UNINASSAU | ser educacional

Front-End Frameworks

Aula 1

Dsc. Francisco Nauber Bernardo Gois

Francisco Nauber Bernardo Gois
Dsc. Informática Aplicada

Experiência Profissional

CGE | CONTROLADORIA E OUVIDORIA GERAL DO ESTADO
GOVERNO DO ESTADO DO CEARÁ

Auditor de Controle Interno e Professor da Especialização em Ciência de Dados

ESCOLA DE SAÚDE PÚBLICA DO CEARÁ

Cientista de Dados (2021-2022)

stratio

Cientista de Dados (Manutenção Preditiva com Modelos de Aprendizado de Máquina e Sistemas Autônomos) - Lisboa, Portugal (2020-2021)

CEARÁ
GOVERNO DO ESTADO
SECRETARIA DA SAÚDE

Lider da Equipe de de Inteligência Artificial da Secretaria de Saúde no Combate a COVID-19 (2019-2020)

VIRTVS VNITA FORTIOR

Professor Adjunto - Campus Russas (2017-2019)

SERPRO

Analista de Sistemas e Cientista de Dados (2004-2017)

Francisco Nauber Bernardo Gois
Dsc. Informática Aplicada

Formação Acadêmica

聖若瑟大學
UNIVERSITY OF SAINT JOSEPH

Pós Doutorado- Laboratório de Neurociência Aplicada (2022-2025)

Doutorado em Informática Aplicada (2012-2017)
Search-based Stress Test: an approach applying evolutionary algorithms and trajectory methods

Mestrado em Informática Aplicada (2012-2017)

UNINASSAU | ser educacional

Objetivo

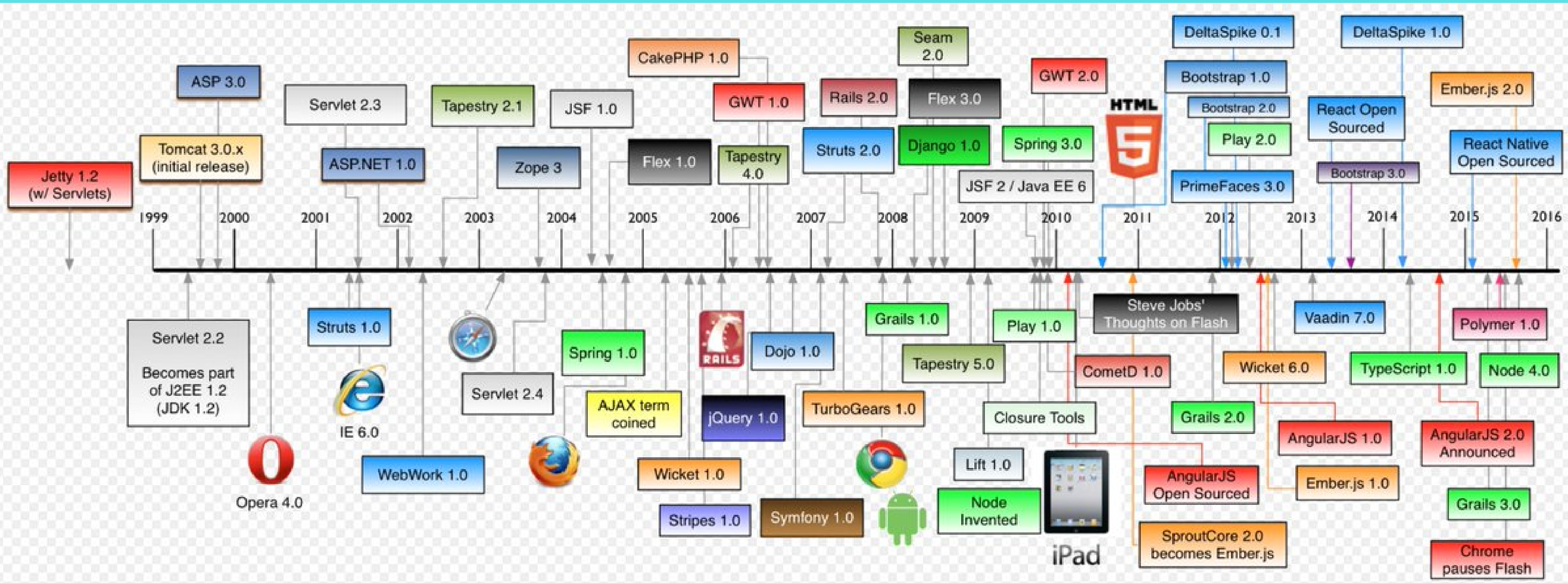
Ao fim dessa aula o aluno será capaz de identificar os principais paradigmas e técnicas de desenvolvimento de aplicações para a internet

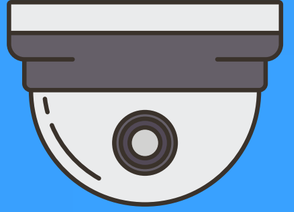
UNINASSAU | ser educacional

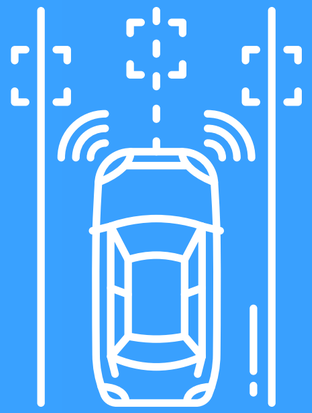
Avaliação

Avaliação Teórica e Prática aplicando conceitos apresentados na modelagem de desenvolvimento de um sistema utilizando uma linguagem de escolha do Aluno. A construção do sistema será incremental durante as aulas.

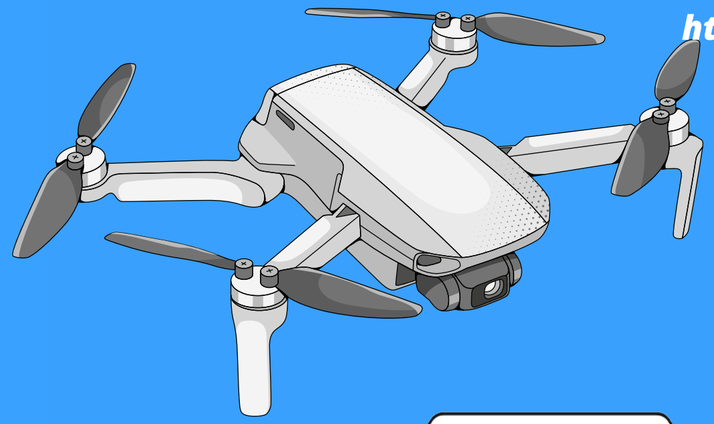
UNINASSAU | ser educacional

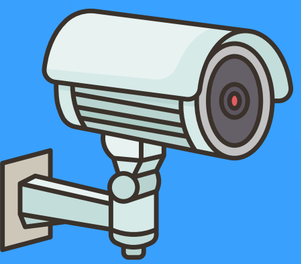


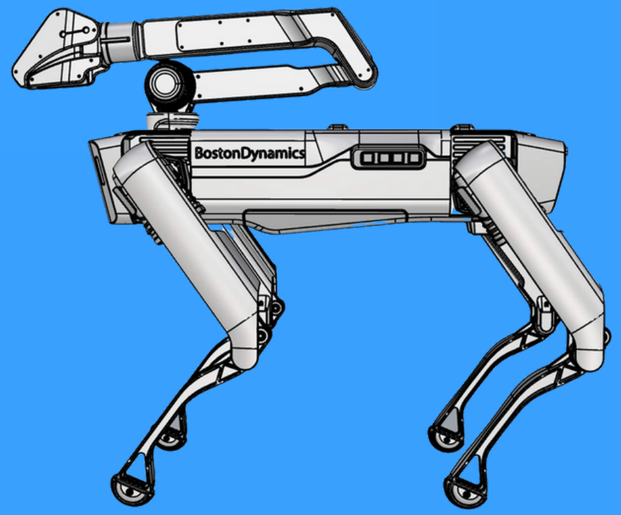




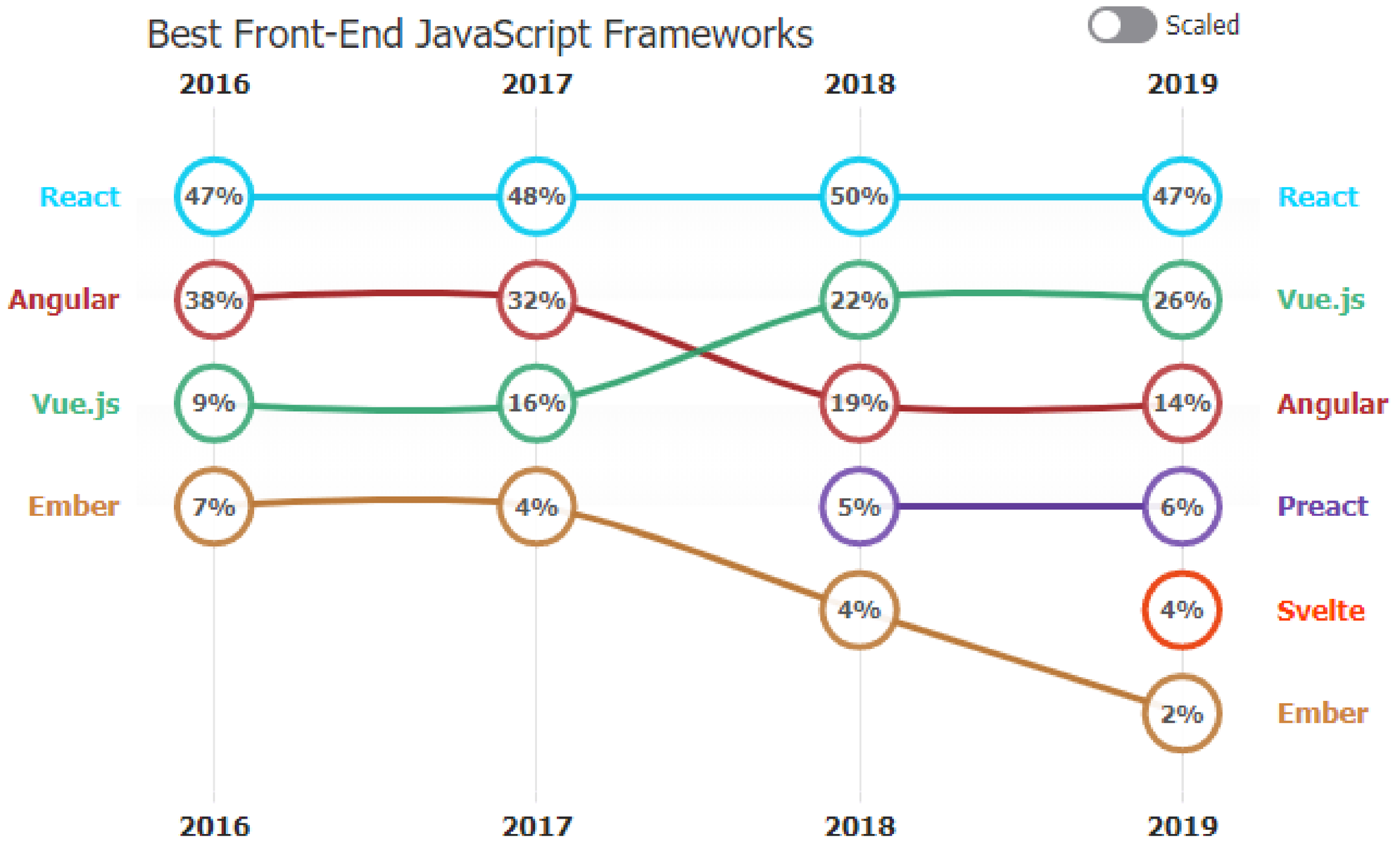
https://www.youtube.com/watch?v=Dy0hJWItsyE







https://www.youtube.com/watch?v=rPul9WKQ6oQ

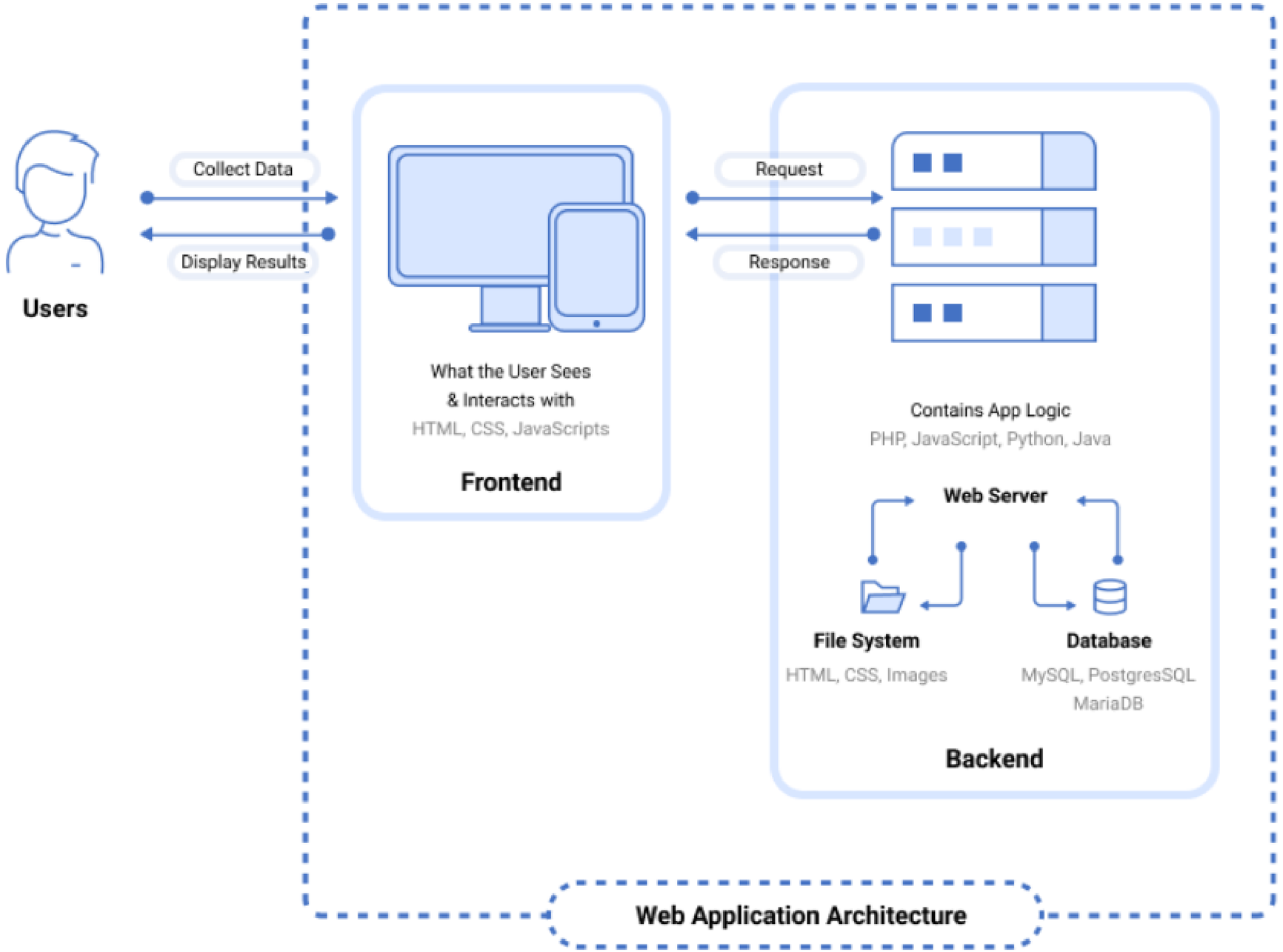

Best Front-End JavaScript Frameworks
Scaled
2016
2017
2018
2019
React
Angular
Vue.js
Ember
47%
48%
50%
47%
38%
32%
22%
26%
9%
16%
19%
14%
7%
4%
5%
6%
4%
2%
React
Vue.js
Angular
Preact
Svelte
Ember
2016
2017
2018
2019

UNINASSAU

Arquitetura da WEB

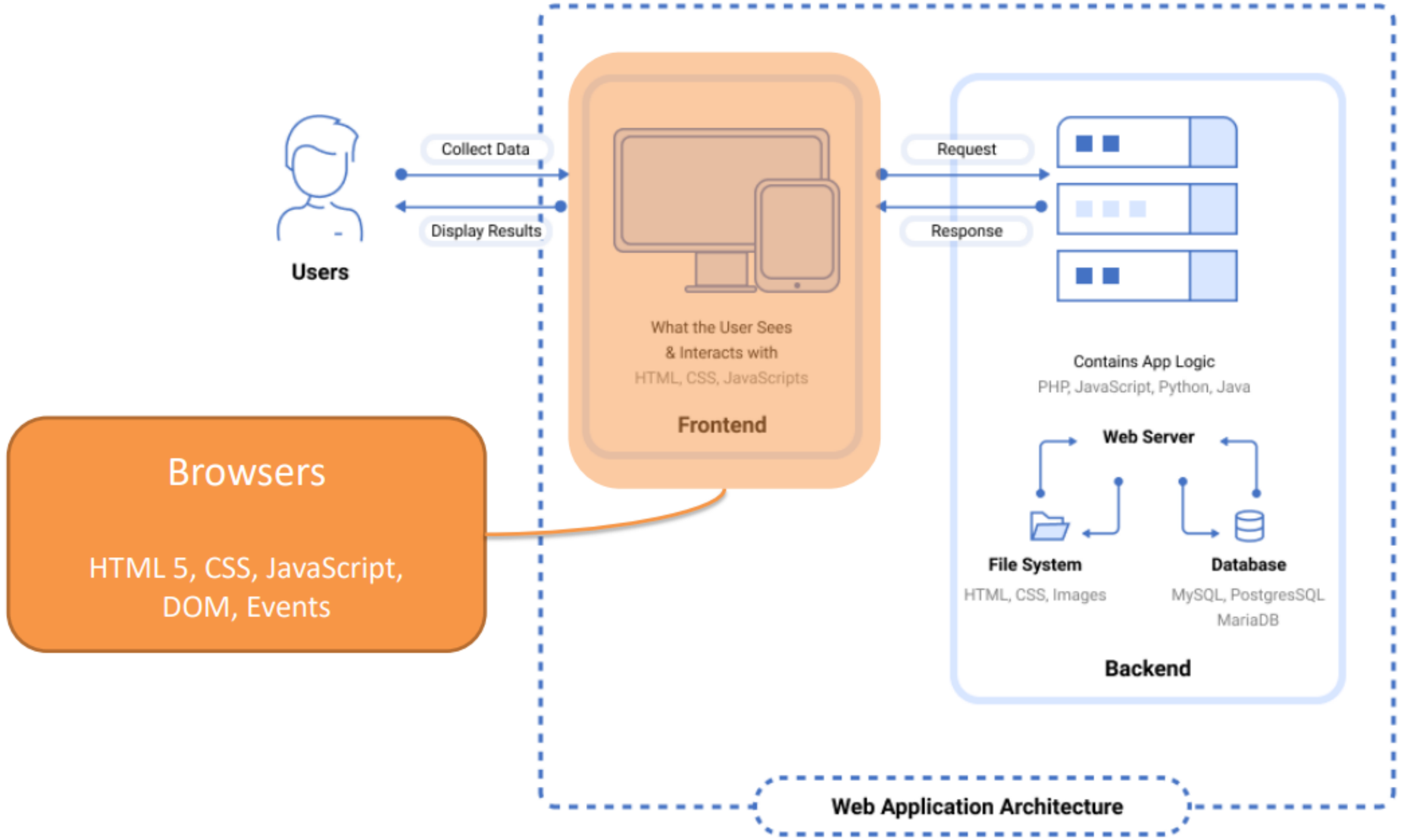
UNINASSAU

Arquitetura da WEB



UNINASSAU

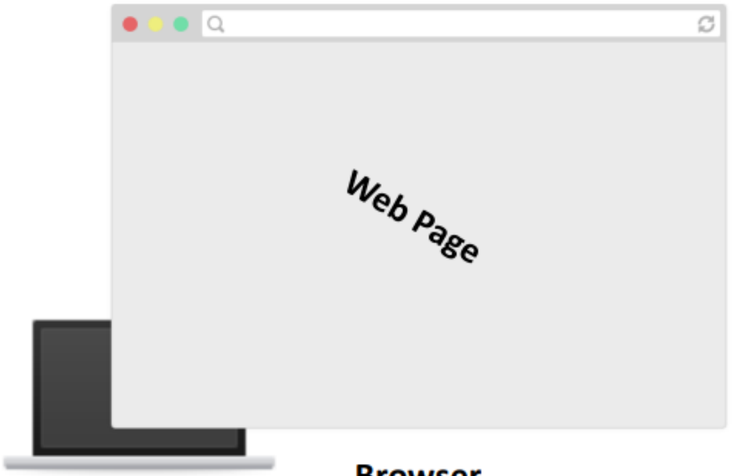
Arquitetura da WEB



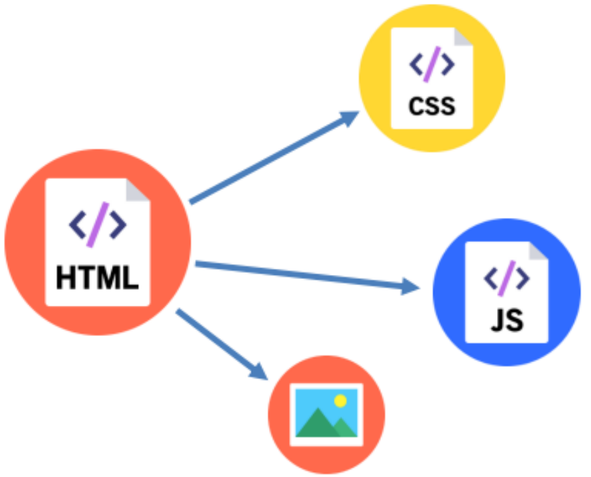
UNINASSAU | ser educacional

Navegadores

Browser



Browser



The HTML file might link to other **resources** (images, videos, …)
as well as **JavaScript** and **CSS** files,
which the browser then also loads

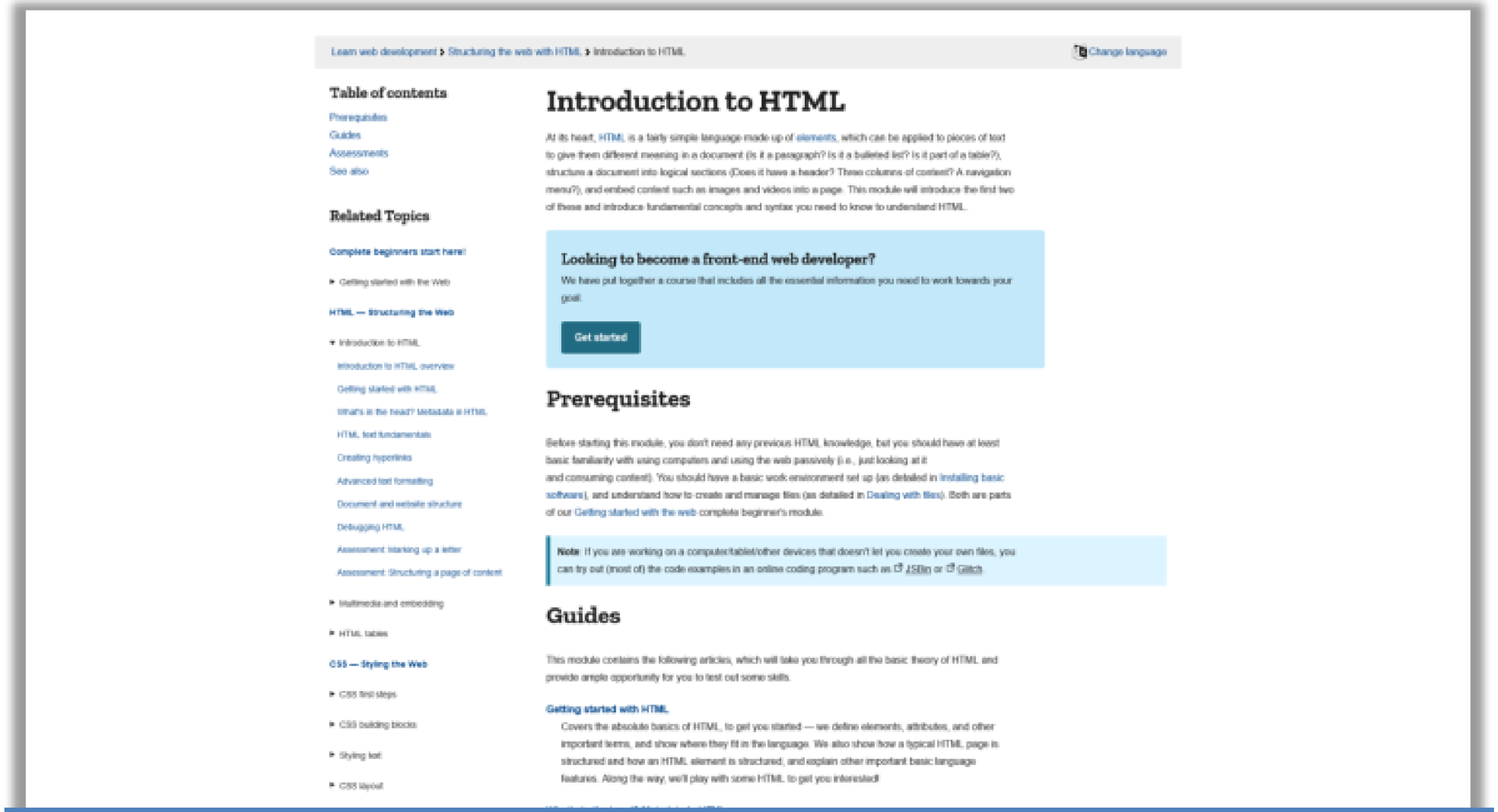
These are stored or generated by a **server**

UNINASSAU

Navegadores

UNINASSAU | ser educacional

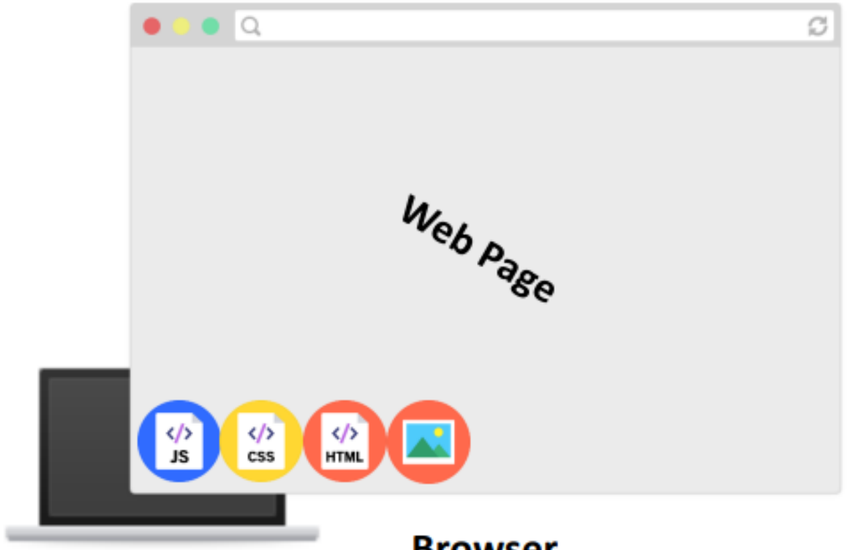
HTML



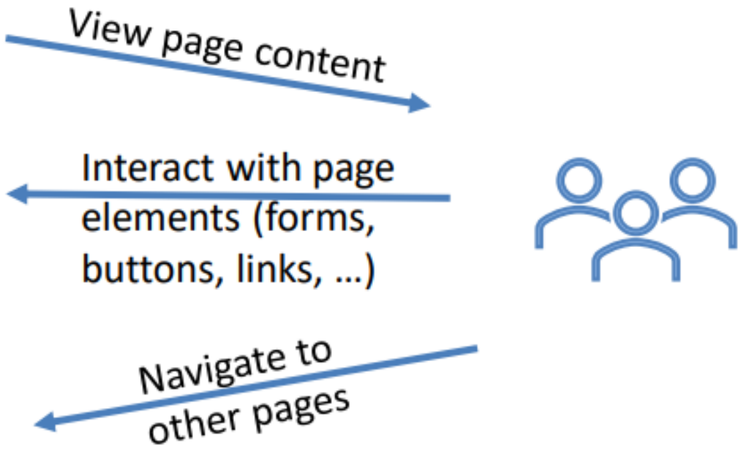
https://developer.mozilla.org/docs/Learn/HTML/Introduction_to_HTML

https://developer.mozilla.org/pt-BR/docs/Learn/HTML/Introduction_to_HTML

Browser



Browser

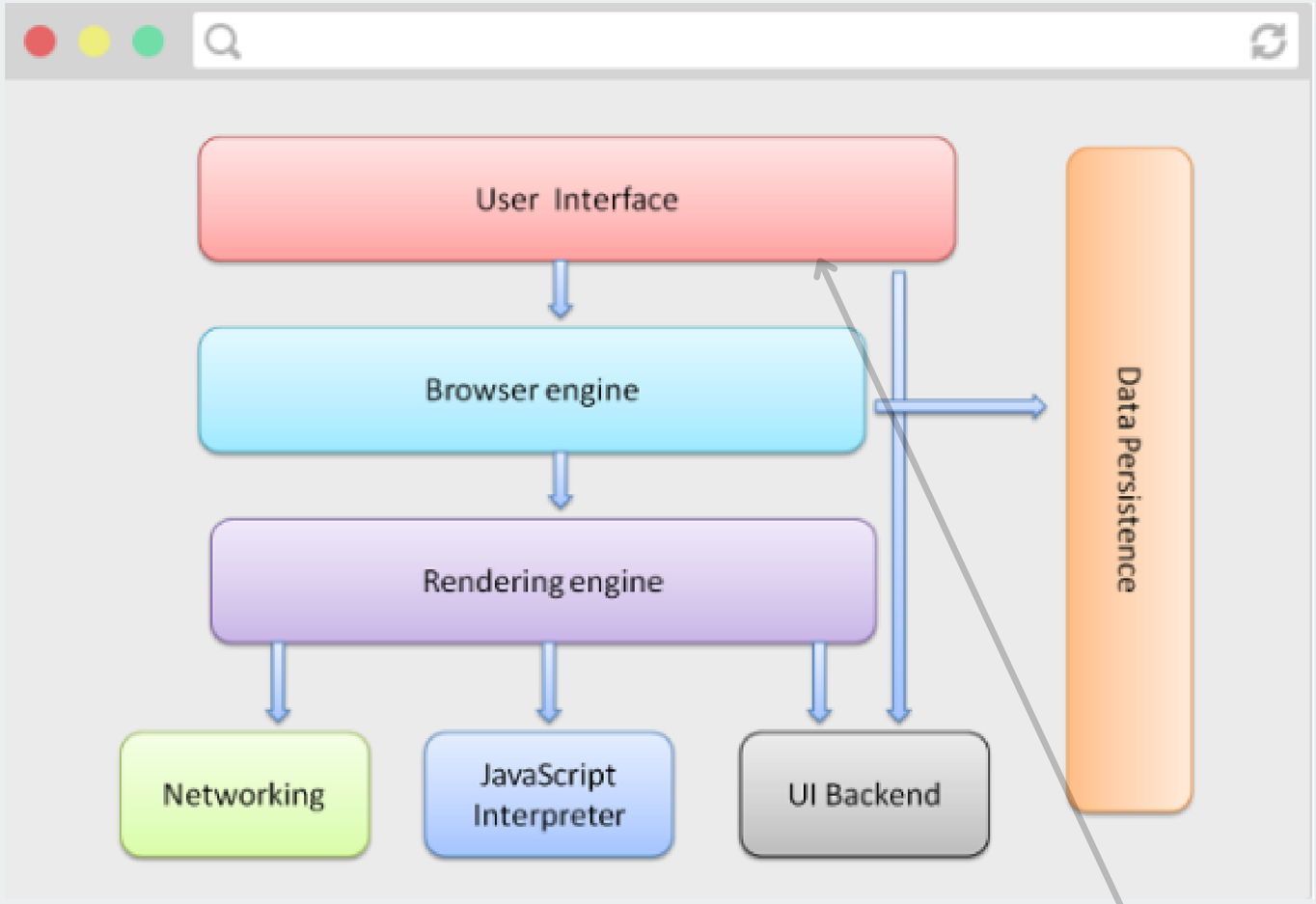


The content of the web page is described by HTML+CSS.

Clicking on a link brings the user to a **new page**.
Interacting with other elements may generate ***Events*** inside the browser.
Such Events are “captured” by JavaScript and may **update the page content**.

UNINASSAU

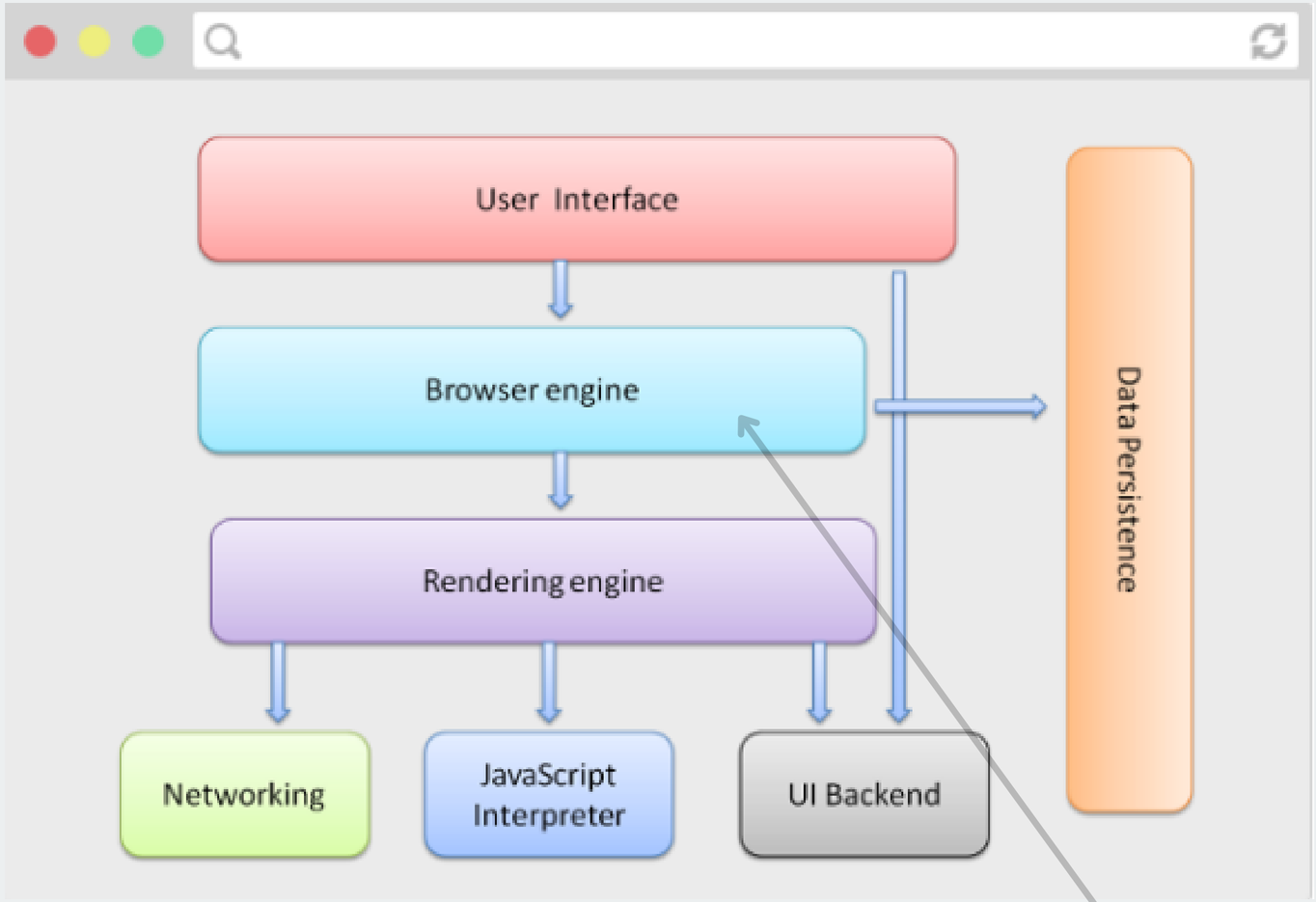
Arquitetura de um Navegador



Interface do usuário: a barra de endereço, botão voltar/avançar, menu de favoritos, etc. Todas as partes do navegador são exibidas exceto a janela onde você vê a página solicitada

