Desenvolvimento para a Internet e Aplicações Móveis

2024/25

REACT 1

ISCTE Instituto Universitário de Lisboa
Gabinete de Comunicação e Imagem

Desenvolvimento de Frontend em REACT

Índice

1. Introdução

Existem bibliotecas utilizadas para o desenvolvimento de um website num servidor, isto é, no *backend*. Outras bibliotecas apresentam tecnologias complementares de *frontend*, de execução rápida nos clientes. Nas arquiteturas mistas o *backend* e o *frontend* estão separados e respeitam um protocolo de comunicação.

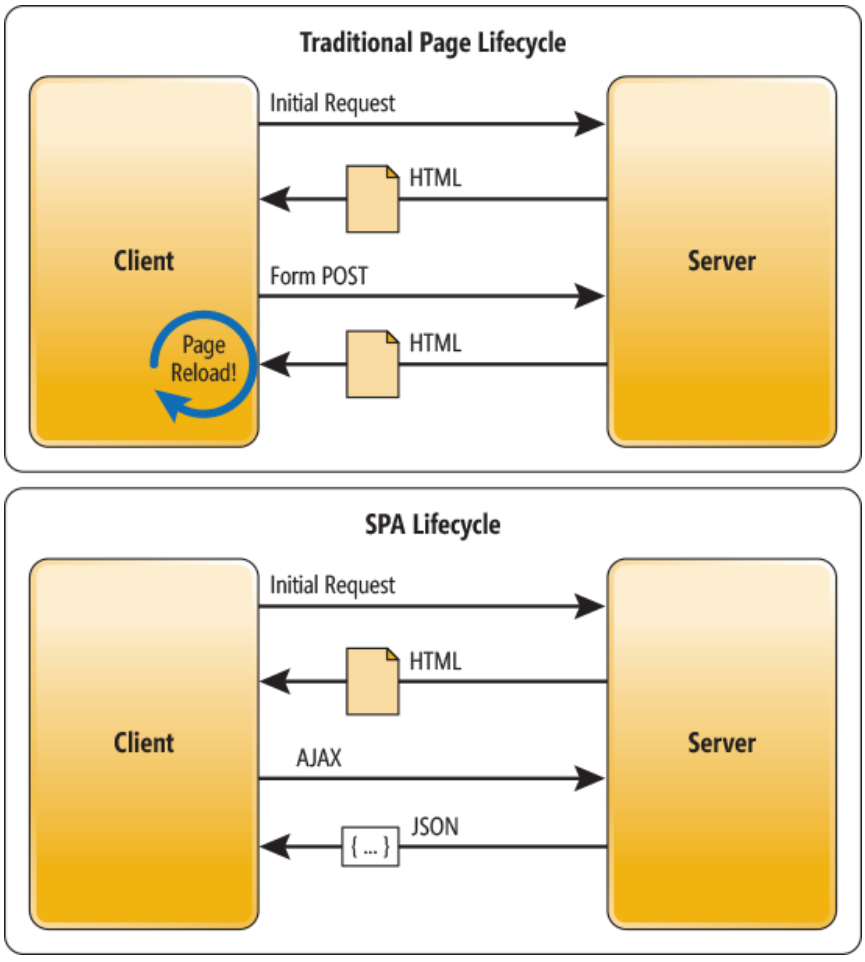
Atualmente, as melhores soluções para o desenvolvimento de *frontend* são as *single-page applications* (SPAs). Estas aplicações comunicam com o *backend* através de pontos de contacto HTTP, por onde recebem e transmitem dados no formato JSON. React é uma biblioteca JavaScript, versão ES6, frequentemente utilizada na programação de SPAs para frontend. React recorre a HTML, CSS e JavaScript. Outras bibliotecas baseadas em JavaScript e também utilizadas para o desenvolvimento de SPAs são Ember, Angular, BackBone e Vue.

Este texto apresenta os conceitos fundamentais de React baseado em HTML, CSS e JavaScript. React é uma biblioteca de programação JavaScript criada pelo Facebook para a programação de interfaces de utilizador. Permite a programação de SPAs (single-page applications) reutilizáveis. React cria e manipula um DOM virtual em memória, antes de passar as alterações para o DOM do browser, sem que seja necessário atualizar a página completa.

