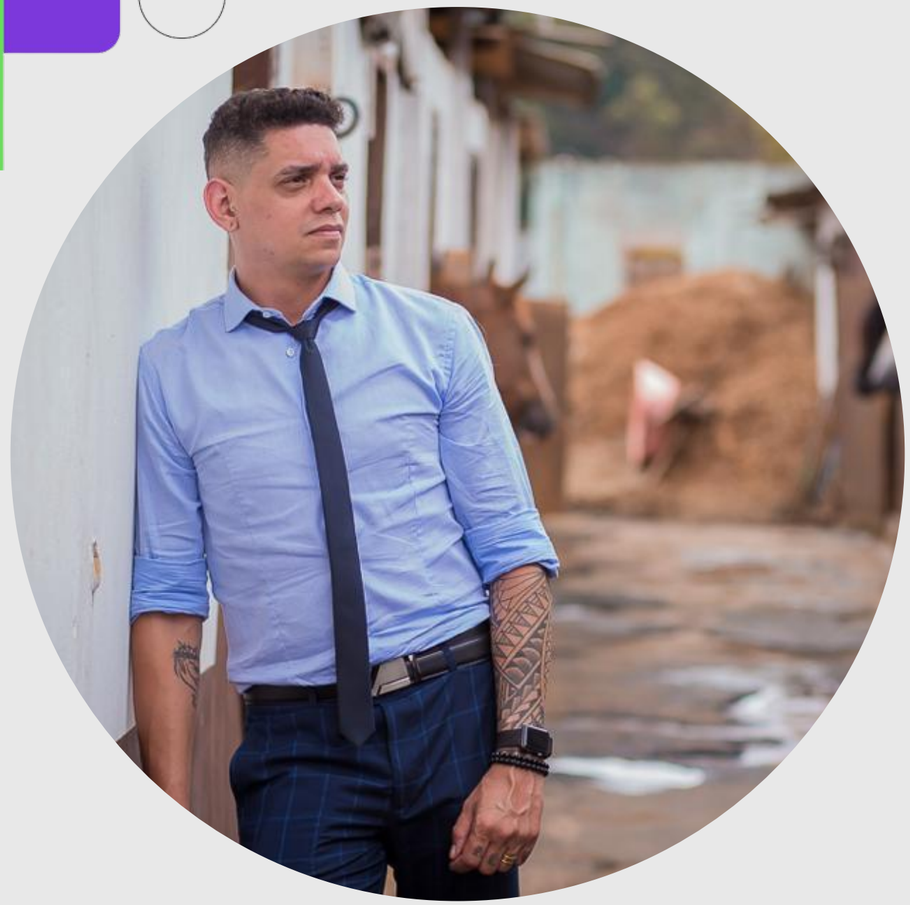
Boas Práticas

Material Complementar



#PraCegoVer: Fotografia do autor Douglas Morais.

Douglas Morais

34 anos, atua no mercado de desenvolvimento a cerca de 14 anos atualmente Techlead na Kenlo.
Apaixonado por Tecnologia e todas as novidades do segmento.
Especialista em desenvolvimento frontend um eterno entusiasta do desenvolvimento mobile.

linkedin.com/in/douglasmoraisdev/

https://github.com/mrdouglasmorais

Índice

- BEM (Bloco Elemento e Modificador)
- Atomic Design
- Design System
- Micro frontend
- MVC (Model View Controller)
- MVVM (Model View View Model)
- Design Patterns

Introdução

Nesta apostila vamos aprender sobre boas prática em desenvolvimento e a importância em torno destes padrões.

Projetos escaláveis e que envolve uma equipe atuando no mesmo projeto, requer certas medidas para que cada entrega seja de fato uma entrega de valor, principalmente a manutenção posterior fique de acordo com as premissas do time e de fácil entendimento. Por este motivo as boas práticas na criação do código é de suma importância.

Os temas abordados serão de padrões de nomenclatura a organização e até mesmo conceitos de média complexidade envolvendo soluções de camada lógica.

#PraCegoVer: Dois notebooks
o primeiro apresentado o teclado
o segundo apresentando o
monitor.

BEM

Bloco, Elemento e Modificador

BEM

O BEM css (Block Element Modifier ou Bloco Elemento Modificador) e uma metodologia ou convenção de nomenclatura muito utilizada no mercado.

Desenvolvido pela *Yandex* (Crowler de busca com maior market share da Rússia), nasceu como parte de um framework completo e é uma das maneiras mais eficientes de tratar nomenclaturas de arquivos *CSS*.

Há quem diga que o BEM faz alusão escrita a programação orientada a objetos, mas eu enxergo este método de maneira mais clara e eficiente a nível organizacional do projeto, intuitivo e de fácil compreensão.

Regras

Como o próprio acrônimo do BEM cada item tem uma missão específica dentro da metodologia.

E são exatamente as regras de escrita associado aos Blocos, Elementos e Modificador.

Bloco: Elemento único representado por ele mesmo.

