

# ESTRUTURA E CONCEITOS DE UM FRAMEWORK FRONTEND WEB

# FRAMEWORK

FRAMEWORKS OFERECE AO  
DESENVOLVEDOR CRIAR UM SOFTWARE A  
PARTIR DE UMA FUNDAÇÃO. SEU OBJETIVO É  
OFERECER AO USUÁRIO A OPORTUNIDADE  
DE NÃO PRECISAR COMEÇAR UM TRABALHO  
DO ZERO.

# BIBLIOTECAS

BIBLIOTECAS SÃO COLEÇÕES DE RECURSOS  
USADOS POR PROGRAMA DE COMPUTADOR.  
SÃO SUBPROGRAMAS UTILIZADOS NO  
DESENVOLVIMENTO DO SOFTWARE.

# E qual é a diferença?

- Um framework é como construir um apartamento modelo. Você tem a planta do imóvel e algumas escolhas limitadas em termos de arquitetura e design. No final, a decoração será diferente, mas as unidades são iguais entre os andares .
- Uma biblioteca é como ir a uma loja de móveis. Você já tem um apartamento, mas precisa de ajuda com os móveis. Não é sua intenção fazer uma mesa do zero, por exemplo. Na loja de móveis, você escolhe coisas diferentes que combinam com a sua casa. Você controla as ações.

# O que é React.JS ?

**React** não é *framework*, **é biblioteca!!**

React é uma biblioteca JavaScript que visa simplificar o desenvolvimento de interfaces visuais. Desenvolvida pelo Facebook e entregue ao mundo em 2013, hoje está presente nas aplicações mais utilizadas da internet, como Instagram, Netflix, Spotify, e por aí vai.

# JSX = JAVASCRIPT E XML



Parece uma mistura de HTML dentro do JavaScript, mas na realidade é tudo JavaScript.

Na verdade, será traduzido para JS por um *transpilador*.

Mas professor, o que é um  
transpilador??

**Qual é a diferença entre um  
compilador e um transpilador?**



O transpilador é um tipo de compilador que transporta (ou traduz) código fonte de uma linguagem de programação para outra.





Ao escrever em JSX, estamos codando algo que se parece com HTML, mas na verdade é uma sintaxe para escrever códigos JavaScript utilizando marcações.

# SYNTATIC SUGAR



Todo código JSX é JavaScript com açúcar! O termo em inglês é esse mesmo: **Syntactic Sugar**. É um código que, se não existir, não impede o dev de programar com React. **JSX não é necessário para usar React, contudo, não existem motivos para não usar**. Ele aumenta a legibilidade do código, é amplamente documentado e usado pela comunidade. Consideramos intuitivo, pois se parece com HTML. Permite que devs que conhecem HTML entendam seu código com pouco esforço.



A comunidade React desenvolveu ferramentas que facilitam muito o trabalho. Vamos ver as mais comuns:

## ***SANDBOX***

É uma aplicação online para colaboração em grupo:

<https://codesandbox.io/>



## ***BOILERPLATE***

É um conjunto de ações prontas para minimizar o esforço de criar algo novo.  
É uma forma de queimar etapas, reduzir esforço repetitivo.

Ao usar **boilerplates** ao invés de configurar do zero nossos apps React, significa que teremos: **Webpack, Babel, HMR updates etc.**, sem precisar nesse momento entender pra quê serve cada um deles.

**DICA:** pesquise o que representa cada uma dessas siglas!



## CREATE REACT APP

O mais conhecido e usado pela comunidade. Demora muito na instalação, mas é completo. É mantido pelo time React.

```
npx create-react-app nome-em-minúsculas
```

## VITE

Muito rápido e atende bem a pequenos projetos, principalmente projetos de aprendizado.

```
npm create vite@latest
```



Vamos abrir o main.jsx e fazer o “Olá mundo” em React.

```
ReactDOM.createRoot(document.getElementById("root")).render("Olá mundo");
```

Uma forma diferente de fazer a mesma coisa, dessa vez usando uma função da biblioteca *React*, ou *ReactDOM*.

**Mas saca só:** Utilizando *React*, essa será nossa única interação direta com o *DOM*! 😊



Single Page Application = Aplicativo de página única

Em SPAs, toda a interface da aplicação é executada pelo navegador. A geração das páginas fica toda no lado cliente, tanto telas, quanto transições e troca de conteúdo. Do servidor, apenas os dados são recuperados, utilizando APIs.



Em SPA, normalmente é empregada a navegação **sem refresh**. Todo o conteúdo da aplicação se concentra em **uma única página**. Ao clicar em links, botões, ou submeter formulários, o navegador se mantém na 'mesma página HTML'. A navegação seria algo como 'ocultar e exibir' elementos da página, ou criar elementos dinamicamente, sem recorrer a outra página ou mudar de endereço.






No SPA a criação ou modificação de elementos na página é feita utilizando *JavaScript* e funções de manipulação do **DOM**. A partir de agora, em React, vamos nos referir a esse processo como **RENDERIZAÇÃO**.

Várias bibliotecas de SPA utilizam a possibilidade de trabalhar com os **endereços na página local para permitir a navegação sem sair do lugar**. A essa forma de endereçamento chamamos de **rotas**. Rotas, portanto, são caminhos locais (**path**, em inglês), mas que se parecem com uma URL completa.



Para que exista o efeito da navegação, ao fazer a transição entre os conteúdos, é utilizada uma navegação local. Você deve conhecer a navegação local em HTML utilizando âncoras  ou desvios de página.



São **links criados em uma página**, que levam para uma parte específica dentro do seu próprio conteúdo. No endereço da barra do navegador é adicionado um **#**, e, após esse marcador, o nome do desvio desejado.

```
<a href="#playlist">Abrir a Playlist</a>
```



## EXEMPLO:

```
http://localhost:3000/#/playlist
```

Nesse exemplo, apenas a página principal (index.html) do site será carregada. Porém, um desvio será feito para a área que estiver nomeada como **/playlist**.

Ainda no exemplo, **/playlist** é uma rota desse site. Não existe no servidor uma página interna **playlist.html**, porém o comportamento de navegação será idêntico.

