

Desenvolvimento em JavaScript

2. Principais Frameworks de
Desenvolvimento

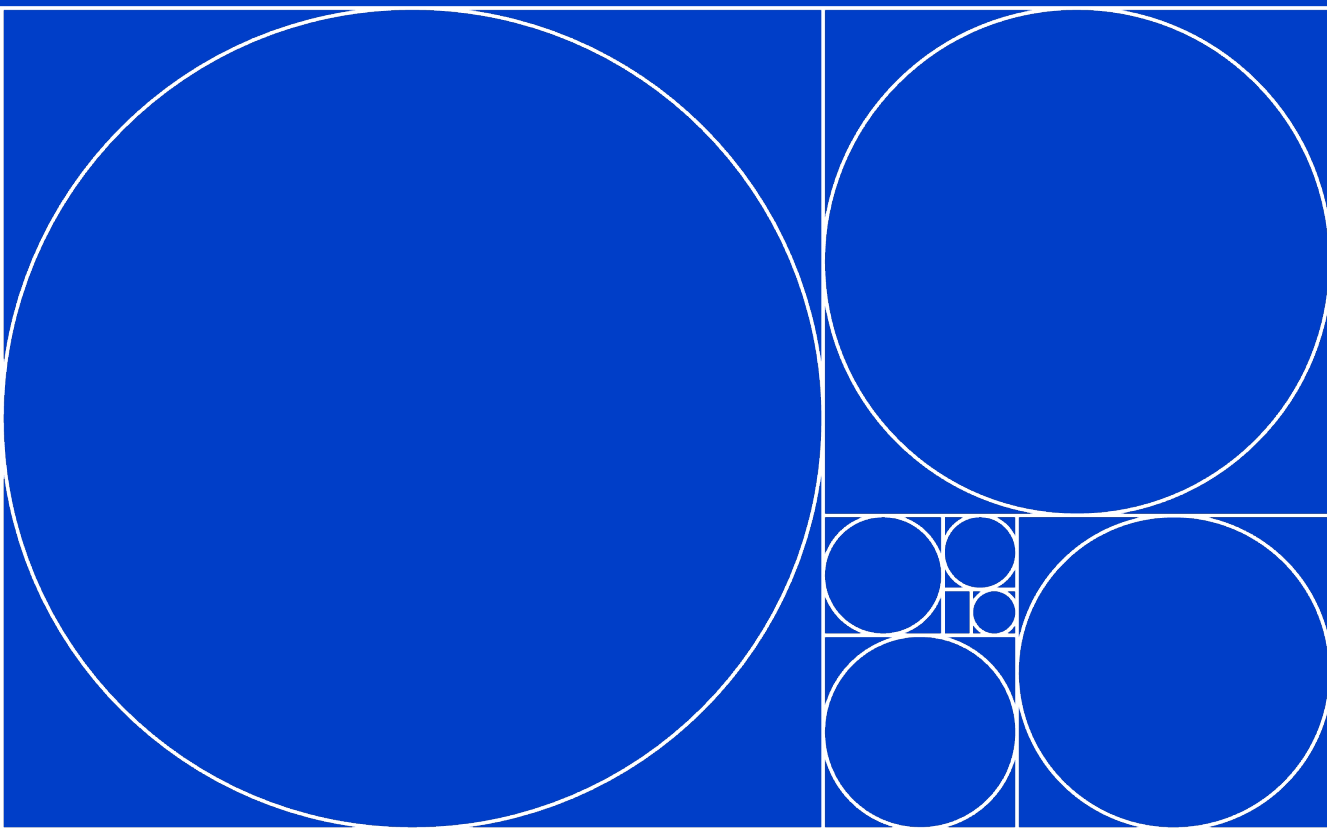
dezembro de 2025



Hoje

- 1. Bibliotecas e *Frameworks*
- 2. jQuery
- 3. Bootstrap
- 4. Bibliotecas a considerar
- 5. TPC

1. Bibliotecas e *Frameworks*



O que é uma Biblioteca?

- Uma biblioteca é um conjunto de código já escrito que resolve problemas específicos e comuns, e que o programador pode usar quando quiser e como quiser.
- Funciona como uma caixa de ferramentas: tens várias funções prontas, mas és tu que decides quando as chamar, onde as usar e como organizar o resto do código.

