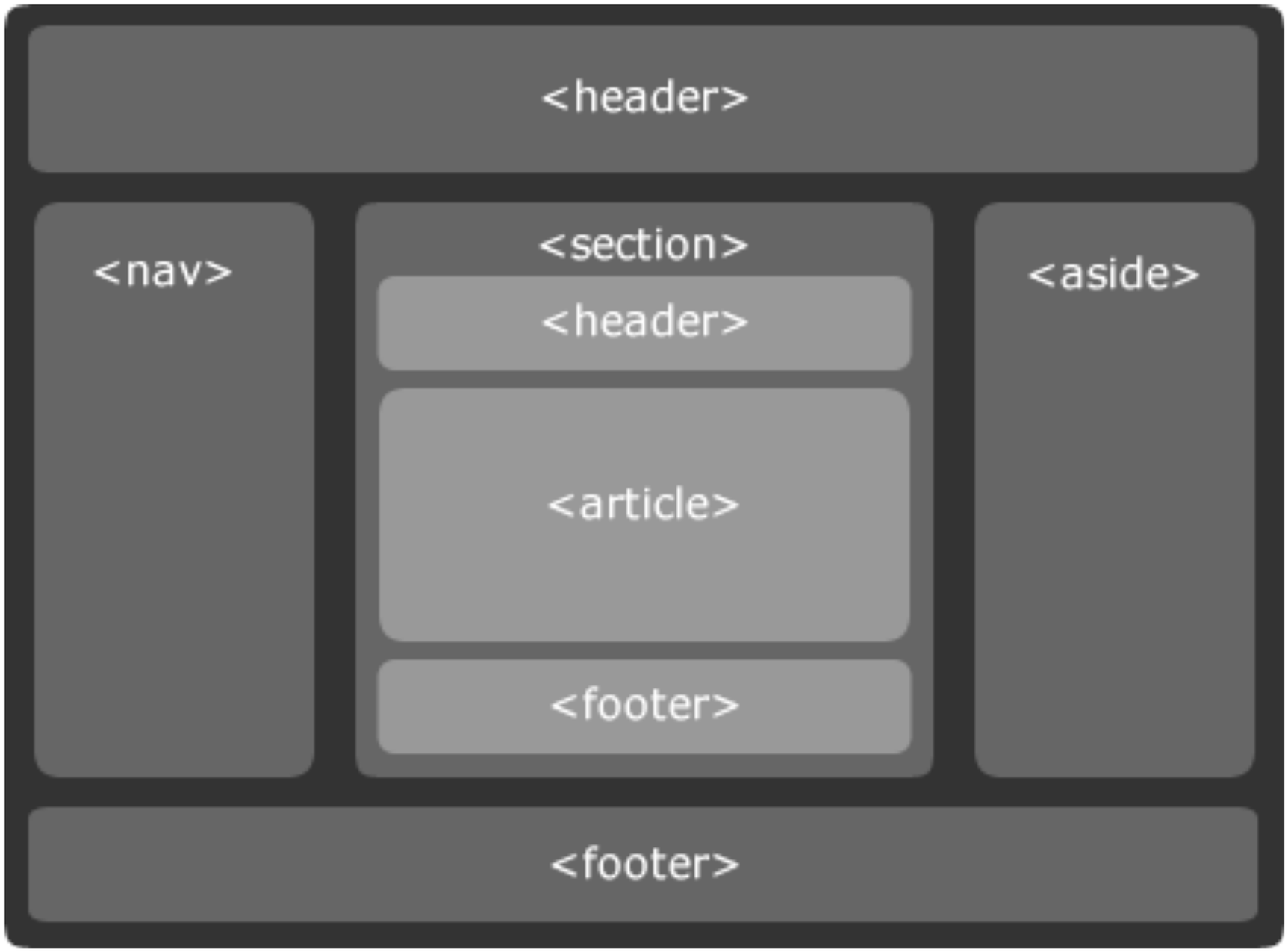
Web Design

Cássio Seffrin 2020