UNINASSAU | ser educacional

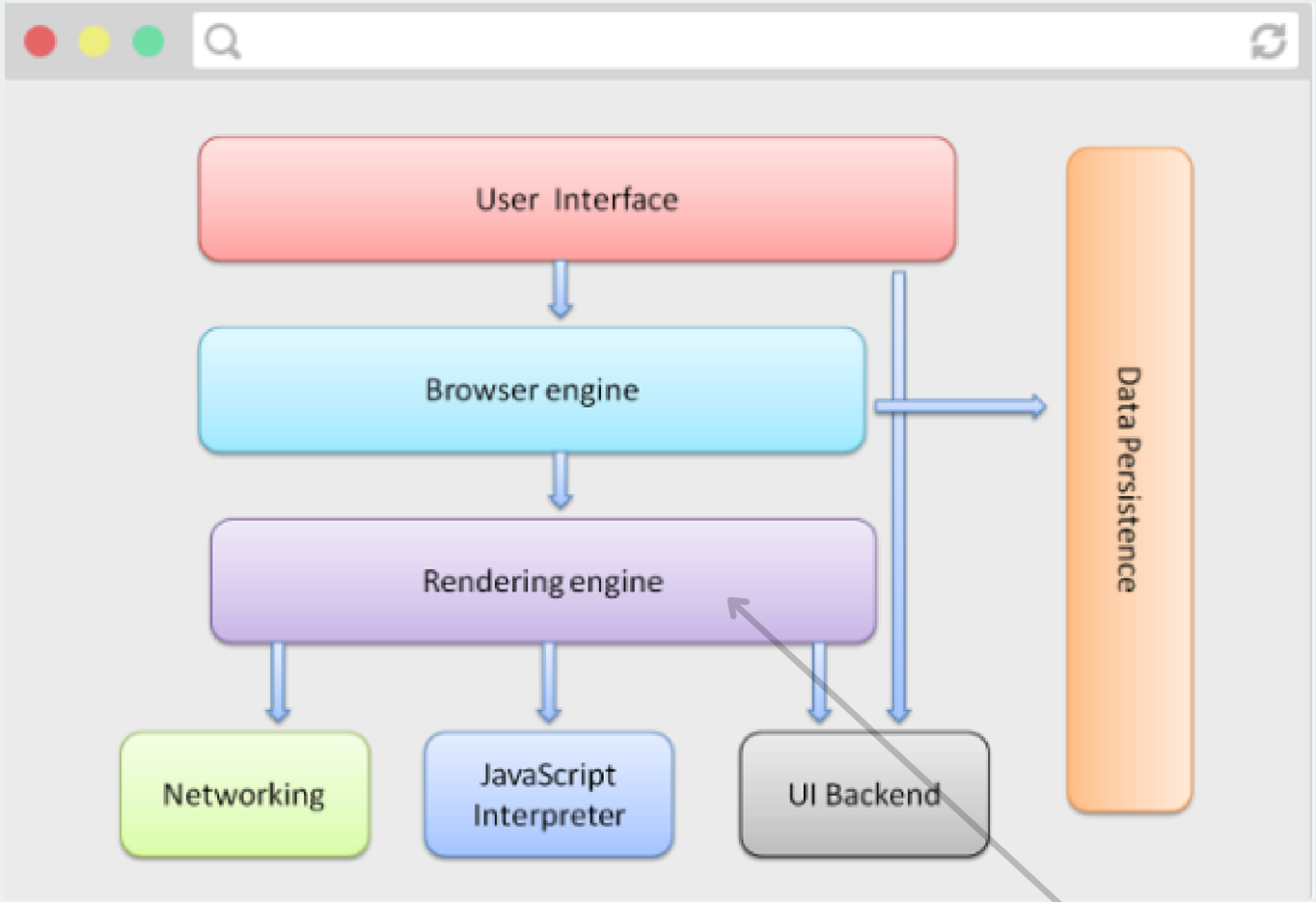
Arquitetura de um Navegador



O Browser Engine empacota ações entre a UI e o mecanismo de renderização

UNINASSAU

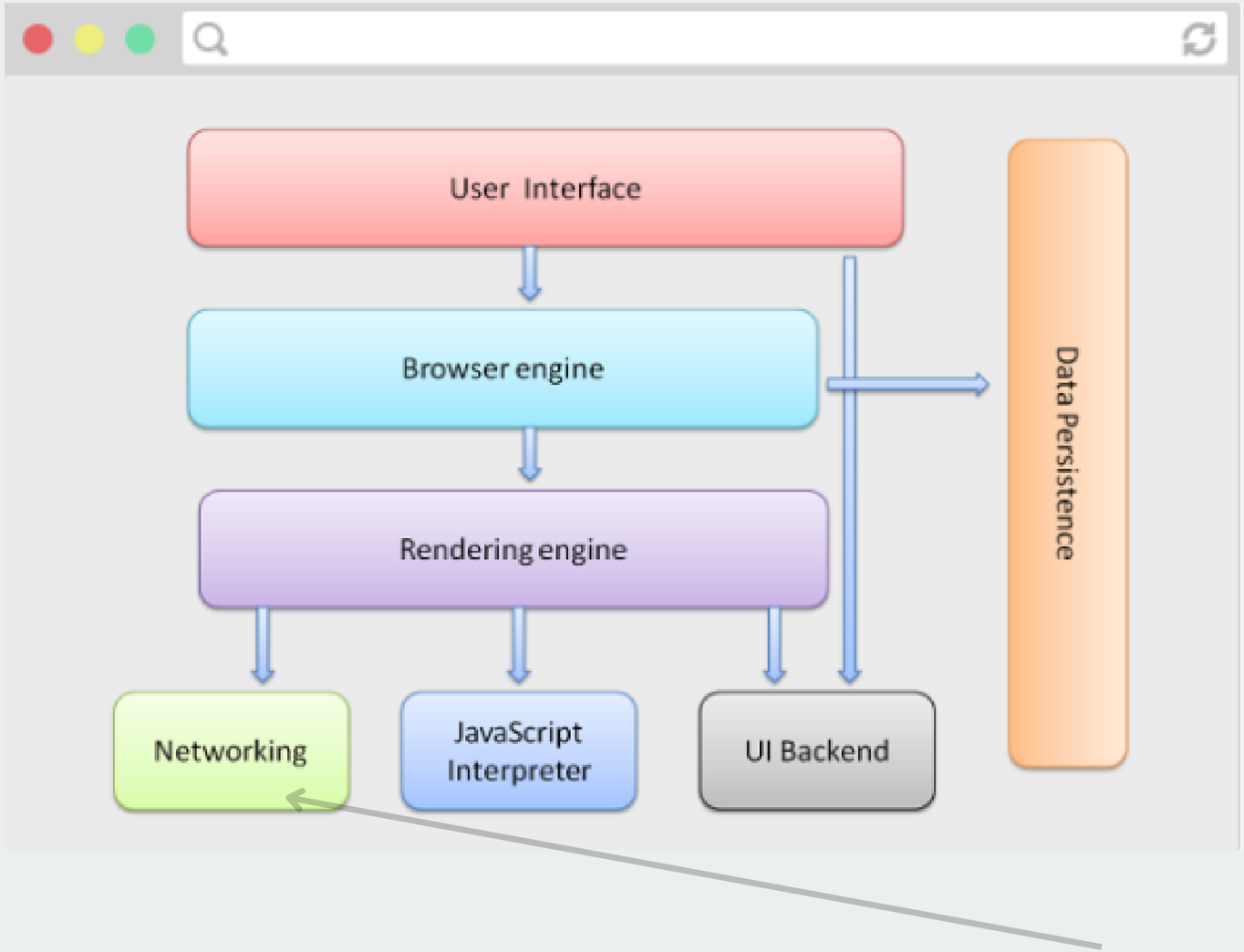
Arquitetura de um Navegador



Rendering Engine: responsável por exibir o contente. Por exemplo, se o conteúdo solicitado for HTML, o mecanismo de renderização analisa HTML e CSS e exibe o conteúdo na tela

UNINASSAU

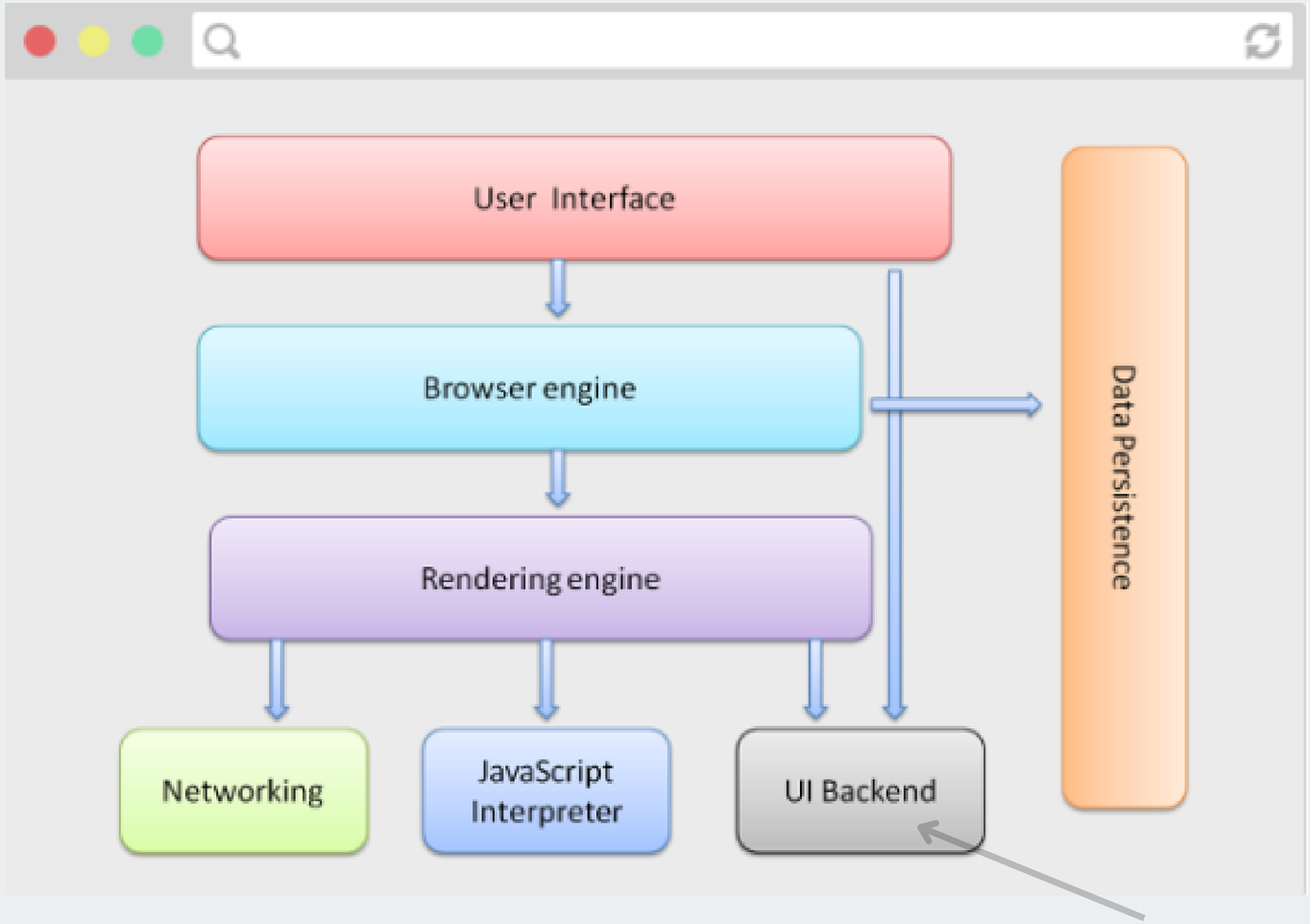
Arquitetura de um Navegador



Rede: para chamadas de rede, como solicitações HTTP, usando diferentes implementações para diferentes plataformas por trás de um interface independente de plataforma

UNINASSAU

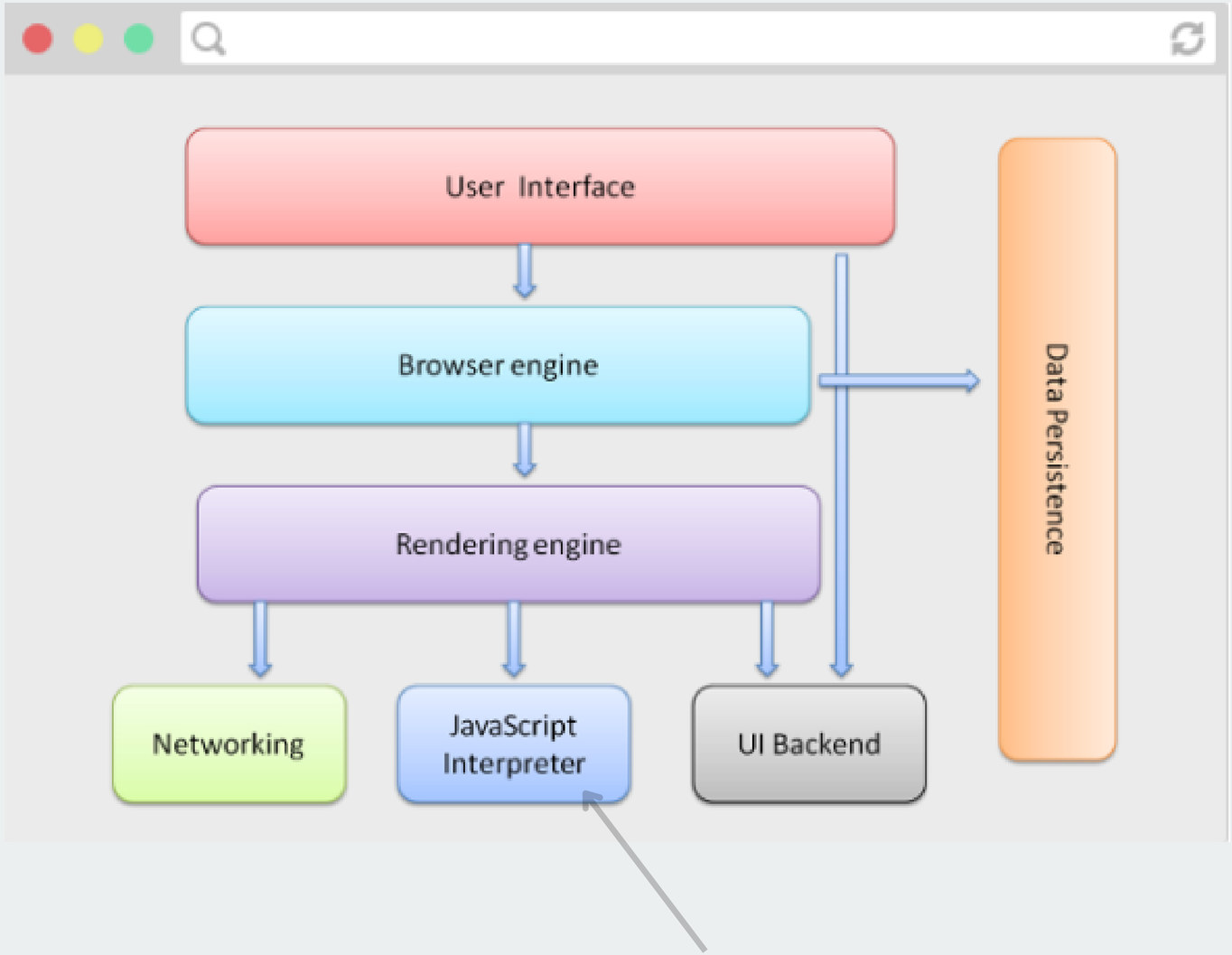
Arquitetura de um Navegador



UI Backend: usado para desenhar widgets básicos como caixas de combinação
e janelas. Este backend expõe uma interface genérica que é não específico da plataforma. Por baixo, usa o usuário do sistema operacional
métodos de interface

UNINASSAU

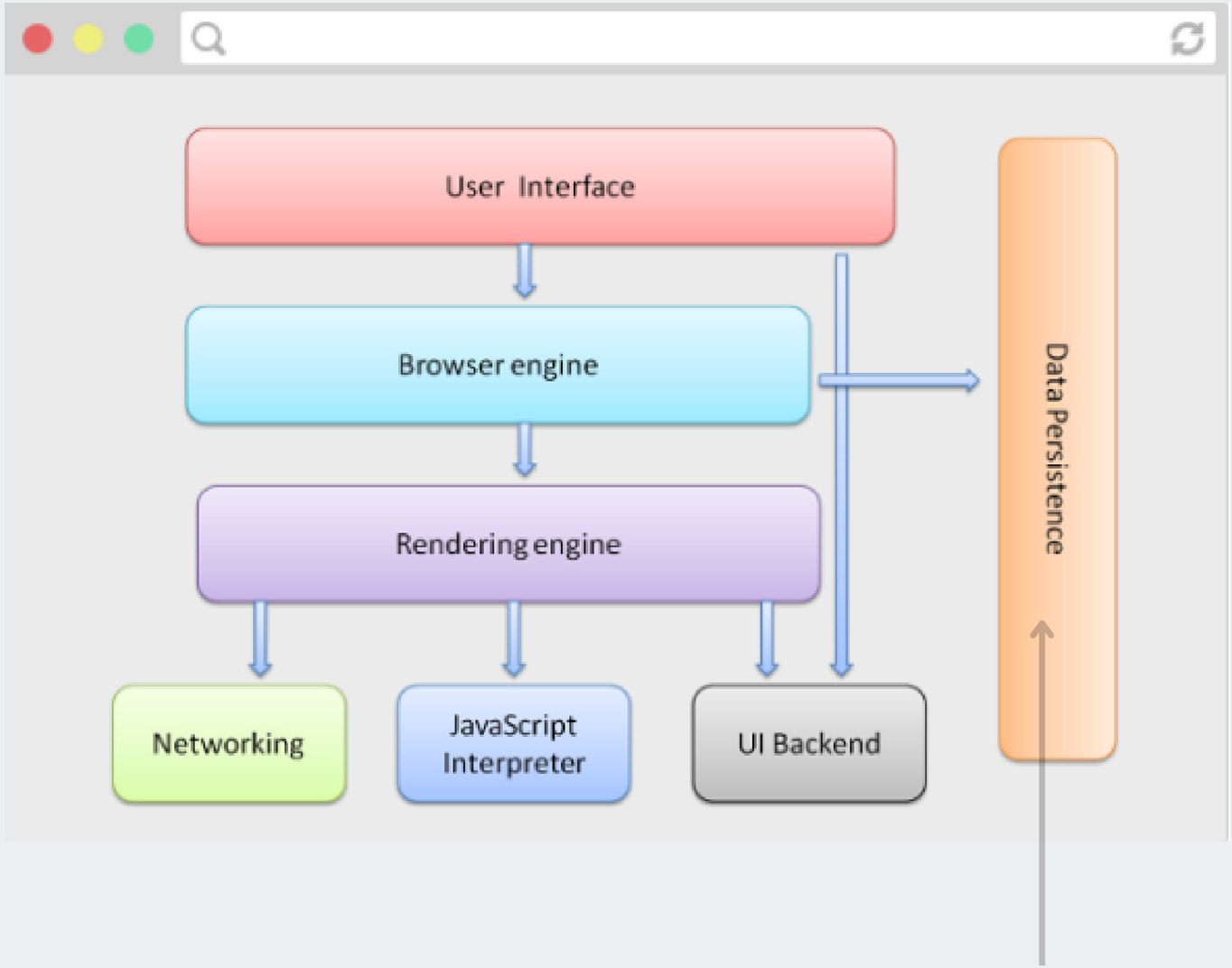
Arquitetura de um Navegador



JavaScript Interpreter: usado para analisar e executar JavaScript código

UNINASSAU

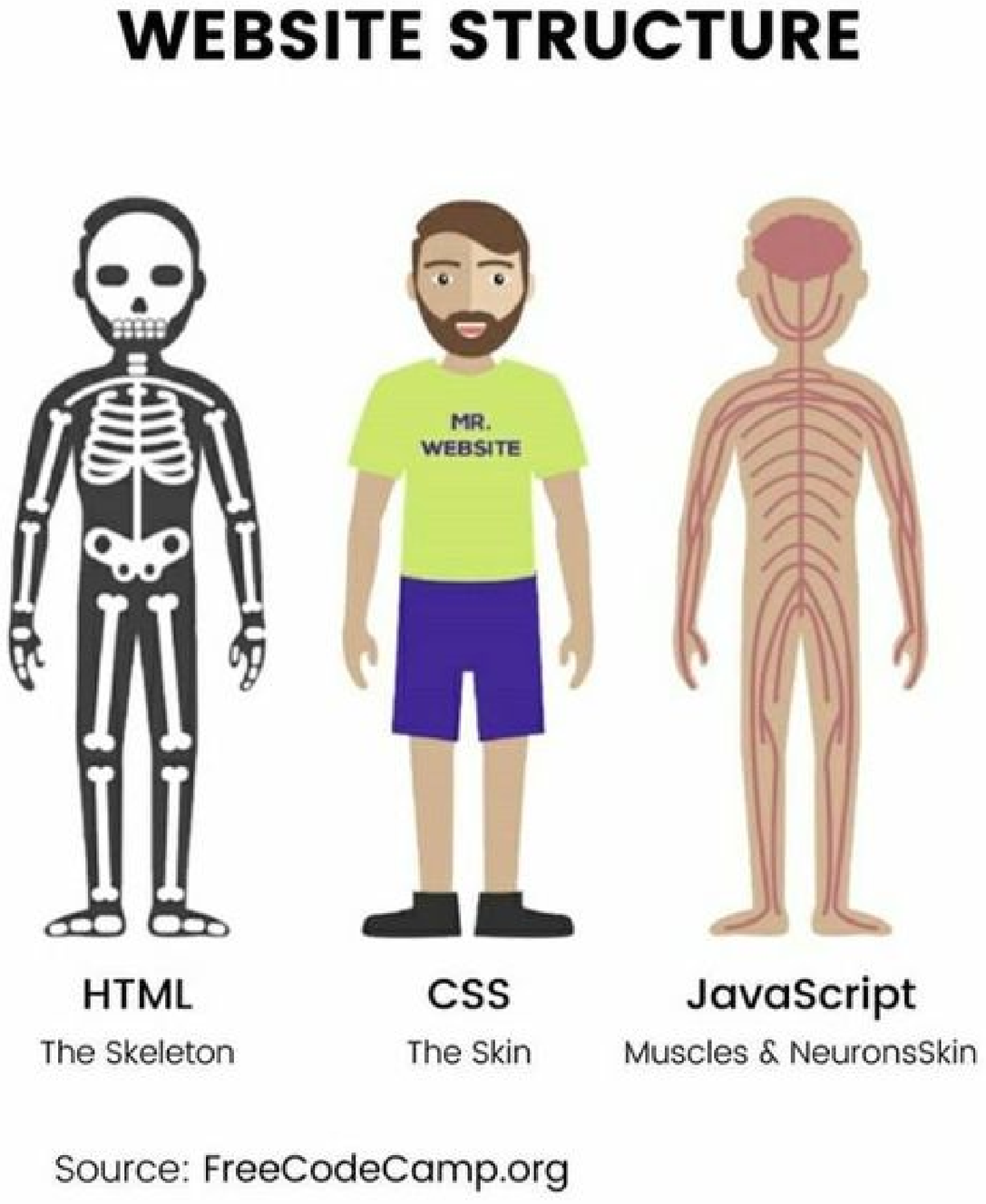
Arquitetura de um Navegador



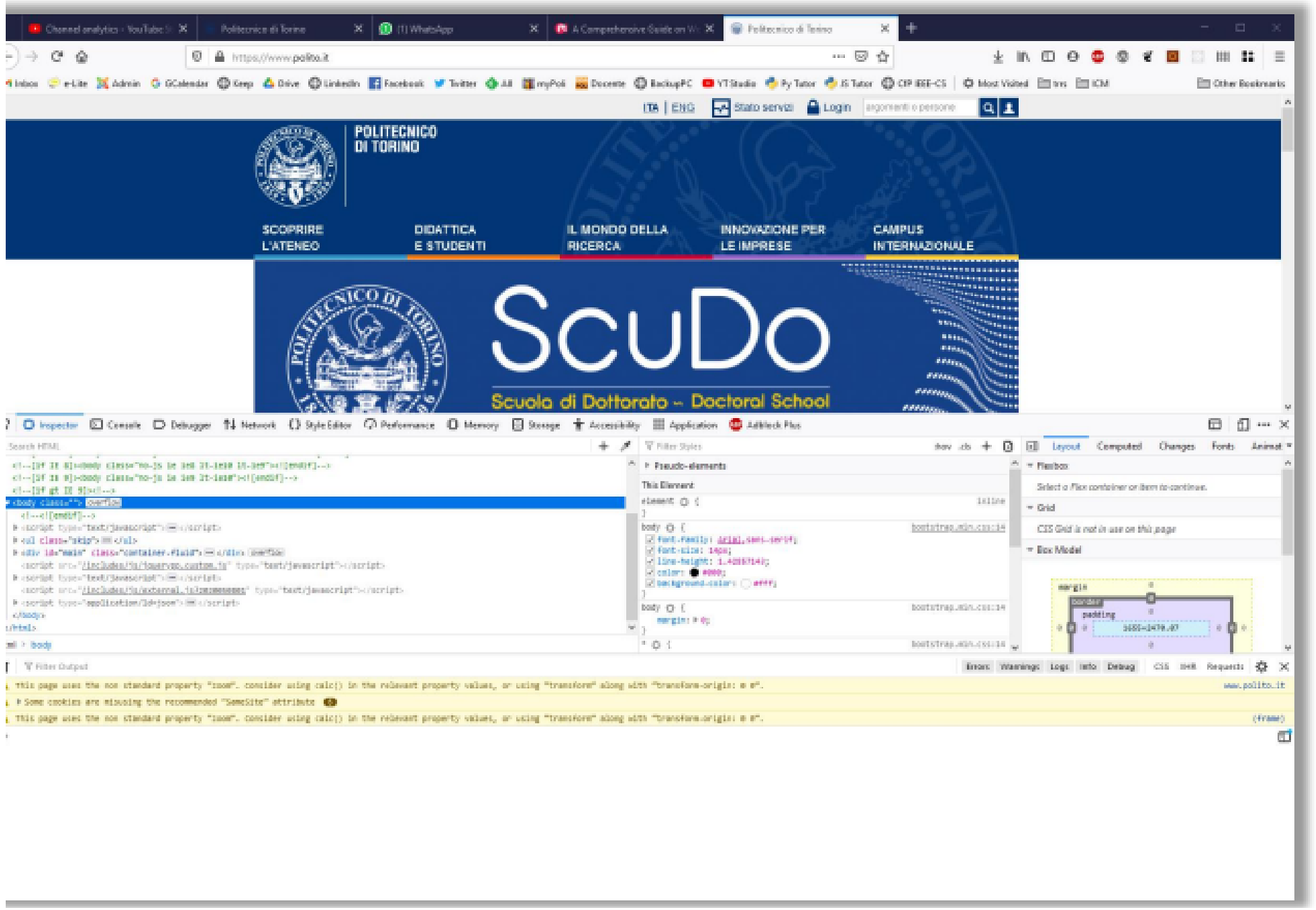
Persistência de dados: uma camada de persistência. O navegador pode precisar salve todos os tipos de dados localmente, como cookies. Os navegadores também suportam mecanismos de armazenamento como LocalStorage, IndexedDB, WebSQL e FileSystem

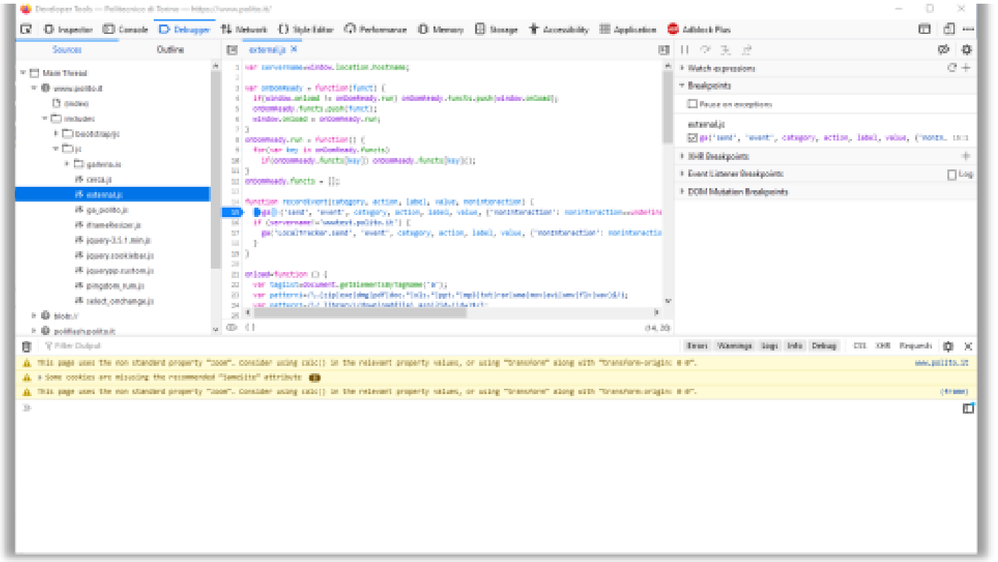
UNINASSAU

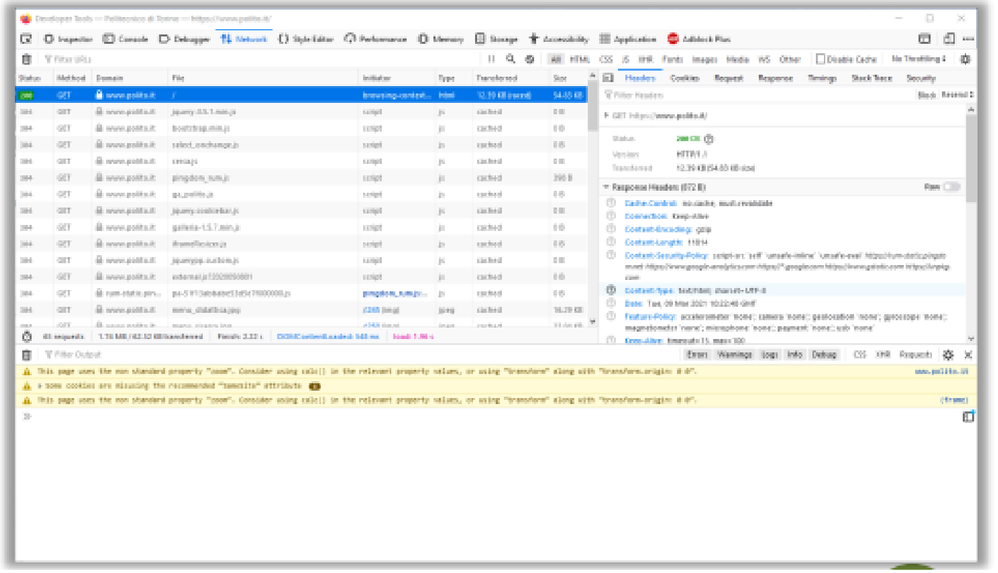
Estrutura de uma Página Web



Browser Development tools







Document Object Model (DOM)

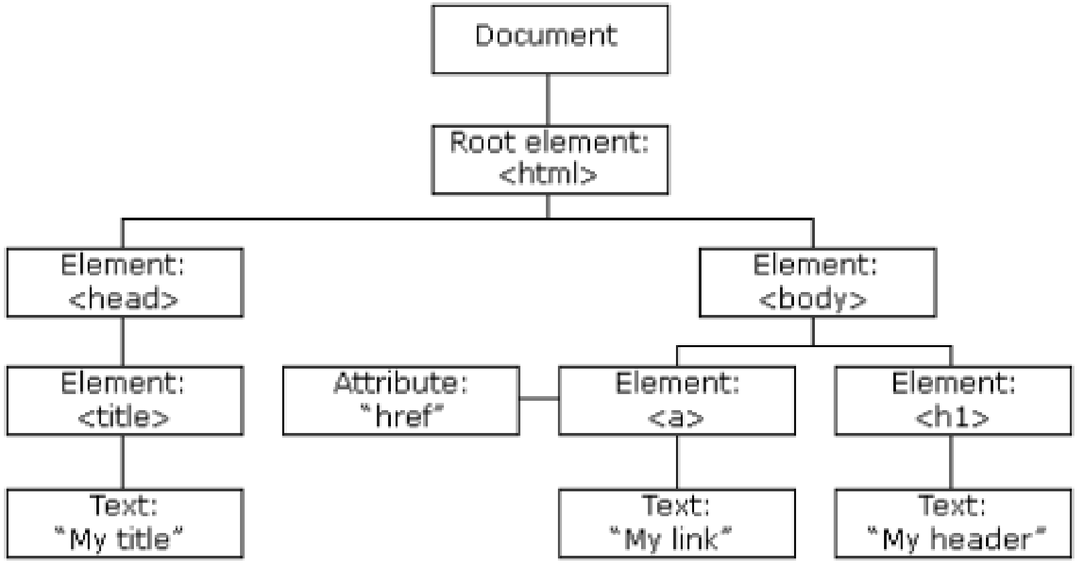
Estrutura de dados padrão para representando o conteúdo da página da web

- **Permite obter, alterar, adicionar ou excluir elementos HTML**
- **Compatível com todos os navegadores**
- **Os programas JavaScript podem ler e modificar o DOM**
- **Abstrai e padroniza APIs para**

- Navegador

– HTML

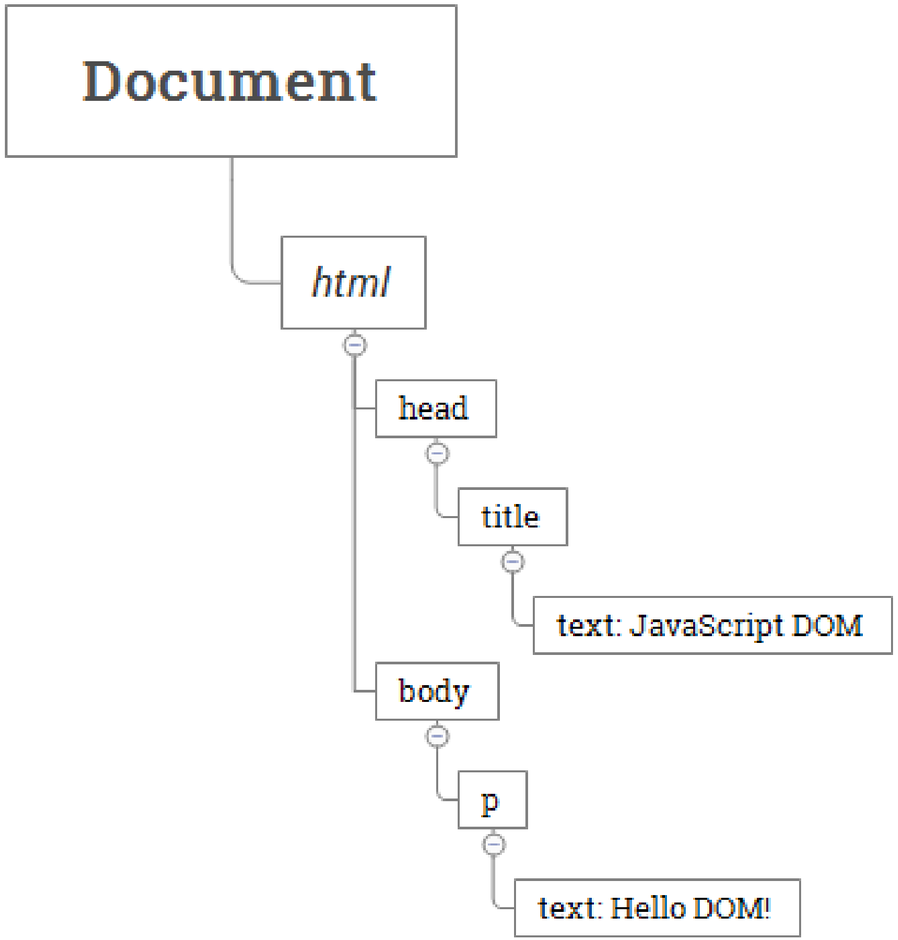
"O modelo de objeto de documento W3C (DOM) é uma plataforma e interface de linguagem neutra que permite que programas e scripts para acessar dinamicamente e atualizar o conteúdo, estrutura e estilo de um documento."



UNINASSAU

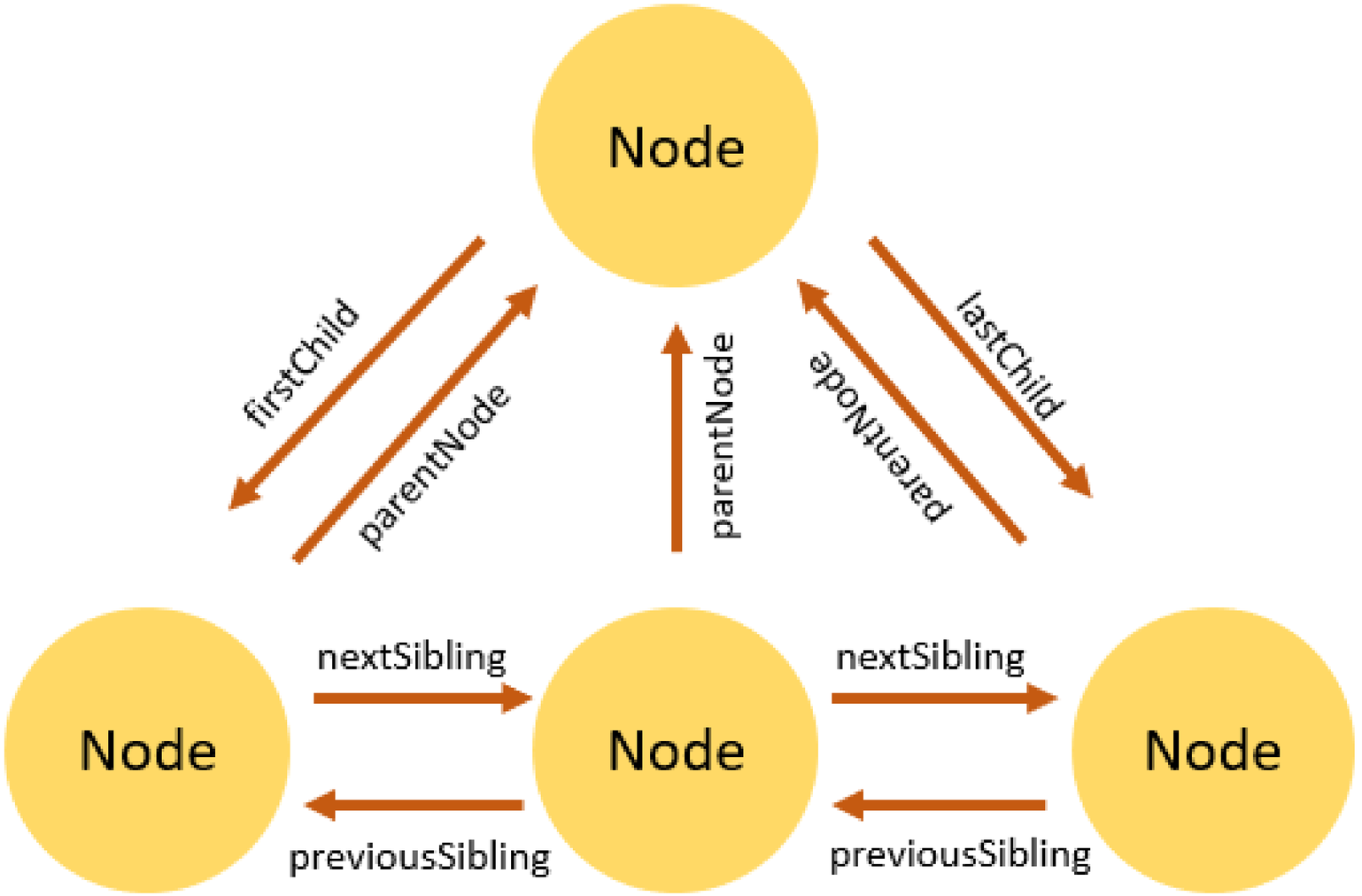
Document Object Model (DOM)

```
<html>
    <head>
        <title>JavaScript DOM</title>
    </head>
    <body>
        <p>Hello DOM!</p>
    </body>
</html>
```



UNINASSAU

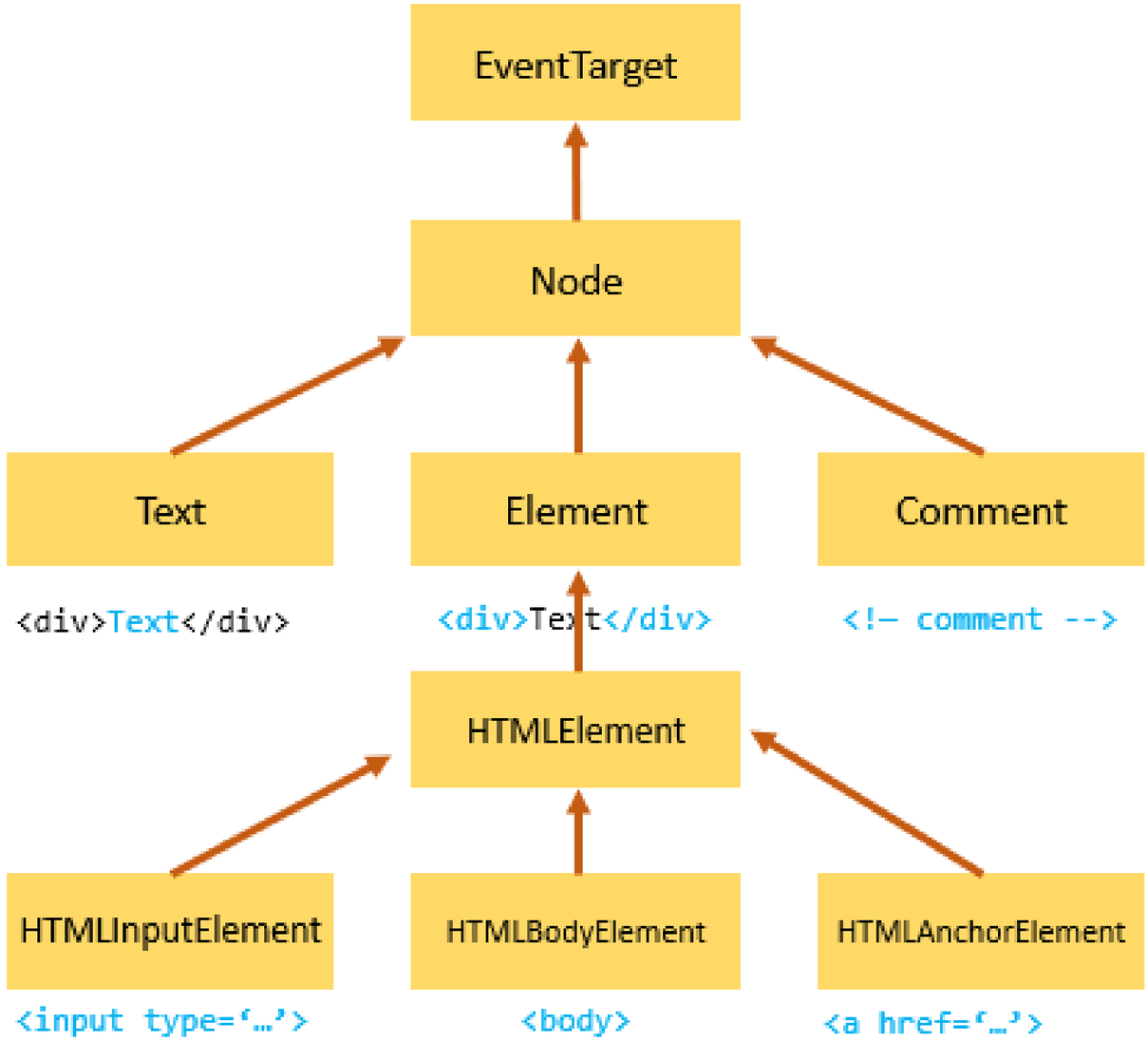
Document Object Model (DOM)



https://www.javascripttutorial.net/javascript-dom/document-object-model-in-javascript/