Quando usas uma biblioteca, **o controlo continua a ser teu**. O teu código é o “protagonista” e a biblioteca entra apenas como apoio.

A biblioteca não te impõe uma estrutura global, nem dita como o projeto deve ser organizado.

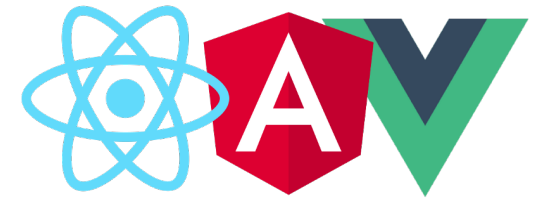
Vantagens das Bibliotecas

- **Ganhos de produtividade:** Ao utilizar código já desenvolvido, testado e documentado, o programador evita reimplementar funcionalidades comuns.
- **Redução de erros:** Bibliotecas populares são usadas por milhares de programadores, o que significa que muitos bugs já foram identificados e corrigidos ao longo do tempo.
- **Consistência no código:** ao seguir uma API bem definida, diferentes partes da aplicação acabam por usar os mesmos padrões e métodos, tornando o código mais previsível e legível.
- **Oferecem soluções otimizadas:** compatibilidade entre browsers, performance e acessibilidade.

Desvantagens das Bibliotecas

- **Custo de aprendizagem associado:** As bibliotecas exigem tempo para ler a documentação e entender as boas práticas.
- **Dependência externa.** O projeto passa a depender de código que não foi escrito pela própria equipa, o que pode ser problemático se a biblioteca deixar de ser mantida ou introduzir alterações.
- **Perda de controlo total:** Ao usar uma biblioteca, o programador aceita as decisões tomadas pelos seus autores.
- **Risco de incluir mais código do que o necessário:** Muitas bibliotecas trazem funcionalidades que não serão usadas, aumentando o tamanho dos ficheiros.

O que é uma *Framework*?



- Uma *framework* é um conjunto muito mais abrangente de ferramentas, regras e estruturas que define como a aplicação deve ser construída.
- É a *framework* que chama o teu código, não o contrário.

Quando escolhes uma *framework*:

- Aceitas uma forma específica de estruturar o projeto.
- Segues convenções rígidas.
- Trabalhas dentro de um “molde” pré-definido.