2. Conceitos Utilizados

1.1 SPA - Single Page Application

Páginas que funcionam por substituição de componentes do lado do cliente (usualmente em resposta a ações do utilizador) e não por refrescamento total da página. Este tipo de soluções emergiu na linha da passagem de parte do peso computacional para o cliente de modo a tornar as páginas mais rápidas (*responsive*). A performance é influenciada pela existência de uma única página HTML, atualizada dinamicamente com dados originários do servidor.

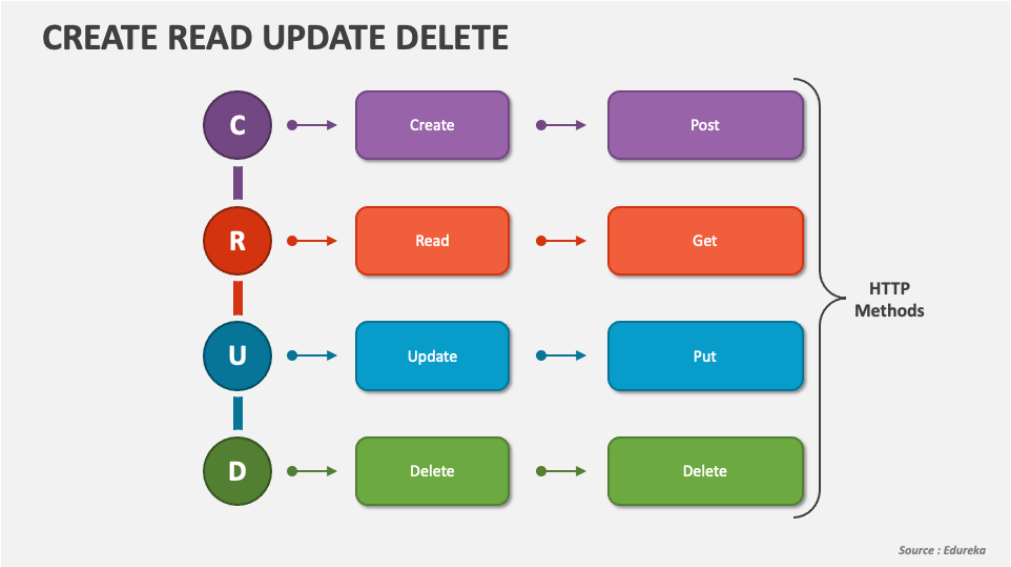


[Wasson, 2013]

1.2 CRUD - Create, Read, Update and Delete

CRUD indica as quatro operações de acesso e de alteração de dados persistentes, isto é, de dados que se mantêm entre várias execuções de um programa. É usado este termo para

interfaces que fornecem as operações de criação, leitura, atualização e eliminação. As APIs REST invocam os quatro tipos de operações CRUD.



1.3 REST - Representational State Transfer Architectural Style

Uma API REST ou RESTful (*representational state transfer architectural style*) permite o acesso a serviços de uma aplicação externa (API) que respeita os seguintes princípios:

- Interface uniforme - o interface é o sempre o mesmo independentemente da fonte do pedido e da linguagem usada pela fonte;