UNINASSAU

Document Object Model (DOM)



UNINASSAU

CSS

Cascading Style Sheets (CSS)

Permitir a definição de layouts complexos

• Adaptar páginas da web para

- resoluções diferentes

– dispositivos diferentes (por exemplo, smartphones)

– preferências diferentes (por exemplo, esquemas de cores)

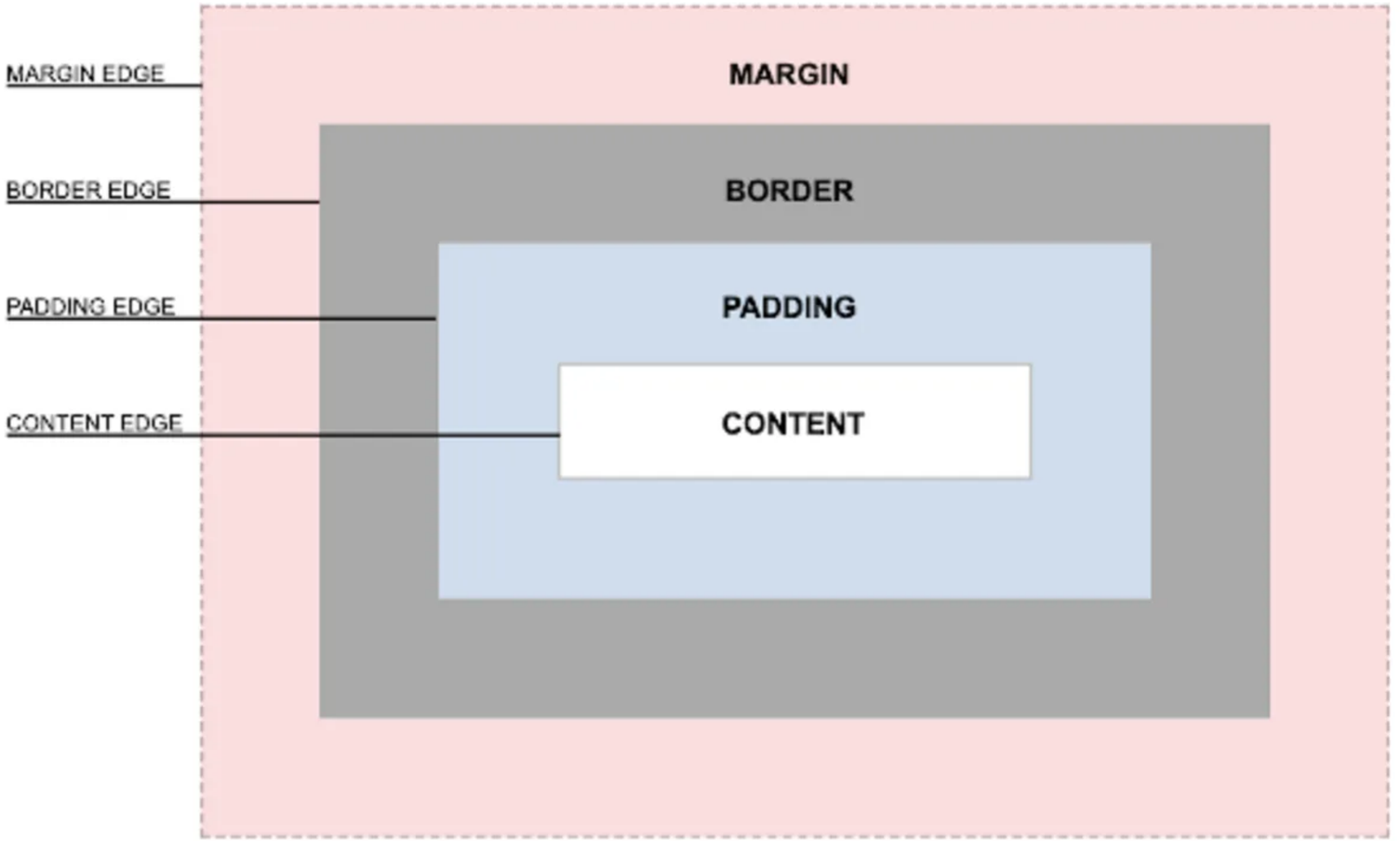
– para diferentes mídias (por exemplo, texto versus vídeo)

- de forma padronizada

UNINASSAU

CSS

CSS BOX MODEL



https://web.dev/learn/css/box-model/

UNINASSAU

CSS

Seletores

https://web.dev/learn/css/selectors/

UNINASSAU

CSS

Seletores

Seletor universal

Um seletor universal - também conhecido como curinga - corresponde a qualquer elemento.

```
* {
  color: hotpink;
}
```

Esta regra faz com que cada elemento HTML na página tenha texto hotpink.

Seletor de tipo

Um seletor de tipo corresponde diretamente a um elemento HTML.

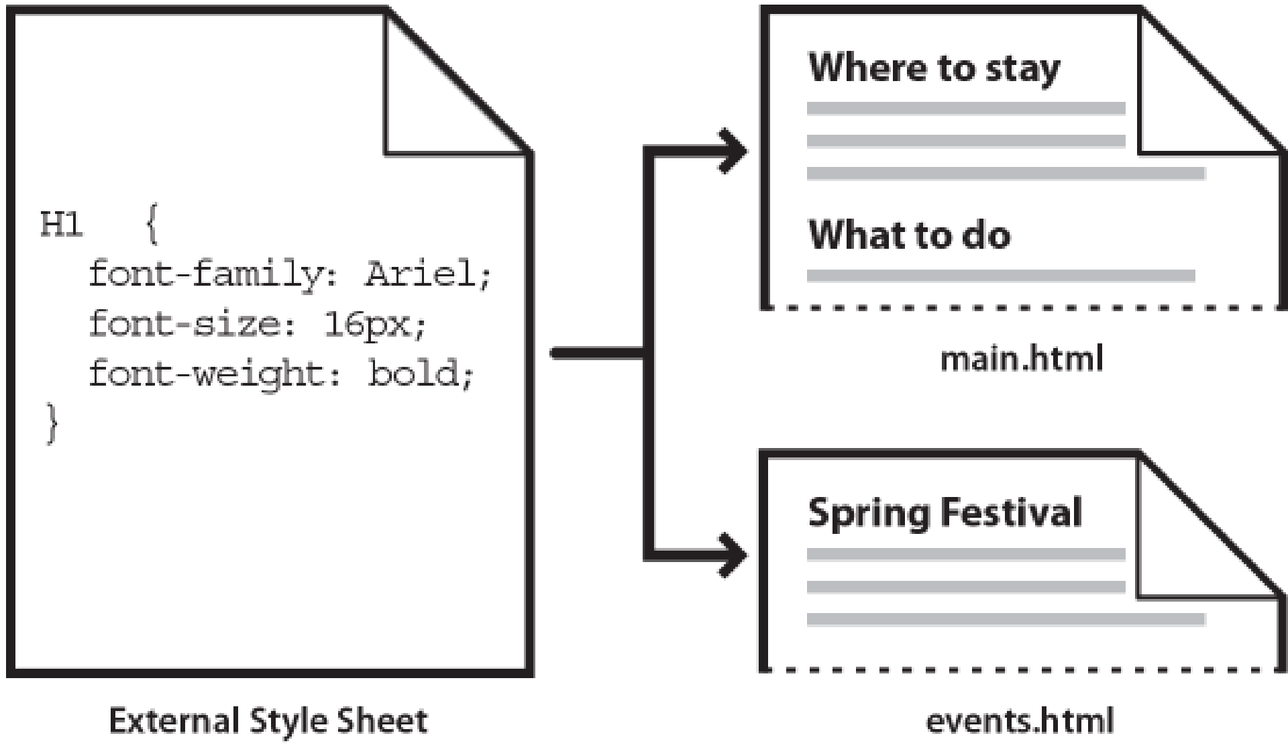
```
section {
  padding: 2em;
}
```

UNINASSAU

CSS

A cascata

A cascata



https://web.dev/learn/css/the-cascade/

UNINASSAU

CSS

A cascata

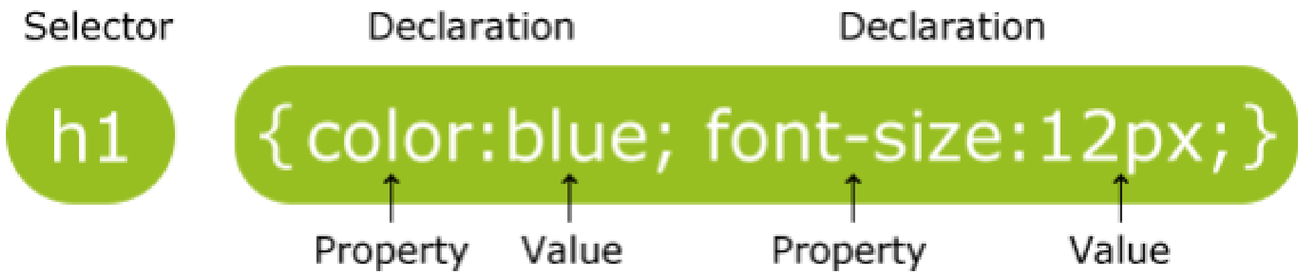
Um conjunto de "declarações" aplicadas a alguns "seletores"

– Seletores identificam partes do DOM

– Declarações definem o valor de algumas propriedades

– Propriedades controlam tudo

• cor, tamanho, fonte, alinhamento, borda, sombra, posição, status de seleção, transições, links, botões, cursores, ...



UNINASSAU

Javascript

JavaScript (JS) é uma linguagem de programação

- Atualmente é a única linguagem de programação que um navegador pode executar nativamente...
- ... e também roda em um computador, como outras linguagens de programação (graças ao Node.js)
- Não tem nada a ver com Java – assim denominado por razões de marketing, apenas
- A primeira versão foi escrita em 10 dias (!)

JAVASCRIPT VERSIONS


Brendan Eich

- **JAVASCRIPT (December 4th 1995)** Netscape and Sun press release
- **ECMAScript Standard Editions:** https://www.ecma-international.org/ecma-262/
  - **ES1 (June 1997)** Object-based, Scripting, Relaxed syntax, Prototypes
  - **ES2 (June 1998)** Editorial changes for ISO 16262
  - **ES3 (December 1999)** Regexps, Try/Catch, Do-While, String methods

10 yrs

  - **ES5 (December 2009)** Strict mode, JSON, .bind, Object mts, Array mts
  - **ES5.1 (June 2011)** Editorial changes for ISO 16262:2011

Main target

  - **ES6 (June 2015)** Classes, Modules, Arrow Fs, Generators, Const/Let, Destructuring, Template Literals, Promise, Proxy, Symbol, Reflect — Also: ES2015
  - **ES7 (June 2016)** Exponentiation operator (**) and Array Includes — Also: ES2016

ES9, ES10, ...

  - **ES8 (June 2017)** Async Fs, Shared Memory & Atomics — Also: ES2017

https://www.slideshare.net/RafaelCasusoRomate/javascript-editions-es7-es8-and-es9-vs-v8

UNINASSAU

JavaScript

JavaScript Engines

- V8 (Chrome V8) by Google
  - used in Chrome/Chromium, Node.js and Microsoft Edge
- SpiderMonkey by Mozilla Foundation
  - Used in Firefox/Gecko
- ChakraCore by Microsoft
  - it was used in Edge
- JavaScriptCore by Apple
  - used in Safari

UNINASSAU

JavaScript

Compatibilidade

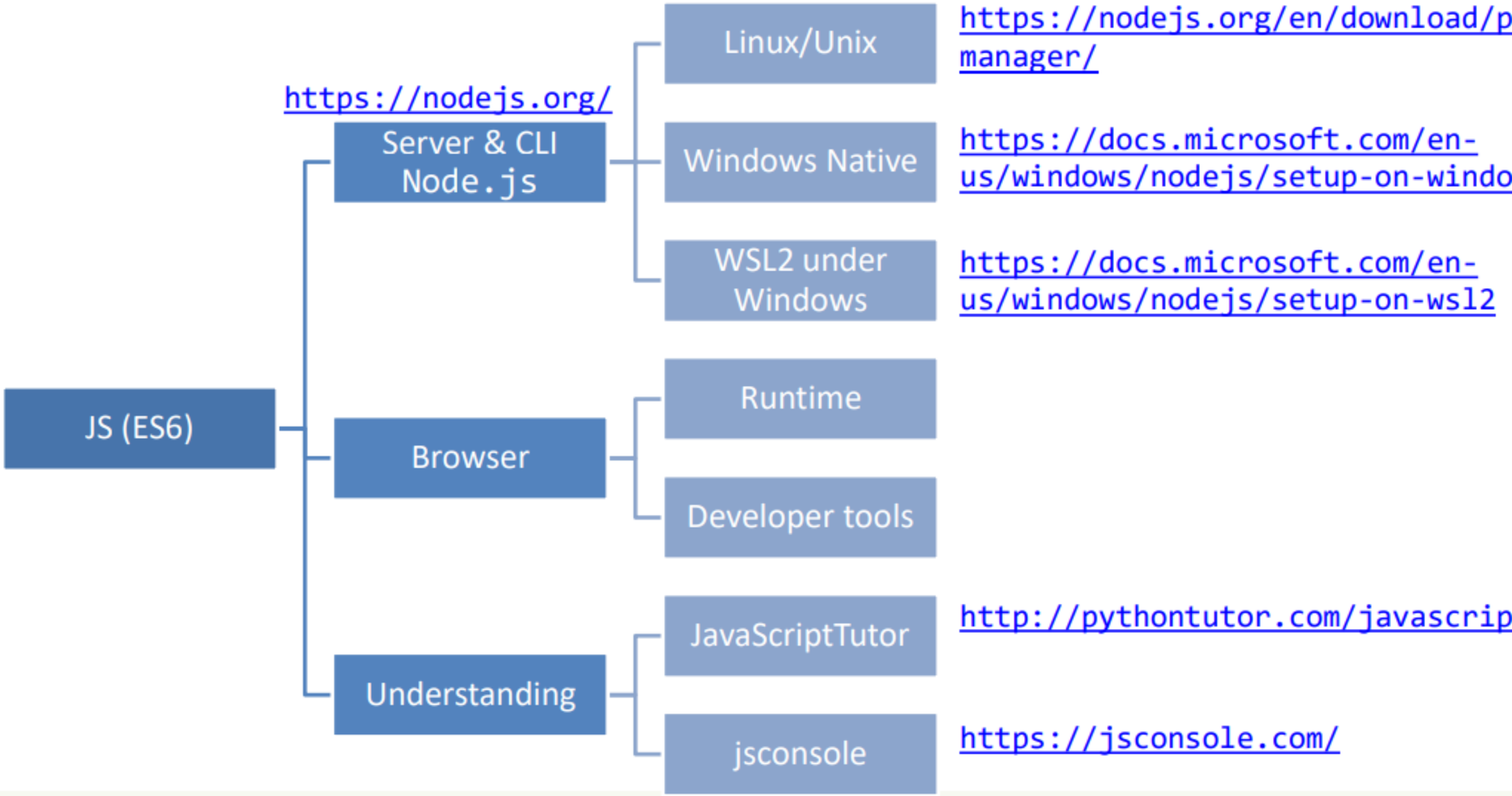
- JS é compatível com versões anteriores
– uma vez que algo é aceito como JS válido, não haverá uma mudança futura no idioma
– Membros do TC39: "nós não quebramos a web!"
- JS não é compatível com versões futuras
– novas adições à linguagem não serão executadas em um mecanismo JS mais antigo e podem travar o programa
- o modo estrito foi introduzido para desabilitar semânticas muito antigas (e perigosas)
- O suporte a várias versões é obtido por:
– Transpiling – Babel (https://babeljs.io) converte da sintaxe JS mais recente para uma equivalente
sintaxe mais antiga
– Polyfilling – funções e métodos definidos pelo usuário (ou biblioteca) que “preenche” a falta de um
recurso implementando o mais novo disponível

UNINASSAU

JavaScript

Ambiente de Execução

JS Execution Environments



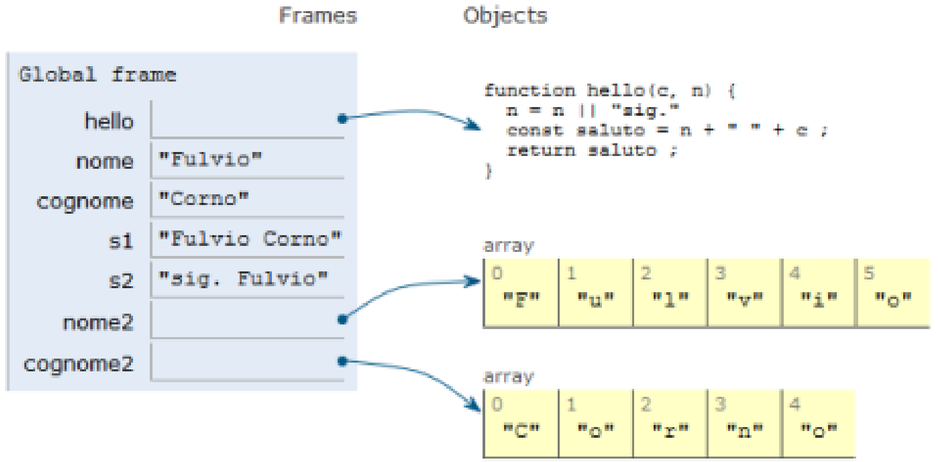
Write code in JavaScript ES6 (drag lower right corner to resize code editor)

```
1  let nome = "Fulvio" ;
2  let cognome = "Corno" ;
3
4  function hello(c, n) {
5    n = n || "sig."
6    const saluto = n + " " + c ;
7    return saluto ;
8  }
9
10 let s1 = hello(cognome, nome)
11 let s2 = hello(nome)
12
13 let nome2 = [...nome]
14 let cognome2 = [...cognome]
```

line that just executed
next line to execute

<< First | < Prev | Next > | Last >>

Done running (16 steps)



http://pythontutor.com/javascript.html

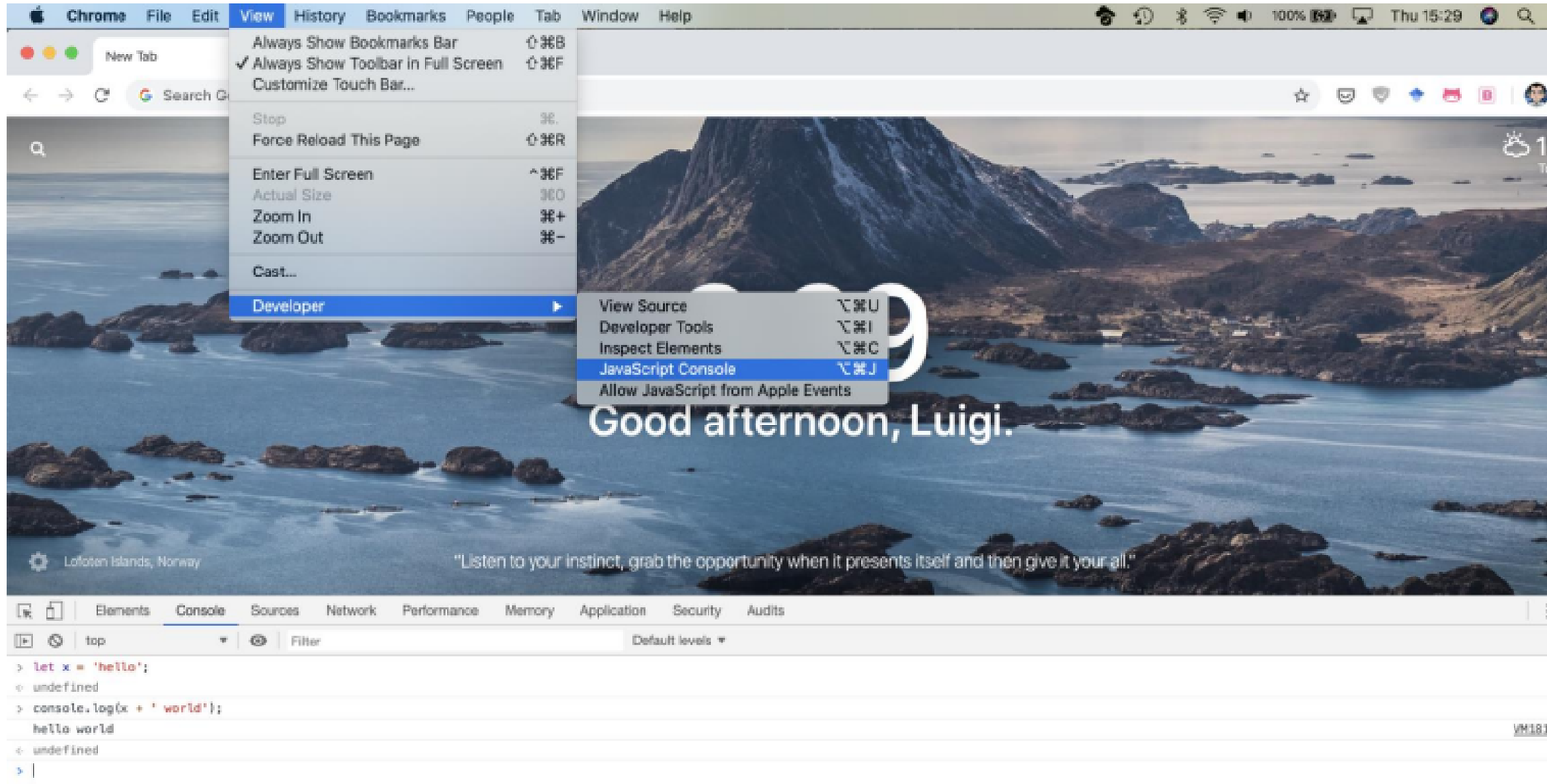
https://pythontutor.com/javascript.html#mode=edit

UNINASSAU

JavaScript

Navegador e JS console

Browser and JS console



UNINASSAU

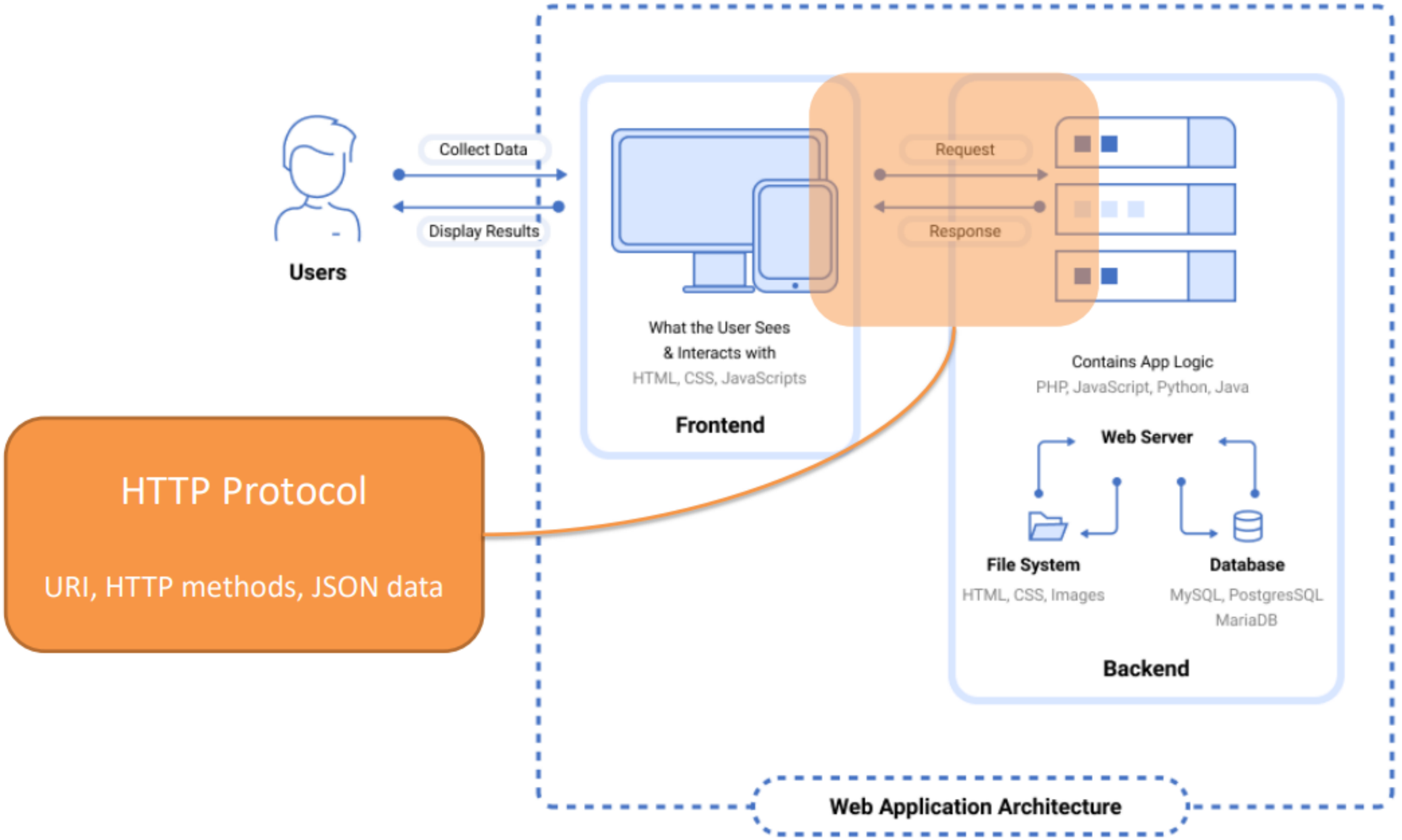
JavaScript

JavaScript

- JS Interpreter Embedded in the Browser
  - Executes within a strict “sandbox”
- JS Scripts loaded by the HTML page
  - `<script src="/js/myscript.js" type="text/javascript"></script>`
- JS Scripts have read-write access to
  - Browser API
  - HTML DOM (including form data)
  - User events and actions

UNINASSAU

JavaScript



UNINASSAU

JavaScript
Protocolo HTTP

HTTP protocol

RFC 2616, RFC 2617
http://www.w3.org/Protocols

```
GET / HTTP/1.1
Host: www.polito.it
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:86.0) Gecko/20100101 Firefox/86.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate, br
DNT: 1
Connection: keep-alive
Cookie: __utma=55042356.701936439.1606736391.1615238467.1615289682.230; __utmz=55042356. [...]
Upgrade-Insecure-Requests: 1
Pragma: no-cache
Cache-Control: no-cache
```

HTTP Request

UNINASSAU

JavaScript
Protocolo HTTP

HTTP protocol

RFC 2616, RFC 2617
http://www.w3.org/Protocols

```
GET / HTTP/1.1
Host: www.polito.it
User-Agent: Mozilla/5.0 (Window
Accept: text/html,application/
Accept-Language: en-US,en;q=0
Accept-Encoding: gzip, defla
DNT: 1
Connection: keep-alive
Cookie: __utma=55042356.7
Upgrade-Insecure-Request
Pragma: no-cache
Cache-Control: no-cache
```

HTTP Response

```
HTTP/1.1 200 OK
Date: Tue, 09 Mar 2021 14:21:35 GMT
Server: Apache
Strict-Transport-Security: max-age=31536000
Content-Security-Policy: script-src 'self' 'unsafe-inline' 'unsafe-eval' [...]
X-Frame-Options: SAMEORIGIN
X-Content-Type-Options: nosniff
X-XSS-Protection: 1; mode=block
Referrer-Policy: no-referrer-when-downgrade
Feature-Policy: accelerometer 'none'; camera 'none'; geolocation 'none'; [...]
Last-Modified: Tue, 09 Mar 2021 14:03:41 GMT
Cache-Control: no-cache, must-revalidate
Vary: Accept-Encoding
Content-Encoding: gzip
Content-Length: 11905
Keep-Alive: timeout=15, max=100
Connection: Keep-Alive
Content-Type: text/html; charset=UTF-8

<!doctype html>
<html xmlns="http://www.w3.org/1999/xhtml" lang="it">
<head>
          <meta charset="UTF-8">
          <title>Politecnico di Torino</title>
. . .
```

Header

Blank line

Body

UNINASSAU

ser educacional

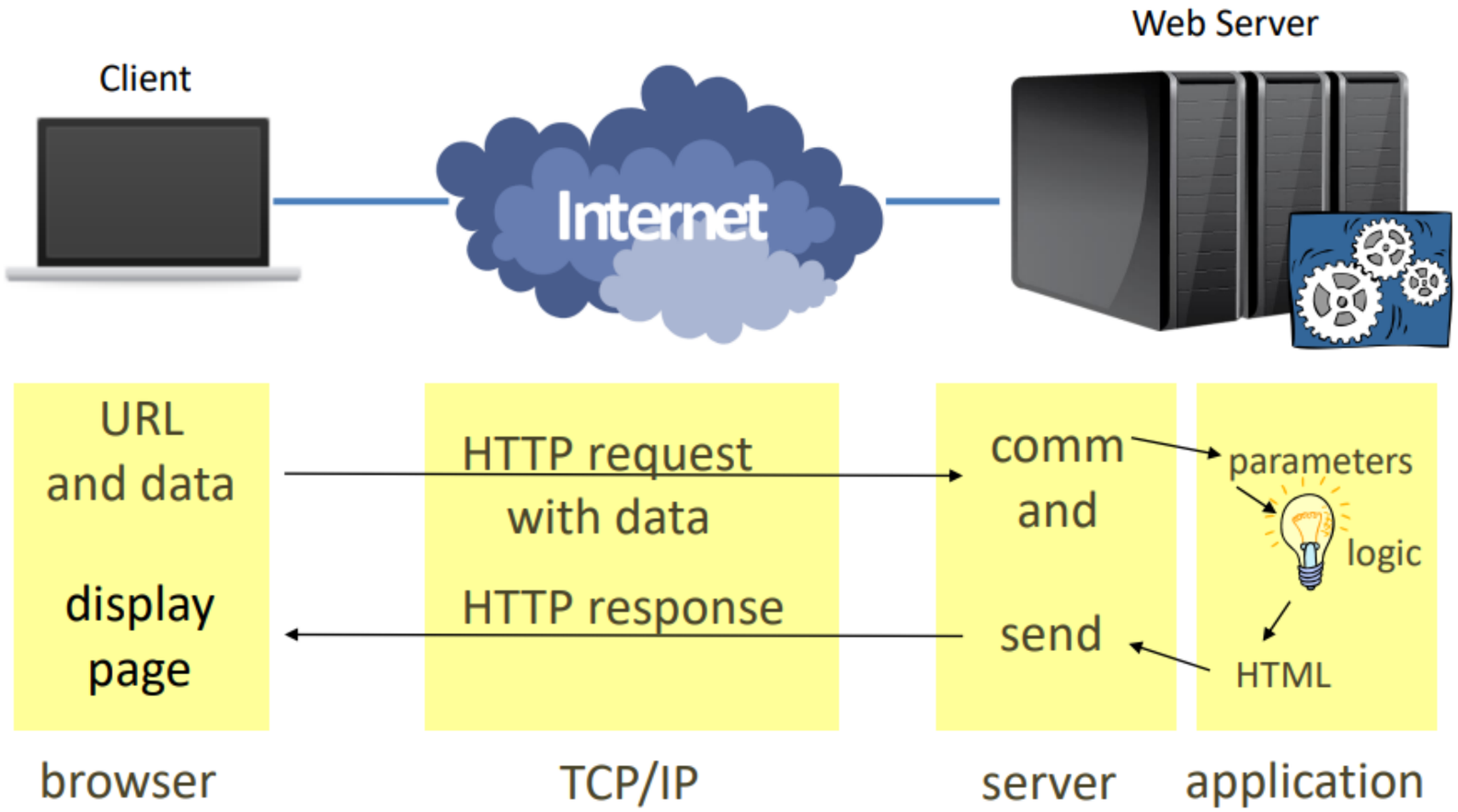
JavaScript
Protocolo HTTP

- Corpo de Resposta Vazio
  - Erros
- Arquivo estático (existe no servidor)
  - HTML (raramente)
  - Imagens, JavaScript, CSS, ...
- Gerado dinamicamente em tempo real pelo servidor
  - HTML (gerado com templates)
  - Dados JSON

UNINASSAU

JavaScript
Protocolo HTTP

Dynamic Web Transaction



UNINASSAU

JavaScript
Protocolo HTTP

HTTP Methods

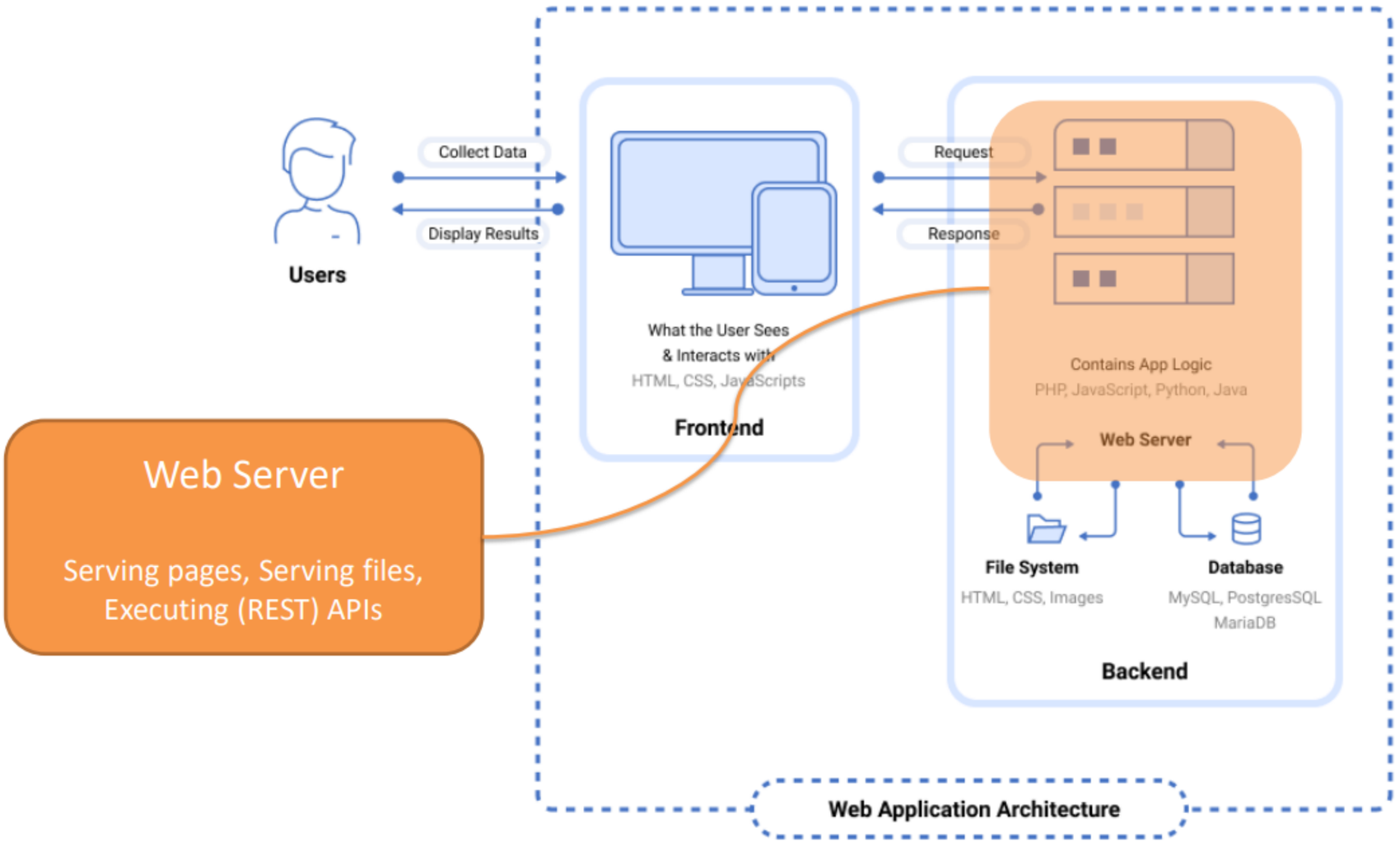
HTTP method	RFC	Request has Body	Response has Body	Safe	Idempotent	Cacheable
GET	RFC 7231	Optional	Yes	Yes	Yes	Yes
HEAD	RFC 7231	Optional	No	Yes	Yes	Yes
POST	RFC 7231	Yes	Yes	No	No	Yes
PUT	RFC 7231	Yes	Yes	No	Yes	No
DELETE	RFC 7231	Optional	Yes	No	Yes	No
CONNECT	RFC 7231	Optional	Yes	No	No	No
OPTIONS	RFC 7231	Optional	Yes	Yes	Yes	No
TRACE	RFC 7231	No	Yes	Yes	Yes	No
PATCH	RFC 5789	Yes	Yes	No	No	No

https://en.wikipedia.org/wiki/Hypertext_Transfer_Protocol#Request_methods

Um método HTTP é idempotente se uma solicitação idêntica puder ser feita uma ou várias vezes seguidas com o mesmo efeito, deixando o servidor no mesmo estado.