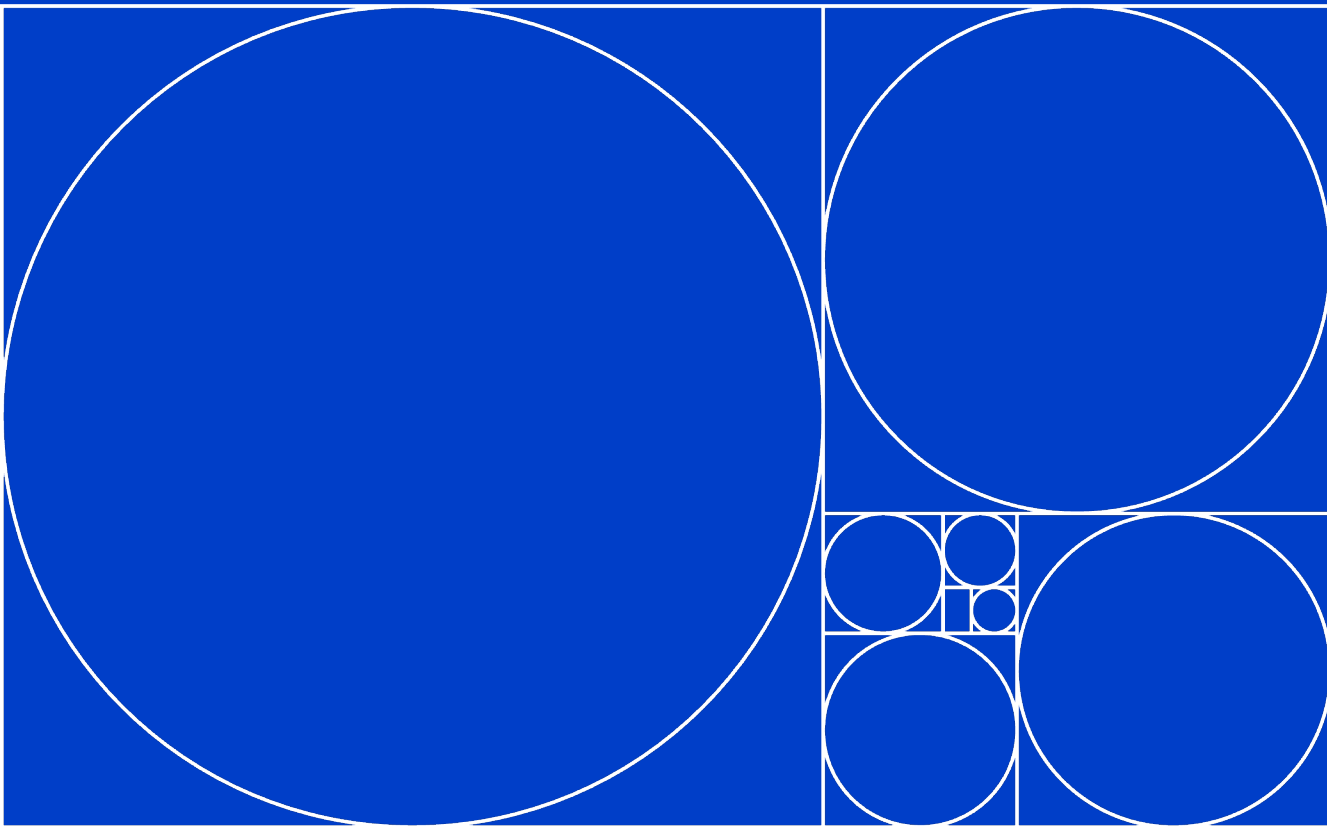
Vantagens das *Frameworks*

- **Estrutura e organização:** Uma *framework* impõe uma arquitetura clara (pastas, ficheiros, padrões), que assegura consistência e facilita o trabalho em equipa.
- **Acelera o desenvolvimento:** Muitas funcionalidades já estão prontas, como gestão de estado, rotas, ou comunicação com APIs.
- **Escalabilidade:** Projetos médios e grandes tornam-se mais fáceis de escalar (ex: adicionar novas funcionalidades).
- **Ecossistema e comunidade:** *Frameworks* populares oferecem documentação extensa, vários exemplos de aplicação, *plugins* e extensões.

Desvantagens das *Frameworks*

- **Curva de aprendizagem:** É necessário aprender conceitos, convenções e ferramentas associadas (como comandos de terminal e *build tools*).
- **Menor controlo sobre os detalhes,** uma vez que a *framework* decide como o código é estruturado e como certas tarefas são feitas.
- **Dependência tecnológica,** porque ficamos dependentes da evolução da *framework* e das decisões de quem a mantém atualizada.
- **Performance nem sempre otimizada:** As *Frameworks* são genéricas por natureza, o que poderá causar maior "overhead" em projetos pequenos. Código escrito "à mão" pode ser mais leve e rápido nestes casos.

2. jQuery



jQuery

- A biblioteca mais popular de JavaScript.
- Lançada em 2006, em 2015 estava no auge, estando presente em 17% de todos os sites da internet, e em 62.7% dos websites mais populares.
- Em 2025, ainda perto de 200 milhões de websites utilizam jQuery.
- A versão 3.0 foi lançada em junho de 2016. Em agosto de 2025 foi, finalmente, lançado o 1º release candidate da versão 4.0.

jQuery – porquê?

- **Sintaxe intuitiva:** fácil de aprender, mesmo para iniciados.
- **Simplificou drasticamente o JavaScript** quando este era inconsistente.
- **Resolveu problemas de compatibilidade** entre *browsers* (IE, Firefox, Chrome, etc.).
- **Grande ecossistema de *plugins*** prontos a usar (sliders, modals, validações de formulários...).

jQuery – principais utilizações

- Manipulação do DOM muito simples (ex: `$('div').hide()` em vez de várias linhas de código).
- Gestão de eventos facilitada (*clicks, hover, submit, etc.*).
- AJAX acessível numa época em que fazer pedidos assíncronos era complexo.

jQuery – faz sentido em 2025?