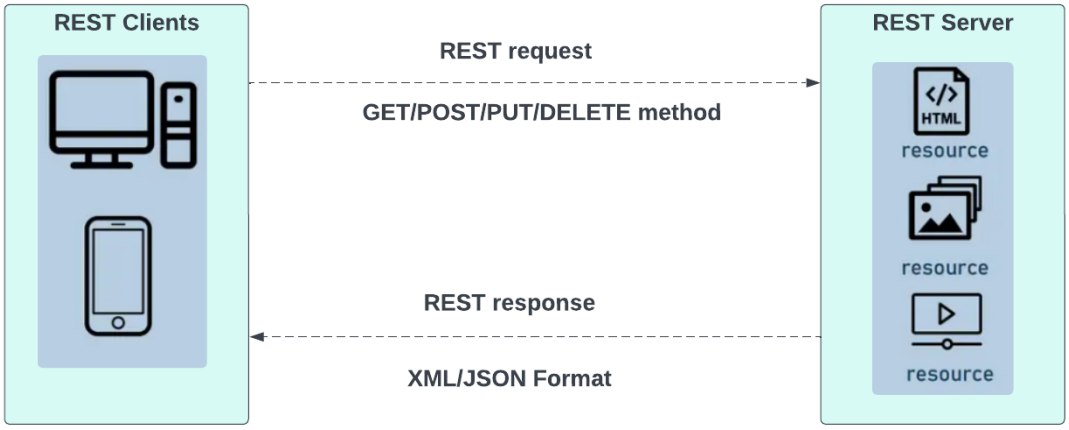
- Independência (desacoplamento) entre cliente e servidor - as aplicações cliente e servido são totalmente independentes. Para permitir o desacoplamento, normalmente as mensagens trocadas têm formatos independentes da linguagem. JavaScript Object Notation (JSON) e eXtensible Markup Language ou (XML) são os formatos mais usados;

- Ausência de estado - o serviço de um pedido não se prolonga por várias chamadas, tem que ser totalmente executável numa chamada da API (para evitar a necessidade de utilização de sessões);

- Possibilidade de usar cache do lado do cliente - deve ser possível manter respostas do lado do cliente para evitar pedidos repetidos;

- Assumir arquitetura por camadas - nem cliente, nem servidor podem assumir que comunicam diretamente um com o outro, mas que pode haver vários intermediários;

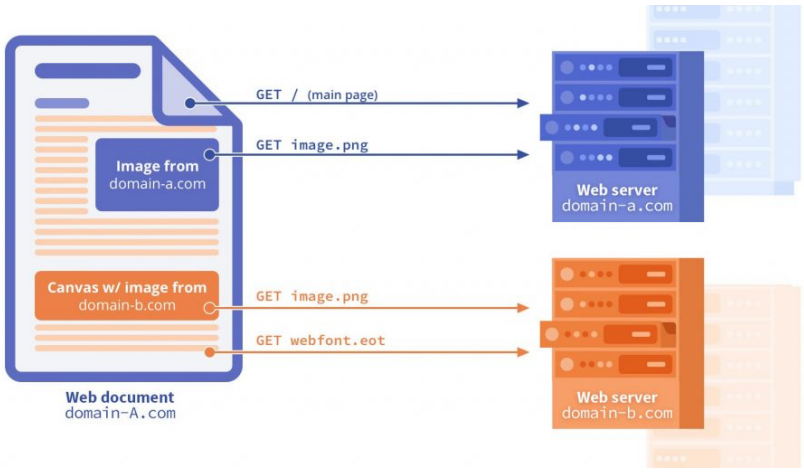
- Passagem de código (opcional) - a informação passada entre cliente e servidor pode ser composta por dados ou código a executar (o código executa a pedido do utilizador).



[Penmetsa, 2024]

1.4 CORS - Cross-Origin Resource Sharing

CORS é o mecanismo que permite a partilha de recursos de origens diferentes do domínio do pedido original, por exemplo imagens de uma página externa. CORS define a forma de um browser interagir com um servidor para determinar se é seguro permitir um pedido de recurso de origem externa.



[4DBlog, 2020]

Por exemplo, um pedido CORS existe quando um servidor https://domain-a.com faz um pedido para obter dados de outro servidor https://domain-b.com através de um script existente no seu código. Por razões de segurança os browsers restringem pedidos CORS originados em scripts.

3. Instalação do Software de Base

Para além do editor PyCharm, deve ter instalado no seu computador:

- A biblioteca Node.js que gere eventos JavaScript de forma concorrente;
- A ferramenta npm, que permite instalar e gerir versões de bibliotecas JavaScript. A instalação *default* do Node.js já inclui a ferramenta npm.

Outras bibliotecas, como Bootstrap para apresentação gráfica, Reactstrap para a programação de componentes React e Axios para gerir os pedidos HTTP, serão instaladas na altura do desenvolvimento do frontend.