UNINASSAU

JavaScript
Protocolo HTTP



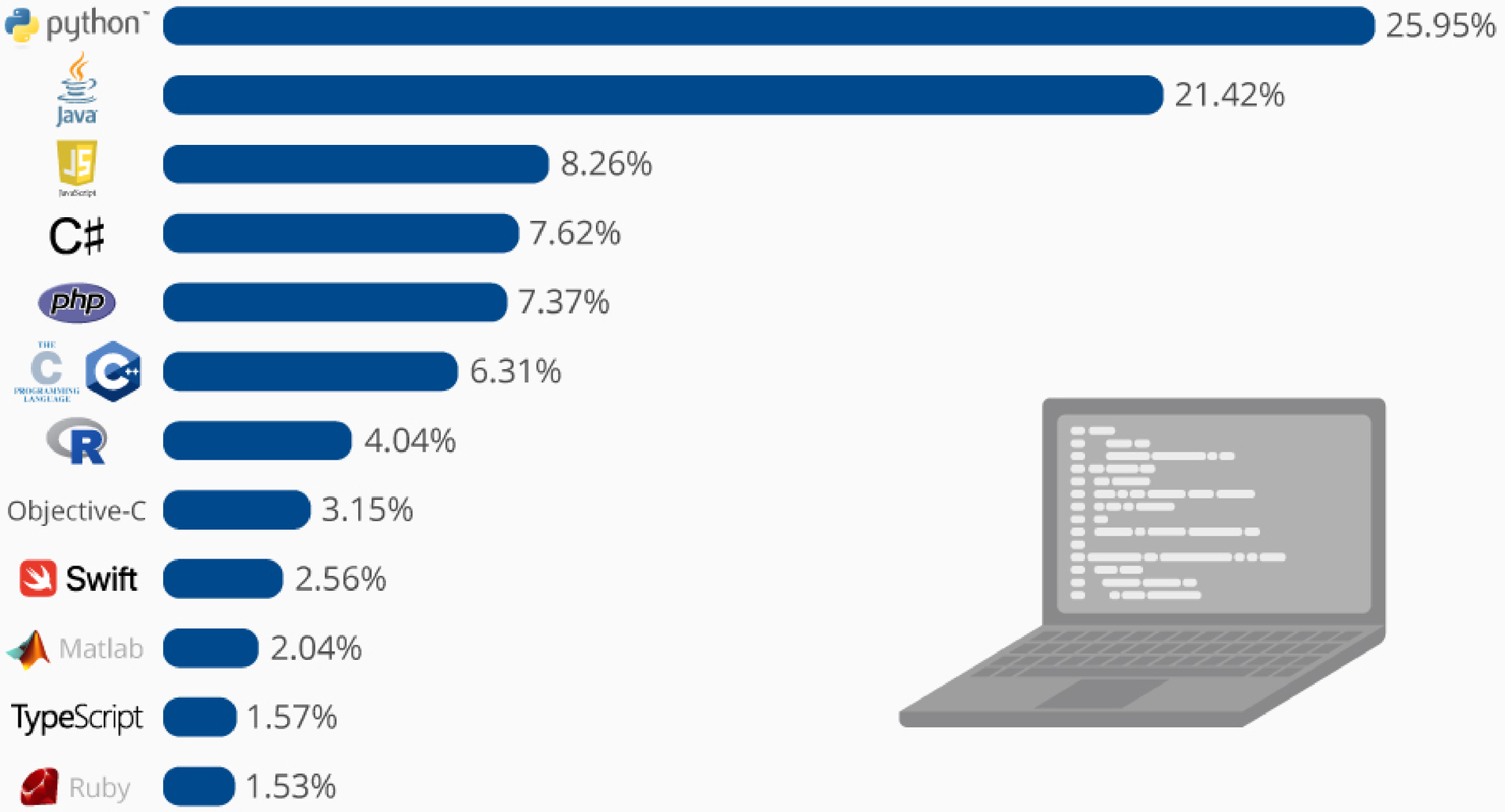
UNINASSAU

JavaScript
Servidor WEB

Um servidor da Web fornece recursos da Web em resposta a uma solicitação
– gerencia o protocolo HTTP para lidar com solicitações e fornecer respostas
• Ele lê ou gera uma página da web
– recebe solicitações de clientes
– lê a página estática do sistema de arquivos
– pede ao servidor de aplicação para gerar páginas dinâmicas (do lado do servidor)
– fornece um arquivo (HTML, CSS, JS, JSON, ...) de volta ao cliente
• Uma conexão HTTP para cada solicitação
• Multi-processo, multi-thread ou pool de processos

The Most Popular Programming Languages

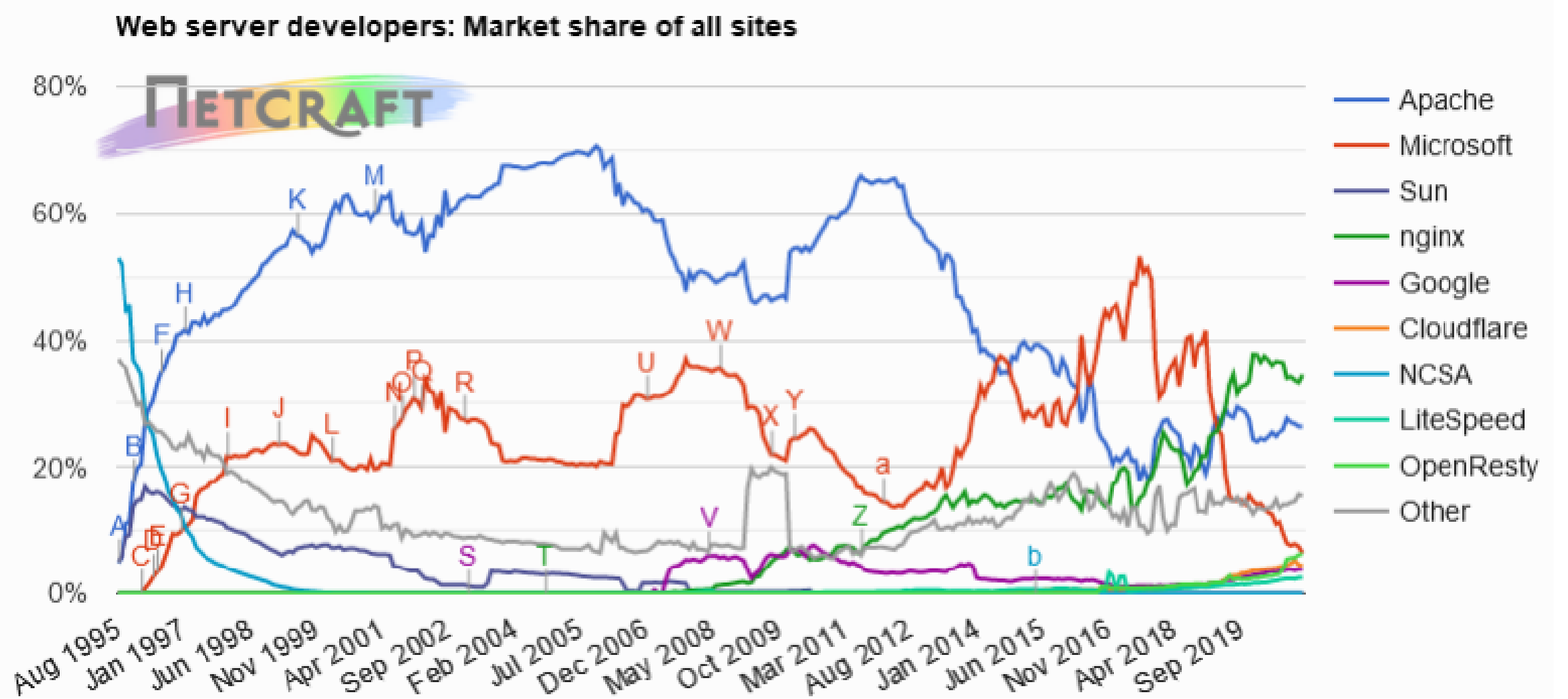
Share of the most popular programming languages in the world*



* Based on the PYPL-Index, an analysis of Google search trends for programming language tutorials.

Source: PYPL

Web Server



Source: http://news.netcraft.com/

https://news.netcraft.com/archives/2021/02/26/february-2021-web-server-survey.html

UNINASSAU

JavaScript
Servidor WEB

EN F5 Sites Search Contact F5

NGINX Port of F5

Solutions & Products Resources Blog Support Partners Get F5 NGINX Free Tri

NGINX

F5 NGINX Sprint

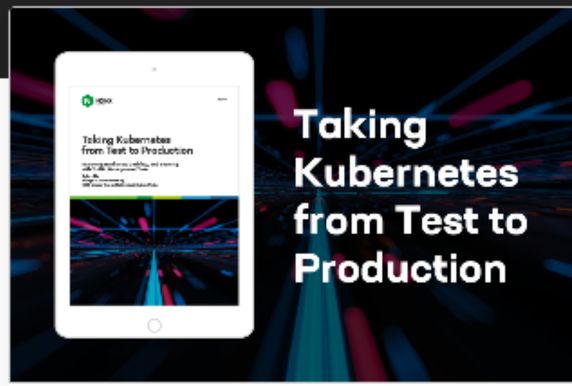
Register for free today and deep dive into the world of NGINX.

Register

The Complete NGINX Cookbook

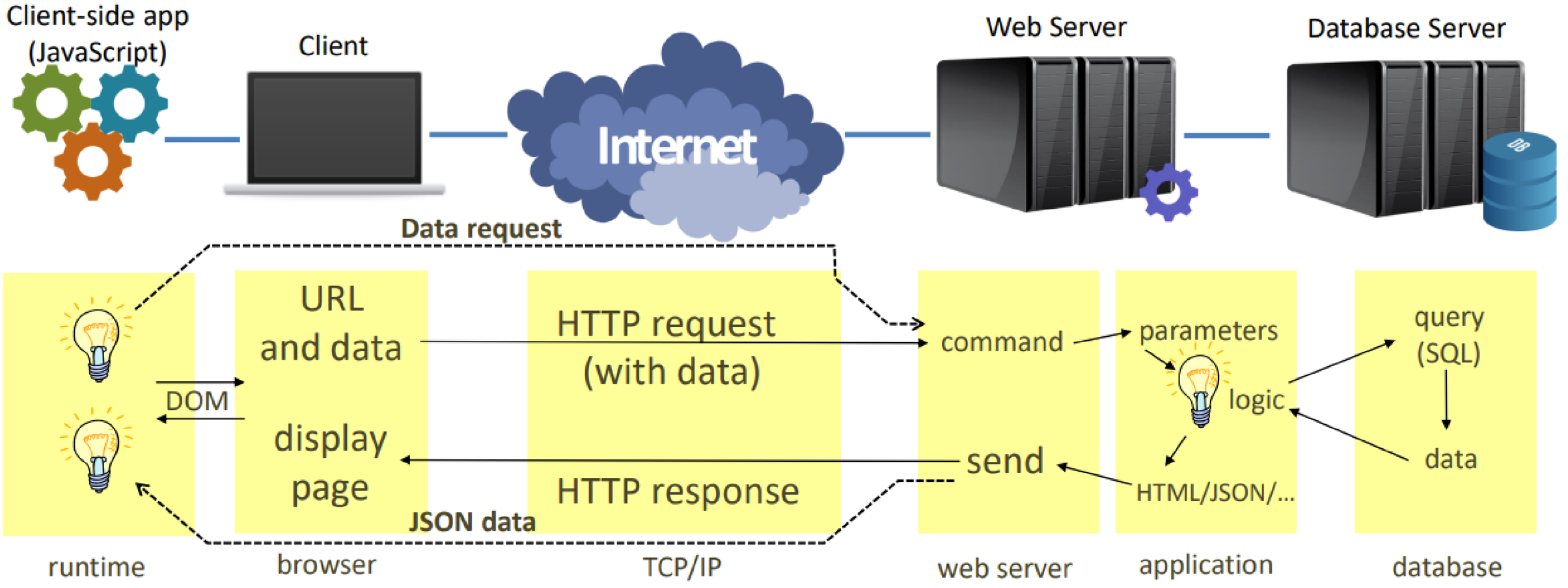
With new and updated recipes for 2022, this free

The Secret to Modern Application Security



Taking Kubernetes from Test to Production

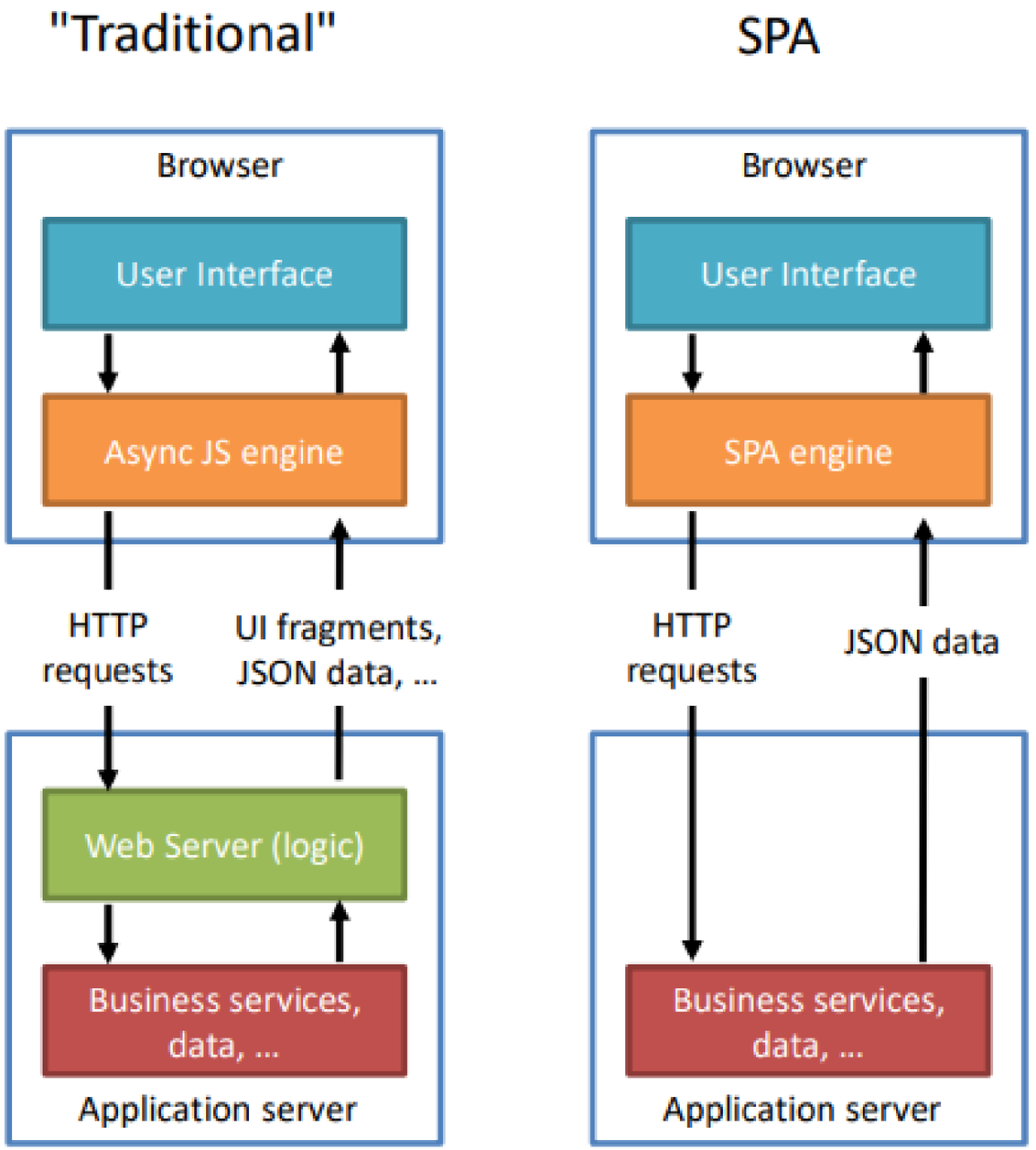
All The Layers At Work…



UNINASSAU

JavaScript

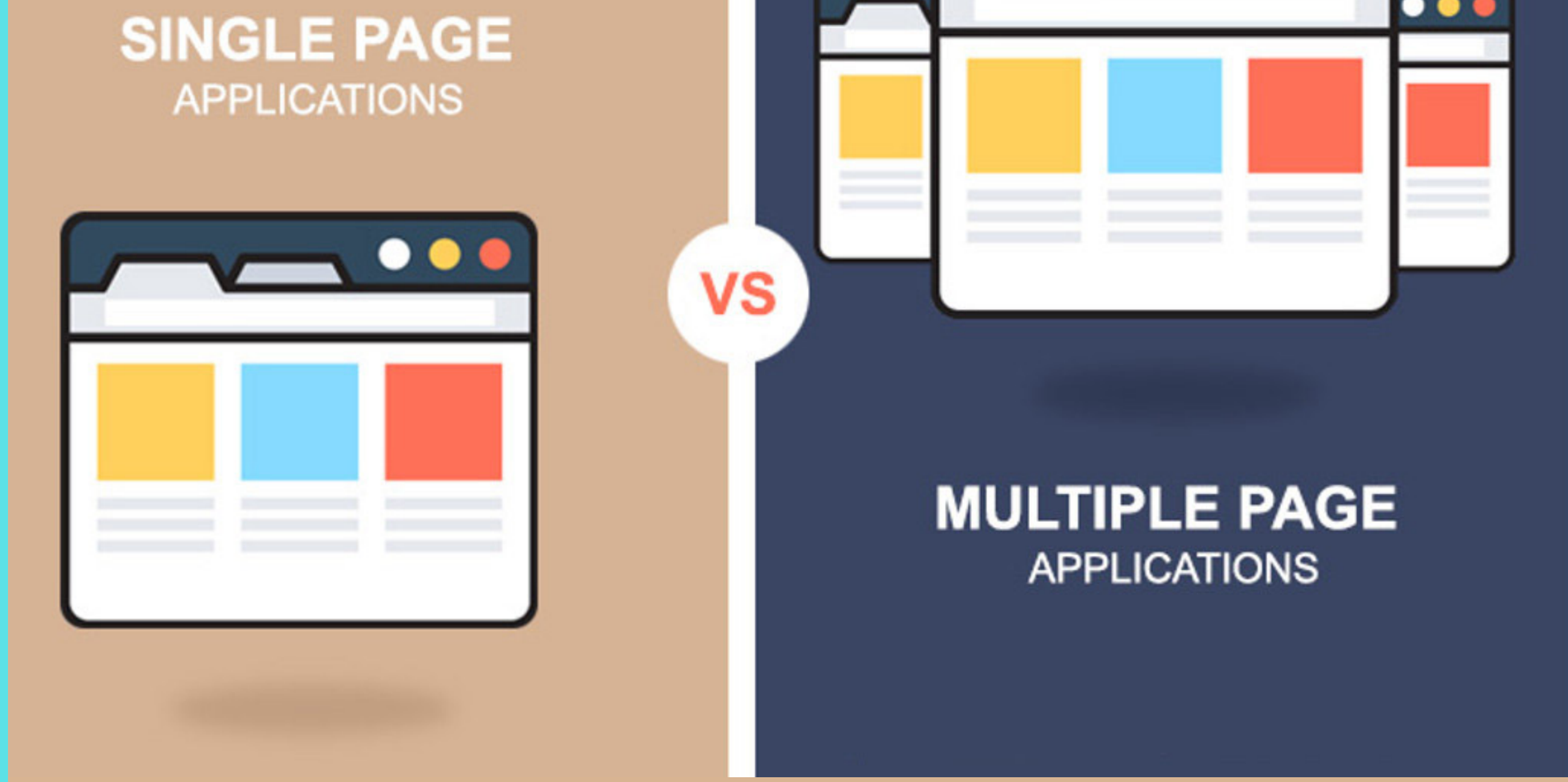
Single Page Application



UNINASSAU

JavaScript

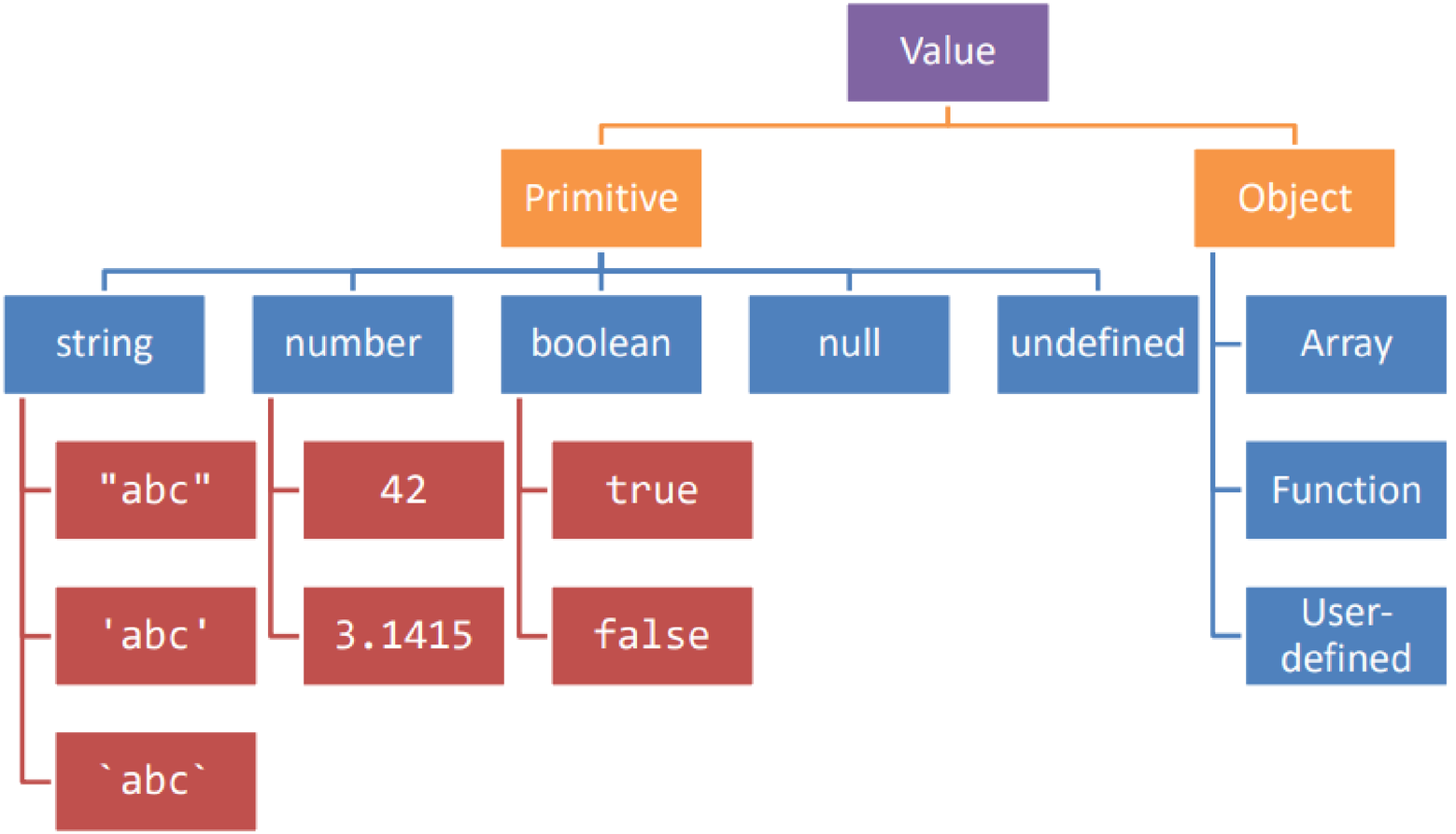
Single Page Application



UNINASSAU

JavaScript

Tipos



UNINASSAU

JavaScript

Booleano

Boolean, true-*truthy*, false-*falsy*, comparisons

- 'boolean' type with literal values: `true`, `false`
- When converting to boolean
  - The following values are 'falsy'
    - `0`, `-0`, `NaN`, `undefined`, `null`, `''` (empty string)
  - Every other value is 'truthy'
    - `3`, `'false'`, `[]` (empty array), `{}` (empty object)

```
> Boolean(3)
true
> Boolean('')
false
> Boolean(' ')
true
```

- Booleans and Comparisons
  - `a == b` *// convert types and compare results*
  - `a === b` *// inhibit automatic type conversion and compare results*

UNINASSAU

JavaScript

Declaração de Variáveis

Variable declarations

Declarator	Can reassign?	Can re-declare?	Scope	Hoisting *	Note
`let`	Yes	No	Enclosing block {…}	No	***Preferred***
`const`	No §	No	Enclosing block {…}	No	***Preferred***
`var`	Yes	Yes	Enclosing function, or global	Yes, to beginning of function or file	*Legacy, beware its quirks, try not to use*
None (implicit)	Yes	N/A	Global	Yes	*Forbidden in strict mode*

§ Prevents reassignment (a=2), does not prevent changing the value of the referred object (a.b=2)

* Hoisting = "lifting up" the definition of a variable (not the initialization!) to the top of the current scope (e.g., the file or the function)

UNINASSAU

JavaScript

Declaração de Variáveis

```
"use strict" ;

let a = 1 ;
const b = 2 ;
let c = true ;

let a = 5 ; // SyntaxError: Identifier 'a' has already been declared
```

https://playcode.io/936965

UNINASSAU

Tipos Primitivos vs Referências

O que são primitivos?

Então vamos começar - o que são "Primitivos"?

Aqui está um exemplo:

```
var age = 28
```

O que são "tipos de referência" então?

Objetos e Arrays!

```
var person = {
  name: 'Max',
  age: 28,
}

var hobbies = ['Sports', 'Cooking']
```

UNINASSAU

Let

A palavra-chave let foi introduzida em ES6 (2015).

Variáveis definidas com let não podem ser redeclaradas.

Variáveis definidas com let devem ser declaradas antes do uso.

Variáveis definidas com let have Block Scope.

Example

```
let x = "John Doe";

let x = 0;

// SyntaxError: 'x' has already been declared
```

https://playcode.io/936799

UNINASSAU

Variáveis e Escopo

Scope

Typically, you don't create a new scope in this way!

```
"use strict" ;

let a = 1 ;
const b = 2 ;
let c = true ;

{ // creating a new scope...
   let a = 5 ;
   console.log(a) ;
}

console.log(a) ;
```

Each { } is called a **block**. 'let' and 'const' variables are *block-scoped*.

They exist only in their defined and inner scopes.

https://codepen.io/naubergois/pen/XWEqBzE

UNINASSAU

Variáveis e Escopo

Scope and Hoisting

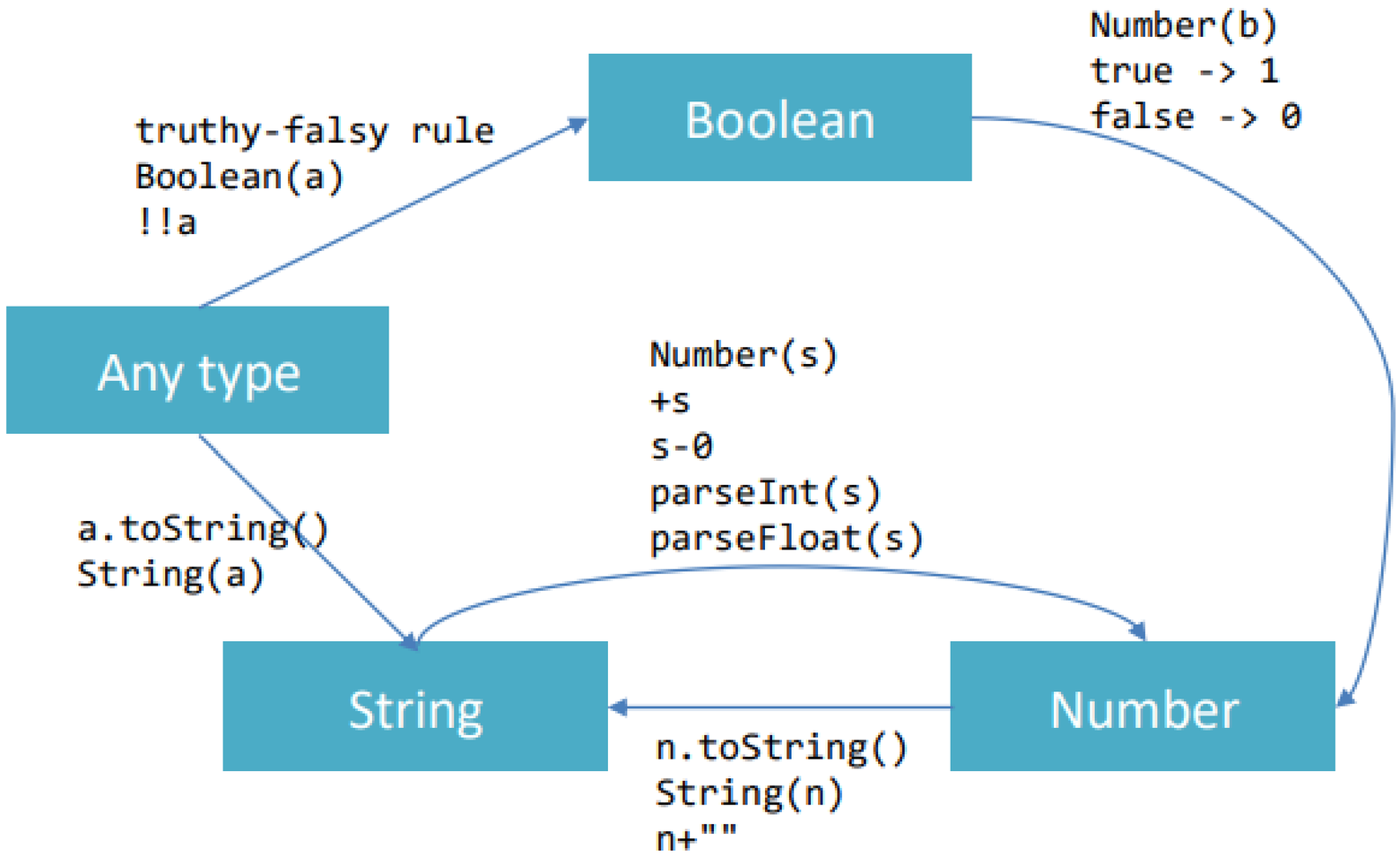
```
"use strict" ;

function example(x) {
    let a = 1 ;

    console.log(a) ;    // 1
    console.log(b) ;    // ReferenceError: b is not defined
    console.log(c) ;    // undefined

    if( x>1 ) {
        let b = a+1 ;
        var c = a*2 ;
    }

    console.log(a) ; // 1
    console.log(b) ; // ReferenceError: b is not defined
    console.log(c) ; // 2
}

example(2) ;
```

```
var c ; // hoisted
```


https://codepen.io/naubergois/pen/rNdvZzR?editors=1111

UNINASSAU

Conversão de Tipos



UNINASSAU

Operadores Lógicos

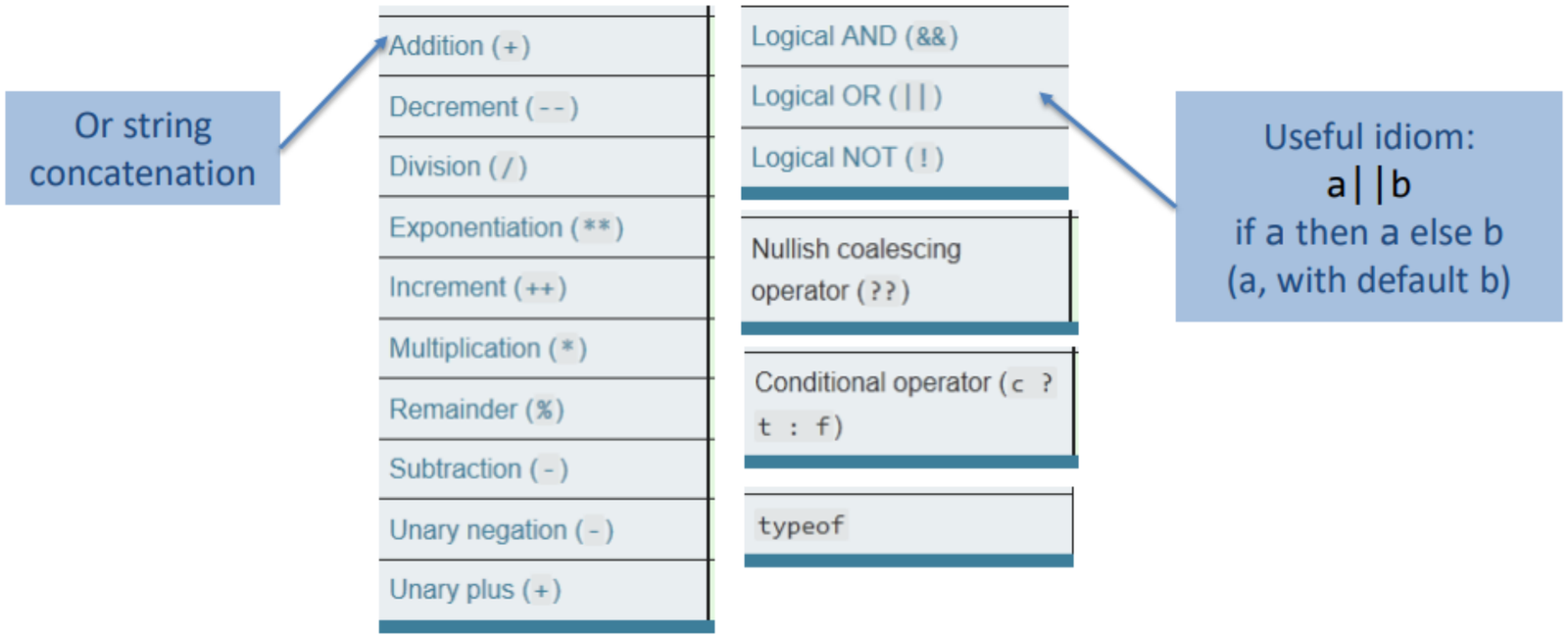
Logical operators

Operator	Usage	Description
Logical AND (`&&`)	`expr1 && expr2`	Returns `expr1` if it can be converted to `false`; otherwise, returns `expr2`. Thus, when used with Boolean values, `&&` returns `true` if both operands are true; otherwise, returns `false`.
Logical OR (`||`)	`expr1 || expr2`	Returns `expr1` if it can be converted to `true`; otherwise, returns `expr2`. Thus, when used with Boolean values, `||` returns `true` if either operand is true; if both are false, returns `false`.
Logical NOT (`!`)	`!expr`	Returns `false` if its single operand that can be converted to `true`; otherwise, returns `true`.

UNINASSAU

Operadores Lógicos

Common operators



UNINASSAU

Funções Matemáticas

Mathematical functions (`Math` global object)

- **Constants**: `Math.E, Math.LN10, Math.LN2, Math.LOG10E, Math.LOG2E, Math.PI, Math.SQRT1_2, Math.SQRT2`
- **Functions**: `Math.abs(), Math.acos(), Math.acosh(), Math.asin(), Math.asinh(), Math.atan(), Math.atan2(), Math.atanh(), Math.cbrt(), Math.ceil(), Math.clz32(), Math.cos(), Math.cosh(), Math.exp(), Math.expm1(), Math.floor(), Math.fround(), Math.hypot(), Math.imul(), Math.log(), Math.log10(), Math.log1p(), Math.log2(), Math.max(), Math.min(), Math.pow(), Math.random(), Math.round(), Math.sign(), Math.sin(), Math.sinh(), Math.sqrt(), Math.tan(), Math.tanh(), Math.trunc()`

Conditional statements

```
if (condition) {
  statement_1;
} else {
  statement_2;
}
```

if truthy (beware!)

```
if (condition_1) {
  statement_1;
} else if (condition_2) {
  statement_2;
} else if (condition_n) {
  statement_n;
} else {
  statement_last;
}
```

May also be a string

```
switch (expression) {
  case label_1:
    statements_1
    [break;]
  case label_2:
    statements_2
    [break;]
    …
  default:
    statements_def
    [break;]
}
```

Loop statements

```
for ([initialExpression]; [condition]; [incrementExpression]) {
  statement ;
}
```

Usually declares loop variable

May use `break;` or `continue;`

```
do {
  statement ;
} while (condition);
```

```
while (condition) {
  statement ;
}
```

Preferred

Special '`for`' statements

```
for (variable in object) {
  statement ;
}
```

- Iterates the variable over all the enumerable **properties** of an **object**
- Do not use to traverse an array (use numerical indexes, or for-of)

```
for( let a in {x: 0, y:3}) {
     console.log(a) ;
}
```

```
x
y
```

```
for (variable of iterable) {
  statement ;
}
```

- Iterates the variable over all values of an *iterable object* (including Array, Map, Set, string, arguments …)
- Returns the *values*, not the keys

```
for( let a of [4,7]) {
     console.log(a) ;
}
```

```
4
7
```

```
for( let a of "hi" ) {
     console.log(a) ;
}
```

```
h
i
```

https://playcode.io/937048

Exception handling

```
try {
  statements ;
} catch(e) {
  statements ;
}
```

```
try {
  statements ;
} catch(e) {
  statements ;
} finally {
  statements ;
}
```

Exception object

```
throw object ;
```

```
EvalError
RangeError
ReferenceError
SyntaxError
TypeError
URIError
DOMException
```

Contain fields: name, message

Executed in any case, at the end of try and catch blocks

UNINASSAU

Arrow Function

5 Differences Between Arrow and Regular Functions

```
const greet = (who) => {
  return `Hello, ${who}!`;
}
```

VS

```
const greet = function(who) {
  return `Hello, ${who}`;
}
```

https://playcode.io/936805

UNINASSAU

Arrow Function

```
// Function declaration
function greet(who) {
  return `Hello, ${who}!`;
}
```

```
// Function expression
const greet = function(who) {
  return `Hello, ${who}`;
}
```

UNINASSAU

Arrow Function

```
function myFunction() {
  console.log(this);
}

// Simple invocation
myFunction(); // logs global object (window)
```

During a *method invocation* the value of `this` is the object owning the method:

```
const myObject = {
  method() {
    console.log(this);
  }
};
// Method invocation
myObject.method(); // logs myObject
```

UNINASSAU

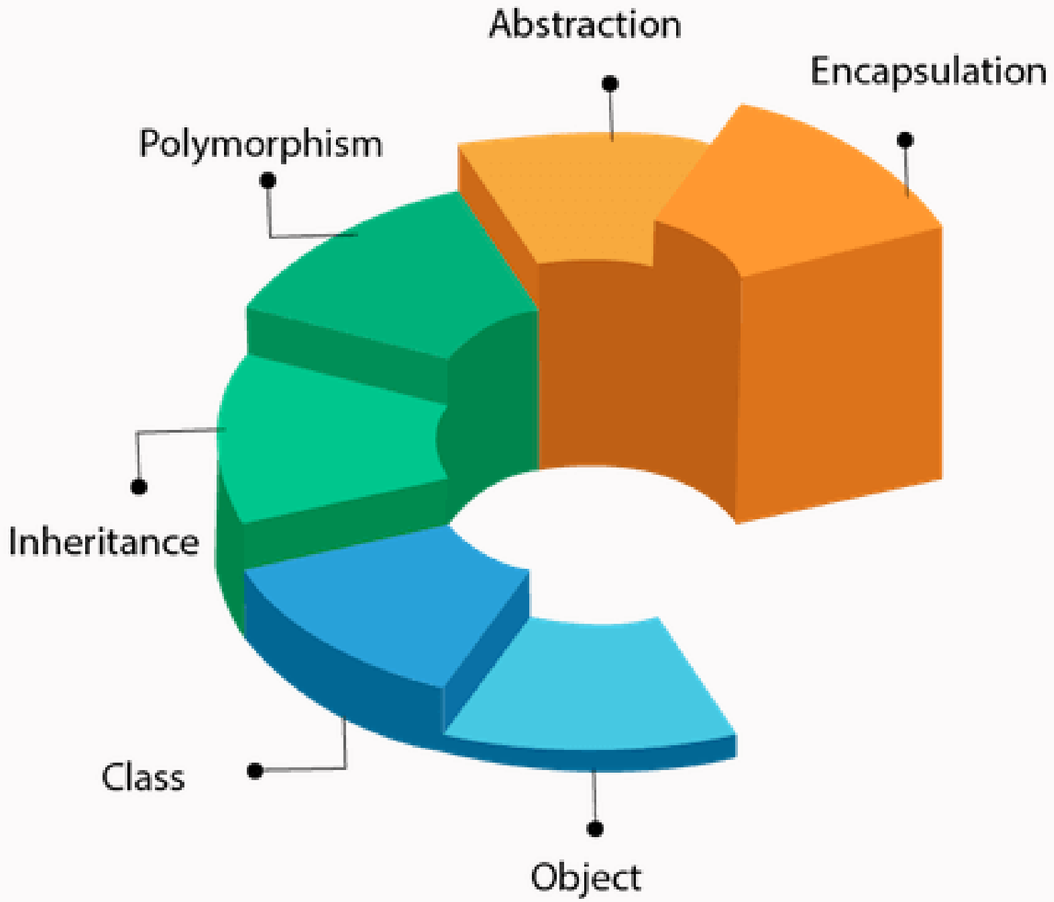
Arrow Function

```
const myObject = {
  myMethod(items) {
    console.log(this); // logs myObject
    const callback = () => {
      console.log(this); // logs myObject
    };
    items.forEach(callback);
  }
};

myObject.myMethod([1, 2, 3]);
```

UNINASSAU

Orientação a Objetos

OOPs (Object-Oriented Programming System)



https://playcode.io/936809

UNINASSAU

ser educacional

Arrays

Arrays in JavaScript

educba.com

https://playcode.io/936810

UNINASSAU

ser educacional

Arrays

Creating an array

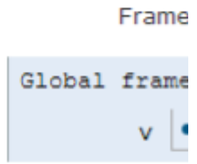
Elements are indexed at positions 0...length-1

Do not access elements outside range

```
let v = [] ;
```

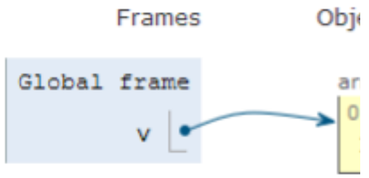
```
let v = [1, 2, 3] ;
```

```
let v = Array.of(1, 2, 3) ;
```




```
let v = [1, "hi", 3.1, true];
```

```
let v = Array.of(1, "hi",
3.1, true) ;
```

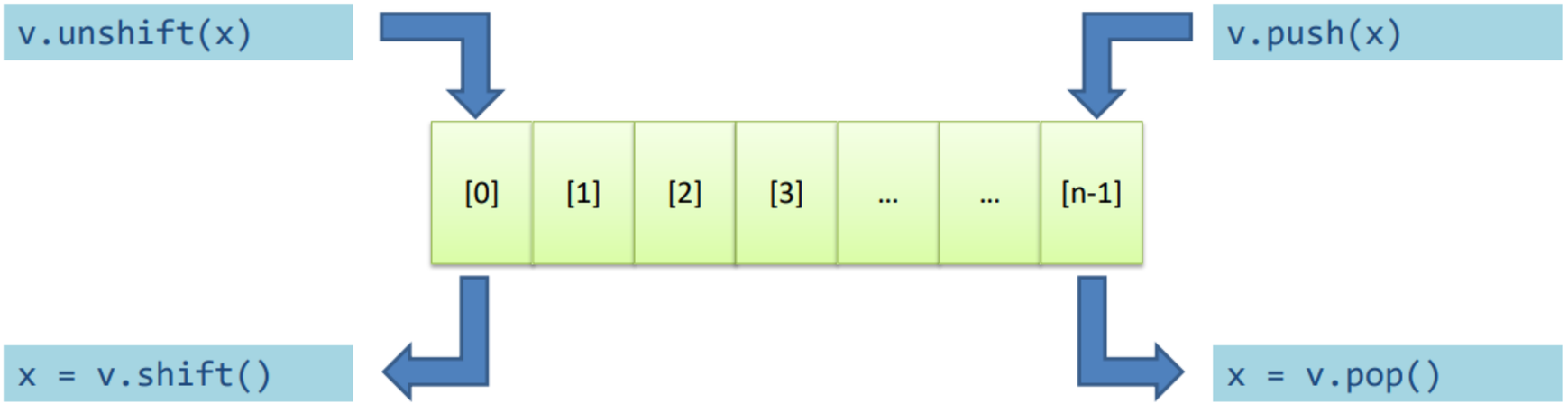


https://playcode.io/937197

UNINASSAU

Arrays

Adding and Removing from arrays (in-place)



UNINASSAU

Arrays

Copying arrays

```
let v = [] ;
v[0] = "a" ;
v[1] = 8 ;

let alias = v ;
alias[1] = 5 ;
```

```
> console.log(v);
[ 'a', 5 ]
undefined
> console.log(alias);
[ 'a', 5 ]
undefined
```

?