As principais funcionalidades que tornaram o jQuery especial já estão integradas nativamente no JS moderno.

- `document.querySelectorAll()` substitui `$()`
- `fetch()` substitui `$.ajax()`
- `element.classList` substitui `.addClass()` e `.removeClass()`
- `element.animate()` permite gerir animações.

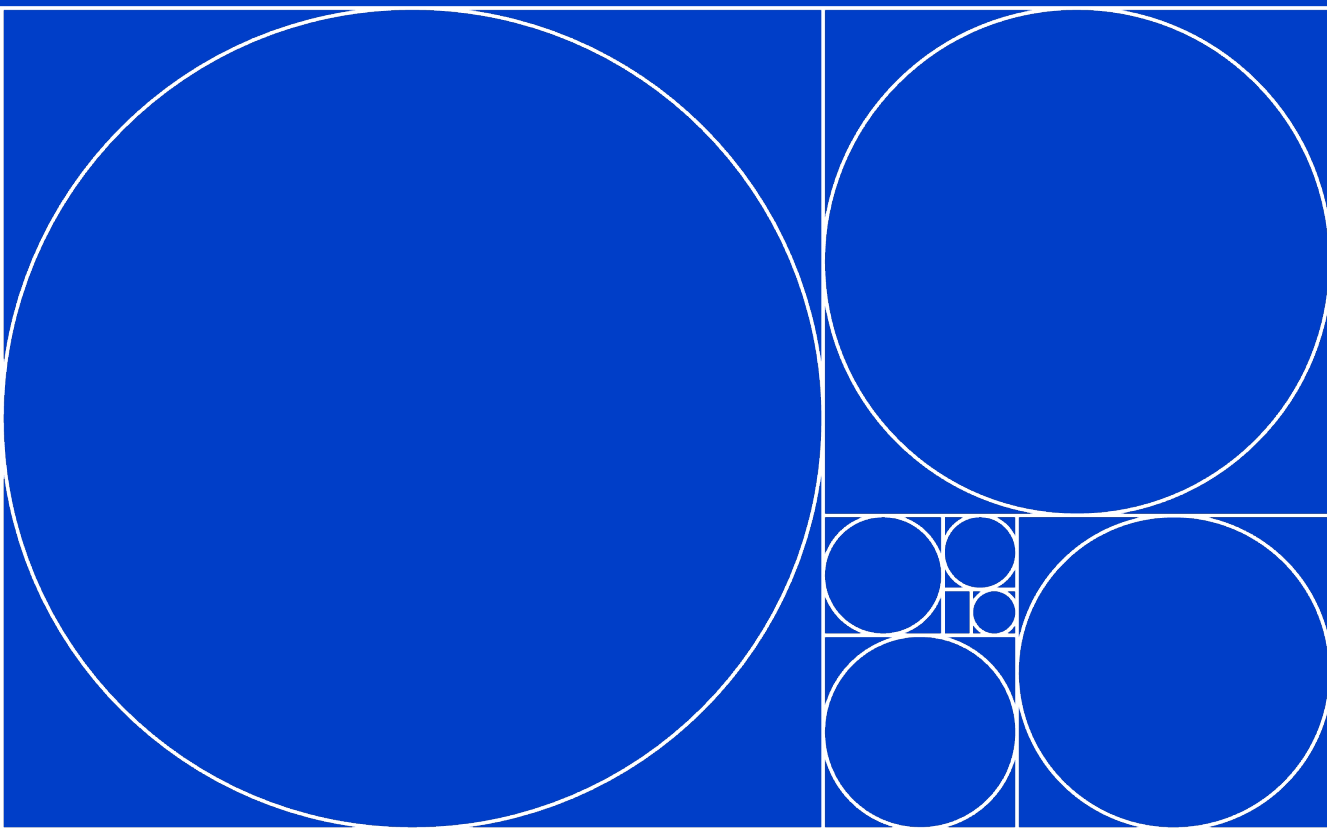
Os *browsers* atuais já são consistentes na forma como interpretam código JS, algo que não acontecia nos tempos do IE6–IE9.

