

9. NPM

NPM (Node Package Manager) é o gerenciador de pacotes padrão do NodeJS, e geralmente é instalado junto com o próprio NodeJS. Para verificar a versão instalada, basta executar o seguinte comando no terminal:

```
npm -v
```

Para iniciarmos um novo projeto, primeiro criamos uma nova pasta, em seguida, utilizamos o comando `cd` para navegar para a pasta criada:

```
cd ./nome-da-pasta
```

Após isso, utilizamos o NPM, é necessário executar o comando de inicialização de projeto:

```
npm init -y
```

O comando acima iniciar o projeto, respondendo confirmando todas as opções padrão do `npm`, um novo arquivo será gerado na nossa pasta, chamado `package.json`, esse arquivo é responsável por armazenar as configurações do nosso projeto.

```
{
  "name": "nome-da-pasta",
  "version": "1.0.0",
  "description": "",
  "main": "index.js",
  "scripts": {
    "test": "echo \"Error: no test specified\" && exit 1"
  },
  "keywords": [],
  "author": "",
  "license": "ISC"
}
```

Note que há uma atributo chamado "main", ele indica o arquivo principal do nosso projeto.

Além disso, há uma propriedade chamada "scripts", nela podemos configurar comandos personalizados para serem executados através da instrução `npm run [nome-do-comando]`, no caso do exemplo acima, o comando `test` pode ser executado da seguinte forma:

```
npm run test
```

O nome do comando é sua chave no objeto "scripts".

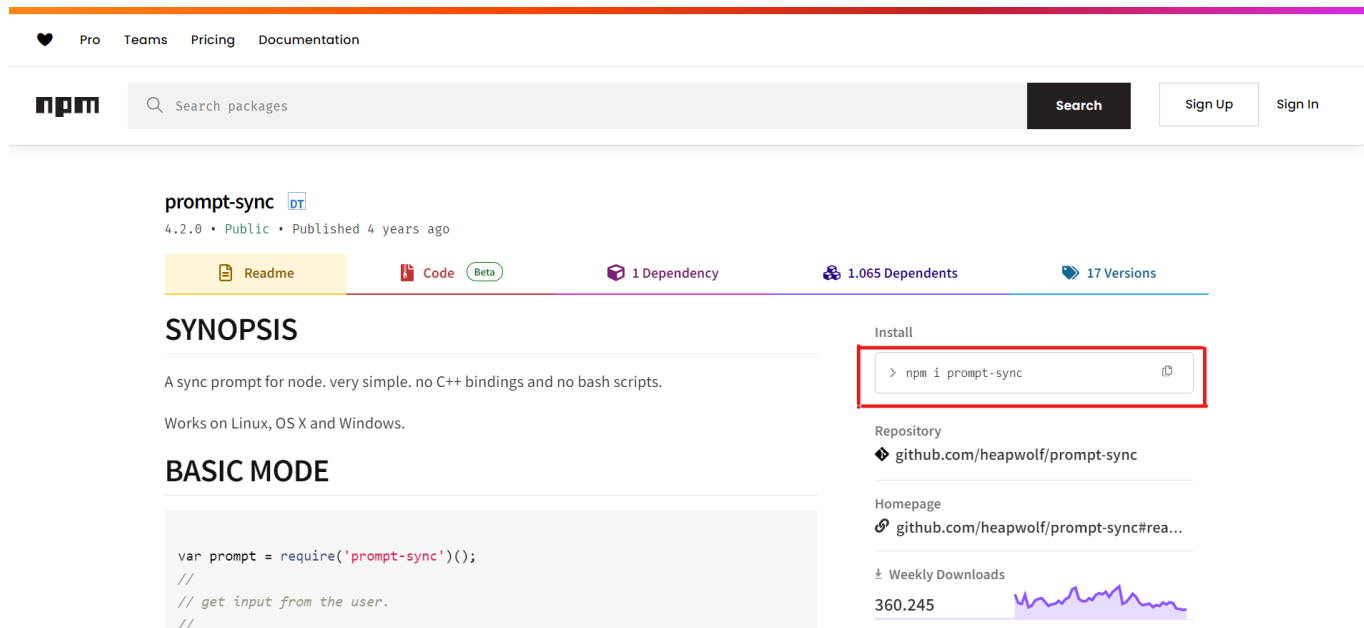
Por fim, precisamos adicionar um atributo "type" para conseguirmos utilizar as importações normalmente, fazemos isso adicionando o seguinte valor:

```
{
  ...
  "type": "module"
  ...
}
```