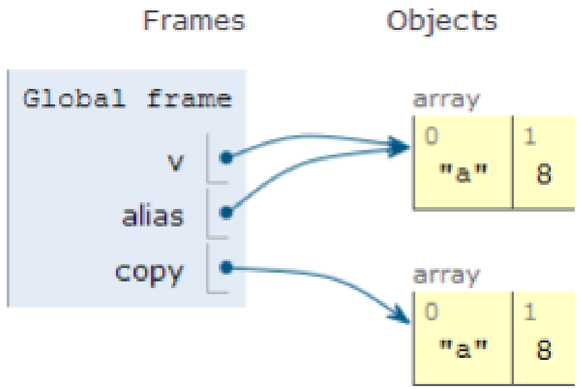
https://playcode.io/937201

UNINASSAU

Arrays

Copying arrays

```
let v = [] ;
v[0] = "a" ;
v[1] = 8 ;

let alias = v ;
let copy = Array.from(v) ;
```



Array.from creates a *shallow copy*

Creates an array from any iterable object

https://playcode.io/937203

UNINASSAU

Objects

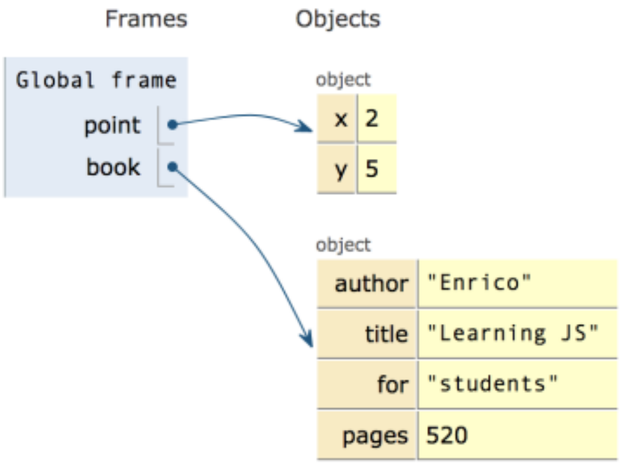
Object

- An object is an unordered collection of properties
  - Each property has a **name** (key), and a **value**
- You store and retrieve *property values*, through the *property names*
- Object creation and initialization:

```
let point = { x: 2, y: 5 };

let book = {
  author : "Enrico",
  title : "Learning JS",
  for: "students",
  pages: 520,
};
```

Object literals syntax:

```
{"name": value,
"name": value, }
        or:
 {name: value,
 name: value, }
```



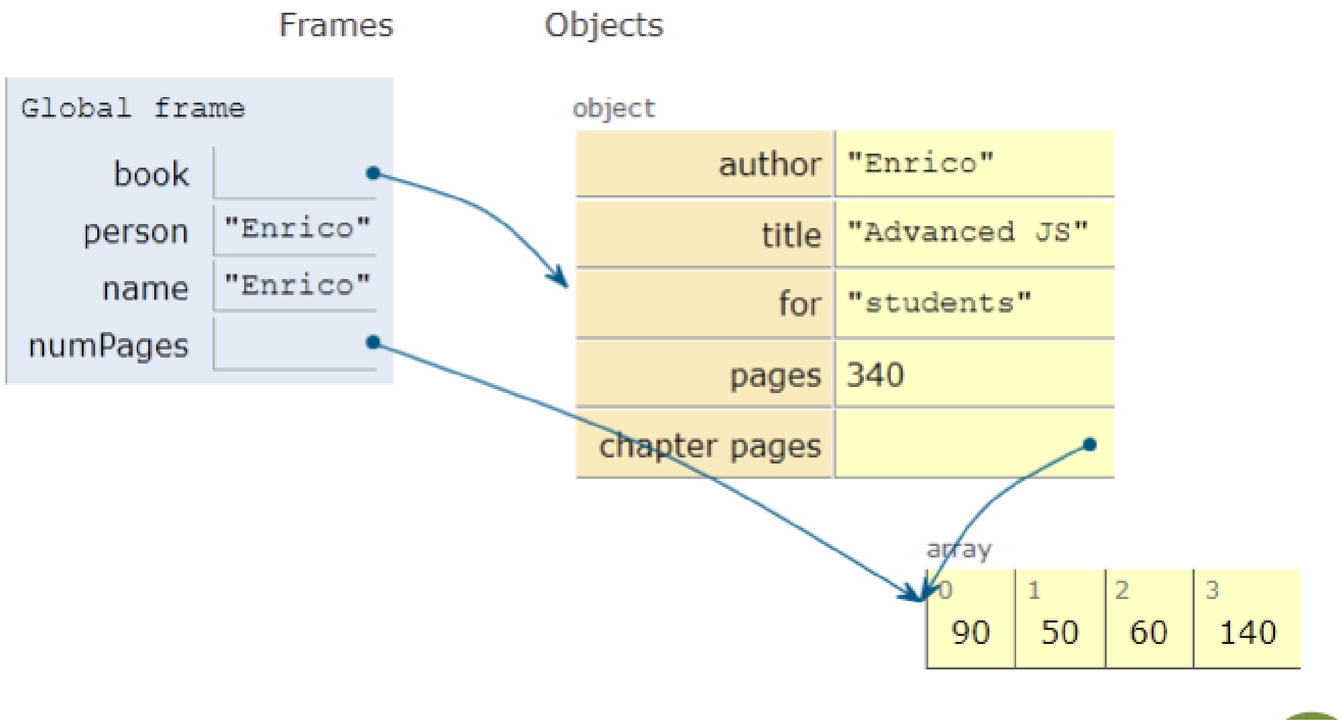
https://playcode.io/937206

Accessing properties

- Dot (.) or square brackets [ ] notation

The . dot notation and omitting the quotes are allowed **when the property name is a valid identifier, only**.
book.title or book['title']
book['my title'] and not ~~book.my title~~

```
let book = {
  author : "Enrico",
  title : "Learning JS",
  for: "students",
  pages: 340,
  "chapter pages": [90,50,60,140]
};

let person = book.author;
let name = book["author"];
let numPages =
    book["chapter pages"];
book.title = "Advanced JS";
book["pages"] = 340;
```



https://playcode.io/937208

UNINASSAU | ser educacional

Objects

Iterating over properties

- `for .. in` iterates over the properties

```
for( let a in {x: 0, y:3}) {
     console.log(a) ;
}
```

```
x
y
```

```
let book = {
  author : "Enrico",
  pages: 340,
  chapterPages: [90,50,60,140],
};

for (const prop in book)
  console.log(`${prop} = ${book[prop]}`)
```

```
author = Enrico
pages = 340
chapterPages = 90,50,60,140
```

https://playcode.io/937210

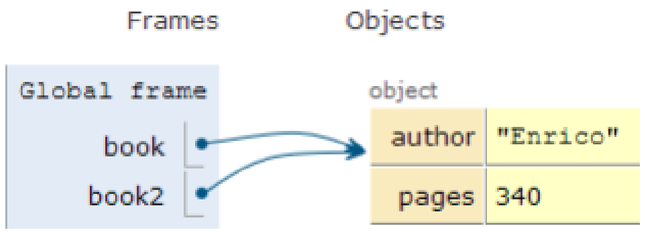
UNINASSAU

Objects

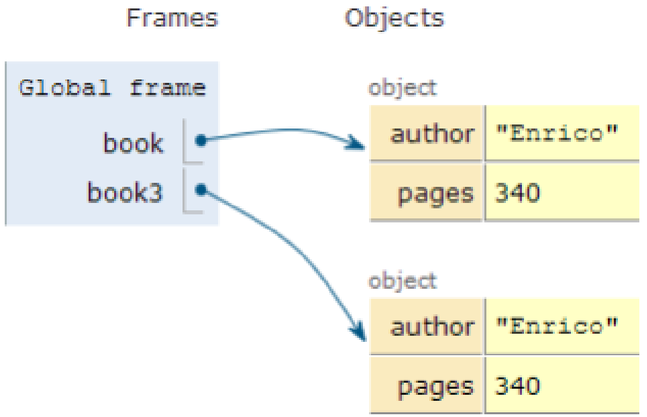
https://playcode.io/937211

Copying objects

```
let book = {
  author : "Enrico",
  pages: 340,
};

let book2 = book;  // ALIAS
```




```
let book = {
  author : "Enrico",
  pages: 340,
};

let book3 =                  // COPY
   Object.assign({}, book);
```



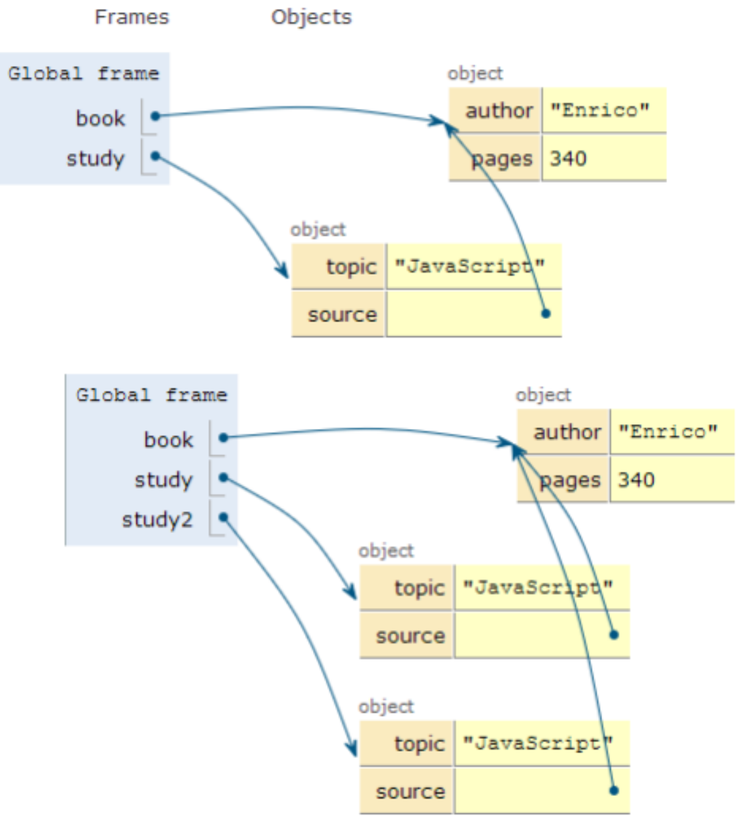
UNINASSAU

Objects

Beware! Shallow copy, only

```
let book = {
  author : "Enrico",
  pages: 340,
};

let study = {
  topic: "JavaScript",
  source: book,
};
```

```
let study2 = Object.assign({},
study);
```



https://playcode.io/937212

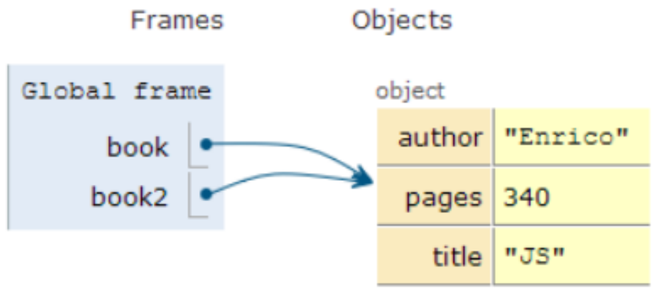
UNINASSAU | ser educacional

Objects

Merge properties (on existing object)

- `Object.assign(target, source, default values, ..);`

```
let book = {
  author : "Enrico",
  pages: 340,
};

let book2 = Object.assign(
 book, {title: "JS"}
);
```



https://playcode.io/937213

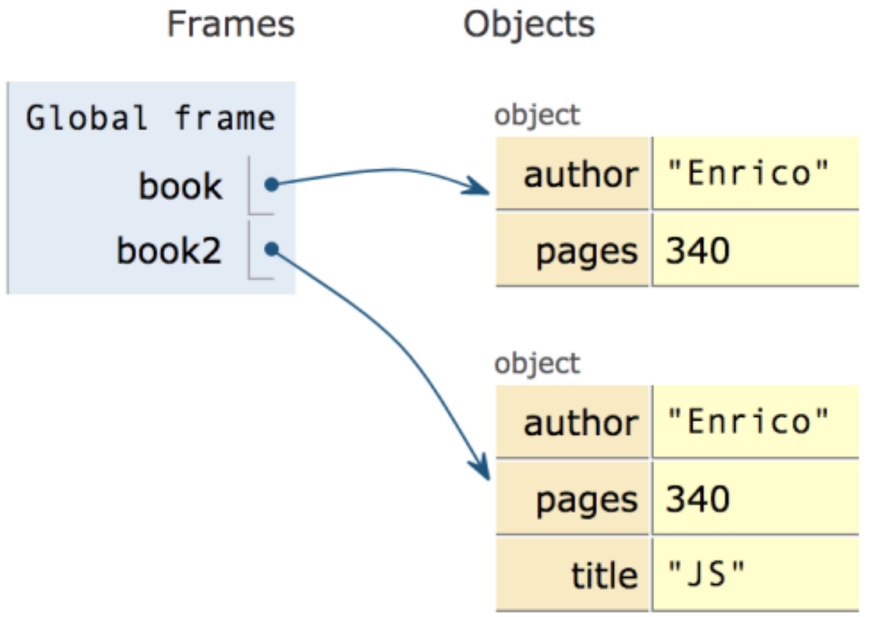
UNINASSAU | ser educacional

Objects

Merge properties (on new object)

- `Object.assign(target, source, default values, ..);`

```
let book = {
  author : "Enrico",
  pages: 340,
};

let book2 = Object.assign(
 {}, book, {title: "JS"}
);
```



https://playcode.io/937214

UNINASSAU | ser educacional

Objects

Copying with spread operator (ES9 – ES2018)

```
let book = {
  author : "Enrico",
  pages: 340,
};

let book2 = {...book, title: "JS"};
let book3 = { ...book2 } ;
console.log(book2);
```

```
{ author: 'Enrico', pages: 340, title: 'JS' }
```

```
const {a,b,...others} =
   {a:1, b:2, c:3, d:4};

console.log(a);
console.log(b);
console.log(others);
```

```
1
2
{ c: 3, d: 4 }
```

https://playcode.io/937215

UNINASSAU

Objects

Checking if properties exist

- Operator `in`
  - Returns true if property is in the object. Do not use with Array

```
let book = {
  author : "Enrico",
  pages: 340,
};

console.log('author' in book);
delete book.author;
console.log('author' in book);
```

```
true
false
```

```
const v=['a','b','c'];

console.log('b' in v);



console.log('PI' in Math);
```

```
false
true
```

https://playcode.io/937216

UNINASSAU

Funções

Declaring functions: 3 ways

1) Classic

```
function do(params) {
  /* do something */
}
```

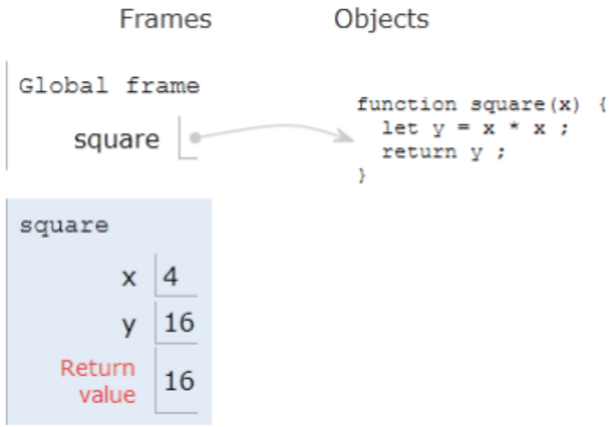
UNINASSAU

Funções

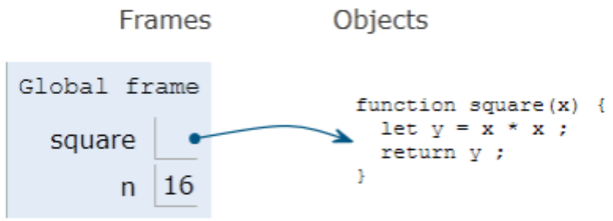
Classic functions

```
function square(x) {
  let y = x * x ;
  return y ;
}

let n = square(4) ;
```

During execution



After execution



https://playcode.io/937217

UNINASSAU

https://playcode.io/937218

Funções

Declaring functions: 3 ways

1) Classic

```
function do(params) {
  /* do something */
}
```

2a) Function expression

```
const fn = function(params) {
  /* do something */
}
```

2b) Named function expression

```
const fn = function do(params) {
  /* do something */
}
```

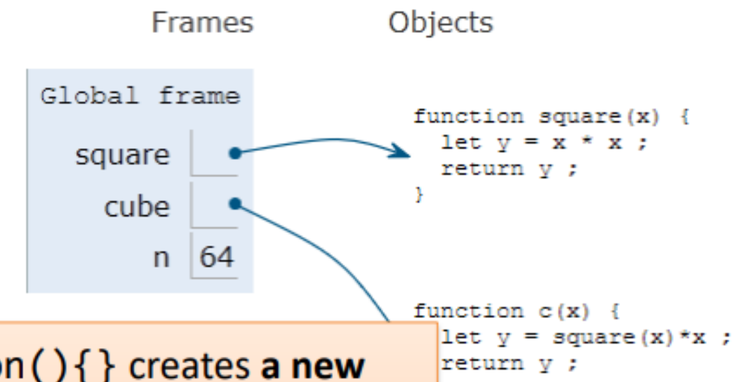
UNINASSAU

Funções

https://playcode.io/937218

Function expression: indistinguishable

```
function square(x) {
  let y = x * x ;
  return y ;
}

let cube = function c(x) {
  let y = square(x)*x ;
  return y ;
}

let n = cube(4) ;
```



The *expression* `function(){}` creates **a new object of type 'function'** and returns the result.

Any variable may "refer" to the function and call it. You can also store that reference into an array, an object property, pass it as a parameter to a function, redefine it, …

method

callback

Declaring functions: 3 ways

1) Classic

```
function do(params) {
  /* do something */
}
```

2a) Function expression

```
const fn = function(params) {
  /* do something */
}
```

3) Arrow function

```
const fn = (params) => {
  /* do something */
}
```

2b) Named function expression

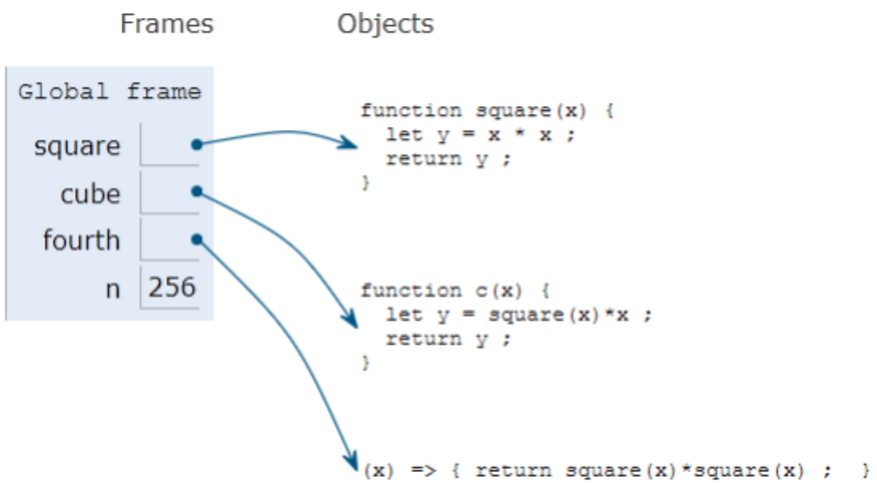
```
const fn = function do(params) {
  /* do something */
}
```

UNINASSAU | ser educacional

Funções

Arrow Function: just a shortcut

```
function square(x) {
  let y = x * x ;
  return y ;
}

let cube = function c(x) {
  let y = square(x)*x ;
  return y ;
}

let fourth = (x) => { return
square(x)*square(x) ;  }

let n = fourth(4) ;
```



https://playcode.io/937219

Parameters in arrow functions

```
const fun = () => {  /* do something */ }          // no params

const fun = param => {  /* do something */ }       // 1 param

const fun = (param) => {  /* do something */ }     // 1 param

const fun = (par1, par2) => {  /* smtg */ } // 2 params

const fun = (par1 = 1, par2 = 'abc') => {  /* smtg */ }   // default values
```

UNINASSAU

Funções

```
function hypotenuse(a, b) {
    const square = x => x*x ;
    return Math.sqrt(square(a) + square(b));
}

function hypotenuse(a, b) {
    function square(x) { return x*x; }
    return Math.sqrt(square(a) + square(b));
}
```

=> Preferred in nested functions

UNINASSAU

Funções

Closure

Um closure é um nome dado a um característica na linguagem pela qual uma função aninhada pode acessar o exterior escopo da função.

Really: one of the most important concepts in JS

UNINASSAU

Funções

Closure

https://playcode.io/937221

JS usa escopo léxico
– Cada nova função define um escopo para as variáveis declaradas dentro
– Funções aninhadas podem acessar o escopo de todas as funções envolventes
• Cada objeto de função lembra o escopo onde está definido, mesmo após a função externa não está mais ativo → Closure

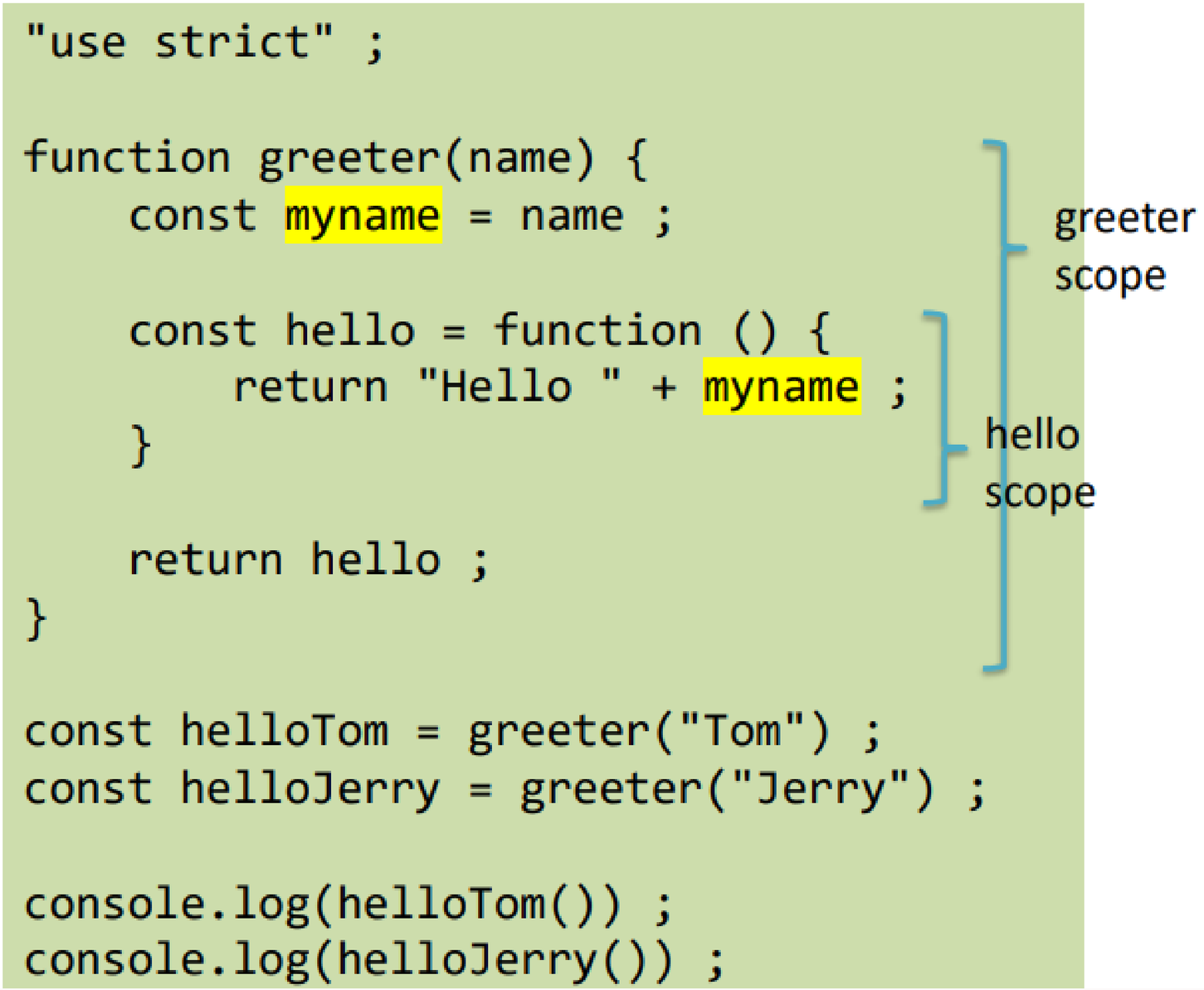
```
"use strict" ;

function greeter(name) {
    const myname = name ;

    const hello = function () {
        return "Hello " + myname ;
    }

    return hello ;
}

const helloTom = greeter("Tom") ;
const helloJerry = greeter("Jerry") ;

console.log(helloTom()) ;
console.log(helloJerry()) ;
```

Warning: not
return hello() ;

hello acessa a variável myname, definido no escopo externo

- **A função é retornada (como OláTom ou OláJerry)**
- **Cada uma das funções “lembra” a referência a myname, quando foi definido**
- **A variável meunome sai de escopo, mas não é destruído**

– Ainda acessível (referido) pelo hello.



Using closures to emulate objects

```
"use strict" ;

function counter() {
    let value = 0 ;

    const getNext = () => {
        value++;
        return value;
    }

    return getNext ;
}
```

```
const count1 = counter() ;
console.log(count1()) ;
console.log(count1()) ;
console.log(count1()) ;

const count2 = counter() ;
console.log(count2()) ;
console.log(count2()) ;
console.log(count2()) ;
```

```
1
2
3
1
2
3
```

https://playcode.io/937222

UNINASSAU

Arrays

https://playcode.io/936812

UNINASSAU

Spread

```
let arr1 = [1, 2, 3]
let arr2 = [...arr1]
console.log(arr2)
// [ 1, 2, 3 ]
```

https://playcode.io/936813

UNINASSAU | ser educacional

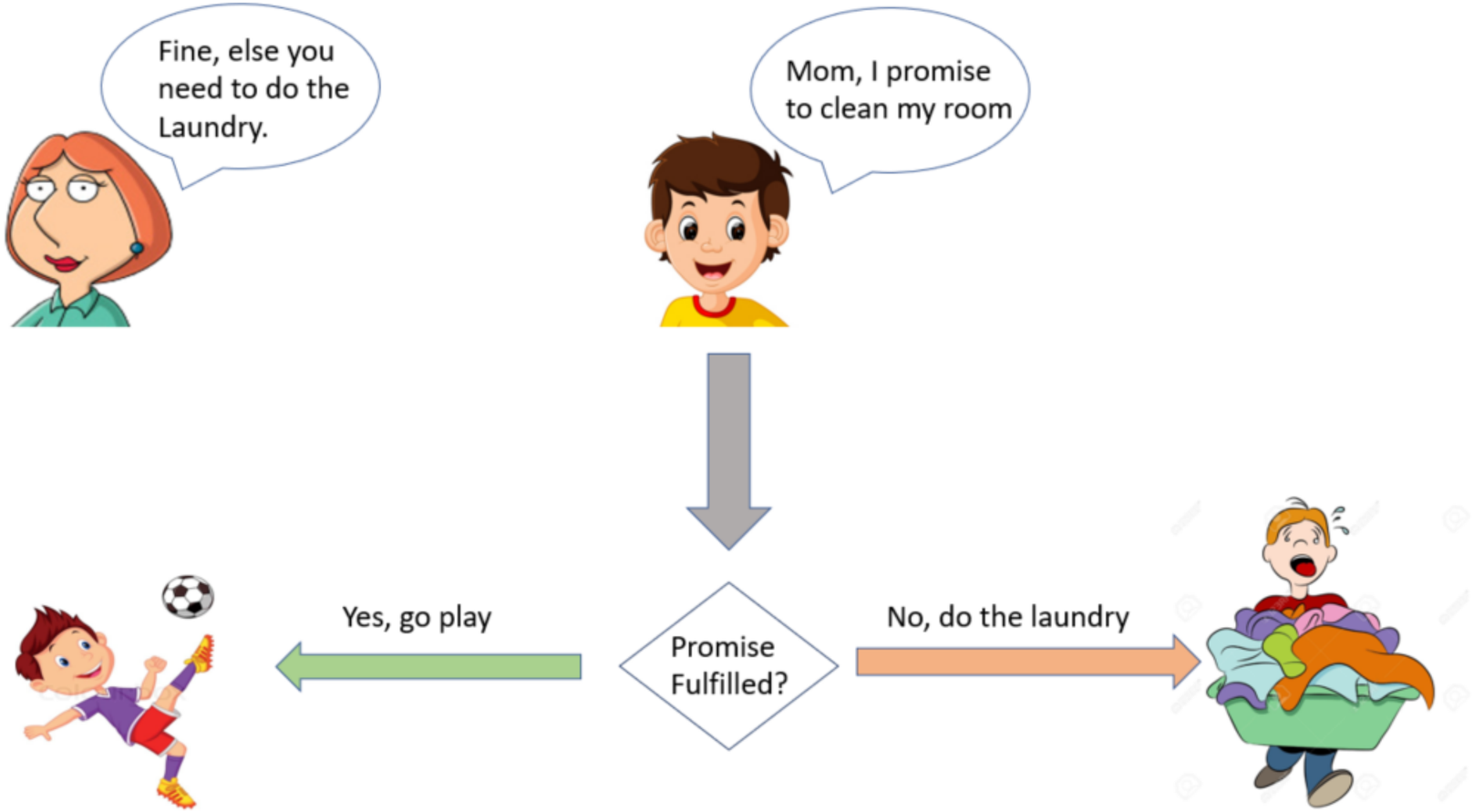
Destructuring

Destructuring in JavaScript

https://playcode.io/936814

UNINASSAU | ser educacional

Promisse



https://playcode.io/936816

UNINASSAU

Alo Mundo

JavaScript Get Element By id, name, class, tag , attribute value

```
<p id="message">A paragraph</p>
```

https://playcode.io/936470

UNINASSAU

Alo Mundo

https://playcode.io/936797

UNINASSAU

Get Element By Name

index.html × style.css × script.js ×

```
24          <label for="very-good">
25              <input type="radio" name="rate" value="Very Good" id="very-good"> Very Good
26          </label>
27      </p>
28      <p>
29          <button id="btnRate">Submit</button>
30      </p>
31      <p id="output"></p>
32      <script>
33          let btn = document.getElementById('btnRate');
34          let output = document.getElementById('output');
35
36          btn.addEventListener('click', () => {
37              let rates = document.getElementsByName('rate');
38              rates.forEach((rate) => {
39                  if (rate.checked) {
40                      output.innerText = `You selected: ${rate.value}`;
41                  }
```

CONSOLE ×

WEBSITE VIEW ×

Please rate the service:

○ Very poor ○ Poor ◉ OK ○ Good ○ Very Good

Submit

You selected: OK

https://playcode.io/936793

UNINASSAU

Get Element By Name

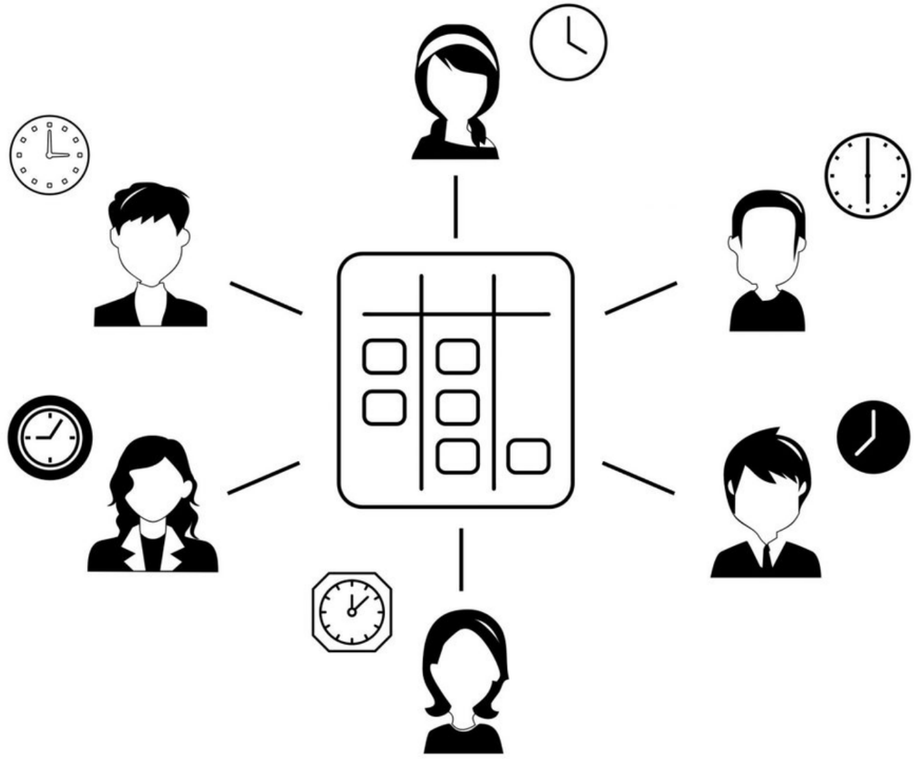
document.getElementsByTagName

< tag >

https://playcode.io/936795

Synchronous
Communication

Asynchronous
Communication



agile
Organization
Development

UNINASSAU

Callback

Uma função de retorno de chamada é uma função
passado para outra função como um argumento, que é então invocado dentro da função externa para completar algum tipo de rotina ou ação.

– Síncrono

– Assíncrono


```
function logQuote(quote) {
  console.log(quote);
}

function createQuote(quote, callback) {
  const myQuote = `Like I always say, '${quote}'`;
  callback(myQuote);
}

createQuote("WebApp I rocks!", logQuote);
```

https://playcode.io/937229

UNINASSAU | ser educacional

Callback

```
setTimeout(function(){
   ...
   ...
   ...

}, 1000);
```

Callback Function

https://scriptverse.academy

https://learntocodetogether.com/callback-functions-in-javascript/

```
let numbers = [4, 2, 5, 1, 3];

numbers.sort(function(a, b) {
  return a - b;
});

console.log(numbers);
```

```
let numbers = [4, 2, 5, 1, 3];

numbers.sort((a, b) => a - b);

console.log(numbers);
```

https://playcode.io/937230

filter() cria um novo array com todos os elementos para os quais o retorno de chamada retorna true

```
const market = [
  { name: 'GOOG', var: -3.2 },
  { name: 'AMZN', var:  2.2 },
  { name: 'MSFT', var: -1.8 }
];

const bad = market.filter(stock => stock.var < 0);
// [ { name: 'GOOG', var: -3.2 }, { name: 'MSFT', var: -1.8 } ]

const good = market.filter(stock => stock.var > 0);
// [ { name: 'AMZN', var: 2.2 } ]
```

https://playcode.io/937231

UNINASSAU

Programação Funcional

```
new_array =
  array.filter ( filter_function ) ;
```

```
new_array = [] ;
for (const el in list)
      if ( filter_function(el) )
            new_array.push(el) ;
```

```
const vehicles = [
  { make: 'Honda',  model: 'CR-V',     type: 'suv',   price: 24045 },
  { make: 'Honda',  model: 'Accord',   type: 'sedan', price: 22455 },
  { make: 'Mazda',  model: 'Mazda 6',  type: 'sedan', price: 24195 },
  { make: 'Mazda',  model: 'CX-9',     type: 'suv',   price: 31520 },
  { make: 'Toyota', model: '4Runner',  type: 'suv',   price: 34210 },
  { make: 'Toyota', model: 'Sequoia',  type: 'suv',   price: 45560 },
  { make: 'Toyota', model: 'Tacoma',   type: 'truck', price: 24320 },
  { make: 'Ford',   model: 'F-150',    type: 'truck', price: 27110 },
  { make: 'Ford',   model: 'Fusion',   type: 'sedan', price: 22120 },
  { make: 'Ford',   model: 'Explorer', type: 'suv',   price: 31660 }
];

const averageSUVPrice = vehicles
  .filter(v => v.type === 'suv')
  .map(v => v.price)
  .reduce( (sum, price, i, array) => sum + price / array.length, 0);

console.log(averageSUVPrice); // 33399
```

https://opensource.com/article/17/6/functional-javascript

https://codepen.io/naubergois/pen/JjLvxxO?editors=1111

React is a JavaScript library for
building user interfaces

HTML, CSS & JavaScript are about building user interfaces as well

React makes building complex, interactive and reactive user interfaces simpler

React is all about “Components”

What is a “Component”?