jQuery – integração

1. A forma mais simples é recorrendo à CDN (recomenda-se o uso do ficheiro minificado).
2. Em alternativa, podes descarregar os ficheiros e adicioná-los diretamente ao teu projeto.
3. Por fim, podes optar pela instalação na linha de comando com **npm** ou **yarn**.

```
1  <!DOCTYPE html>
2  <html lang="en">
3  <head>
4    <meta charset="UTF-8">
5    <meta name="viewport" content="width=device-width, initial-scale=1.0">
6    <title>Hello World!</title>
7  </head>
8  <body>
9    <h1>Hello World!</h1>
10   <script src="https://code.jquery.com/jquery-3.7.1.min.js"></script>
11   <script>
12     var h1 = $('h1');
13     console.log(h1);
14   </script>
15 </body>
16 </html>
```

3. Bootstrap



Bootstrap

- A *framework* front-end (CSS-first) mais popular do mundo. Não é uma framework de JS.
- Criado em 2011 por Mark Otto e Jacob Thornton no Twitter, tornou-se rapidamente um standard para desenvolvimento de interfaces responsivas.
- Durante a década de 2010, foi utilizado por milhões de websites. Em 2025, continua a ser amplamente usado, especialmente em protótipos rápidos, backoffices e aplicações empresariais.
- O Bootstrap 4 foi lançado em 2018, introduzindo Flexbox como base do layout.
- O Bootstrap 5 foi lançado em 2021, removendo a dependência do jQuery e adotando JavaScript moderno.

Bootstrap – porquê?

- Reduz a necessidade de escrever CSS “do zero”.
- Acelera drasticamente o desenvolvimento de interfaces web.
- Fornece um sistema de layout responsivo pronto a usar (Grid).
- Oferece componentes UI pré-construídos (modals, navbars, carousels, accordions, alerts...).
- Grande comunidade, documentação clara e ampla adoção no mercado.

Bootstrap – principais utilizações

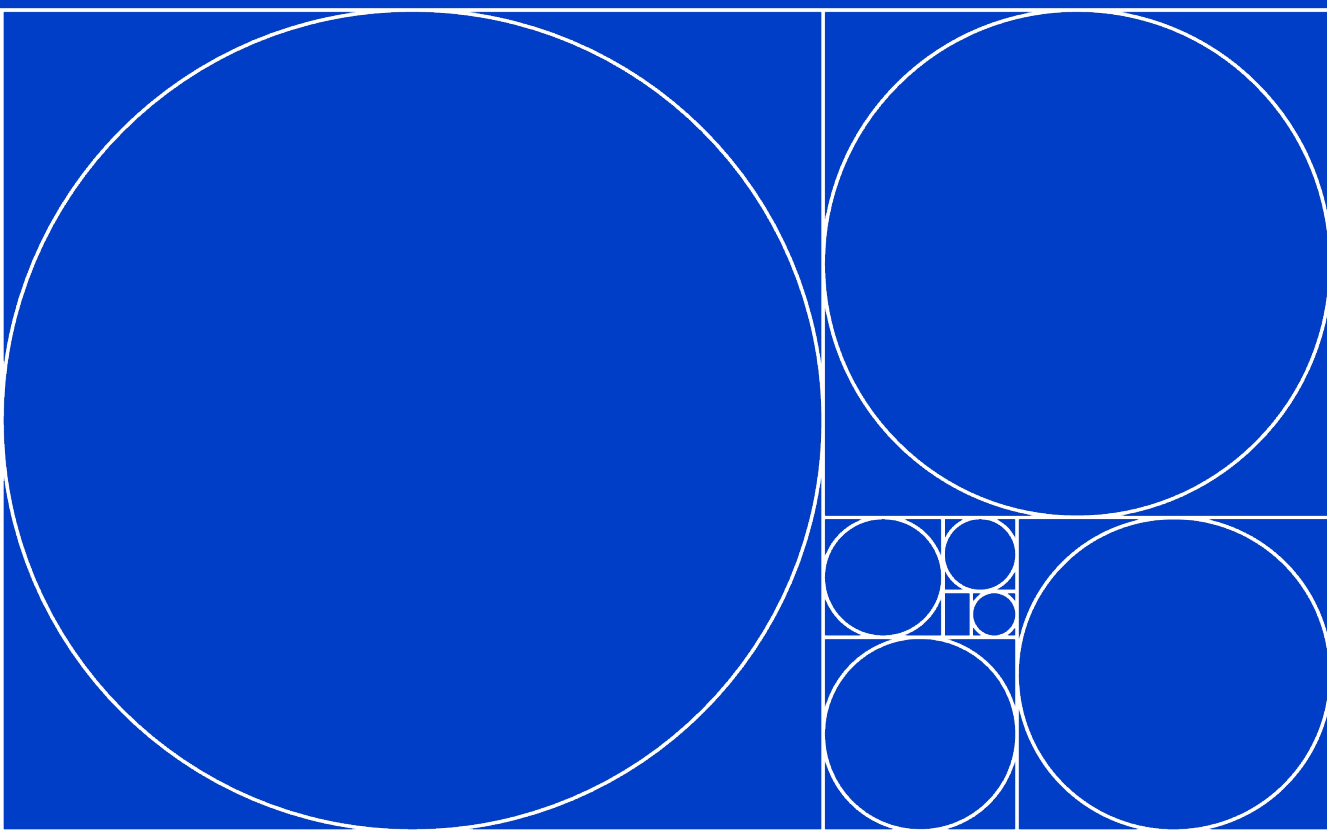
- Criação rápida de interfaces com classes pré-definidas (sem CSS extenso).
- Gestão de layout responsivo de forma simples (grid, breakpoints).
- Componentes interativos prontos (modals, dropdowns, tooltips).
- Consistência visual entre browsers e tamanhos de ecrã.