4. App React

4.1 Criação

Na linha de comando do seu sistema operativo, ou num terminal do seu IDE:

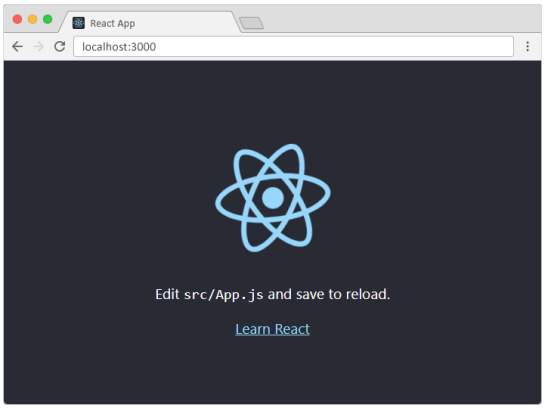
```
npx create-react-app my-react-app
```

4.2 Execução

Entre na diretoria my-react-app e execute a *app* React:

```
cd my-react-app
npm start
```

Deverá obter o seguinte resultado no seu browser. Por omissão, a *app* React corre na porta 3000 do localhost.



Para interromper o servidor React basta fazer CTRL-C.

4.3 Modificação

Entre na subdiretoria src e edite o ficheiro App.js . Substitua o código que encontrar pelo seguinte:

```
function App() {
  return (
    <section className="App">
      <h1>Os alunos do ISCTE são excelentes!</h1>
    </section>
  );
}

export default App;
```

Execute o servidor e verifique o resultado no browser.

Agora apague todos os ficheiros na diretoria src e crie nesta diretoria o ficheiro index.js com o código seguinte:

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

const oMeuPrimeiroElemento = <h1>Cá está o React de novo!</h1>

const root = ReactDOM.createRoot(document.getElementById('root'));
root.render(oMeuPrimeiroElemento);
```

Execute o servidor de novo e verifique o resultado no browser. Criou o seu primeiro elemento React e fez render deste elemento no document, isto é na sua página gerada por JavaScript. O código será explicado em detalhe no ponto seguinte.

5. Renderização de HTML

5.1 createRoot()

Função que define em que elemento HTML o componente React será renderizado.

5.2 render()

Função que define o componente React a renderizar. O resultado da renderização é um ficheiro HTML gravado na diretoria public. Frequentemente o resultado da renderização tem efeito no componente "root" do ficheiro HTML.

```
<body>
  <section id="root"></section>
</body>
```

5.3 JSX

JSX significa JavaScript XML. O código JSX permite escrever HTML dentro de código JavaScript.

JSX avalia as expressões dentro de { }. Exemplo:

```
const umElemento = <h1>React é {999 + 1} vezes melhor com JSX</h1>;
```

Um componente deve ter apenas um elemento HTML. Se quiser colocar vários elementos HTML, coloque-os dentro de um bloco (`<div>`, `<section>`, etc.):

```
const umElemento = (
  <section>
    <p>Este é um parágrafo.</p>
    <p> Este é outro parágrafo.</p>
  </ section >
);
```

O atributo class do HTML em JSX é denominado className.

```
const umElemento = <h1 className="aMinhaClasse">Cá está o React!</h1>;
```

JSX não admite condições usando if. Alternativa: programar o if fora do JSX.

```
const x = 5;
let texto = "Até breve";
if (x < 10) {
  texto = "Olá";
}

const umElemento = <h1>{texto}</h1>;
```

Teste e verifique o resultado no browser.

Exemplo completo de um componente definido com JSX:

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

const umElemento = (
  <table>
    <tr>
      <th>Nome</th>
    </tr>
    <tr>
      <td>Joana</td>
    </tr>
    <tr>
      <td>João</td>
    </tr>
  </table>
);

const container = document.getElementById('root');
const root = ReactDOM.createRoot(container);
root.render(umElemento);
```

Verifique o resultado no browser.

6. Estilos CSS

Existem várias formas de associar estilos CSS a um ficheiro React.

