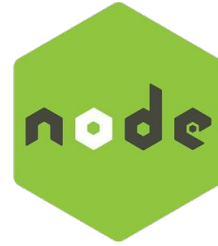


# NodeJS



Prof. Celso Henrique Masotti

# Necessidades Prévias

- Ter o node.JS instalado;
- Ter o gerenciador de pacotes (NPM) instalado;
- Ter um editor de códigos IDE<sup>(\*)</sup>, dê preferência ao “[Visual Studio Code](#)”
- Conhecer JavaScript básico.

(\*) IDE, do inglês Integrated Development Environment ou Ambiente de Desenvolvimento Integrado, é um programa de computador que reúne características e ferramentas de apoio ao desenvolvimento de software com o objetivo de agilizar este processo, como um editor de texto para a digitação de códigos.

# Você verá nesta aula:

Objetivo: imprimir texto no terminal (Prompt).

Estrutura de Pastas no Prompt

Estrutura de Pastas Windows Explorer

Acessando Pasta com VSCode

Controles: expandir pasta

Controle: ícones

Criando Arquivo de JavaScript

Salvando o Arquivo

Uso do ponto e vírgula

Imprimindo no Terminal

Aviso de Erro

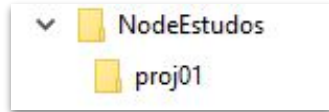
# Estrutura de Pastas no Prompt

- Abra o terminal do windows: [Teclas] Win + R; [digite] cmd.exe; [Enter]
- No terminal iremos à raiz do drive. digite: cd c:\
- Criar pasta (diretório) “NodeEstudos”: md NodeEstudos
- “Entrar” na pasta: cd NodeEstudos
- Criar pasta (diretório) “proj01”: md proj01
- “Entrar” na pasta: cd proj01

```
C:\Users\Samsung>cd c:\  
c:\>md NodeEstudos  
c:\>cd NodeEstudos  
c:\NodeEstudos>md proj01  
c:\NodeEstudos>cd proj01  
c:\NodeEstudos\proj01>
```

# Estrutura de Pastas Windows Explorer

- Abra o Windows Explorer: [teclas] win + E
- Na raiz do drive “c:” vamos criar uma pasta: [teclas] Ctrl + Shift + N
- Digite o nome da pasta “NodeEstudos” e aperte a tecla Enter;
- Abra a pasta recém criada e crie nova pasta: [teclas] Ctrl + Shift + N
- Digite o nome da pasta “proj01”

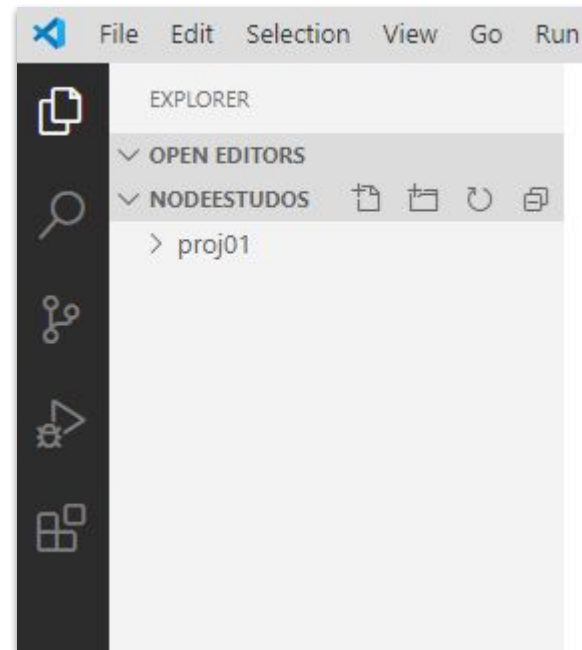


# Acessando Pasta com VSCode

- Abra o Visual Studio Code.
- Abrir pasta do projeto pelo teclado é Ctrl + K e Ctrl + O, ou use o mouse no menu “File” -> “Open Folder”
- Indique a pasta recém criada “NodeEstudos”



Caso não esteja visualizando os nomes das pastas, clique com o mouse no botão “Explorer” (explorar).