Bootstrap – integração

1. A forma mais simples é recorrendo à CDN (recomenda-se o uso do ficheiro minificado).
2. Em alternativa, podes descarregar os ficheiros e adicioná-los diretamente ao teu projeto.
3. Por fim, podes optar pela instalação na linha de comando com **npm** ou **yarn**.

```
1 <!DOCTYPE html>
2 <html lang="en">
3 <head>
4   <meta charset="UTF-8">
5   <meta name="viewport" content="width=device-width, initial-scale=1.0">
6   <title>Hello World!</title>
7   <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.8/dist/css/bootstrap.min.css" rel="stylesheet" integrity="
  sha384-sRII4kxILFvY47J16cr9ZwB07vP4J8+LH7qKQnuqkuIAvNWLzeN8tE5YBujZqJLB" crossorigin="anonymous">
8 </head>
9 <body>
10  <div class="container text-center mt-5">
11    <h1 class="mb-4">Hello World!</h1>
12    <button class="btn btn-primary">Hello Bootstrap 🙌</button>
13  </div>
14  <script src="https://cdn.jsdelivr.net/npm/bootstrap@5.3.8/dist/js/bootstrap.bundle.min.js" integrity="
  sha384-FKyoEForCGlyvw9Hj09JcYn3nv7wiPVLz7YYwJrWVcXK/BmnVDxM+D2scQbITxI" crossorigin="anonymous"></script>
15 </body>
16 </html>
```