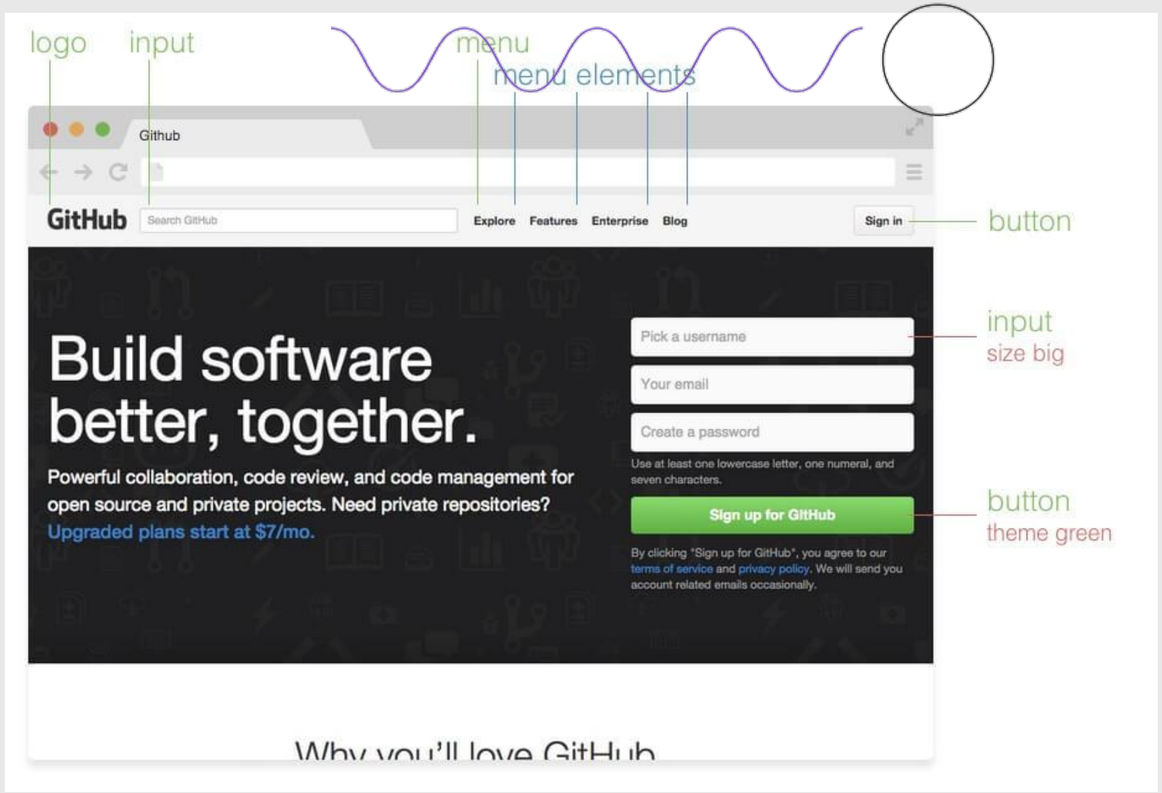
ex: header, container, menu, checkbox, input

Elemento: Trecho de um bloco que não tem um significado de forma isolada, mas sim associada ao bloco.

ex: menu item, list item, checkbox caption, header title

Modificador: Flags de mudança de comportamento ou aparência.

ex: disabled, highlighted, checked, fixed, size big, color yellow



#PraCegoVer: Imagem ilustrando uso do BEM com interface do GitHub

Uso

HTML

```html
<button class="button">
 Normal button
</button>
<button class="button button--state-success">
 Success button
</button>
<button class="button button--state-danger">
 Danger button
</button>
```

CSS

```css
.button {
 display: inline-block;
 border-radius: 3px;
 padding: 7px 12px;
 border: 1px solid #D5D5D5;
 background-image: linear-gradient(#EEE, #DDD);
 font: 700 13px/18px Helvetica, arial;
}
.button--state-success {
 color: #FFF;
 background: #569E3D linear-gradient(#79D858, #569E3D) repeat-x;
 border-color: #4A993E;
}
.button--state-danger {
 color: #900;
}
```

Benefícios

Modular: Estilizando em escopo de bloco nunca dependem de outros elementos em uma página assim, você não terá problemas com herança ou especificidade dos elementos.

Reutilizável: O comportamento de cada bloco deve ser independente e possibilita "N maneiras" de reutilizar em seu projeto, reduz a quantidade de CSS para dar manutenção.

Com um conjunto de diretrizes de estilo, você pode construir uma biblioteca de blocos, tornando o CSS mais eficiente.

Estruturado: O BEM deixa o seu CSS com uma estrutura mais limpa e de fácil entendimento.

BEM CSS

Neste módulo tivemos uma introdução ao BEM CSS e o mais importante entendemos os ganhos com uso deste padrão de nomenclatura, melhor que ter um projeto escalando é crescer com organização e contribuindo para o fácil entendimento do demais desenvolvedores envolvidos.

#PraCegoVer: Dois notebooks
o primeiro apresentado o teclado
o segundo apresentando o
monitor.

Atomic Design

Atomic Design e sua importância

Introdução

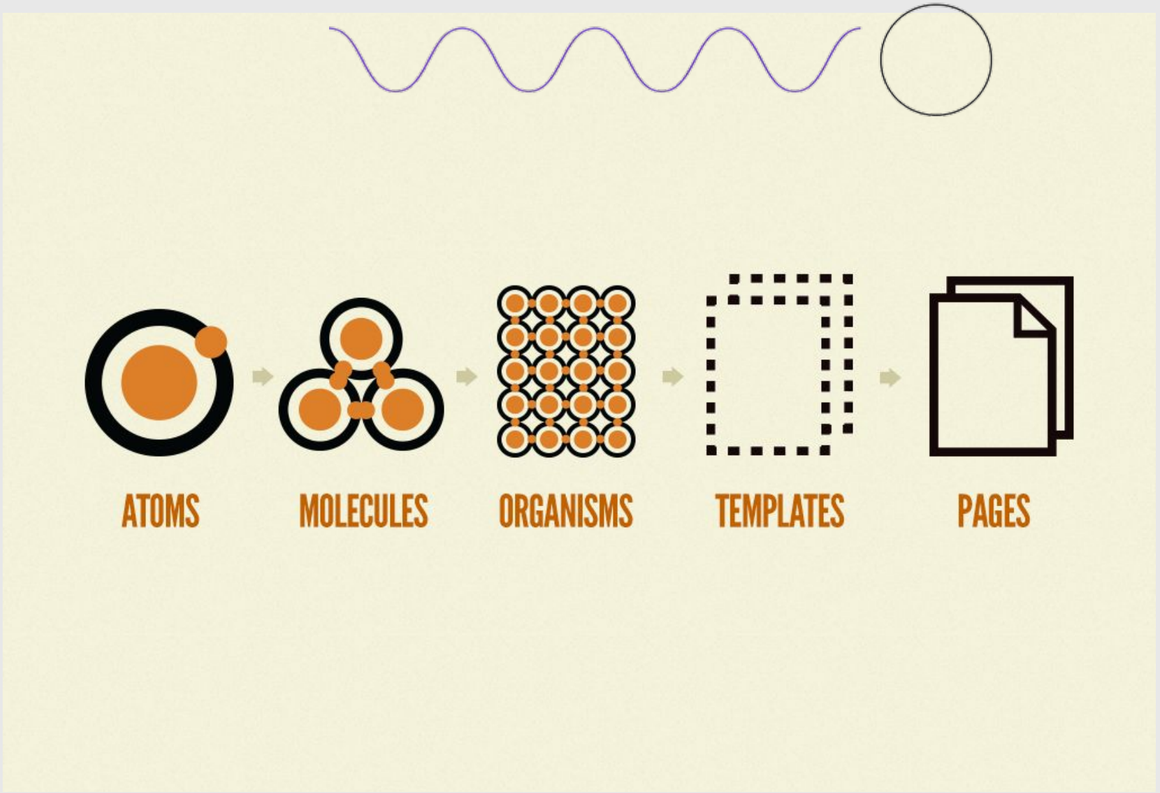
Atomic Design foi criado em meados de 2013, ele tem como forte referência química para elaborar uma metodologia eficiente para utilização em design e interfaces.