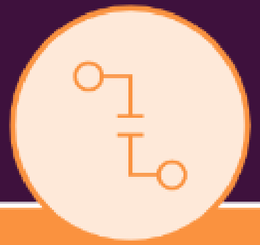
React is all about “Components”

Because all user interfaces in the end are made up of components

Why Components?

Reusability

Don’t repeat yourself

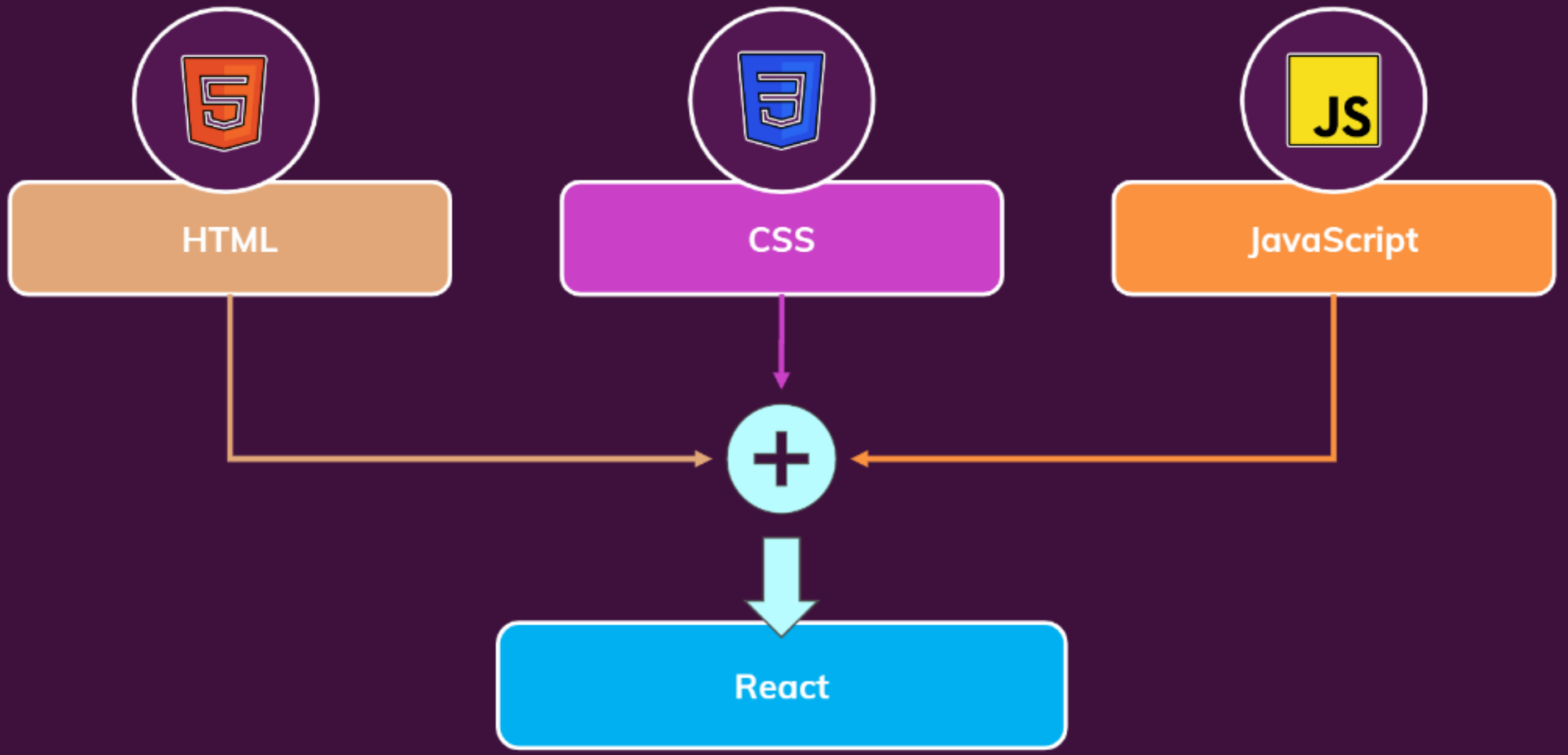


Separation of Concerns

Don’t do too many things in one and the same place (function)

Split big chunks of code into multiple smaller functions

How Is A Component Built?



Build your own, custom HTML Elements

JSX = “HTML in JavaScript”

Understanding JSX

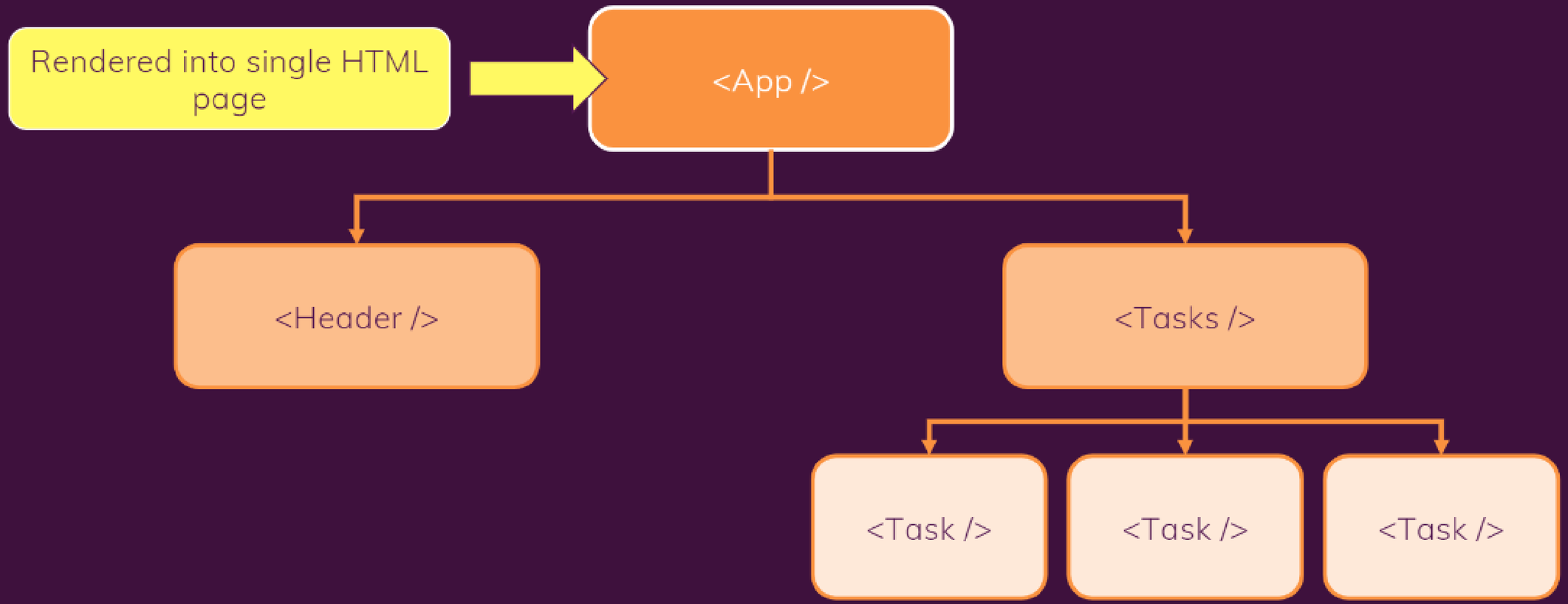
```
<p title="Intro text">
React.js is a library for
building user interfaces.
</p>
```

```
React.createElement(
        'p',
 { title: 'Intro text' },
'React.js is a library for
building user interfaces.'
        );
```

"**Syntactic sugar**", does **not run** in the browser like this!

Real JavaScript code, would run in the browser like this. Not nice to use for more complex than "HTML code".

You Build A Component Tree



UNINASSAU | ser educacional

React

Objetivo

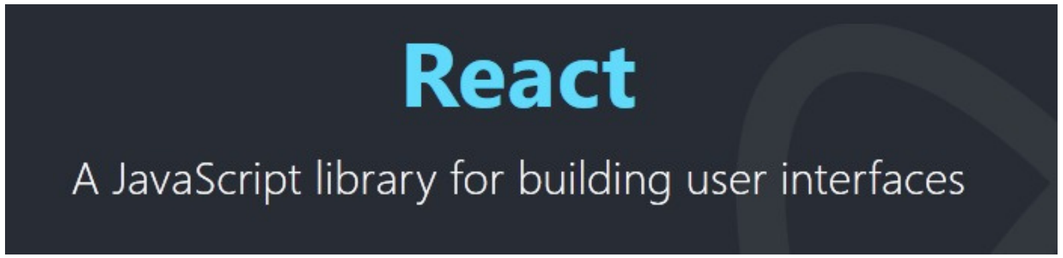
Version 18.0.0
Releasedon March 29, 2022

Aprender um dos frameworks de front-end mais populares

Princípios básicos

Arquitetura de aplicativos Técnicas de programação

Aproveite o conhecimento dos conceitos de JS



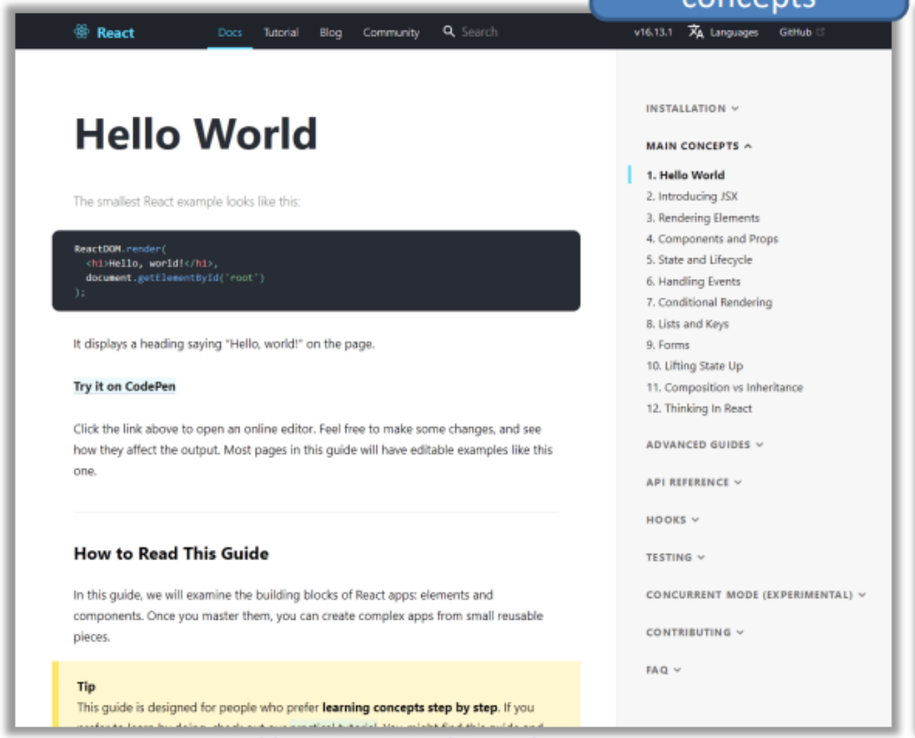
https://reactjs.org/

UNINASSAU | ser educacional

Main Resources

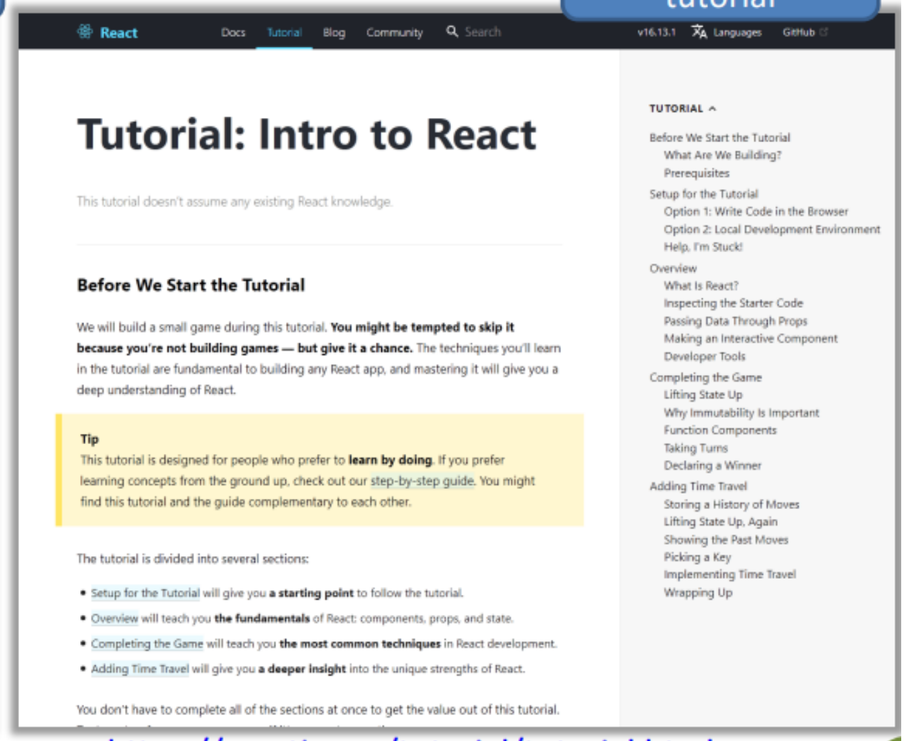
React

Learning the main concepts



https://reactjs.org/docs/hello-world.html

Learn by doing tutorial



https://reactjs.org/tutorial/tutorial.html

https://reactjs.org/docs/hello-world.html

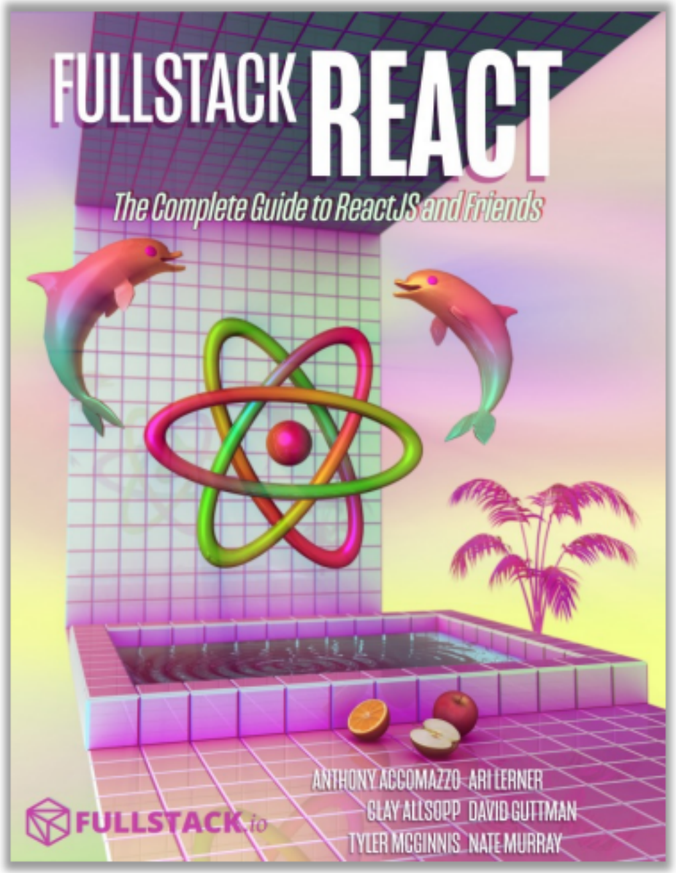
https://codepen.io/naubergois/pen/ExELrBv?editors=1111

UNINASSAU | ser educacional

React

Livros

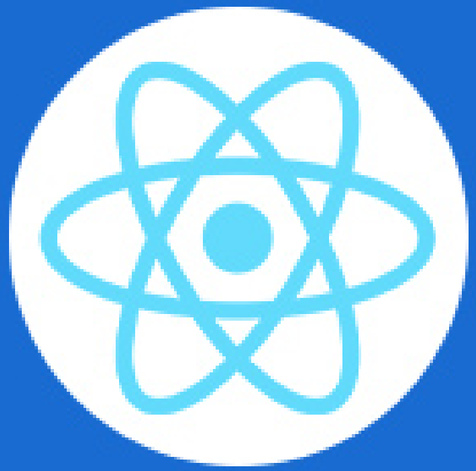
Written Resources



https://www.newline.co/fullstack-react/

https://flaviocopes.com/page/react-handbook/

React Vs JavaScript



React

Vs

JavaScript

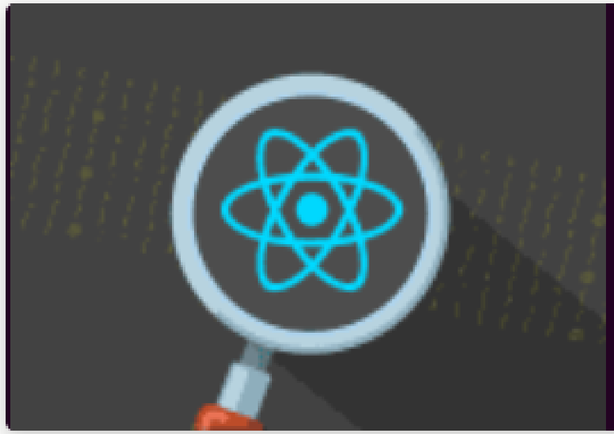
https://github.com/academind/react-complete-guide-code/tree/01-getting-started/code/react-vs-vanilla-js-example

UNINASSAU | ser educacional

Frameworks

Angular - The Complete Guide

Learn Angular from A - Z with this best-selling, 5* rated complete course!



React - The Complete Guide

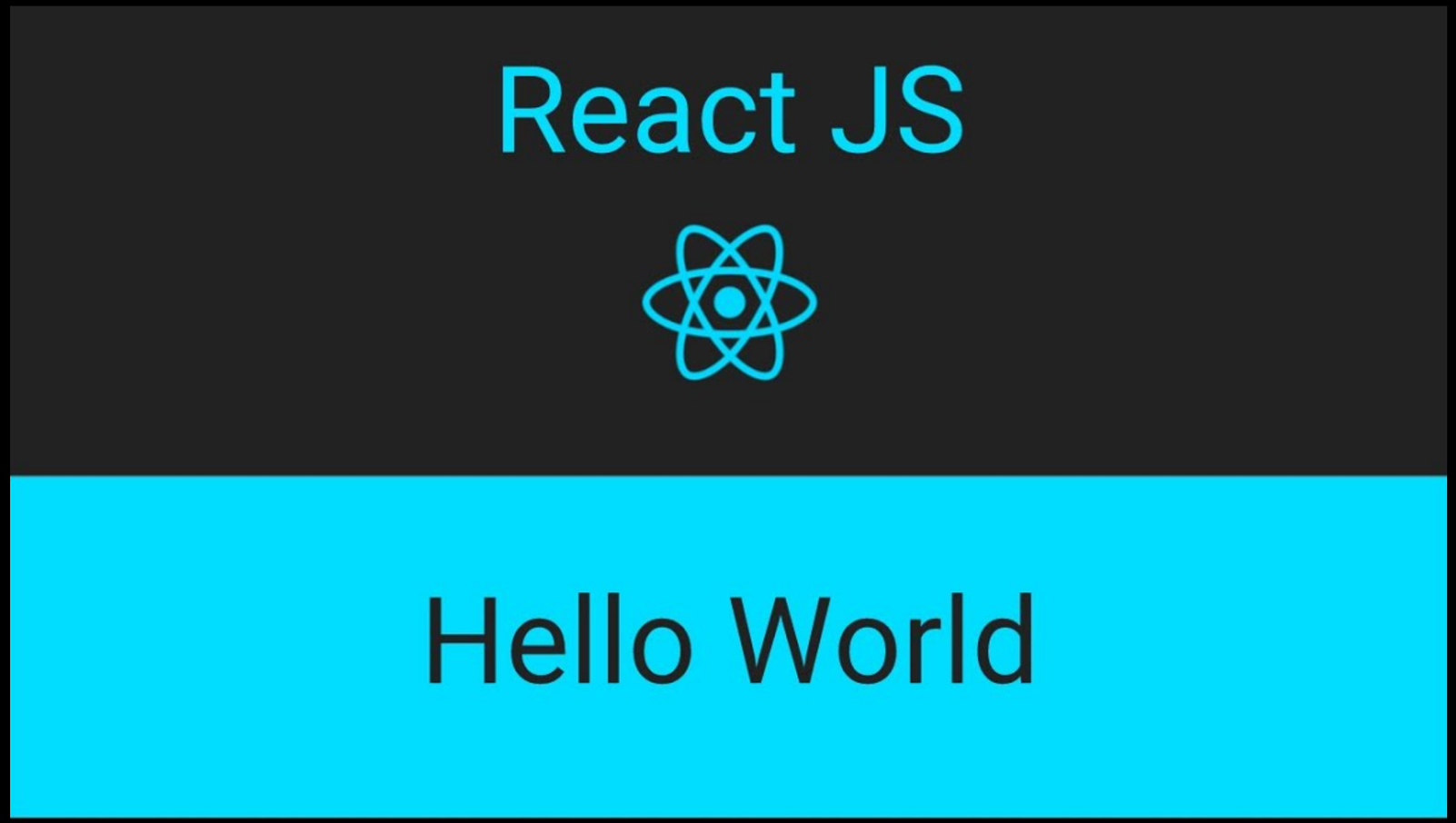
In this course, you'll learn React from scratch. And then even further - including React Hooks, Redux & more!

VueJS - The Complete Guide

Learn Vue.js from from the ground-up and join thousands of other happy students!

UNINASSAU | ser educacional

React



https://playcode.io/936819

UNINASSAU

React


```
function App() {
  return (
    <div>
      <h2>Let's get started!</h2>
    </div>
  );
}

export default App;
```

https://github.com/naubergois/datasets/tree/master/ForReact/01-starting-setup

UNINASSAU

React

Criando Primeiro Componente

```
import ReactDOM from 'react-dom/client';

import './index.css';
import App from './App';

const root = ReactDOM.createRoot(document.getElementById('root'));
root.render(<App />);
```

https://github.com/naubergois/datasets/tree/master/ForReact/02-building-a-first-custom-component

UNINASSAU

React

Criando Primeiro Componente

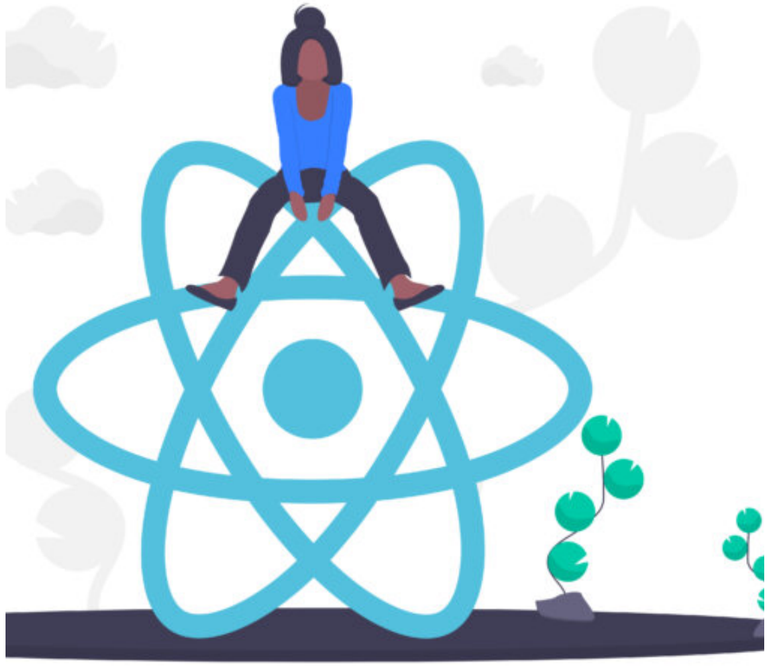
```jsx
import ExpenseItem from './components/ExpenseItem';

function App() {
  return (
    <div>
      <h2>Let's get started!</h2>
      <ExpenseItem></ExpenseItem>
    </div>
  );
}

export default App;
```

```jsx
function ExpenseItem() {
  return <h2>Expense item!</h2>;
}

export default ExpenseItem;
```

UNINASSAU

React

Criando Primeiro Componente



```jsx
function ExpenseItem() {
  return (
    <div>
      <div>March 28th 2021</div>
      <div>
        <h2>Car Insurance</h2>
        <div>$294.67</div>
      </div>
    </div>
  );
}

export default ExpenseItem;
```

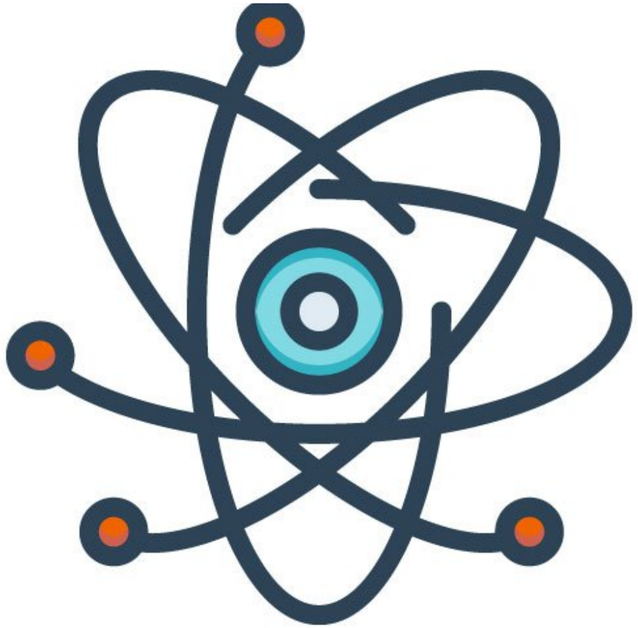
https://github.com/naubergois/datasets/tree/master/ForReact/03-writing-more-complex-jsx-code

UNINASSAU

React

Criando Primeiro Componente

Adicionando CSS



```jsx
import './ExpenseItem.css';

function ExpenseItem() {
  return (
    <div className="expense-item">
      <div>March 28th 2021</div>
      <div className="expense-item__description">
        <h2>Car Insurance</h2>
        <div className="expense-item__price">$294.67</div>
      </div>
    </div>
  );
}

export default ExpenseItem;
```

https://github.com/naubergois/datasets/tree/master/ForReact/04-adding-basic-css-styling

UNINASSAU

React

Criando Primeiro Componente

Incluindo Dados de forma Dinâmica

```
import './ExpenseItem.css';

function ExpenseItem() {
  const expenseDate = new Date(2021, 2, 28);
  const expenseTitle = 'Car Insurance';
  const expenseAmount = 294.67;

  return (
    <div className='expense-item'>
      <div>{expenseDate.toISOString()}</div>
      <div className='expense-item__description'>
        <h2>{expenseTitle}</h2>
        <div className='expense-item__price'>${expenseAmount}</div>
      </div>
    </div>
  );
}

export default ExpenseItem;
```

https://github.com/naubergois/datasets/tree/master/ForReact/05-outputting-dynamic-data

UNINASSAU

React

Criando Primeiro Componente

Usando propriedades

```
import './ExpenseItem.css';

function ExpenseItem(props) {
  return (
    <div className='expense-item'>
      <div>{props.date.toISOString()}</div>
      <div className='expense-item__description'>
        <h2>{props.title}</h2>
        <div className='expense-item__price'>${props.amount}</div>
      </div>
    </div>
  );
}

export default ExpenseItem;
```

https://github.com/naubergois/datasets/blob/master/ForReact/06-passing-data-via-props/src/components/ExpenseItem.js

UNINASSAU

React

Criando Primeiro Componente

Incluindo Javascript

```
import './ExpenseItem.css';

function ExpenseItem(props) {
  const month = props.date.toLocaleString('en-US', { month: 'long' });
  const day = props.date.toLocaleString('en-US', { day: '2-digit' });
  const year = props.date.getFullYear();

  return (
    <div className='expense-item'>
      <div>
        <div>{month}</div>
        <div>{year}</div>
        <div>{day}</div>
      </div>
      <div className='expense-item__description'>
        <h2>{props.title}</h2>
        <div className='expense-item__price'>${props.amount}</div>
      </div>
    </div>
  );
}

export default ExpenseItem;
```

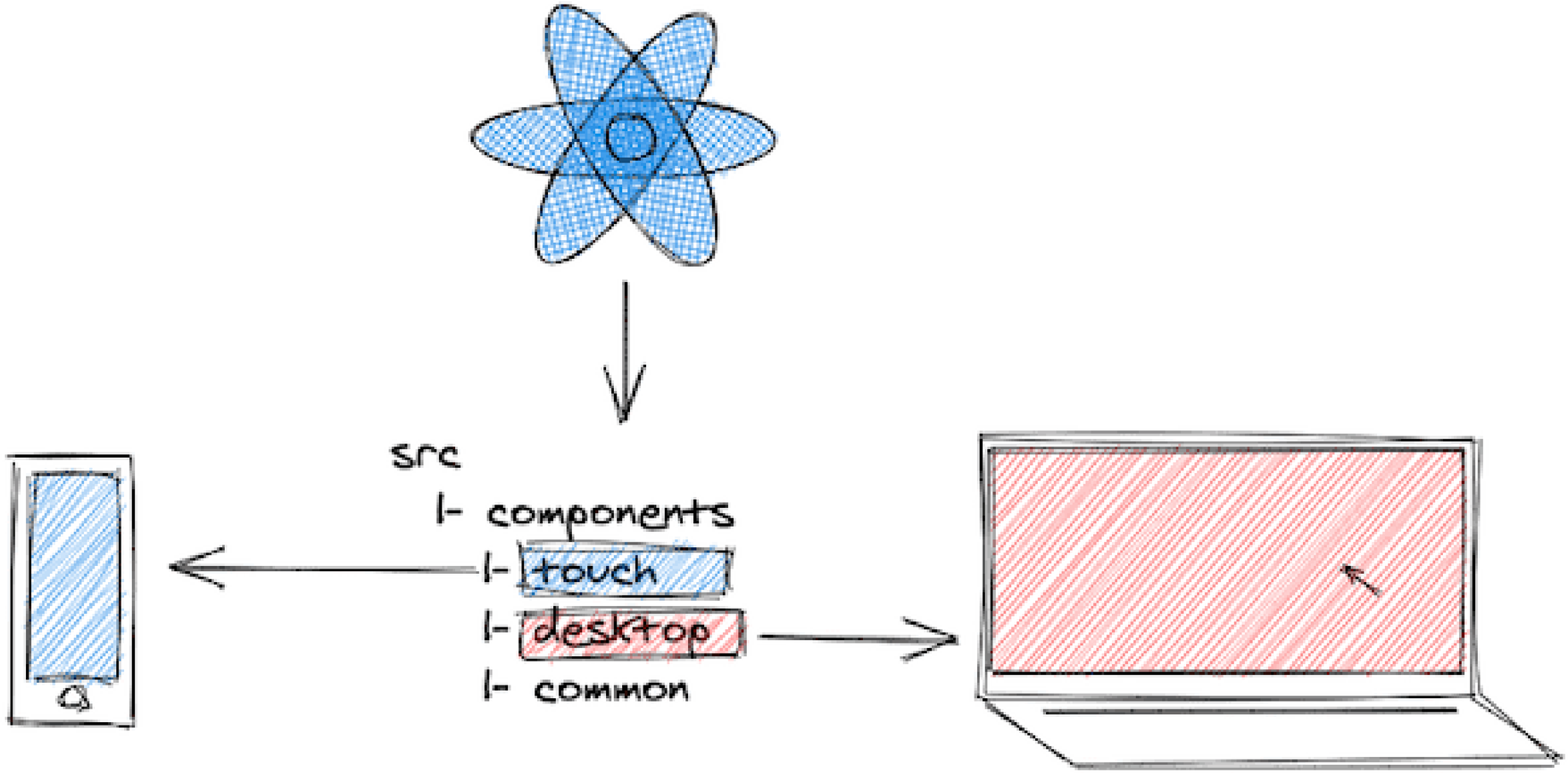
https://github.com/naubergois/datasets/tree/master/ForReact/07-adding-normal-javascript-logic/src

UNINASSAU | ser educacional

React

Criando Primeiro Componente

Dividir em Múltiplos Componentes



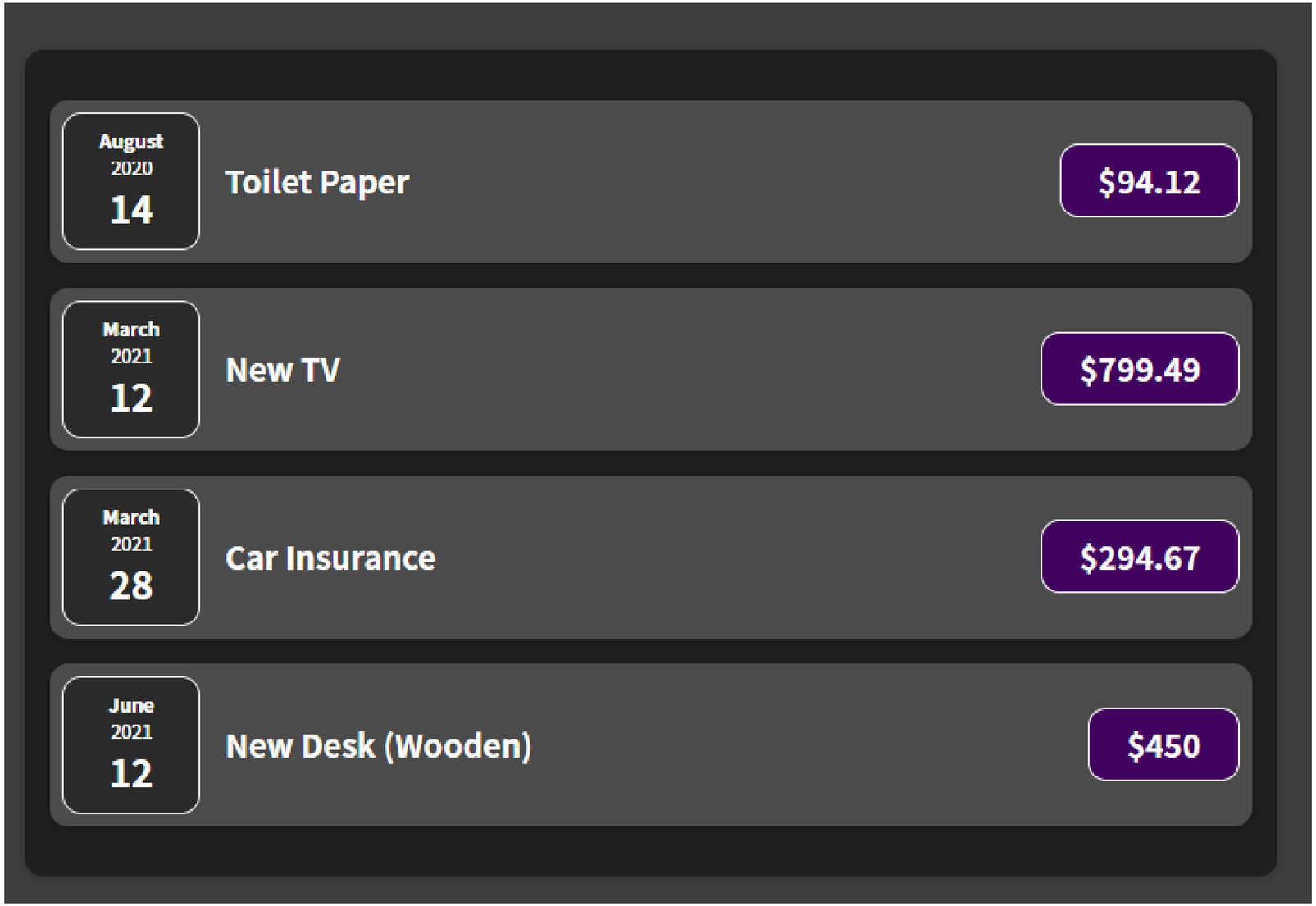
https://github.com/naubergois/datasets/tree/master/ForReact/08-splitting-components-into-multiple-components

UNINASSAU

React

Criando Primeiro Componente

Primeira Solução em React



https://github.com/naubergois/datasets/tree/master/ForReact/09-assignment-1-solution/src

https://github.com/academind/react-complete-guide-code/tree/03-react-basics-working-with-components/code/11-finished

UNINASSAU

React

Criando Primeiro Componente

Composição de Componentes

```
import ExpenseDate from './ExpenseDate';
import Card from './Card';
import './ExpenseItem.css';

function ExpenseItem(props) {
  return (
    <Card className='expense-item'>
      <ExpenseDate date={props.date} />
      <div className='expense-item__description'>
        <h2>{props.title}</h2>
        <div className='expense-item__price'>${props.amount}</div>
      </div>
    </Card>
  );
}

export default ExpenseItem;
```

UNINASSAU

React

Criando Primeiro Componente

Composição de Componentes

Alex Lobera

Jan 30, 2019 · 12 min read · Listen

React is all about composition

Software development is, in essence, the process of breaking a problem down into smaller problems, implementing solutions for those smaller problems, and then **composing** tho
eventually completin

What do React comp
independent compor
to create greater com
whole UI.

O desenvolvimento de software é, em essência, o processo de dividir um problema em problemas menores, implementar soluções para esses problemas menores e, em seguida, compor essas soluções para formar soluções parciais, e assim por diante, até completar a solução inteira.

https://medium.com/@alex_lobera/react-is-all-about-composition-f9f49dec183c

UNINASSAU

React

Criando Primeiro Componente

Composição de Componentes

Function composition

Function composition is the act of applying a function to the output of another function. In algebra, given two functions, f and g, $(f \circ g)(x) = f(g(x))$. The circle is the composition operator and it's commonly pronounced "f composed with g" or "f after g". I pronounce it "composed with", maybe because in React I often see code like the following where I literally read "compose with":

```
export default compose(
   withRouter, withApollo, withEtcEtc
)(Component)
```

You can compose functions
input of the next function. Y
array and the next one's inp
the following image:

Composição de função é o ato de aplicar uma função à saída de outra função. Em álgebra, dadas duas funções, f e g, (f ∘ g)(x) = f(g(x)).