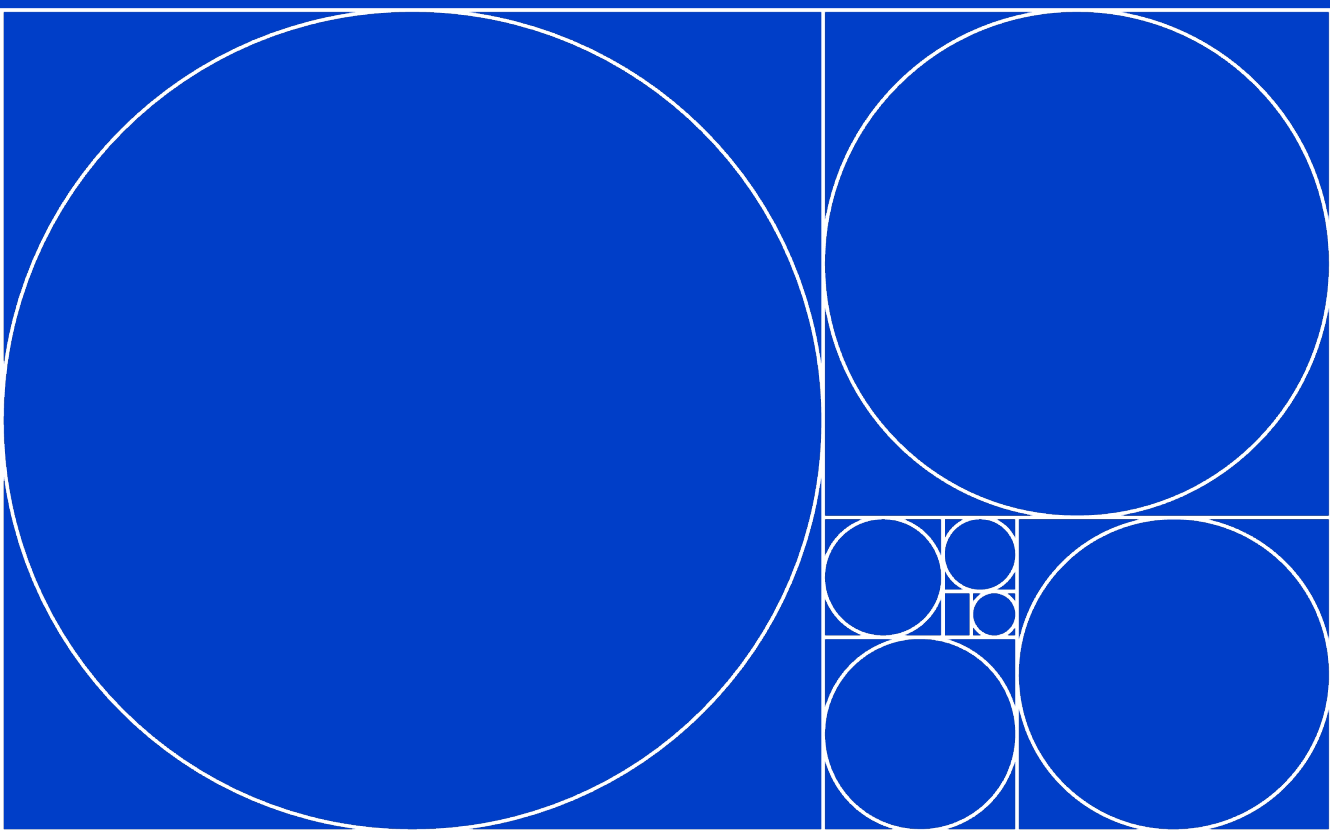
4. Bibliotecas a considerar



Bibliotecas a considerar

- [Chart.js](#) – gráficos simples e rápidos.
- [D3.js](#) – visualizações de dados avançadas e customizadas.
- [Day.js](#) – manipulação de datas.
- [GSAP](#) e [Motion](#) – animações avançadas.
- [Swiper.js](#) – *sliders* e *carousels* modernos.
- [Three.js](#) – gráficos e experiências 3D.

5. TPC



Bootstrap

Cria uma landing page (tema livre!) com os seguintes componentes Bootstrap:

- Layout com várias colunas
- Carousel com 3 imagens
- Secção de Perguntas Frequentes com Accordion
- Um modal (ativado por um clique num botão)

Torna a página única com CSS a teu gosto!

jQuery – exercício 1

Objetivo: Explorar eventos e addClass.

- Criar uma página com:
 - Um <p> com texto
 - Um botão com o texto “Destacar texto”
- Ao clicar no botão:
 - Adicionar uma classe CSS ao parágrafo (.highlight)
- A classe deve:
 - Alterar cor, fundo ou tamanho do texto (com CSS)
- Eventos e métodos a explorar:
click, addClass()

jQuery – exercício 2

Objetivo: Trabalhar efeitos básicos.


- Criar:
 - Um bloco <div> com conteúdo
 - Dois botões: “Mostrar” e “Esconder”
- Usar jQuery para:
 - Esconder o bloco
 - Voltar a mostrá-lo
- Métodos a explorar:
hide(), show() ou toggle()

jQuery – exercício 3

Objetivo: Introduzir animações jQuery.

- Criar um botão “Animar”
- Aplicar um efeito ao elemento:
 - `fadeIn` / `fadeOut`
 - ou `slideToggle`
- Métodos a explorar:
`fadeIn()`, `fadeOut()`, `slideToggle()`



Faculdade de Design,
Tecnologia e Comunicação
 Universidade Europeia

Obrigado!

Até à próxima aula! 🖐️