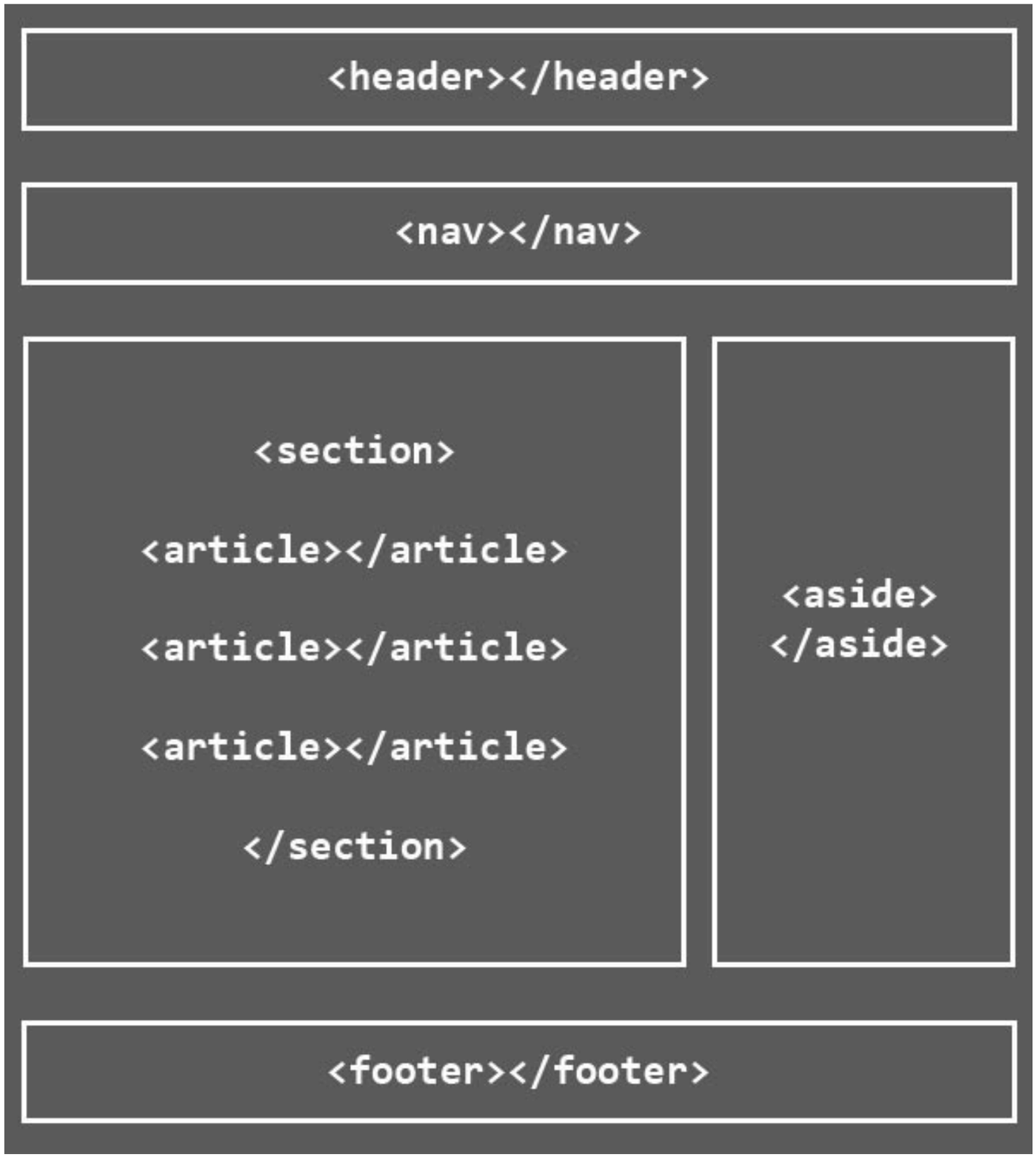
https://github.com/cassioseffrin/webdesign