6.1 Definição do CSS no próprio ficheiro, em constantes

A forma mais simples e direta consiste na escrita do código CSS no próprio ficheiro, em constantes depois utilizadas num elemento JSX, da seguinte forma:

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

const separador = {
    color: "white", backgroundColor: "DodgerBlue",
    margin: "0", height: "5vh",
};
const contentor = {
    color: "DodgerBlue", backgroundColor: "LightBlue",
    margin: "0", height: "20vh",
    display: "flex", justifyContent: "center", alignItems: "center",
```

```
};

return (
    <>
        <div style={separador}/>
        <div style={contentor}>
            <h1>Este é o texto dentro do contentor</h1>
        </div>
        <div style={separador}/>
    </>
)

const container = document.getElementById('root');
const root = ReactDOM.createRoot(container);
root.render(umElemento);
```

6.2 Definição do CSS num ficheiro

Outra forma consiste na escrita do código CSS num ficheiro de texto com extensão .css . Este ficheiro é depois importado no seu respetivo ficheiro React. Exemplo:

index.css

```
body {
  background-color: #282c34;
  color: white;
  padding: 40px;
  font-family: Sans-Serif;
  text-align: center;
}

#umId {
  background-color: yellow;
}
```

index.js

```
import React from 'react';
import ReactDOM from 'react-dom/client';
import './index.css';

const umElemento = (
  <section>
    <p id="umId">Este é um parágrafo.</p>
    <p> Este é outro parágrafo.</p>
  </ section >
);
```

```
const container = document.getElementById('root');
const root = ReactDOM.createRoot(container);
root.render(umElemento);
```

6.3 Instalação de uma biblioteca de CSS pré-definida (exemplo com Bootstrap)

Instale o Bootstrap na diretoria do projeto React:

```
npm install bootstrap
```

Importe o Bootstrap no ficheiro inicial no projetp React (src/index.js ou src/App.js):

```
import "bootstrap/dist/css/bootstrap.min.css";
```

Importe também os componentes Bootstrap para JavaScript:

```
import "bootstrap/dist/js/bootstrap.bundle.min";
```

6.4 Instalação de uma biblioteca de CSS pré-definida (exemplo com React-Bootstrap)

Instale o React-Bootstrap na diretoria do projeto React:

```
npm install react-bootstrap
```

Exemplo de utilização de um botão com estilo React-Bootstrap:

```
import Button from "react-bootstrap/Button";

function App() {
      return <Button variant="primary">Prima aqui</Button>;
}

export default App;
```

Consulte os components disponíveis em react-bootstrap:
https://react-bootstrap.netlify.app/

7. Bibliografia

"React Top-Level API", https://legacy.reactjs.org/docs/react-api.html . Acedido em Acedido em 16-04-2024.

"React Tutorial", https://www.w3schools.com/REACT/default.asp . Acedido em 16-04-2024.

Purohit, Rakesh: "A Comprehensive Guide to React Practical Exercises and Coding Challenges", https://www.dhiwise.com/post/a-comprehensive-guide-to-react-practical-exercises-and-coding-challenges . Acedido em 18-04-2024.

Souza, "Using React with Django to create an app: Tutorial" https://blog.logrocket.com/using-react-django-create-app-tutorial/ . Acedido em 24-04-2023.

Singhal, "How to create a REST API with Django REST framework" https://blog.logrocket.com/django-rest-framework-create-api/ . Acedido em 24-04-2023.

Parker, "How to NPM Start for React Tutorial Project" https://www.pluralsight.com/guides/npm-start-for-react-tutorial-project . Acedido em 24-04-2023.

Basques and Emelianova, "Simulate mobile devices with Device Mode" https://developer.chrome.com/docs/devtools/device-mode/ . Acedido em 24-04-2023.

Wasson, Mike, "ASP.NET - Single-Page Applications: Build Modern, Responsive Web Apps with ASP.NET", MSDN Magazine, Volume 28 Numero 11, Novembro 2013.

Penmetsa, Chaitanya, "Representational State Transfer (REST) and Design Principles", Medium, Janeiro 2024, https://medium.com/codenx/representational-state-transfer-rest-and-design-principles-98640faa1ab4 . Acedido em 16-02-2025.

"Support of Cross-Origin Resource Sharing (CORS)", 4DBlog, Agosto 2020, https://blog.4d.com/support-of-cross-origin-resource-sharing-cors/ , acedido em 16-02-2025.