

React sem magia

Agora que já conhecemos **React**, vamos pensar um pouco sobre o que acontece de forma "mágica"

Esse código é JavaScript, certo?

```
import React from 'react';

const Button = () => (
  <button onClick={() => console.log('clicou')}>
    botão
  </button>
);

export default Button
```

Então, se executar esse trecho de código no
DevTools, funciona?

UÉ



	30°	45°	60°
sin	$\frac{1}{2}$	$\frac{\sqrt{2}}{2}$	$\frac{\sqrt{3}}{2}$
cos	$\frac{\sqrt{3}}{2}$	$\frac{\sqrt{2}}{2}$	$\frac{1}{2}$
tan	$\frac{\sqrt{3}}{3}$	1	$\sqrt{3}$



$$y = ax^2 + bx + c$$

$$(x_1, x_2) = \frac{-b \pm \Delta}{2a}$$

Esse código não é "entendível" pelo navegadores

```
import React from 'react';

const Button = () => (
  <button onClick={() => console.log('clique')}>
    botão
  </button>
);

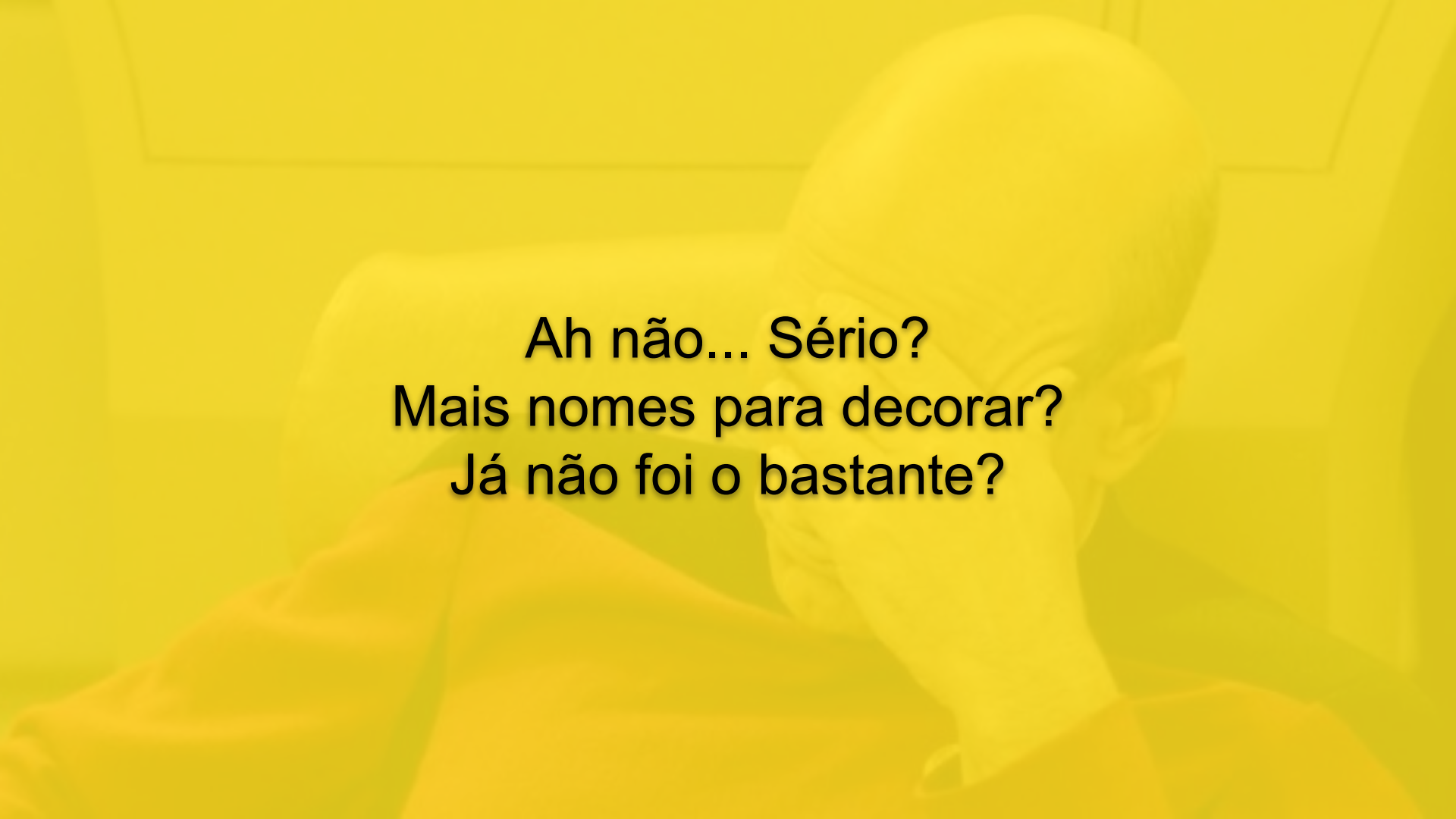
export default Button
```

E pra que eles sejam transformados, outros dois pacotes entram em ação

BABEL



webpack



Ah não... Sério?
Mais nomes para decorar?
Já não foi o bastante?

Essas ferramentas são mais **conceituais**.
Não é necessário estudar elas em detalhes agora,
mas é **importante saber pra que elas servem** :)

Babel



O [Babel](#) é um transpilador (ou compilador) de JS. Seu trabalho é permitir que as pessoas desenvolvedoras possam escrever um código JavaScript com funcionalidades mais modernas, sem precisar ficar se preocupando se determinada funcionalidade funciona em ambiente X ou Y.

