Javascript

A linguagem Javascript não tem palavras reservadas para definição de visibilidade de uma classe, no caso js não trabalha com modificadores de acesso. Definição de classe em javaScript:

JavaScript é uma linguagem baseada em protótipos e não contém a declaração , como vemos em C++ ou Java. Isso, às vezes, causa confusão em programadores(as) acostumados(as) a linguagens com uma declaração para classes. Em vez disto, JavaScript usa funções como classes. Definir uma classe-função é tão fácil quanto definir uma função. Para criar propriedades podemos usar a palavra var ou this. Se utilizarmos a palavra var o atributo vai ficar privado e se utilizarmos a palavra this o atributo vai ficar publico. BASICAMENTE não existe o conceito *class* em javaScript.

function MyClasse() {

var nome;

this.idade;

}

function document\_OnLoad() {

oMyClasse = new MyClasse();

oMyClasse.idade = '10';

}

Para passar parâmetros para o construtor da classe:

function MyClasse(value) {

this.idade = value;

}

function document\_OnLoad() {

oMyClasse = new MyClasse('10');

}

O **React** (também denominado **React**. js ou **ReactJS**) é uma biblioteca JavaScript de código aberto com foco em criar interfaces de usuário (frontend) em páginas web. É mantido pelo Facebook, Instagram, outras empresas e uma comunidade de desenvolvedores individuais.

O React adota o fato de que a lógica de renderização é inerentemente acoplada com outras lógicas de UI: como eventos são manipulados, como o state muda com o tempo e como os dados são preparados para exibição.

**O React NÃO REQUER o uso do JSX.** Porém, a maioria das pessoas acha prático como uma ajuda visual quando se está trabalhando com uma UI dentro do código em JavaScript. Ele permite ao React mostrar mensagens mais úteis de erro e aviso.

A biblioteca React trabalha com um DOM virtual, baseia-se em componentes reutilizáveis e no paradigma declarativo (programação funcional). É possível desenvolver aplicativos móveis com React, especificamente React Native.

React Native utiliza componentes nativos em vez de componentes da Web como blocos de construção, existindo dois tipos de dados que controlam um componente:

* Refs - fornecem uma forma de acessar os nós do DOM ou elementos React criados no método render.
* Props - (Propriedades): São recebidas pelo componente pai, são fixas (imutáveis) por toda vida do componente. É uma palavra-chave especial no React, que significa propriedades e está sendo usada para passar dados de um componente para outro .
* State  - (Estado): É o estado interno do componente. É definido e controlado pelo próprio componente e muda ao longo de sua vida. É gerenciado de dentro do componente (como variáveis declaradas dentro de uma função).
* Key são úteis ao trabalhar com componentes criados dinamicamente ou quando suas listas são alteradas pelos usuários. A definição do valor da chave manterá seus componentes identificados de maneira única após a alteração.
* Elements um elemento é um objeto simples que descreve uma instância de componente ou nó DOM e suas propriedades desejadas.

**Laravel**

**View Routes** Se sua rota precisar apenas retornar uma visualização, você poderá usar o método Como o método, esse método fornece um atalho simples para que você não precise definir uma rota ou controlador completo. O método aceita um URI como seu primeiro argumento e um nome de exibição como seu segundo argumento. Além disso, você pode fornecer uma matriz de dados para passar para a exibição como um terceiro argumento opcional: Route::viewredirectview

**Route Model Binding** Ao injetar um ID de modelo em uma ação de rota ou controlador, você frequentemente consulta para recuperar o modelo que corresponde a esse ID. A ligação do modelo de rota do Laravel fornece uma maneira conveniente de injetar automaticamente as instâncias do modelo diretamente em suas rotas. Por exemplo, em vez de injetar o ID de um usuário, você pode injetar toda a Userinstância do modelo que corresponde ao ID fornecido.

**Required Parameters** Às vezes, você precisará capturar segmentos do URI em sua rota. Por exemplo, pode ser necessário capturar o ID de um usuário a partir da URL. Você pode fazer isso definindo parâmetros de rota:

**Route Groups** -  permite compartilhar atributos de rotas, tais como *middle-ware* ou *namespaces*, entre várias rotas, sem a necessidade de definir esses atributos para cada rota individualmente