E a essência do Atomic design é as referências de átomos e moléculas em sua divisões e composição criando elementos uniformes e que são muito utilizados em padrões de Design System.

Divisão

Ele é divido em 5 partes ou componentes que juntos compõem interfaces ordenadas por hierarquias.

- **Átomos**
- **Moléculas**
- **Organismos**
- **Templates**
- **Páginas**

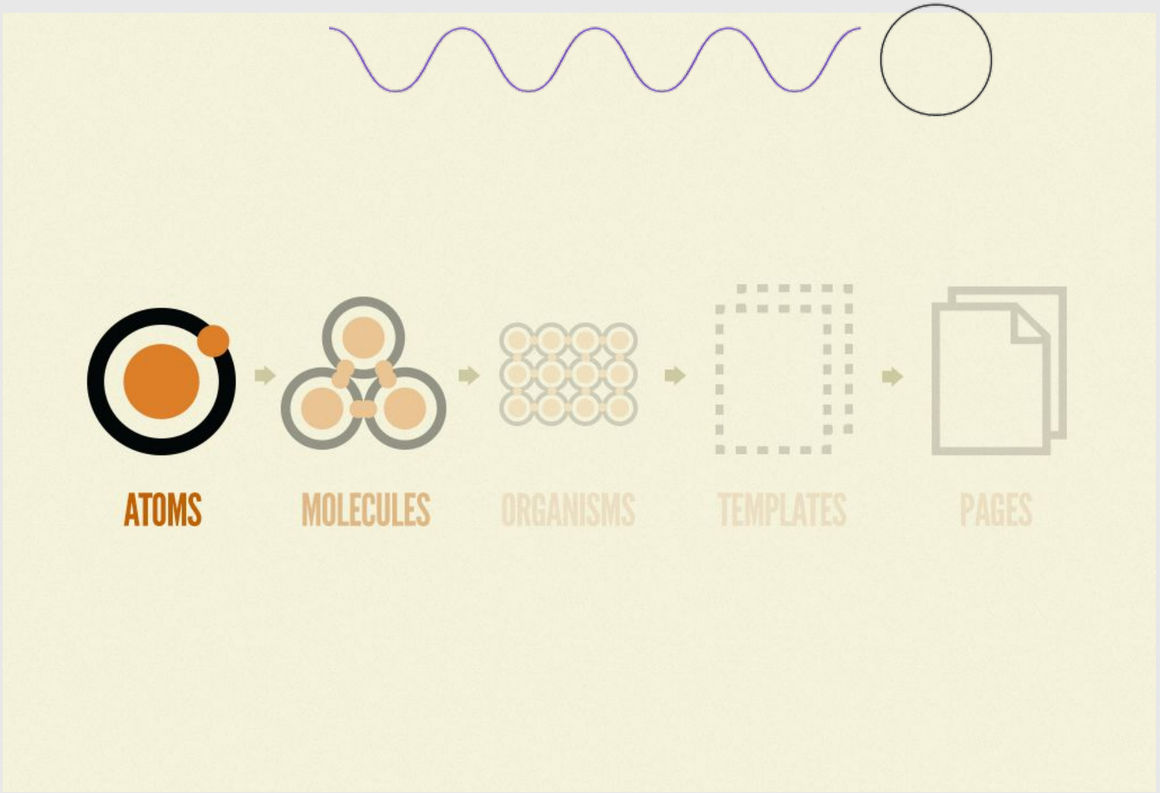


#PraCegoVer: Imagem ilustrando a divisão hierárquica do atomic design.

Átomos

E elementos ou blocos que formam a interface.

Exemplo: Elementos isolados, botões ou até mesmo inputs.