https://www.youtube.com/watch?v=9hIQjrMHTv4

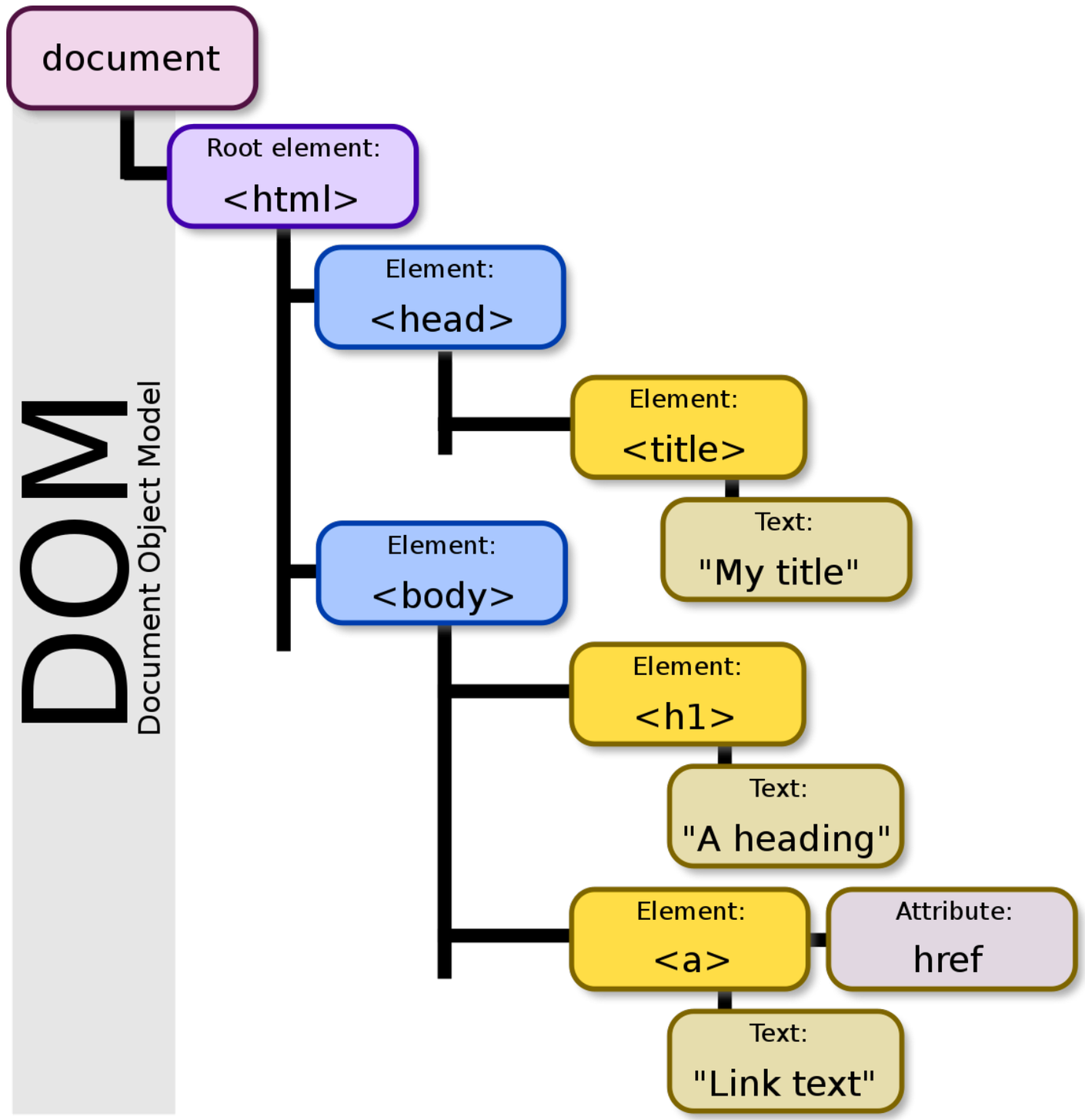
Semântica HTML5



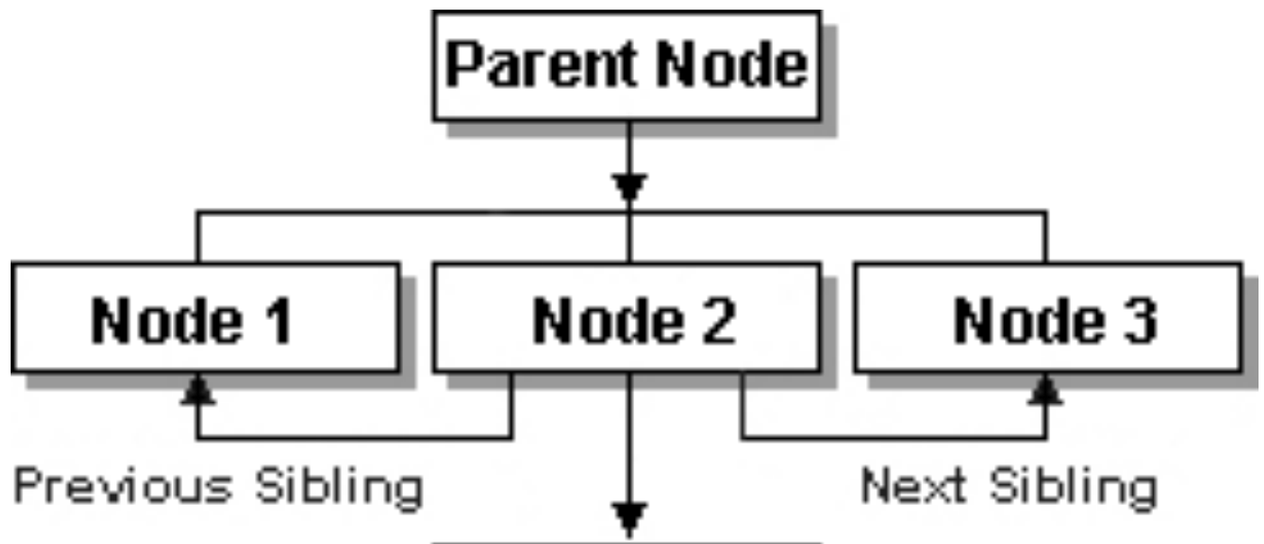
Semântica HTML5

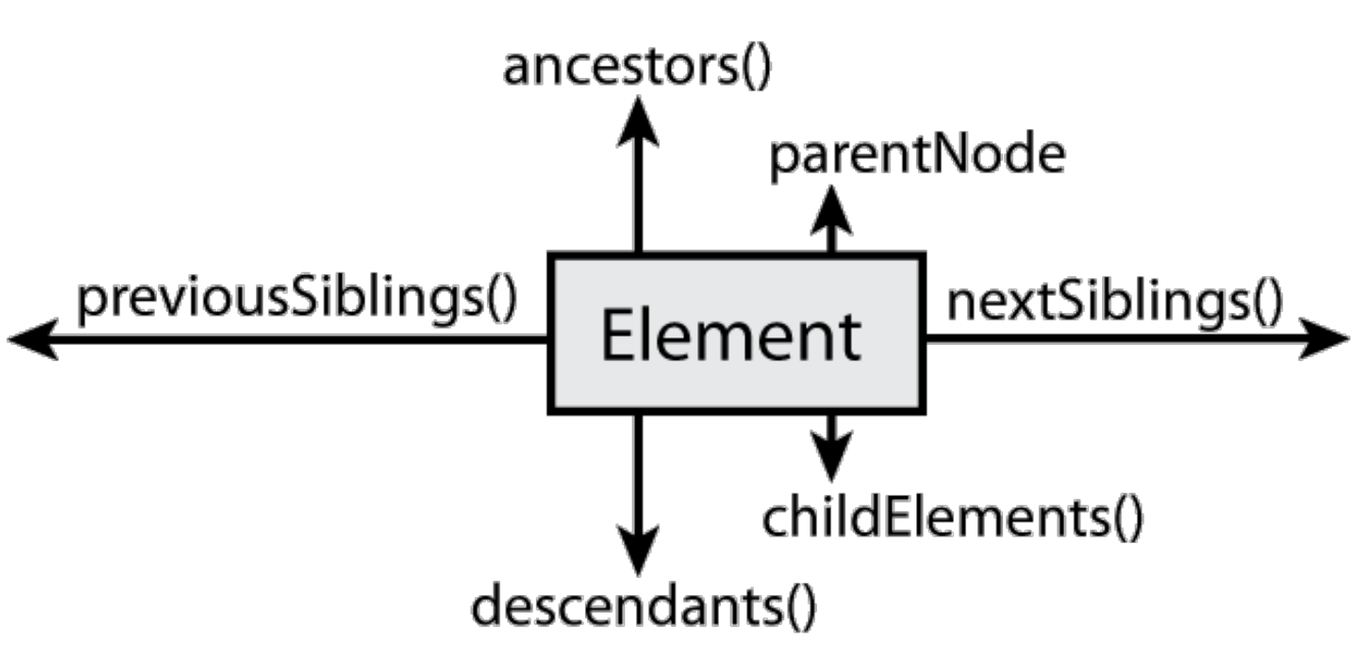


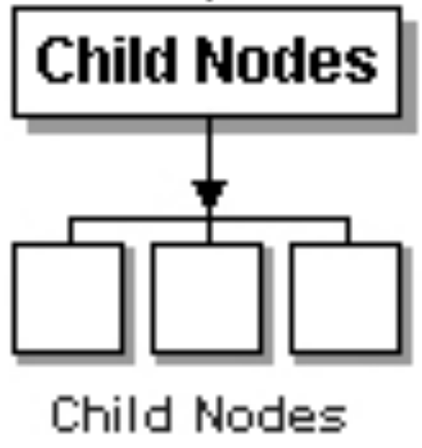
DOM - Document Object Model



DOM - Document Object Model







#action-menu-1-menu > a:nth-child(3) {display: none;}

DOM - Document Object Model

Representa os objetos de uma página HTML
em forma de uma árvore.

O DOM é importante pois é através dele que acessamos os
elementos dentro de uma página HTML.

Em outra palavras, obtemos acesso ao documento HTML através do DOM.

HTML5

Layout
<article>: Define um artigo;
<aside>: Define o conteúdo além do conteúdo da página;
<embed>: Define o conteúdo interativo ou plugin externo;
<figcaption>: Define o caption de uma imagem;
<figure>: Define um grupo de média e seus captions;
<footer>: Define o rodapé de uma página;
<header>: Define o cabeçalho de uma página;
<nav>: Define os links de navegação;