Vamos ver o que acontece com um código React quando usamos o Babel? Podemos [brincar no REPL](#) do próprio Babel.

Babel

```
1 const button = (props) => {  
2   return (  
3     <button className={props.className}>  
4       <div>conteudo</div>  
5     </button>  
6   )  
7 }
```

```
1 "use strict";  
2  
3 var button = function button(props) {  
4   return /*#__PURE__*/React.createElement("button", {  
5     className: props.className  
6   }, /*#__PURE__*/React.createElement("div", null, "conteudo"));  
7 };
```

Webpack

O [Webpack](#) é um empacotador de módulos (ou "*module bundler*") e seu trabalho é ainda mais simples: apenas "resolver" os módulos e arquivos da sua aplicação e gerar arquivos unificados no final.

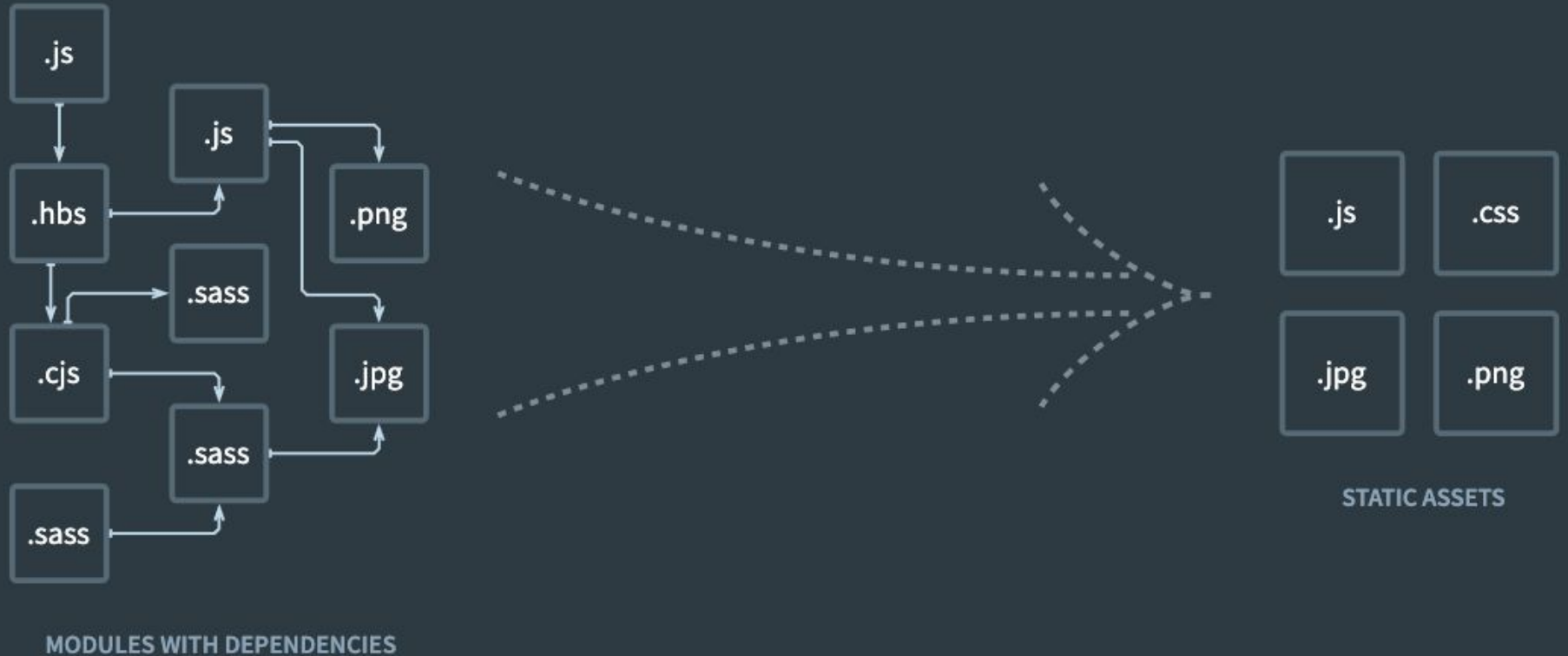
E o que é um módulo?

É qualquer arquivo/trecho de JS que importamos e exportamos funções para utilizar em outros arquivos.



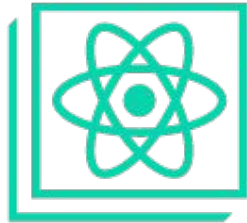
webpack

Webpack



E por que a gente não consegue ver essa magia
acontecer?

Por causa do [create-react-app](#)



Em resumo

O **create-react-app** facilita o início de um projeto e faz a abstração de algumas configurações (como webpack e o babel).

Mas é importante que a gente saiba o que acontece "por baixo dos panos", pois nem sempre utilizaremos **create-react-app** para desenvolver.

A light-colored dog is captured mid-jump, with its front paws raised high and its head tilted upwards. The entire image is overlaid with a semi-transparent yellow filter. The word "DÚVIDAS" is centered in the middle of the image in a bold, black, sans-serif font.

DÚVIDAS

Issaê

Links úteis:

- [Babel](#)
- [Webpack](#)