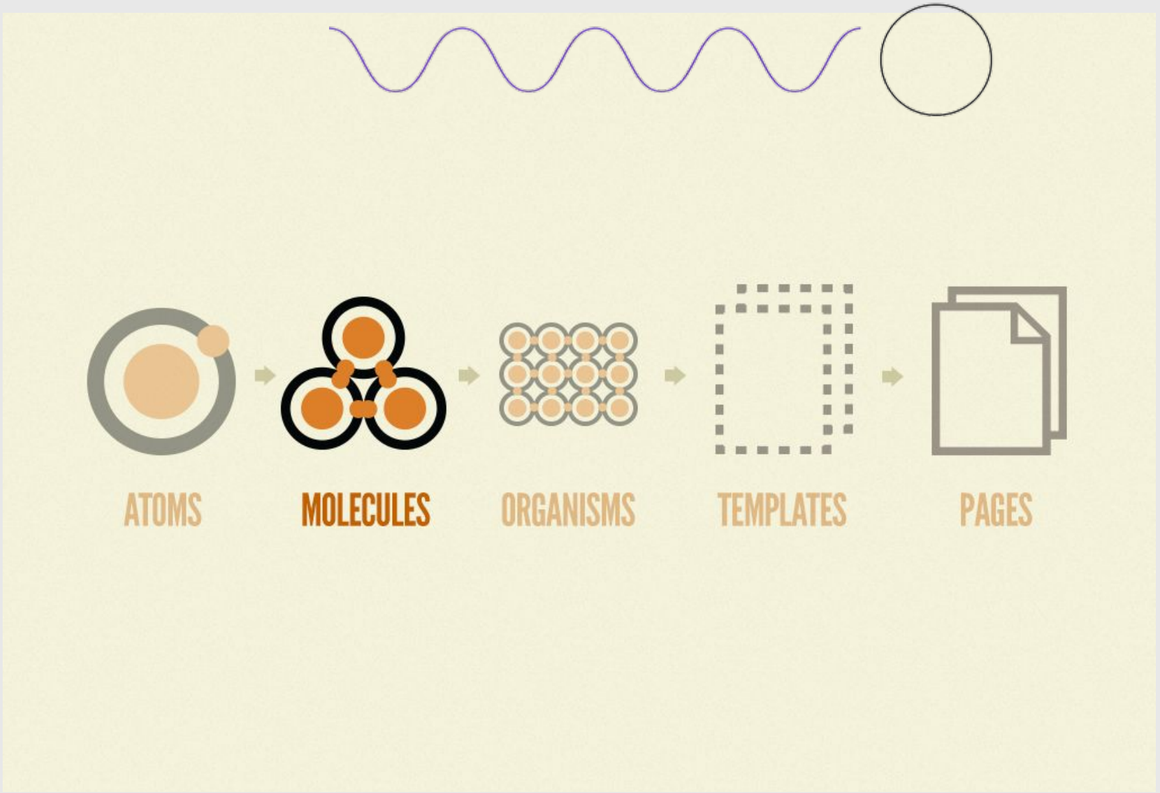


#PraCegoVer: Imagem ilustrando a divisão hierárquica do atomic design o primeiro item (átomo).

Moléculas

Grupo de elementos que formam de interface e funcionam de maneira agrupada.

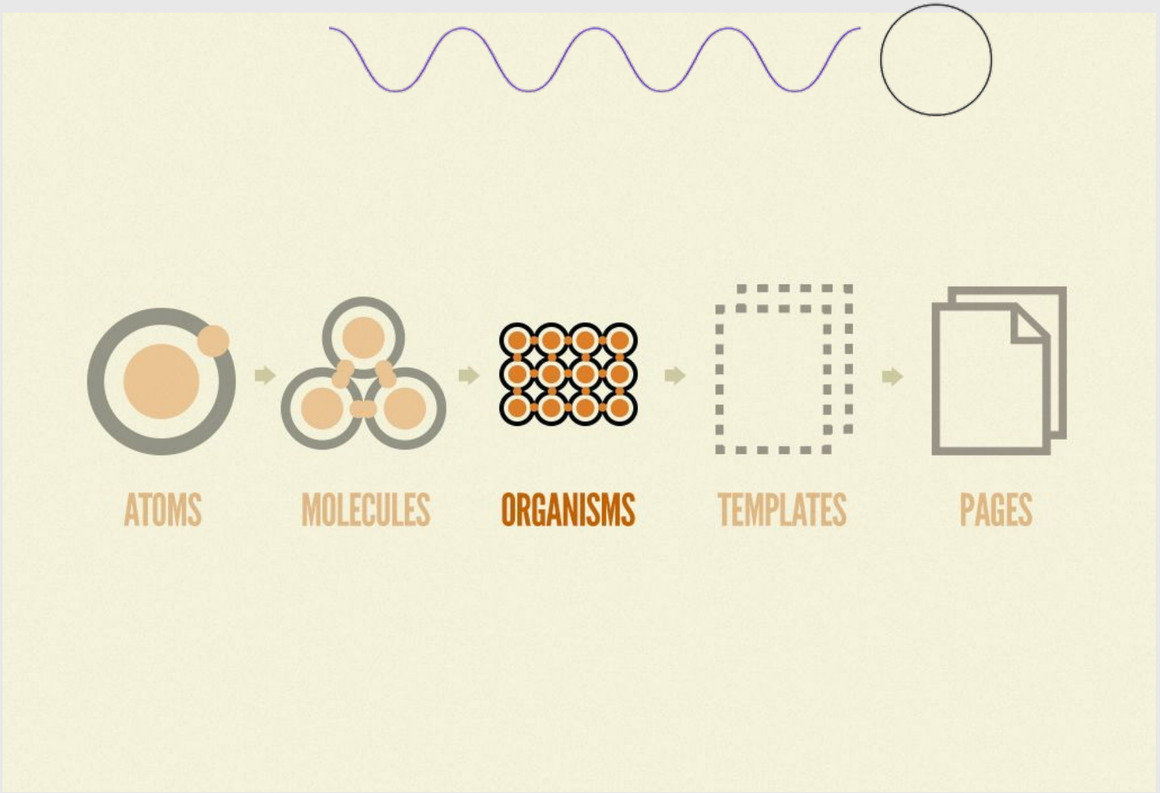
Exemplo: um formulário composto por inputs e botões.



#PraCegoVer: Imagem ilustrando a divisão hierárquica do atomic design.