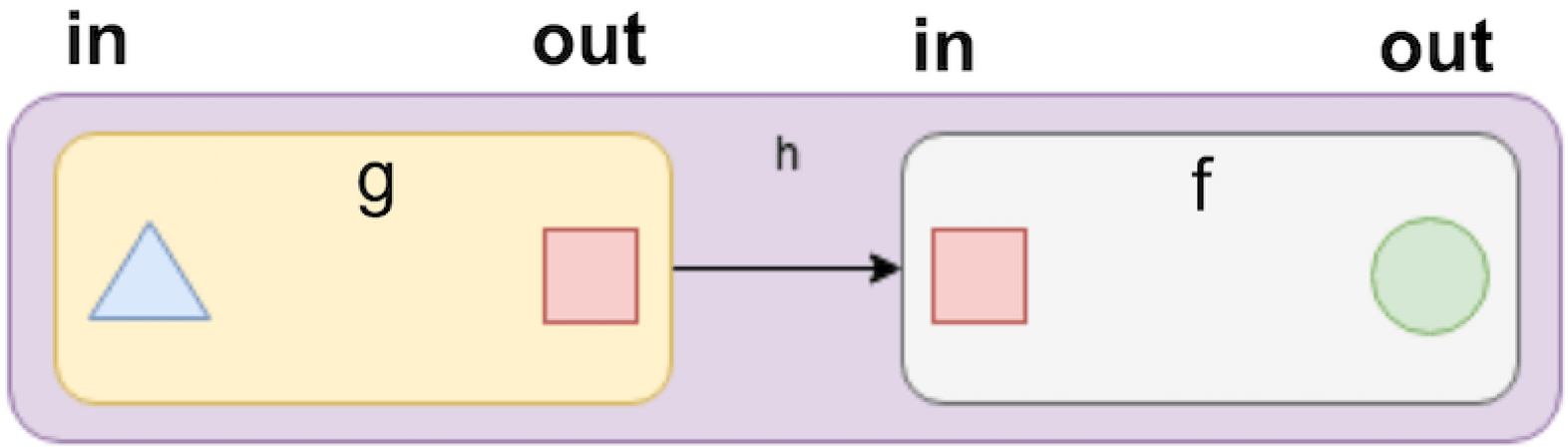
UNINASSAU

ser educacional

React

Criando Primeiro Componente

Composição de Componentes



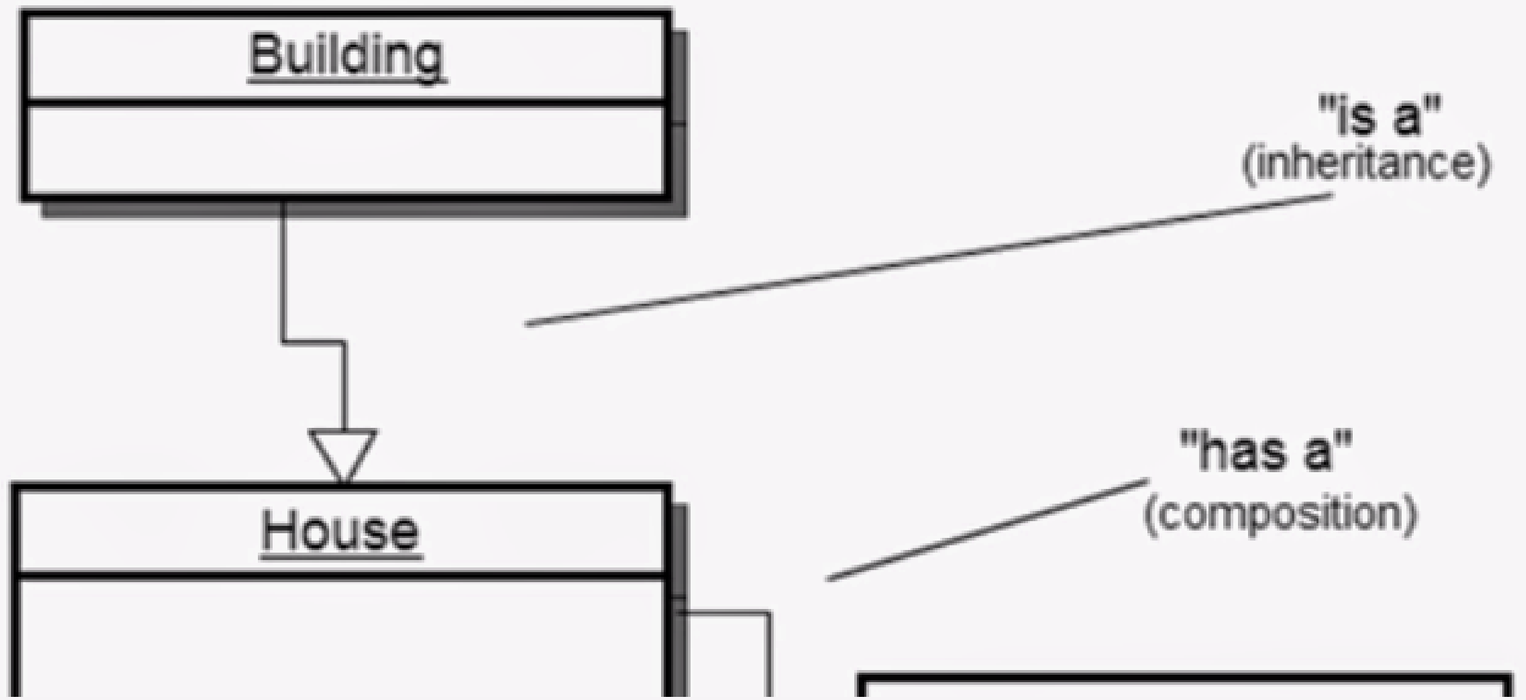
UNINASSAU

React

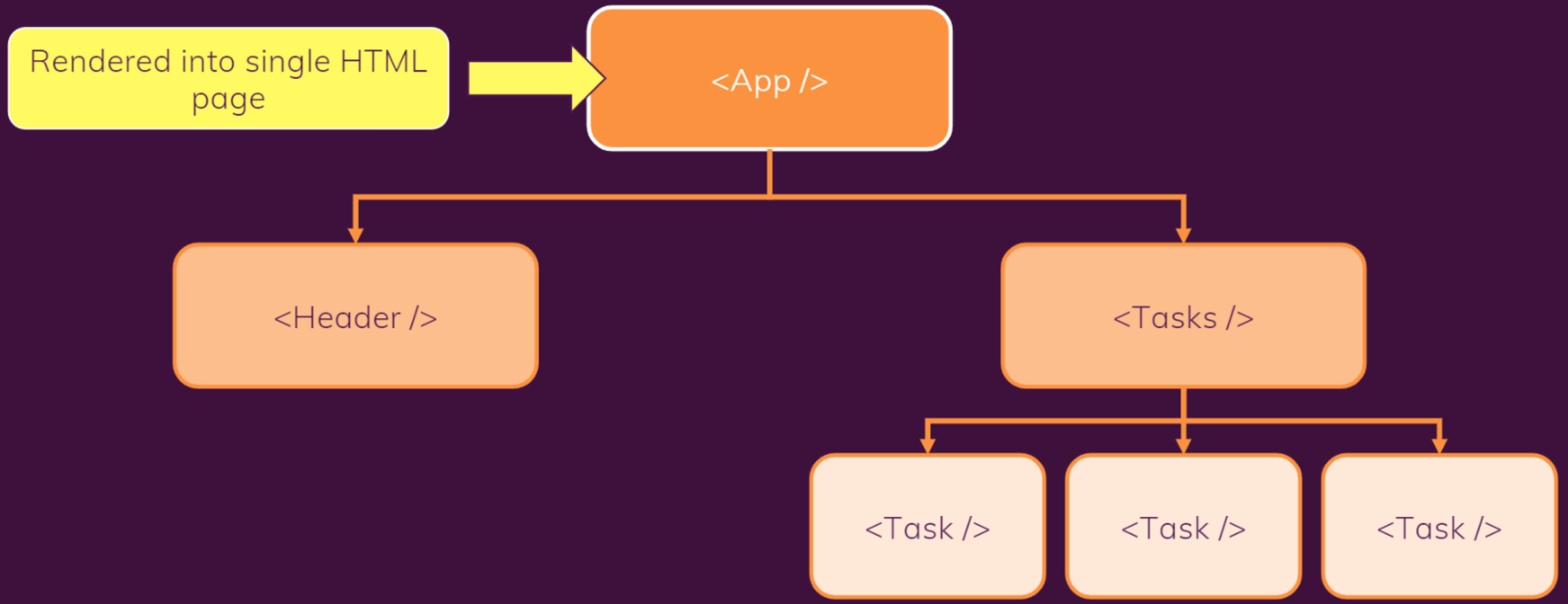
Criando Primeiro Componente

Composição de Componentes

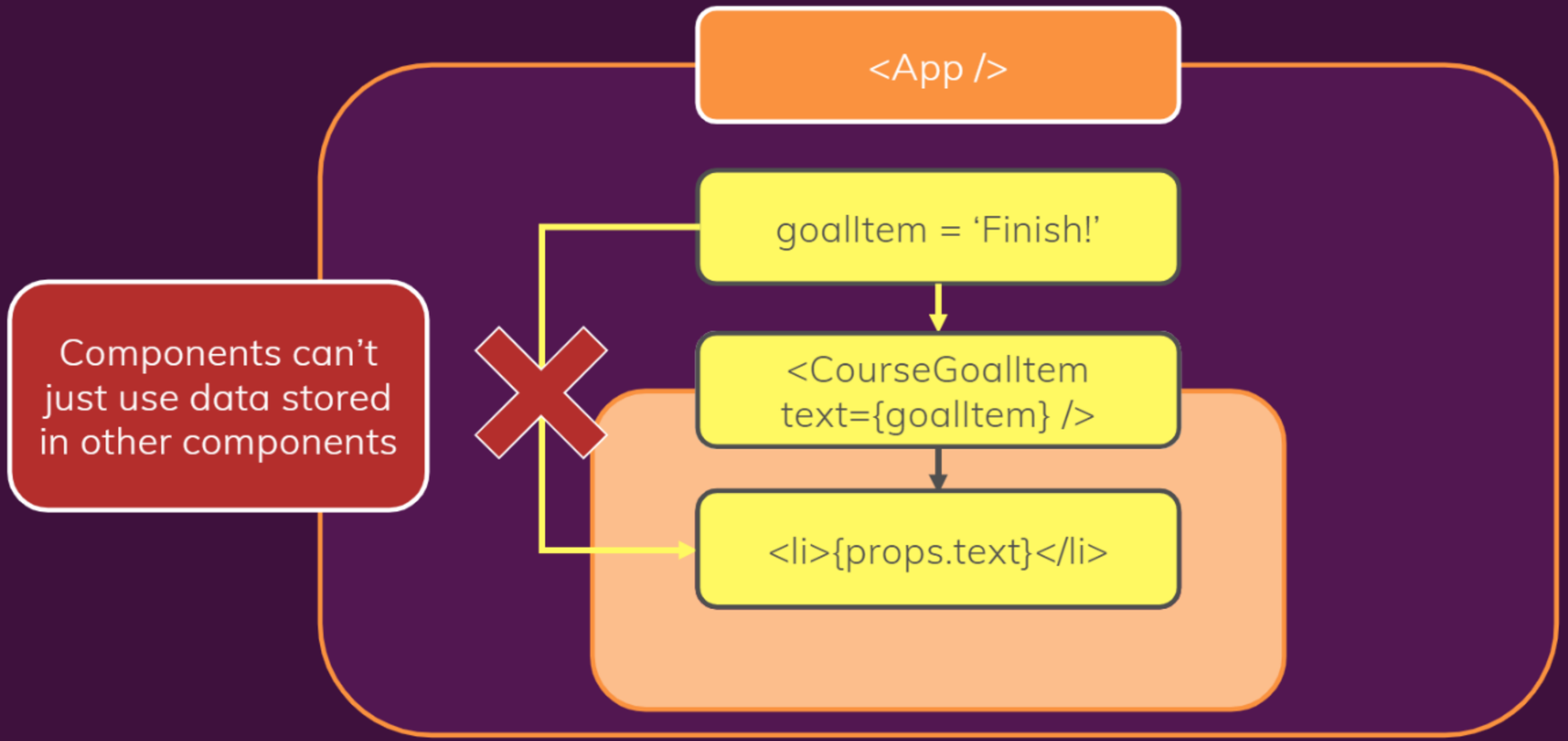
Composition vs. Inheritance



https://codepen.io/discountry/pen/aJPypv


Rendered into single HTML page
<App />
<Header />
<Tasks />
<Task />
<Task />
<Task />

Passing Data via “Props”



UNINASSAU

ser educacional

React

Imutabilidade

Imutabilidade

- Reacts explora a imutabilidade de objetos, para facilitar a programação e eficiência de processamento
- Os 'props' do componente são imutáveis (somente leitura pelo componente)
- O 'estado' do componente não é diretamente mutável (pode ser alterado apenas através de chamadas especiais)
- As funções são 'puras' (não têm efeitos colaterais além de calcular o retorno valor)

– Idempotência (re-renderizar o mesmo componente sempre produz o mesmo resultado)

– Previsibilidade

UNINASSAU

React

Re-renderização

Re-renderização

- A aplicação é feita de Componentes
- Todo o aplicativo é renderizado novamente:
  - Toda vez que um estado é alterado
  - Toda vez que uma propriedade é alterada
- Cada Componente se reconstruirá do zero
  - Com pequenas variações, ou radicalmente diferente
- Performance?

UNINASSAU

ser educacional

React

Re-renderização

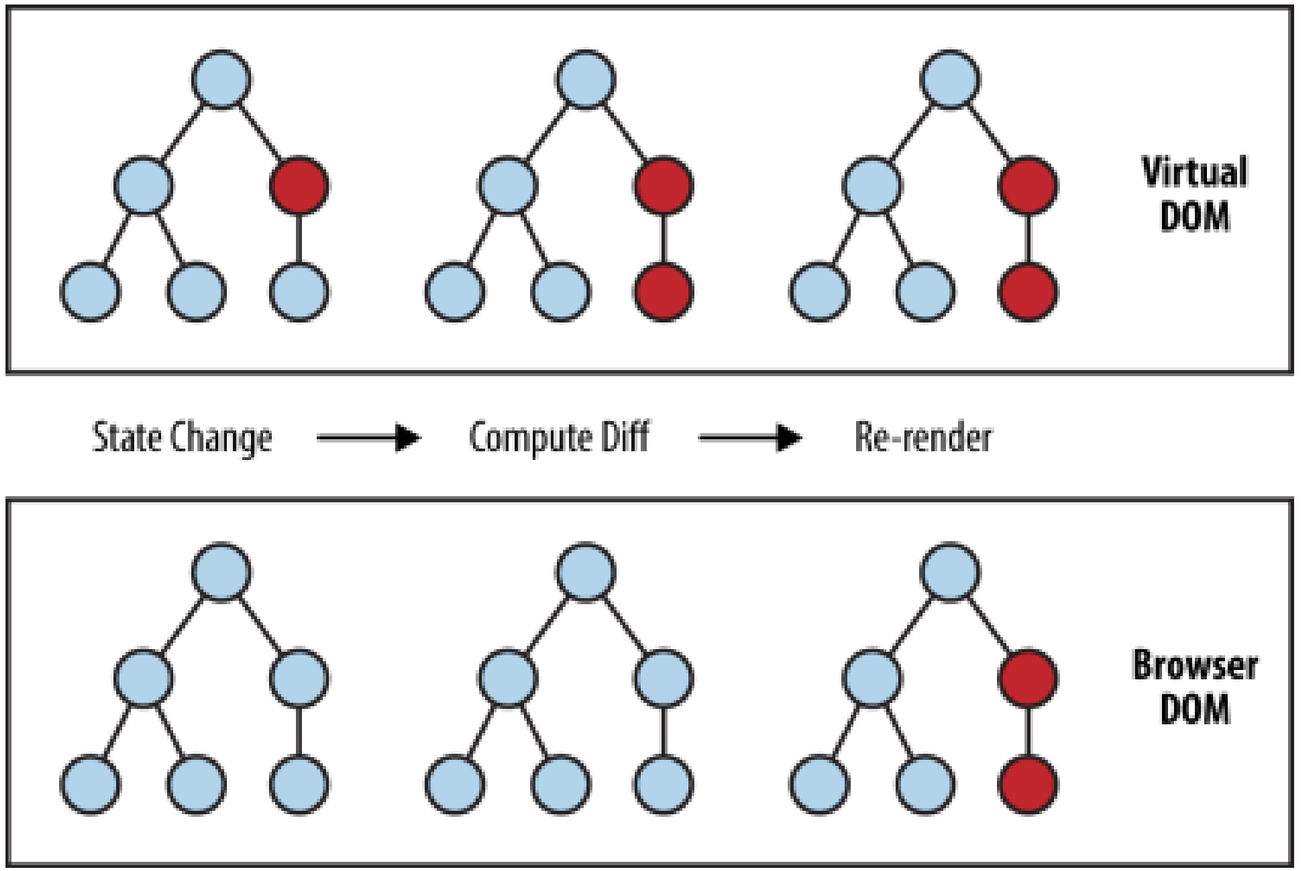
Desempenho de Rerenderização

- **Modificações no DOM são caras (recomputação de layout e atualizando GUI)**
- **React implementa uma camada Virtual DOM**
  - **Estrutura de dados interna na memória, otimizada e muito rápida de atualizar**
  - **Corrige algumas anomalias e assimetrias do DOM**
  - **Gerencia seu próprio conjunto de eventos “sintéticos”**
  - **Após os componentes renderizarem novamente, o React calcula a diferença entre o “antigo” DOM e o novo DOM Virtual modificado**
  - **Somente modificações e diferenças são aplicadas seletivamente ao DOM do navegador, em lote**

UNINASSAU

React

Re-renderização



https://www.oreilly.com/library/view/learning-react-native/9781491929049/ch02.html

UNINASSAU

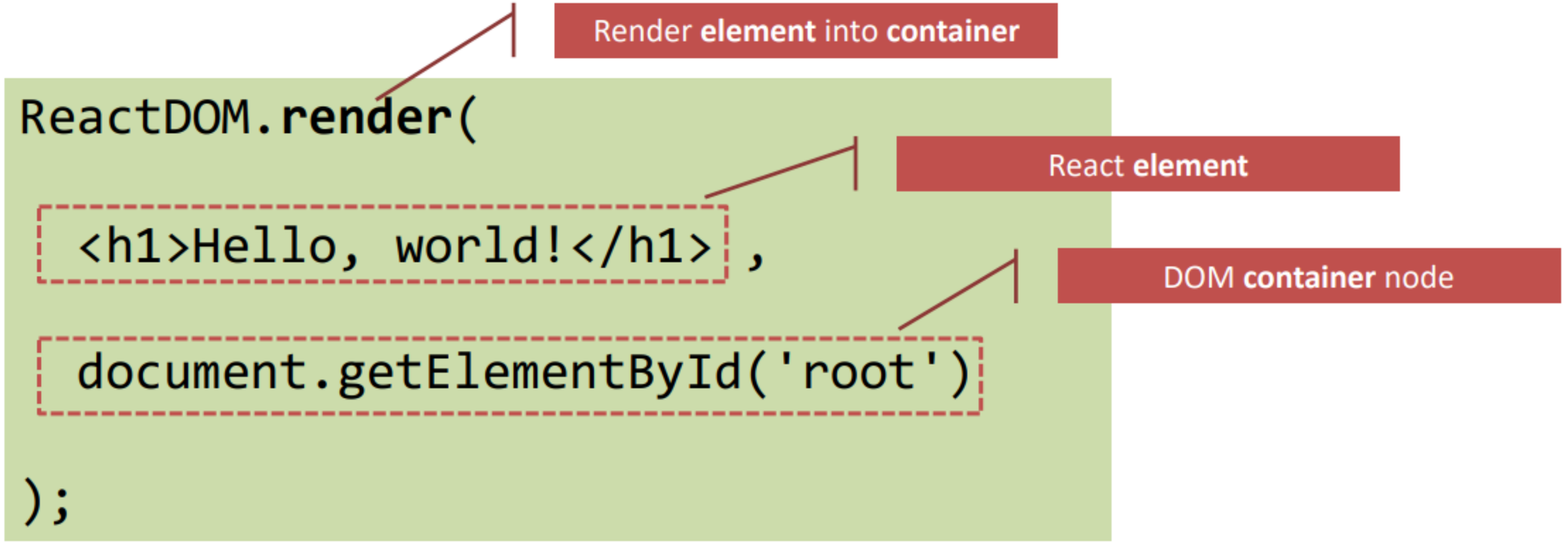
ser educacional

React

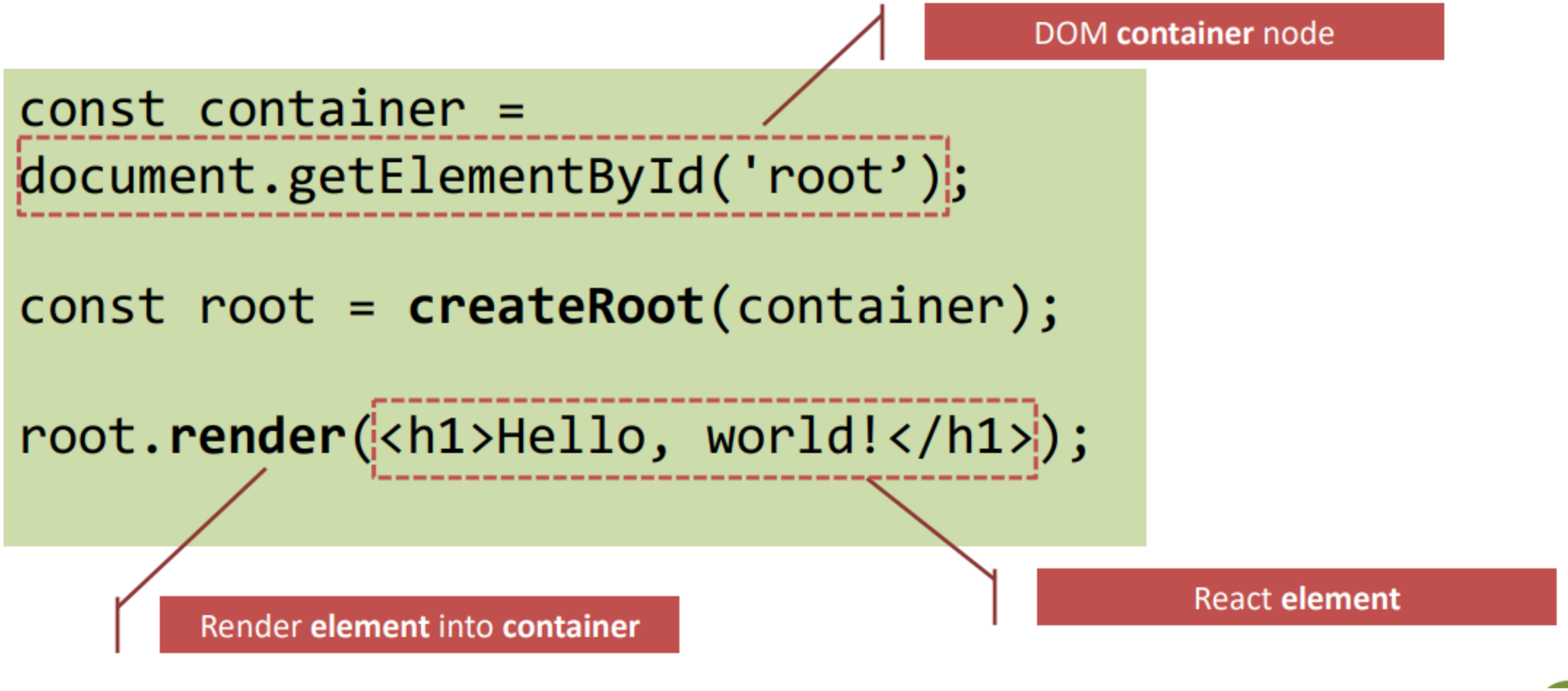
Re-renderização

Ciclo de atualização

- Crie uma nova árvore DOM Virtual
- Diferenciar com o antigo
- Calcular o conjunto mínimo de alterações
- Coloque-os em uma fila
- Renderização em lote de todas as alterações no navegador


Render element into container
ReactDOM.render(
<h1>Hello, world!</h1>,
React element
document.getElementById('root')
DOM container node
);

How React Code Looks Like (Since Version 18)



JSX Syntax

```
const container =
document.getElementById('myapp');
const root = createRoot(container);

root.render(
```

JSX Syntax

```
  <div id="test">
    <h1>A title</h1>
    <p>A paragraph</p>
  </div>
```

```
);
```

Equivalent

Transpiling
(Babel)

```
const container =
document.getElementById('myapp');
const root = createRoot(container);

root.render(
```

JS calls to React.createElement

```
  React.DOM.div(
    { id: 'test' },
    React.DOM.h1(null, 'A title'),
    React.DOM.p(null, 'A paragraph')
  )
```

```
);
```

UNINASSAU

ser educacional

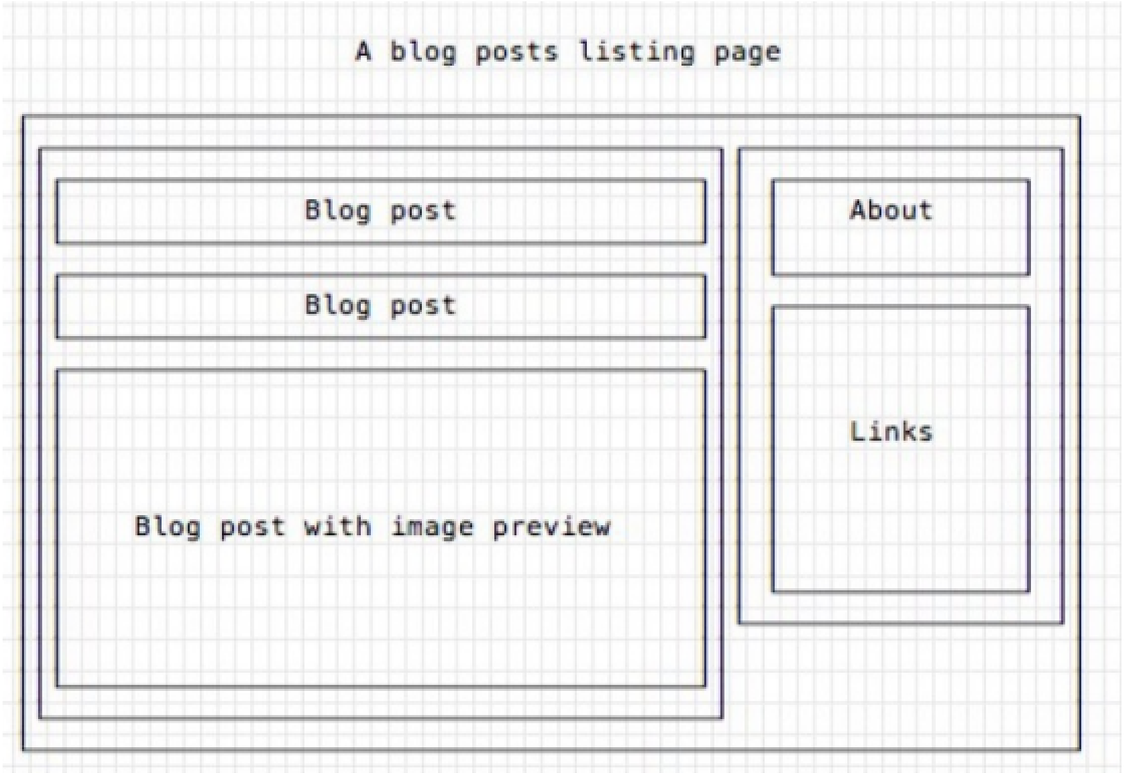
React

Componentes

Tudo em uma página é um Componente

– Mesmo simples tags HTML (React.DOM.element)

- **Os componentes podem ser aninhados**
- **ReactDOM.createRoot().render() constrói um componente e anexa para um contêiner DOM**



As a function, returning DOM elements

```
const BlogPostExcerpt = () => {
  return (
    <div>
      <h1>Title</h1>
      <p>Description</p>
    </div>
  )
}
```

As a class, with a render() method

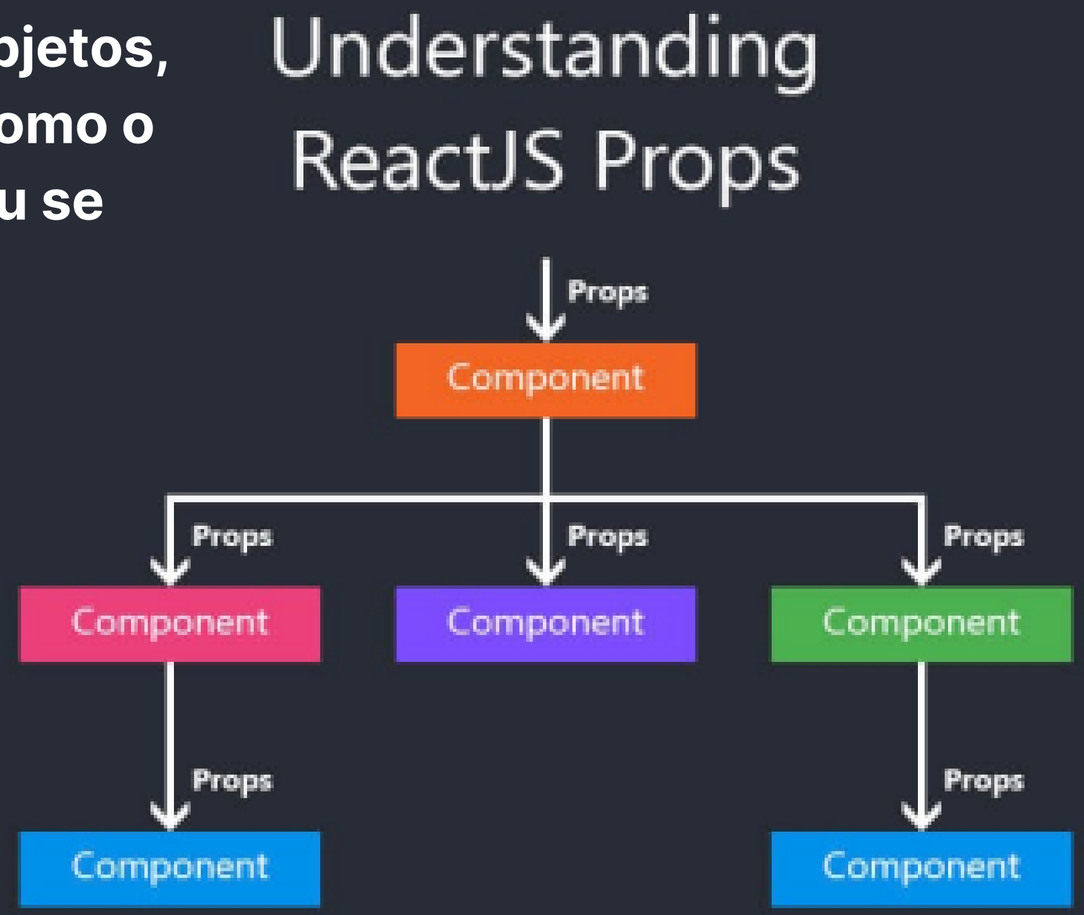
```
import React, { Component } from 'react'

class BlogPostExcerpt extends Component {
  render() {
    return (
      <div>
        <h1>Title</h1>
        <p>Description</p>
      </div>
    )
  }
}
```

UNINASSAU

React

Propriedades

Props (propriedades) são passados para um componente
por seu pai
– Valores (strings, objetos, ...) para configurar como o componente exibe ou se comporta



UNINASSAU

ser educacional

React

State

Estado é um conjunto de variáveis locais ao componente

– Inicializado com valor padrão ou por valores de props

– Pode ser modificado apenas chamando métodos específicos

• Assíncrono

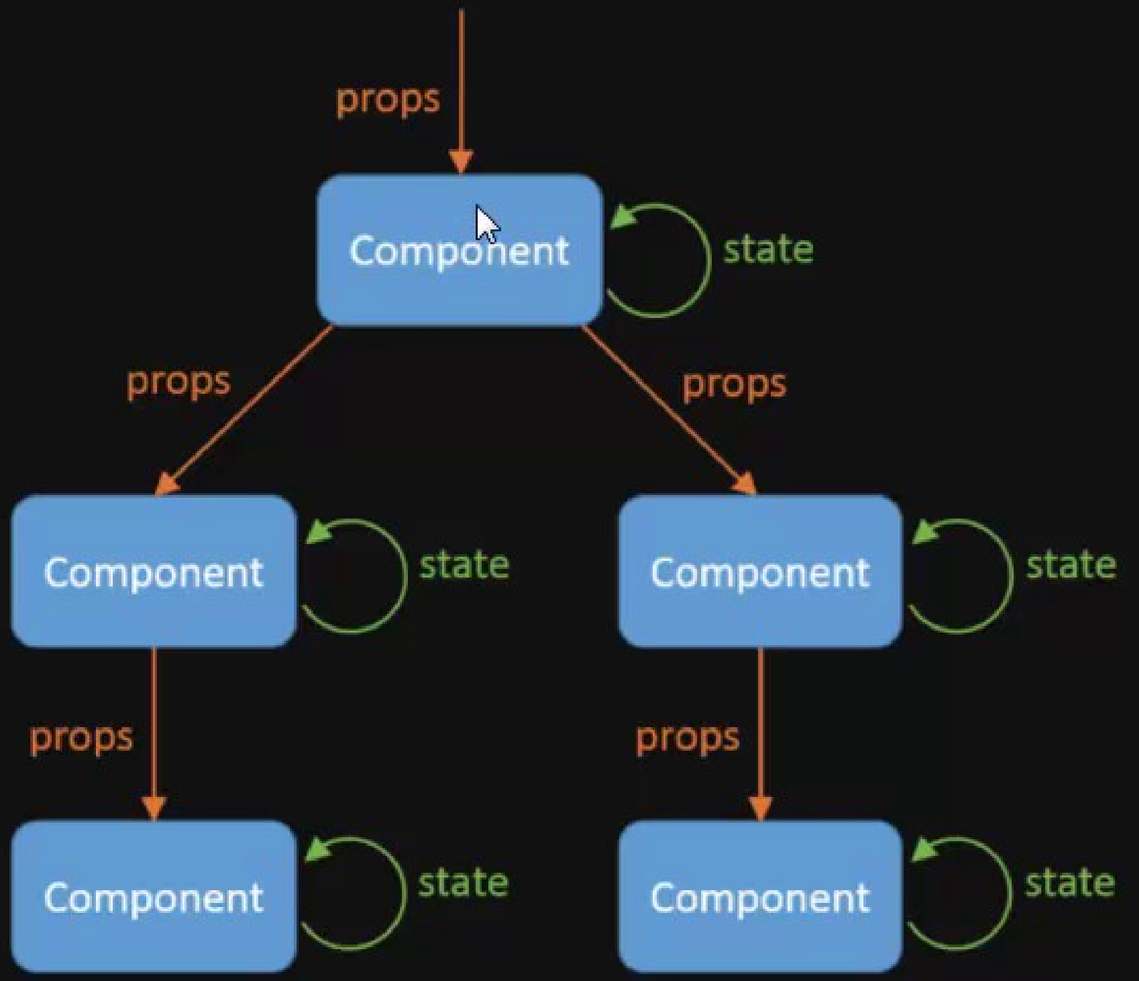
• Iniciará a re-renderização do DOM Virtual

– O valor do estado atual pode ser passado para os filhos (como adereços)

UNINASSAU

React

Propriedades e Estados



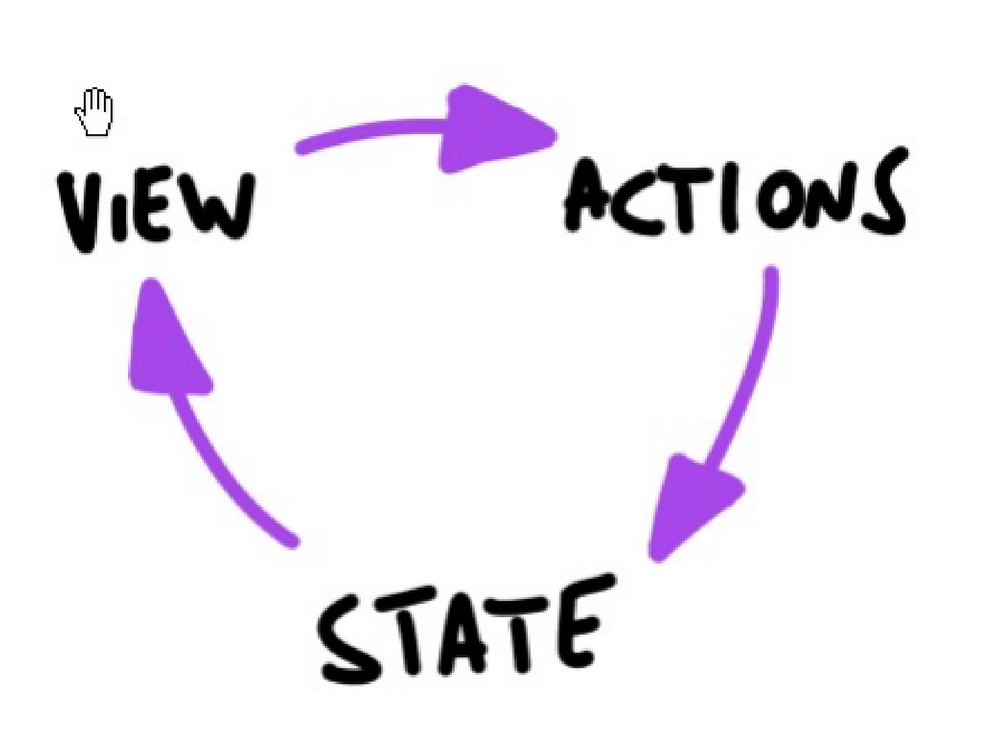
UNINASSAU

ser educacional

React

Propriedades e Estados

- O estado é passado para a visualização e para componentes filho
- As ações são acionadas pela visualização
- As ações podem atualizar o estado
- A mudança de estado é passada para o visualizar e para o componente filho



UNINASSAU

React

Criando aplicação React do Zero

Starting With All The Needed Infrastructure

1. `npx create-react-app my-app`
2. ⌛ *... 270 Megabytes later ...* ⌛
3. `cd my-app`
4. `npm start`
5. Visit http://localhost:3000

`npx` downloads the npm module and runs it immediately.

It will save the downloaded package in a local cache, but outside the current project.

