências informadas no `package.json` serão instaladas no seu projeto.
- Em seguida, digite: `npm start`
- O seu servidor disponível em `./bin/www` será inicializado.
- No seu browser, acesse: `localhost:3000`

Executando sua aplicação