Organismos

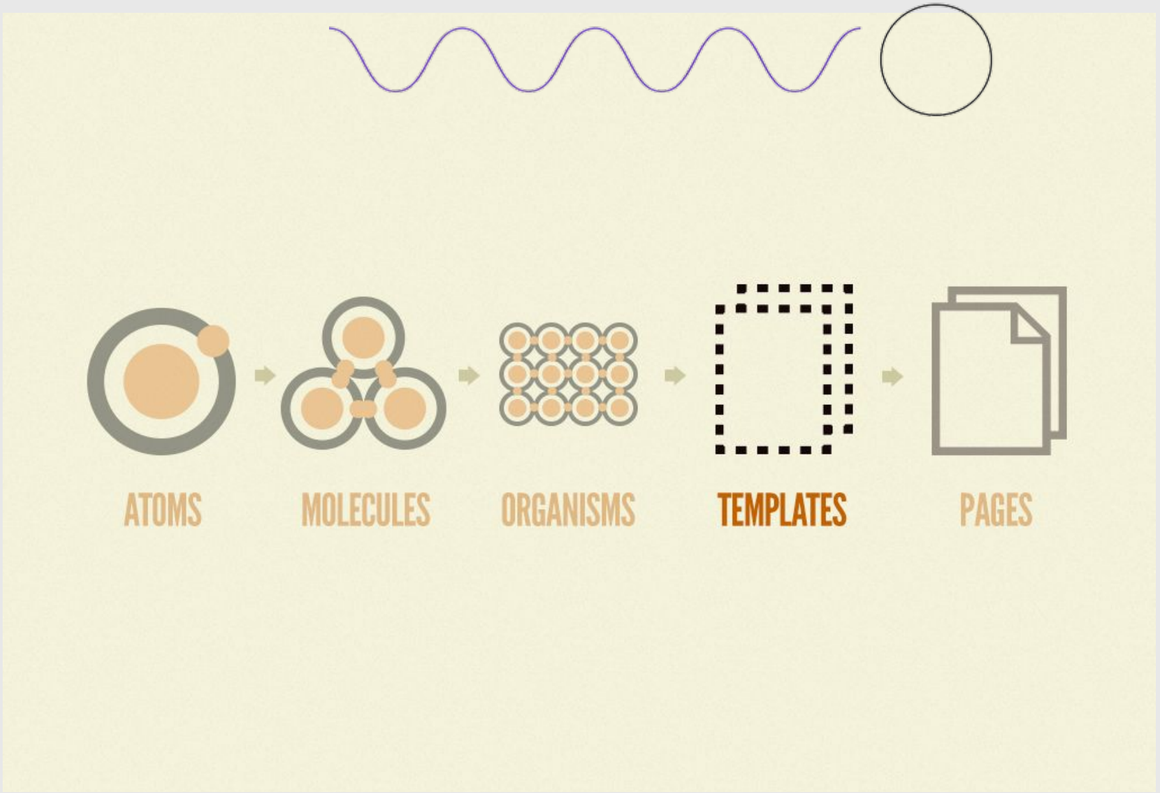
Conjunto de moléculas que tratam um contexto de interface.



#PraCegoVer: Imagem ilustrando a divisão hierárquica do atomic design.

Templates

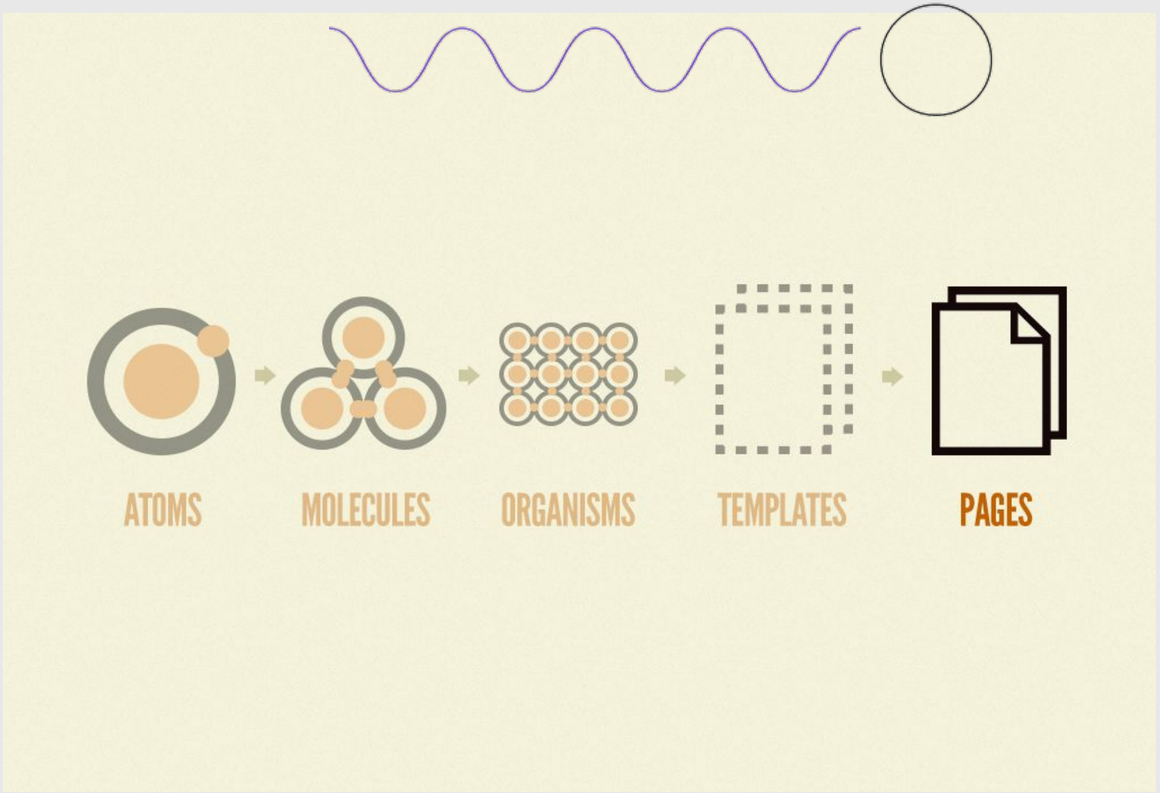
São elementos do nível de páginas onde no formato de componentes formando a estrutura de páginas.



#PraCegoVer: Imagem ilustrando a divisão hierárquica do atomic design.

Páginas

E o conjunto dos elementos mencionados formando o resultado final.



#PraCegoVer: Imagem ilustrando a divisão hierárquica do atomic design.

Atomic Design

Eficiente e fácil de compor o Atomic design é um aliado poderoso na construção de interfaces.

Até aqui entendemos como compor e até mesmo como organizar os nossos projetos utilizando este método.