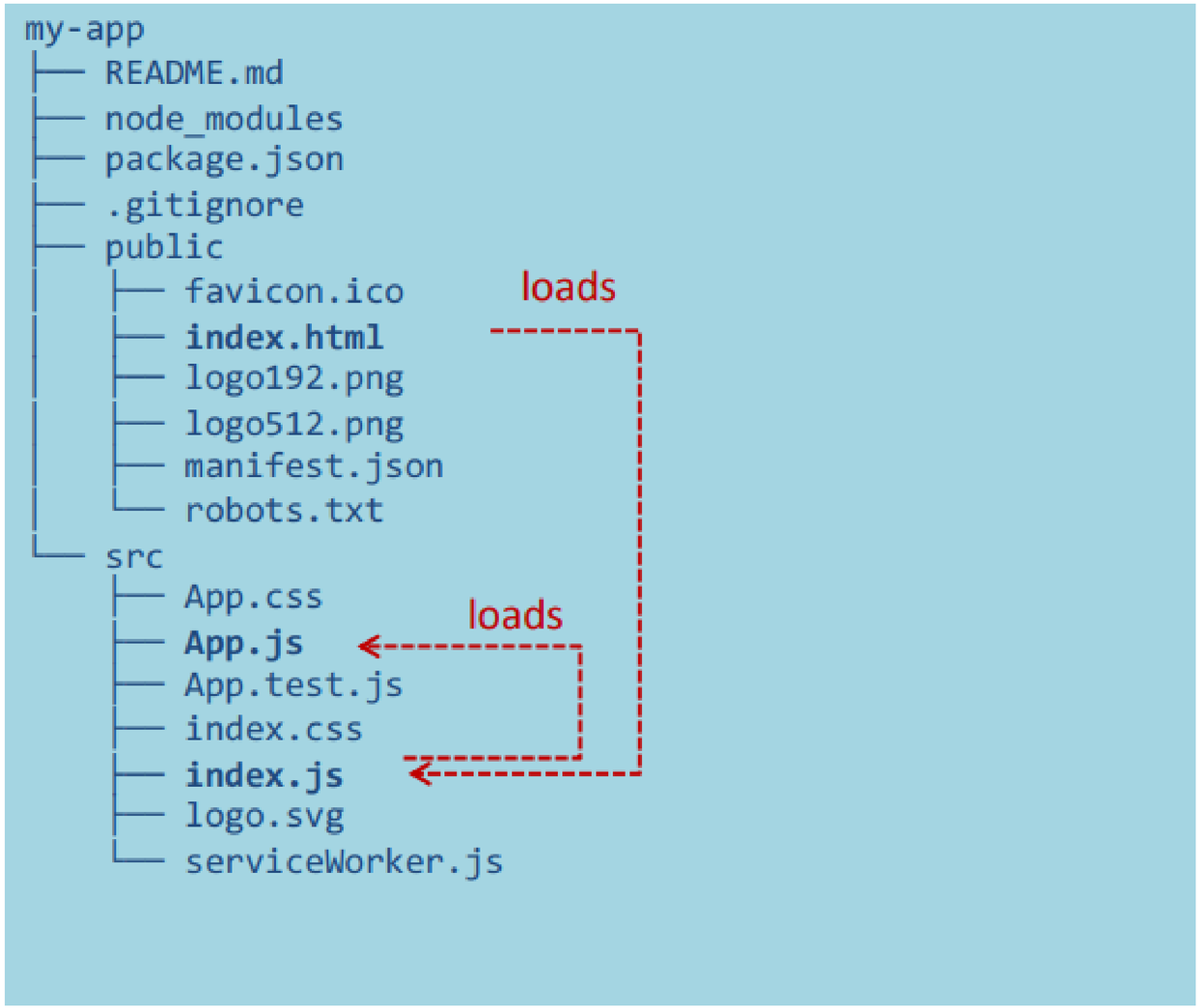
https://create-react-app.dev/

UNINASSAU

React

Estrutura de Pastas

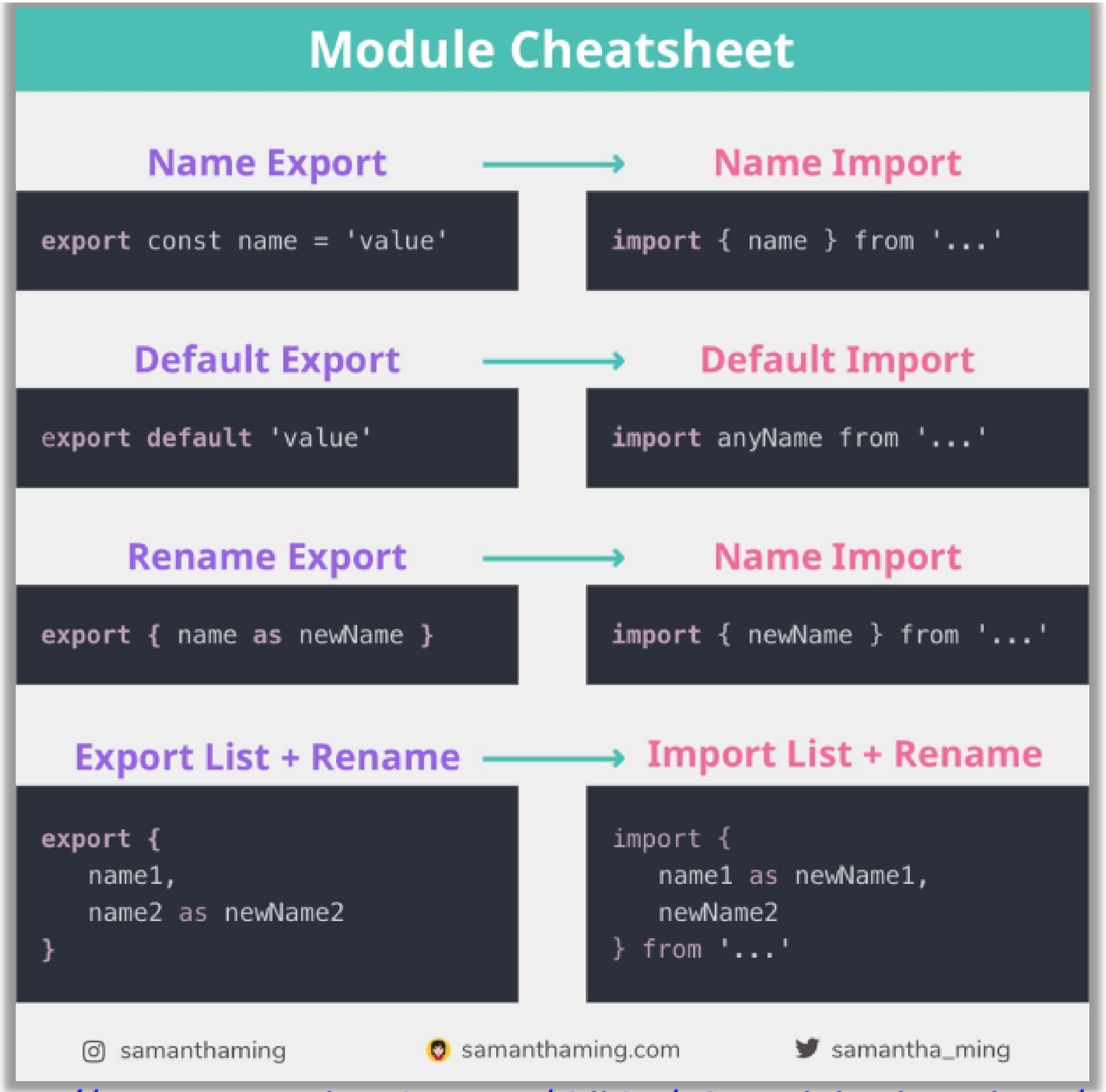
Folder Structure



UNINASSAU | ser educacional

React

Importando e exportando Módulos



https://www.samanthaming.com/tidbits/79-module-cheatsheet/

UNINASSAU

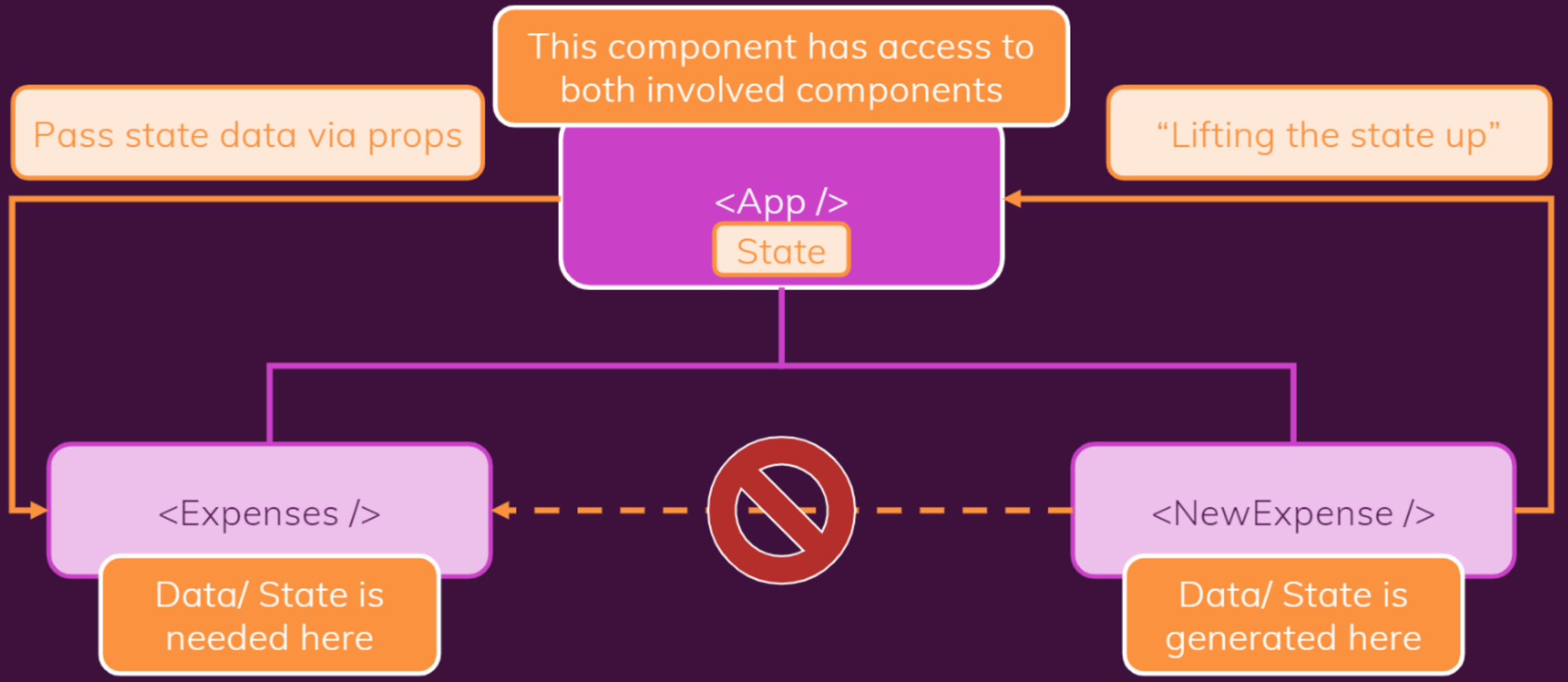
ser educacional

React

Componentes

App.js

```
function Button(props) {
  if (props.lang === 'it')
    return <button>Ciao!</button>;
  else
    return <button>Hello!</button>;
}

function App() {
  return (
    <p>
      Premi qui: <Button lang='it' />
    </p>
  );
}

export default App;
```


This component has access to both involved components
Pass state data via props
“Lifting the state up”
<App />
State
<Expenses />
<NewExpense />
Data/ State is needed here
Data/ State is generated here

UNINASSAU

ser educacional

React

Trabalhando com Estados

```
import ExpenseDate from './ExpenseDate';
import Card from '../UI/Card';
import './ExpenseItem.css';

const ExpenseItem = (props) => {
  // function clickHandler() {}
  const [title, setTitle] = useState(props.title);
  console.log('ExpenseItem evaluated by React');

  const clickHandler = () => {
    setTitle('Updated!');
    console.log(title);
  };
```

https://github.com/naubergois/datasets/tree/master/ForReact/01-working-with-state

UNINASSAU

ser educacional

React

useState

React Hooks

useState

UNINASSAU

React

useState

https://codesandbox.io/s/usestate-yrsbi?file=/src/index.js

UNINASSAU | ser educacional

React

useState

https://codesandbox.io/s/usestate-hskqs

UNINASSAU

React

Formulários

```
return (
  <div>
    <NewExpense />
    <Expenses items={expenses} />
  </div>
);
```

```
import React from 'react';

import ExpenseForm from './ExpenseForm';
import './NewExpense.css';

const NewExpense = () => {
  return (
    <div className='new-expense'>
      <ExpenseForm />
    </div>
  );
};

export default NewExpense;
```

https://github.com/naubergois/datasets/tree/master/ForReact/02-adding-form-inputs/src

UNINASSAU

React

Formulários e Estado

https://codesandbox.io/s/react-form-basic-form-with-inputfield-wrybd

UNINASSAU

React

Multiplos Estados

```
import React, { useState } from 'react';

import './ExpenseForm.css';

const ExpenseForm = () => {
  const [enteredTitle, setEnteredTitle] = useState('');
  const [enteredAmount, setEnteredAmount] = useState('');
  const [enteredDate, setEnteredDate] = useState('');
```

https://github.com/naubergois/datasets/tree/master/ForReact/03-working-with-multiple-states

UNINASSAU

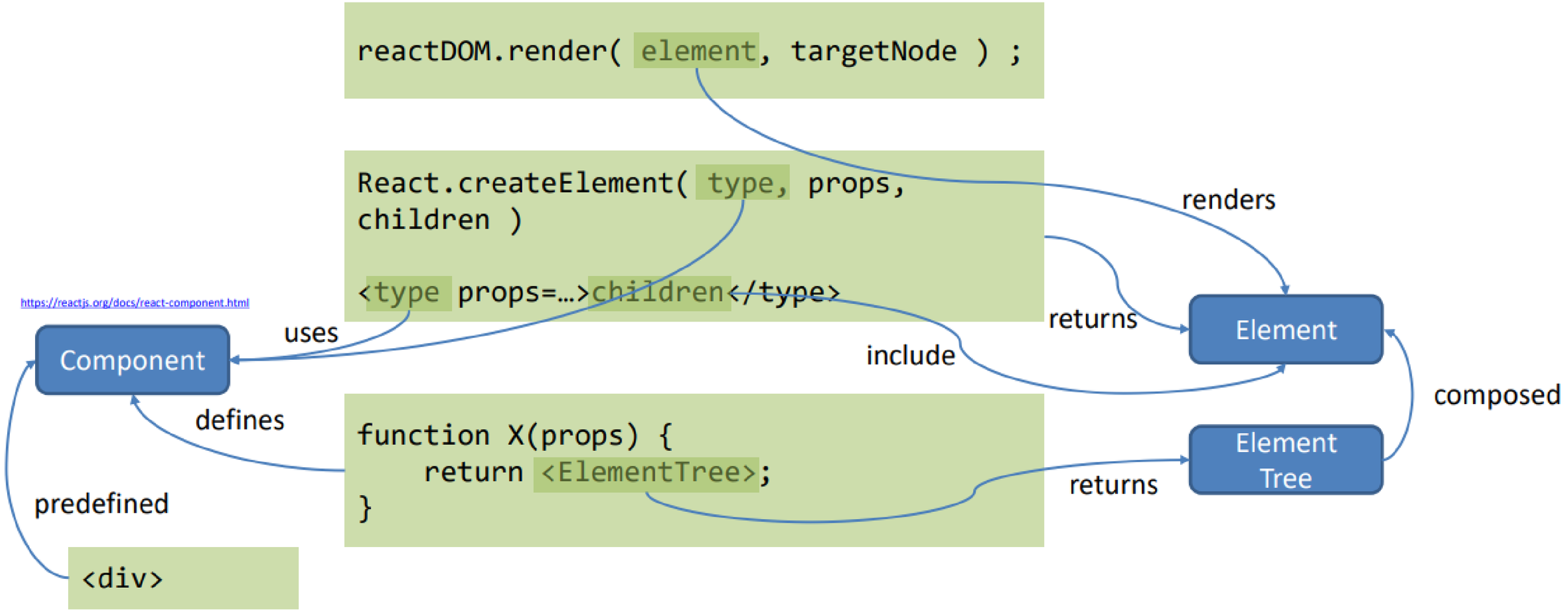
ser educacional

React

Elementos

Um elemento é um objeto simples que descreve uma instância de componente ou DOM
nó e suas propriedades desejadas

- Um ReactElement é uma representação de um elemento DOM no Virtual DOM.
- Contém apenas informações sobre
  – o tipo de componente (por exemplo, um Button)
  – suas propriedades (por exemplo, sua cor)
  – quaisquer elementos filhos dentro dele.
- Não é uma instância de uma parte de uma página, mas uma descrição sobre como
  construí-lo.


reactDOM.render(element, targetNode) ;
React.createElement(type, props, children)
<type props=…>children</type>
renders
returns
Element
https://reactjs.org/docs/react-component.html
uses
Component
include
composed
defines
function X(props) {
return <ElementTree>;
}
Element Tree
returns
predefined

UNINASSAU | ser educacional

React

Elementos

React.createElement (1/3)

- `React.createElement( type, props, children )`

- Type
  - String: a DOM node identified by the tag name (e.g., `'div'`)
  - React component class/function: a user-defined component

UNINASSAU

ser educacional

React

Elementos

https://codesandbox.io/s/ov66pz79zy

UNINASSAU

React

Elementos

https://codesandbox.io/s/react-elements-167t5

UNINASSAU | ser educacional

React

JSX

JSX – JavaScript Syntax Extension

- Alternative syntax for React.createElement
- XML fragments inside the JS code
  - Syntax details: all tags must be </closed> or <selfclosing/>
- Transpiled by Babel into plain JS

```
<MyButton color="blue" shadowSize={2}>
  Click Me
</MyButton>
```

Element/Component name
Props
Children / Text content

```
React.createElement(
  MyButton,
  {color: 'blue', shadowSize: 2},
  'Click Me'
) ;
```

UNINASSAU | ser educacional

React

JSX

Components are expanded during *rendering*

```
<Button color='blue'>
OK!
</Button>
```

render

```
<button class='button button-blue'>
  <b>
    OK!
  </b>
</button>
```

- Components encapsulate element trees (generated given their properties).
- React asks the Button component to render itself. It will generate a tree of elements, to replace this one.
- Repeat until only DOM nodes are present.

UNINASSAU | ser educacional

React

JSX

JSX Syntax

- May use <tag>…</tag> or <tag/> anywhere a JS expression is syntactically valid
  - Not only in Components
  - May also store in Arrays/Objects
  - After all, they are just ReactElements generated by React.createElement!
- May enclose in (…) for clarity

```
const element = <div className="main">Hello world</div>;
```

Note: use <tag/> if the component doesn't have any children

```
const element2 =  (<Message text="Hello world" />);
```

UNINASSAU | ser educacional

React

JSX

JSX Tag Name

- <Foo> is just React.createElement(Foo,...)
  - Foo must be in scope (imported or declared)

```
import CustomButton from './CustomButton';

function WarningButton() {
  return <CustomButton color="red" />;
}
```

UNINASSAU | ser educacional

React

JSX

https://codesandbox.io/s/react-jsx-r89jp?file=/src/Weather.js

UNINASSAU

React

JSX

```
<MyComponent>Hello
world!</MyComponent>
```

```
<MyContainer>
  <MyFirstComponent />
  <MySecondComponent />
</MyContainer>
```

UNINASSAU | ser educacional

React

JSX

Render Children Components

- In the component, you may render `{props.children}` to include the nested elements

```
return (
  <Container>
    <Article headline="An interesting
Article">
      Content Here
    </Article>
  </Container>
)
```

```
function Container (props) {
    return (<div className="container">
      {props.children}
    </div>);
}
```

UNINASSAU

React

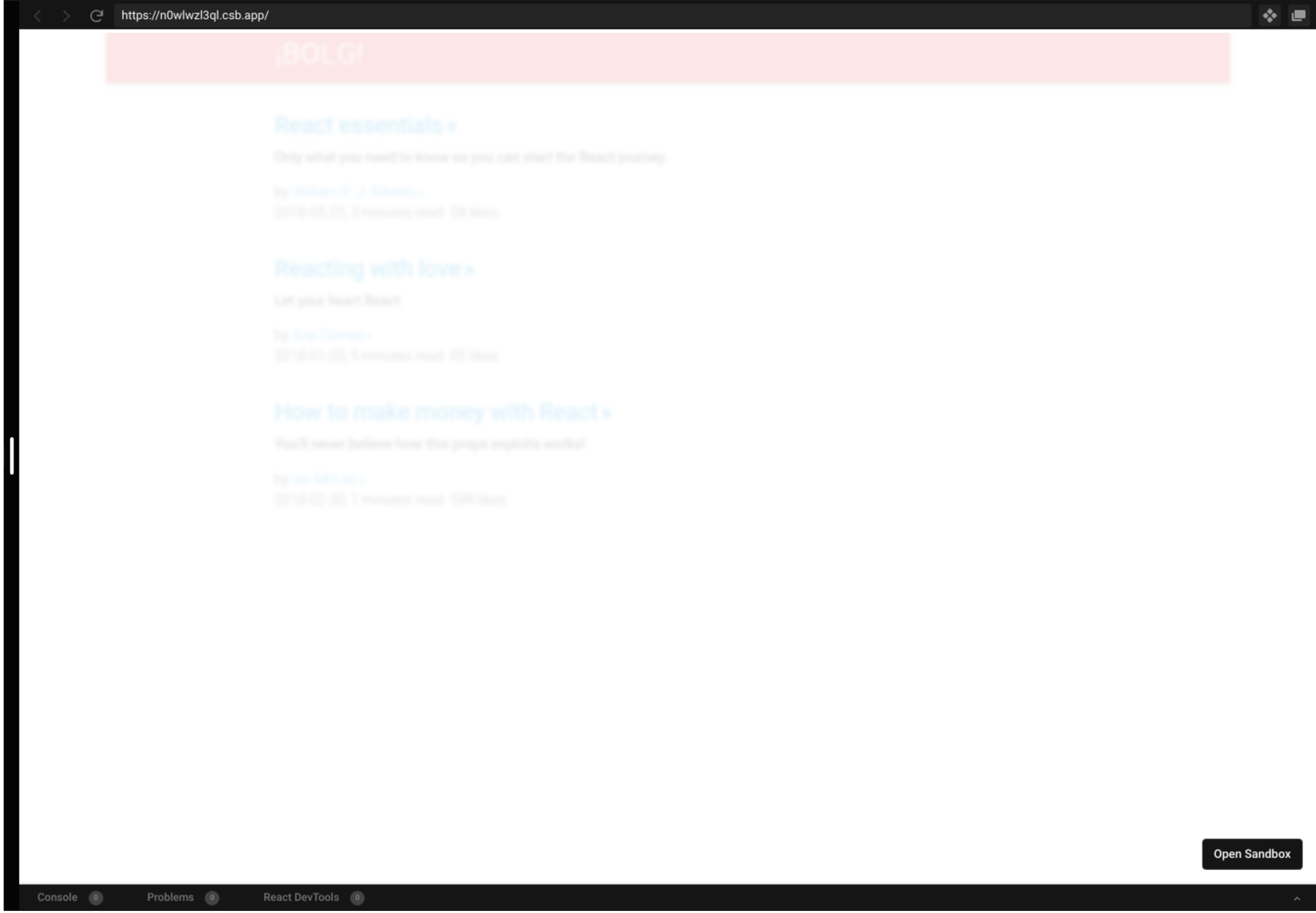
Props.children

https://codesandbox.io/s/7403046mvx?file=/src/index.js

UNINASSAU | ser educacional

React

Exemplo de Site com React

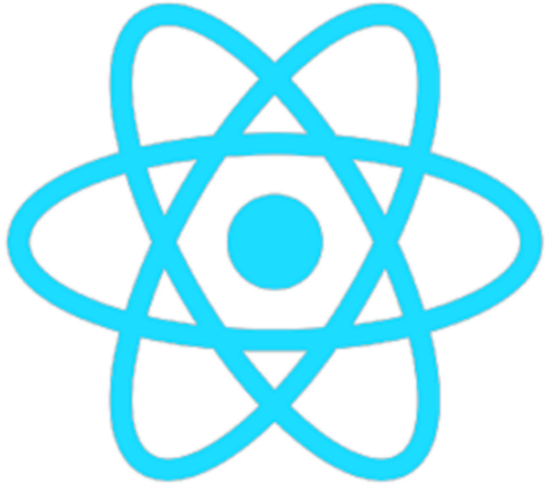


https://codesandbox.io/s/n0wlwzl3ql?file=/src/index.js

UNINASSAU | ser educacional

React

Material



React

UNINASSAU | ser educacional

React

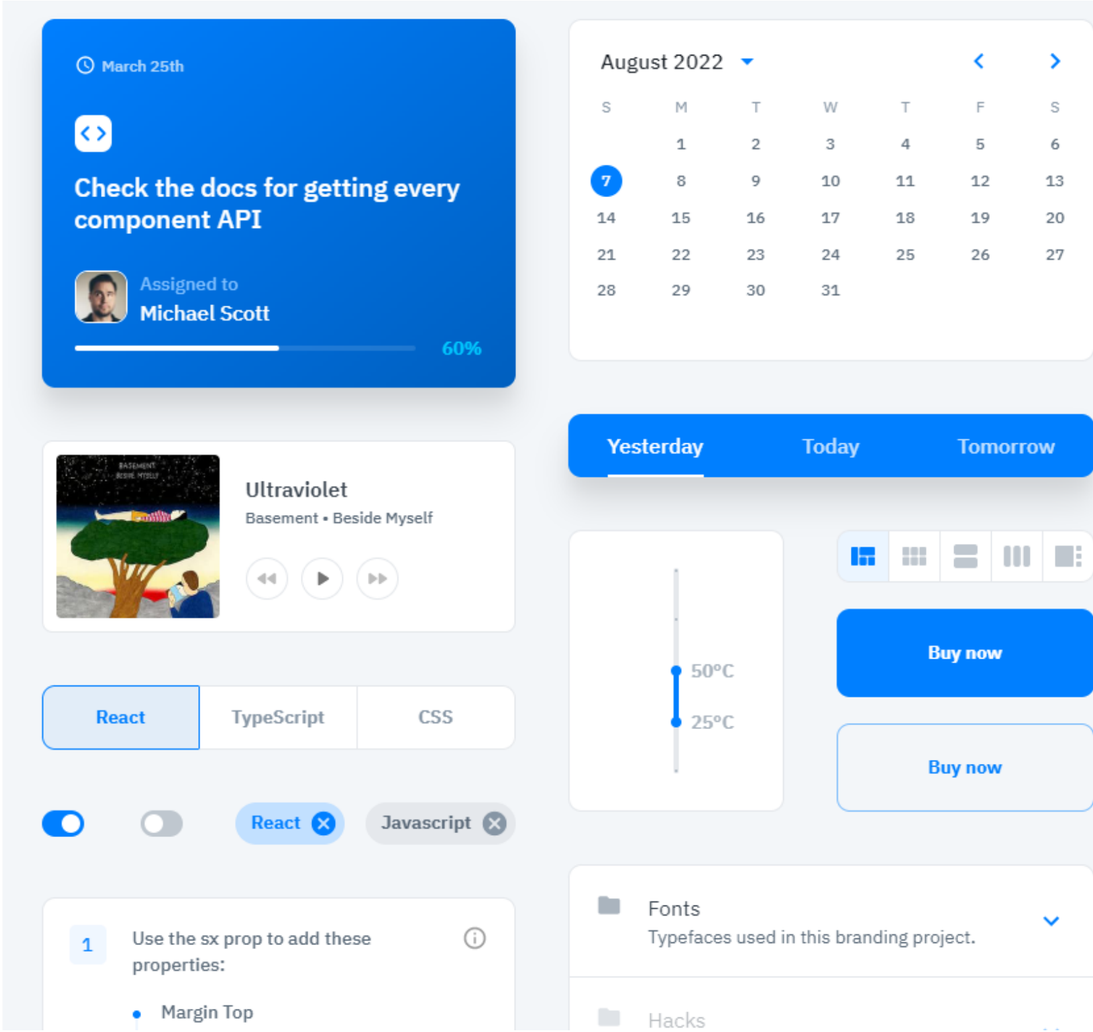
Material

Move faster with intuitive React UI tools

MUI offers a comprehensive suite of UI tools to help you ship new features faster. Start with Material UI, our fully-loaded component library, or bring your own design system to our production-ready components.

Get started

npm install @mui/material @emotion…



Spotify amazon NASA

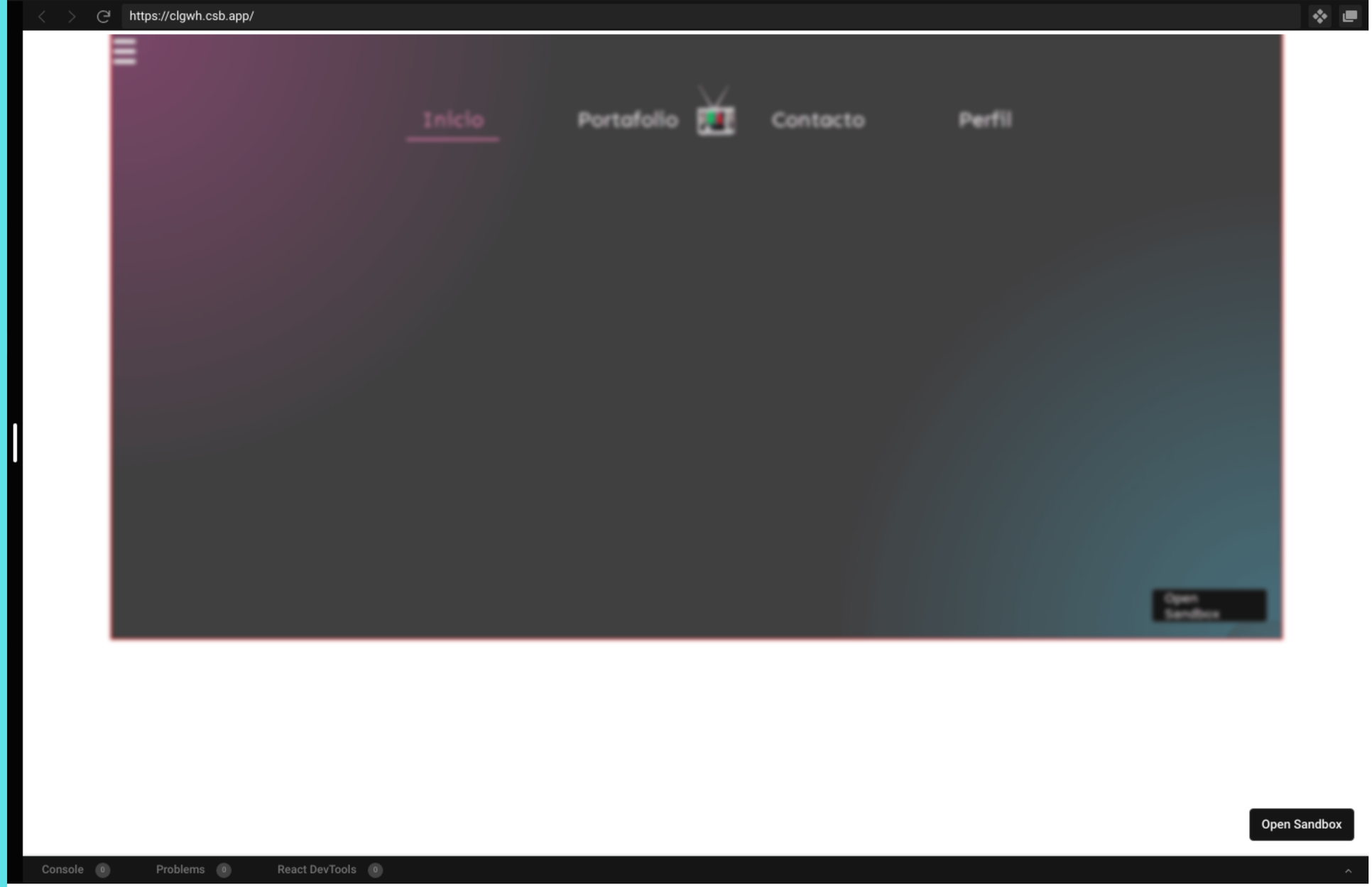
https://mui.com/pt/

UNINASSAU

ser educacional

React

Material

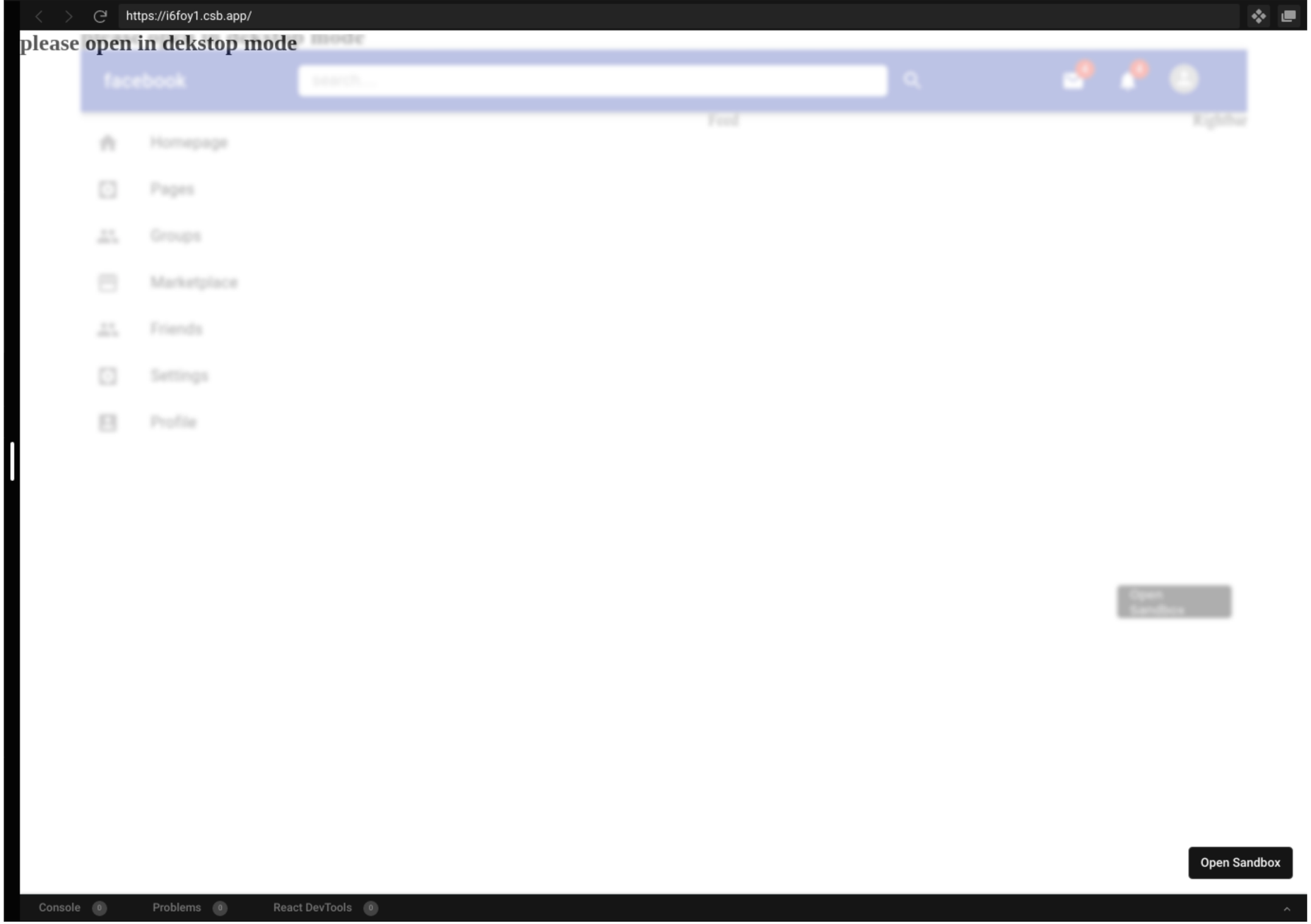


https://codesandbox.io/s/material-clgwh

UNINASSAU

React

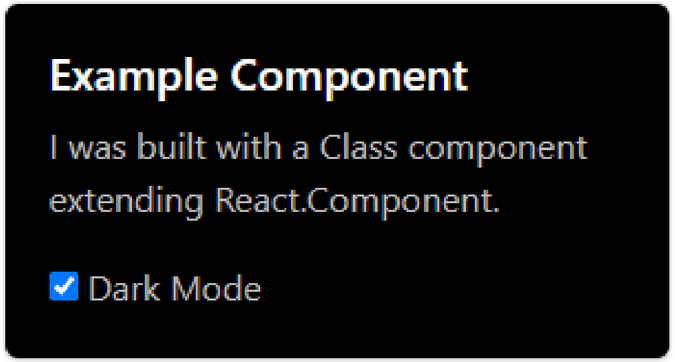
Material



https://codesandbox.io/s/material-ui-basic-i6foy1?file=/src/components/App.jsx

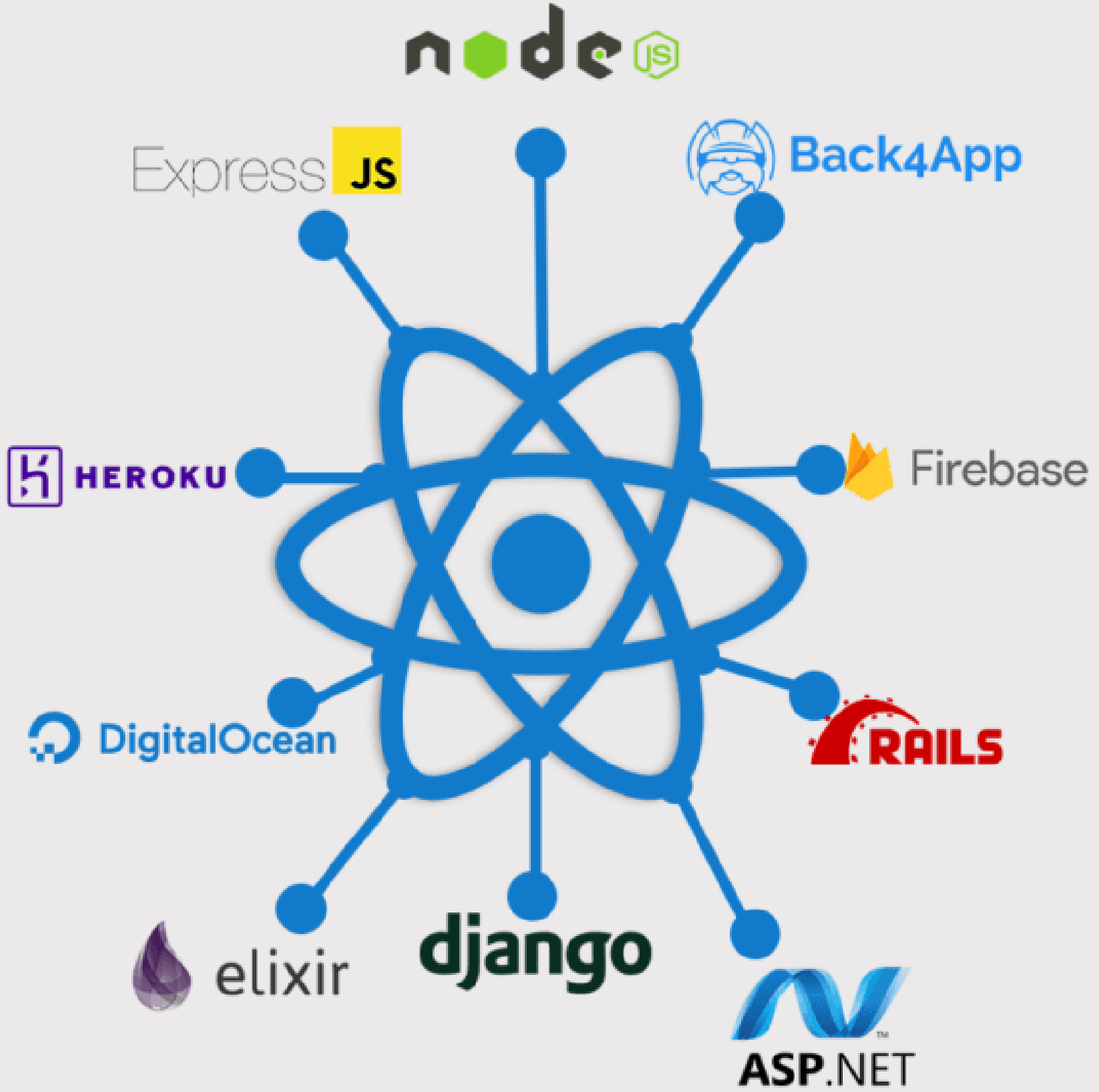
UNINASSAU | ser educacional

React Componente




```
class Card extends React.Component {

  constructor(props) {
    super(props)
    this.state = {
      darkMode: true
    }
    this.handleChange = this.handleChange.bind(this);
  }

  handleChange(event) {
    this.setState({ darkMode: !this.state.darkMode });
  }

  componentWillMount() {
    // could do something like pull state from API
  }
```

https://codepen.io/team/codepen/pen/qgJdQw


node
Express JS
Back4App
HEROKU
Firebase
DigitalOcean
RAILS
elixir
django
ASP.NET