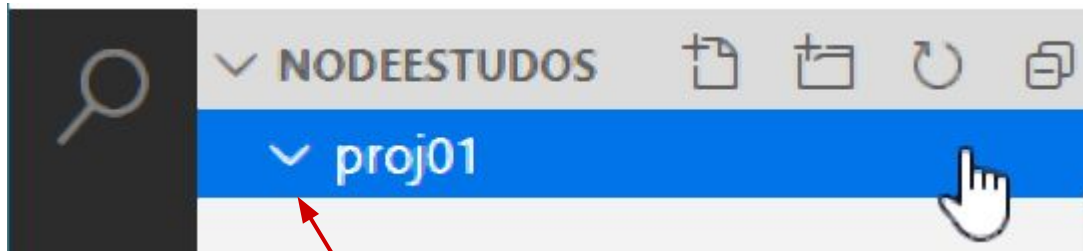
# Controles: expandir pasta

Expanda a pasta “proj01” clicando sobre sua nomenclatura.

Observe a sinalização do símbolo “>” para “v”.



Pasta não expandida

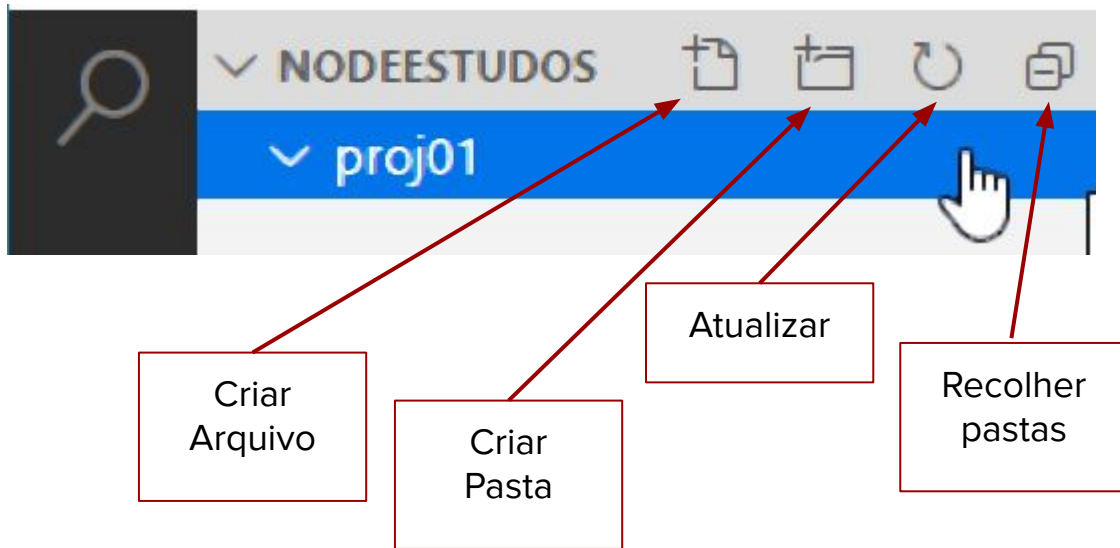


Pasta expandida

# Controle: ícones

Atente-se aos controles que aparecem ao aproximar o mouse:

- criar arquivo
- criar pasta
- atualizar
- recolher pastas



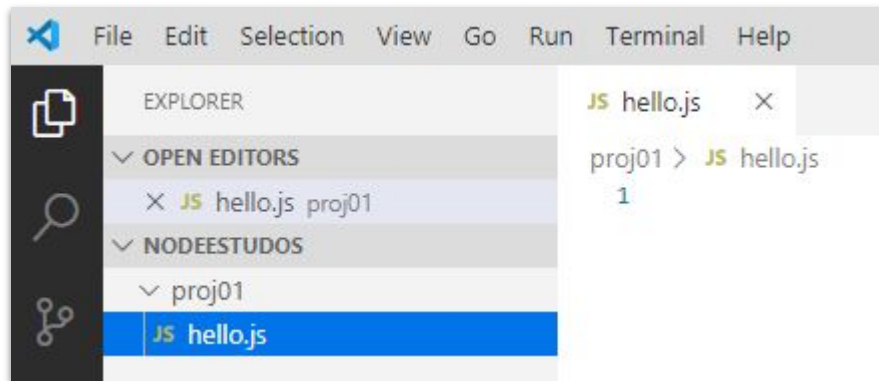


# Criando Arquivo de JavaScript

Quando a pasta “proj01” estiver selecionada, clique no ícone “Criar arquivo” (Create File).

Dê o nome “hello.js” para o arquivo.

A extensão “js” especifica arquivos de JavaScript.



# Imprimindo em Console

No interior do arquivo vamos digitar código para imprimir no console “console.log()” e entre os parênteses inseriremos o texto que desejamos “Hello World”. Como se trata de texto, ou seja, de “string”, este deve ser digitado sempre entre aspas. Veja:

```
console.log(“Hello World”)
```





# Uso do ponto e vírgula

É comum usarmos o “;” (ponto e vírgula) no final de uma linha de comando do javascript para informar que a linha foi finalizada. Mas, desde a versão ES6 esta pontuação não é mais obrigatória, ou seja, você pode ou não fazer uso dela. A ausência está relacionada com ASI (Automatic Semicolon Insertion) [[saiba mais](#)].



```
JS hello.js  X
proj01 > JS hello.js
1 console.log("Hello World");
```



```
JS hello.js  X
proj01 > JS hello.js
1 console.log("Hello World");
```

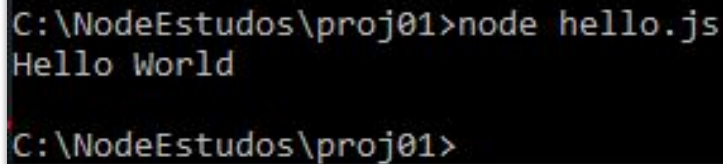
# Imprimindo no Terminal

Agora, no terminal (prompt), digite:

`node hello.js`

e aperte o “enter” do teclado.

Se você fez tudo direito, a impressão do texto ocorrerá.



```
C:\NodeEstudos\proj01>node hello.js  
Hello World  
C:\NodeEstudos\proj01>
```

# Aviso de Erro

Caso seu algoritmo tenha erros, normalmente o node informa onde provavelmente o erro se encontra. Exemplo. Vamos fazer uma chamada para uma função que não existe.

Provável linha

```
proj01 > JS soma.js > ...  
1   function somar(a,b){  
2     |   return a + b;  
3   }  
4  
5   console.log(sub(12,25))
```

```
C:\NodeEstudos\proj01\soma.js:5  
console.log(sub(12,25))  
      ^
```

erro

```
ReferenceError: sub is not defined  
    at Object.<anonymous> (C:\NodeEstudos\proj01\soma.js:5:9)  
[90m    at Module._compile (internal/modules/cjs/loader.js:1123:30)[39m  
[90m    at Object.Module._extensions..js (internal/modules/cjs/loader.js:1143:10)[39m  
[90m    at Module.load (internal/modules/cjs/loader.js:972:32)[39m  
[90m    at Function.Module._load (internal/modules/cjs/loader.js:872:14)[39m  
[90m    at Function.executeUserEntryPoint [as runMain] (internal/modules/run_main.js:71:12)[39m  
[90m    at internal/main/run_main_module.js:17:47[39m
```

Obrigado

Prof° Celso H. Masotti