Agora é hora de colocar em prática, até o próximo módulo.

#PraCegoVer: Dois notebooks o primeiro apresentado o teclado o segundo apresentando o monitor.

Design System

Vamos entender um pouco mais sobre os tipos

Introdução

Antes de qualquer coisa, o **Design System** é um grande aliado quando o assunto é estruturar produtos digitais, atuar com gestão e padronização de elementos para escalar e atualizar.

O uso do Design System não se limita apenas a elementos web, mas se estende a elementos Mobile podendo ser agnóstico de frameworks ou até mesmo linguagens de programação.

Vamos entender um pouco mais sobre este assunto.

Mais sobre Design System

Temos diversos produtos e soluções que fazem uso deste método.

Mas no final das contas o que é o Design System?

Trata-se de uma documentação composta por elementos e informações associadas a marca onde o mesmo deve ser implementado, ou seja tokens com medidas de espaçamento, cores e tom de de voz da marca. Onde se aplica até mesmo a bibliotecas e padronização de componentes.

A seguir vamos entender um pouco mais deste assunto visualizando um case que se tornou referência no mercado.

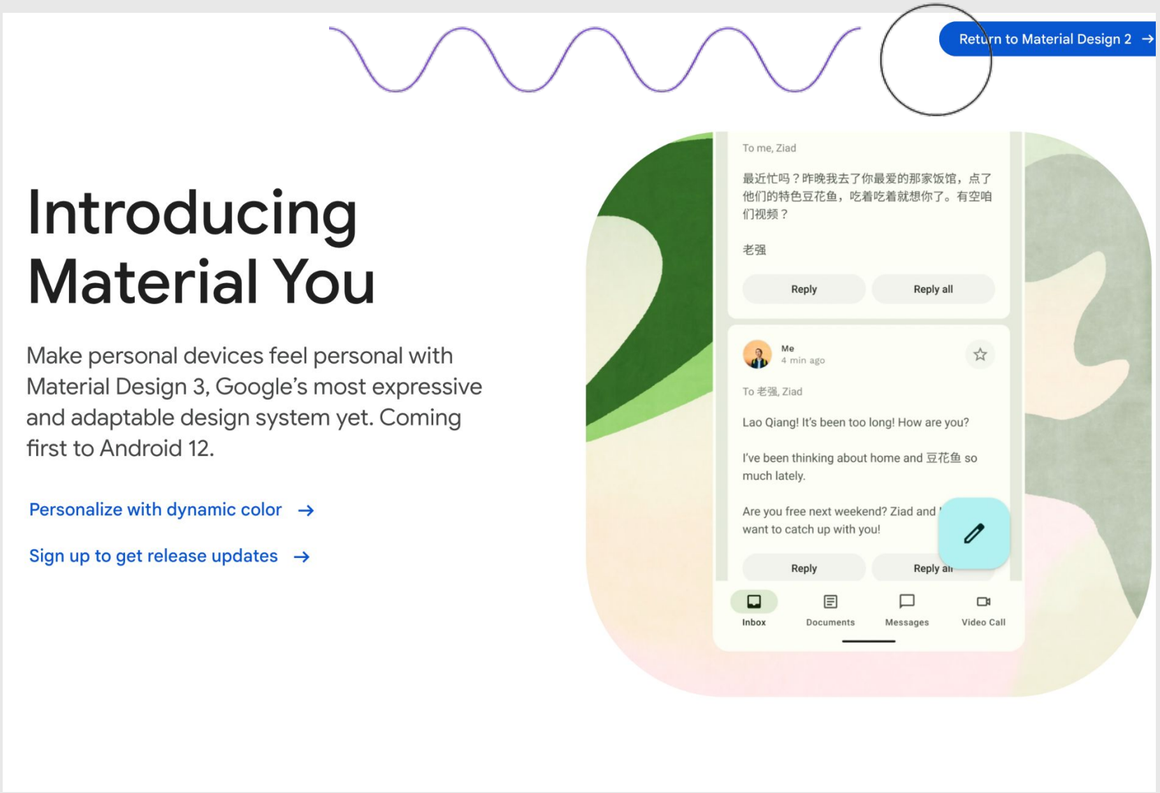
Material Design

Um dos cases mais utilizados do mercado é o Material Design. Criado e mantido pelo Google o mesmo é utilizado em todos os seus produtos e atualmente adotado por muitas outras empresas.

Mas de fato o que é o Material Design?
Uma biblioteca Disponível para os frameworks mais utilizados do mercado que possibilita a utilização de elementos padronizados.

Ex: Botões, Inputs, Cards, Tooltips e muitos outros elementos.

Todos os estes elementos estão devidamente encapsulados nesta biblioteca facilitando o uso e escala em produtividade.



#PraCegoVer: imagem com site oficial do Material Design.

Design System

Design System é um assunto extenso e cheio de nuances, neste módulo tivemos uma breve introdução aos principais conceitos e um dos casos de uso mais conhecidos do mercado.

Caso este assunto tenha te chamado atenção, e queira entender um pouco mais sobre o Material Design, acesse aqui o site oficial e fique por dentro do assunto, clique aqui.

#PraCegoVer: Dois notebooks o primeiro apresentado o teclado o segundo apresentando o monitor.