Instalação de Dependências

Outra função do NPM é adicionar dependências ao nosso código, quando precisamos realizar uma tarefa mais complexa, como gerar tokens para autenticação, ou simplesmente ler inputs do console, podemos buscar um pacote no [site do NPM](#) que satisfaça nossas necessidades.

Após encontrarmos um pacote com as funções que precisamos, basta executar o comando de instalação exibido na aba direita do site:



The screenshot shows the NPM website interface. At the top, there's a navigation bar with links like 'Pro', 'Teams', 'Pricing', and 'Documentation'. Below that is the NPM logo and a search bar. The main content area displays the details for the 'prompt-sync' package. It includes the package name, version (4.2.0), and publication date. There are buttons for 'Readme', 'Code', and 'Beta'. Statistics show 1 dependency, 1.065 dependents, and 17 versions. The 'SYNOPSIS' section describes it as a sync prompt for node. The 'BASIC MODE' section shows a code snippet. On the right, the 'Install' section is highlighted with a red box, showing the command 'npm i prompt-sync'. Below that, the repository and homepage are listed, along with a graph of weekly downloads showing 360,245 downloads.

O pacote acima é o prompt-sync, iremos utilizá-lo para ler informações inseridas no console.

Após realizarmos sua instalação, podemos criar um arquivo index.js e adicionar o seguinte código:

```
import promptSync from "prompt-sync";

const prompt = promptSync();
```

É possível também utilizar a sintaxe de `require`, porém por questões estéticas prefiro `import` 😊

Esse código foi extraído dessa [pergunta no StackOverflow](#)

Após realizarmos essa configuração inicial, podemos utilizar a variável `prompt` para enviar "perguntas" ao console, o valor digitado será retornado para o código, e podemos armazená-lo em uma variável da seguinte forma:

```
const nome = prompt("Qual é o seu nome? ");
```

A partir desse momento, podemos adicionar alguma interação entre o console e nossos códigos.

Agora, façamos os seguintes testes, vamos inserir um valor numérico e um booleano, em seguida, vamos verificar o tipo dos dados inseridos:

```
const nome = prompt("Qual é o seu nome? ");
const idade = prompt("Qual é a sua idade? ");
const dia = prompt("Hoje é sexta-feira? ");

console.log(typeof nome);
console.log(typeof idade);
console.log(typeof dia);
```

Isso aconteceu pois os valores lidos pelo console são somente de texto, por isso o JavaScript entende que o tipo do valor (`typeof`) é `string`. Podemos realizar conversões de dados utilizando funções como `parseInt`, ou no caso de booleano, verificar se o texto inserido é igual a `true`, essas são algumas formas de converter os valores inseridos.

Exercícios

Iremos refazer os exercícios da aula passada, porém dessa vez iremos utilizar valores inseridos no console.

Caso os exercícios sejam concluídos, vamos realizar o seguinte exercício:

Calculadora

Vamos criar uma calculadora simples utilizando o console, vamos declarar duas variáveis, `a` e `b`, que receberão dois valores por padrão, vamos iniciar com 5 e 9.

Em seguida, vamos pedir que o usuário escolha a operação a ser realizada com o `prompt`:

```
const operacao = prompt(`Escolha uma operação:  
1. Soma  
2. Subtração  
3. Divisão  
4. Multiplicação  
5. Sair `);
```

Podemos utilizar um `switch` ou um `if` para selecionar a operação desejada, caso o usuário escolha a opção "5", o código deve simplesmente encerrar a execução.

Por fim, iremos implementar cada uma das opções possíveis para cada uma das operações.