Micro Frontend

Vamos entender um pouco mais sobre os tipos

Introdução

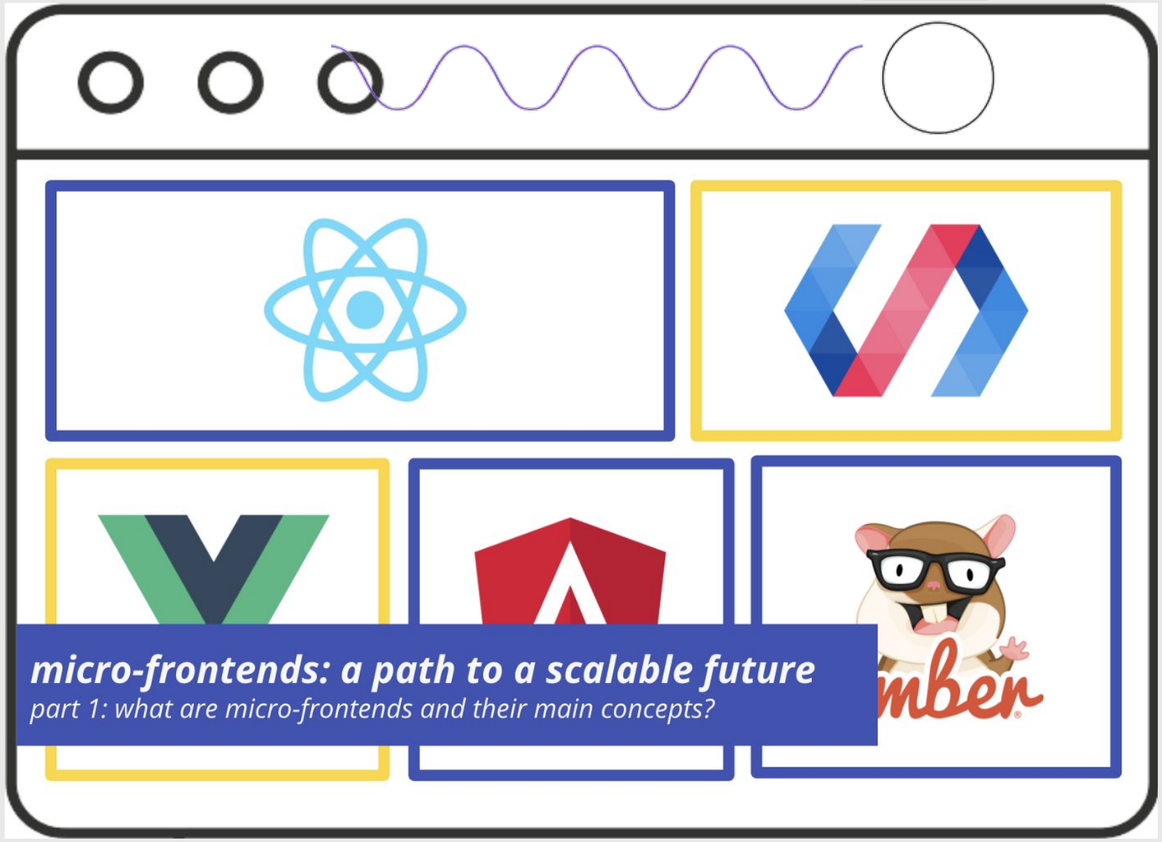
Falando sobre os métodos e conceitos mais atuais de desenvolvimento frontend, o Micro Frontend é sem sombra de dúvidas um dos assuntos mais mencionados do mercado.

Mas de fato o que é o **Micro Frontend**?
Ele é um método onde é possível mesclar frameworks e tecnologias em uma mesma aplicação, possibilitando e auxiliando trabalho de times em features ou até mesmo funcionalidades de produtos de maneira isolada e contida, vamos entender mais exemplos a seguir.

Micro frontend

Ao lado temos uma imagem ilustrando a composição de uma página utilizando libs e frameworks distintos na mesma página. Exatamente esta e a proposta do **Micro Frontend** oferecer maior autonomia para o desenvolvimento e escala por squads (formação de equipes para atuar com desenvolvimento).

São infinitas a possibilidades para uso deste formato, tanto na entrega de melhorias quanto na evolução de cada feature, imagine que no ato de publicar cada atualização os módulos são completamente isolados, ou seja não afetam o funcionamento dos demais.



#PraCegoVer: Ilustração de funcionalidade de micro frontend possibilitando a união de diversos frameworks.

Micro Frontend

Aprendemos sobre um conceito bastante relevante nos dias de hoje, e quer saber como implementar este solução?

Vou compartilhar duas das soluções mais utilizadas do mercado para você aprender mais sobre o assunto:

Webpack: Module federation, clique aqui para ler a documentação oficial.
Hydra Micro Service, clique aqui para acessar a documentação oficial.

#PraCegoVer: Dois notebooks
o primeiro apresentado o teclado
o segundo apresentando o
monitor.

MVC

Model View e Controller

MVC

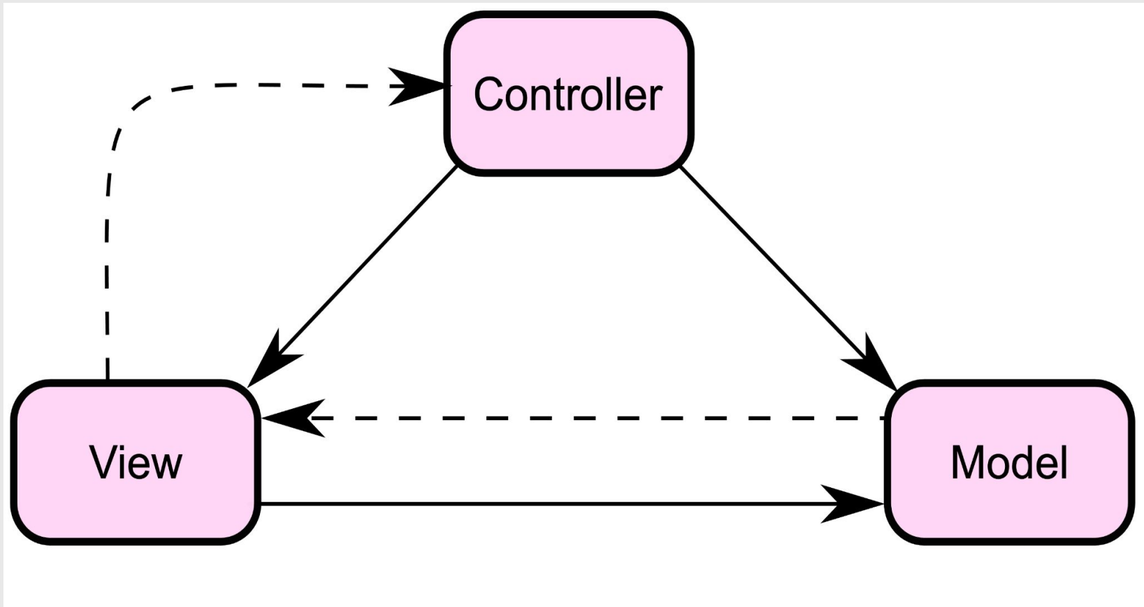
Antes de qualquer coisa, MVC é uma arquitetura de solução que tem por conceito padrão a divisão do serviço em 3 camadas. **MODEL, VIEW e CONTROLLER**.

Ele foi formulado por volta da década de 70 e seus conceitos são utilizados até hoje, por ser um grande facilitador na divisão de responsabilidades de uma aplicação.

O seu uso tem como premissa, dividir níveis interconectados como interação com banco de dados e respostas para os usuários.

Diagrama

Este diagrama tem como objetivo expor de maneira simples as relações entre as camadas, trazendo um pouco mais de clareza quanto ao conceito como um todo.



#PraCegoVer: Diagrama de solução arquitetural MVC.

MVC

Neste módulo tivemos uma breve introdução ao MVC e agora é hora de começar a entender um pouco mais a fundo esta arquitetura.

Vou compartilhar um artigo na Wikipedia onde a solução, conceitos, abordagem e até mesmo a história em torno deste método é compartilhada, clique aqui para saber mais.

#PraCegoVer: Dois notebooks
o primeiro apresentado o teclado
o segundo apresentando o
monitor.

MVVM

Model View View Model

Introdução

O **MVVM** Model-View-viewmodel é um padrão arquitetural de desenvolvimento que auxilia no separação desenvolvimento da interface gráfica do usuário sendo por meio de linguagens de marcação ou até mesmo por código de toda a camada de lógica.

Desenvolvido pela Microsoft por volta de 2005 trata-se de uma variação do padrão de projeto Presentation Model e arrisco em dizer que este padrão "é o primo mais novo do **MVC**".

Este padrão é utilizado por diversos frameworks, tanto no desenvolvimento mobile, quanto no desenvolvimento WEB:

Mobile: Java, Swiftkl, Flutter, Xamarin Forms... Entre outros.

WEB: Angular, VueJS, Ember... Entre outros.

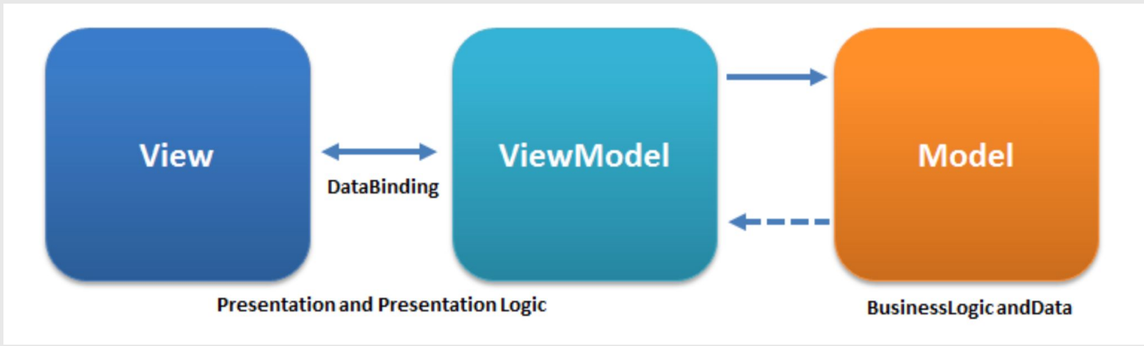
Diagrama

Conforme o diagrama ao lado, podemos enxergar o conceito de maneira simples.

Principal conceito por de trás desta solução é o desacoplamento de de código trazendo muitos benefícios para o desenvolvimento.

Componentes do padrão **MVVM**:

Modelo
Visualizador
Ver modelo
Encadeador



#PraCegoVer: Diagrama de solução arquitetural MVVM.

MVVM

Neste módulo tivemos um introdução ao MVVM e desta e não para por aqui, caso queira entender mais a fundo este padrão arquitetural, você pode visualizar mais, acessando a documentação oficial fornecida pela Microsoft, clicando aqui.

#PraCegoVer: Dois notebooks o primeiro apresentado o teclado o segundo apresentando o monitor.

Design patterns

Padrões de código e organização de projeto.

Introdução

Design patterns são padrões de desenho ou padrões de projeto.

Para ser mais claro, um padrão de projeto encapsula alguns parâmetros importantes, sendo assim é composto por 4 elementos:

Nome do padrão;
Problema a ser resolvido;
Solução fornecida pelo padrão;
Consequência;

Por tanto os padrões de projetos (**Design Patterns**) tem como objetivo:
Facilitar a reutilização de soluções (códigos ou afins);
Estabelecer um vocabulário entre o desenho e solução;

Design Patterns

Neste módulo aprendemos sobre Design Patterns e o conceito por de trás desta solução.

Em nossa jornada falamos um pouco sobre padrões específicos e de maneira isolada, e agora é hora de colocar em prática o aprendizado.

Fechamento

Sobre o nosso aprendizado.

Agora é hora de colocar em prática de maneira organizada o nosso aprendizado com boas práticas, vamos à diante e não deixe este conhecimento tão importante cair no esquecimento ein.

Siga-nos no Linkedin clique aqui.

E os estudos não param por aqui.

Aproveite ao máximo para praticar lembrando que...

"a repetição sem exaustão leva a perfeição" - Autor desconhecido.

Prof Douglas Morais

Referência Bibliográfica

GETBEM, Documentação oficial - BEM CSS - Site oficial.

Brad Frost, Documentação oficial - Atomic Design. Site oficial.

MEDIUM, Khachatur Tovmassian, Ilustração de imagem Micro frontends.

Wikipedia, Autor Desconhecido. MVC Diagrama de solução.

Microsoft, Documentação oficial - MVVM - Site oficial.