<section>: Define uma área ou seção;
<wbr>: Define uma possível quebra de linha;

Media
<audio>: Define o conteúdo de som;
<source>: Define recursos de mídia;
<video>: Define um vídeo;

HTML5 - Tags

Aplicativos Web

<canvas>: Define gráficos;
<command>: Define um botão de comando;
<datagrid>: Referências aos dados dinâmicos em Tree View ou tabelas;
<datalist>: Define uma lista suspensa (DropDown);
<details>: Define detalhes de um elemento;
<output>: Define os tipos de saída (outputs);
<progress>: Define o progresso de uma tarefa qualquer;

Outros

<dialog>: Define uma conversa ou pessoas falando;
<hgroup>: Define informações sobre uma determinada área do documento;
<keygen>: Define a key (chave) do formulário;
<mark>: Define a marcação de um texto;
<meter>: Define a medição dentro de um intervalo pré-definido;
<summary>: Define o cabeçalho de dados “detalhe”;
<time>: Define uma data ou hora;

HTML 5 - Novos Atributos e Eventos

Atributos Globais
contenteditable - especifica se o usuário está autorizado a editar um conteúdo ou não
contextmenu - especifica um menu contexto para um elemento. menu_id.
draggable - especifica se um usuário tem permissão para arrastar um elemento.
dropzone - especifica o que acontece quando um dado arrastado é solto.
hidden - especifica que o elemento não é relevante: hidden (oculto).
spellcheck - especifica se o elemento deve ter sua grafia verificada:
Eventos

Janelas:
onafterprint - executa após o documento ser impresso.
onbeforeprint - executa antes do documento ser impresso.
onbeforeonload - executa antes do documento ser carregado.
onerror - executa quando ocorre um erro.
onhaschange - executa quando o documento sofre alteração.
onmessage - executa quando uma mensagem é disparada.
onoffline - executa quando o documento é desconectado da internet.
ononline - executa quando o documento é conectado à internet.
onpagehide - executa quando a janela é ocultada.
onpageshow - executa quando a janela se torna visível.
onpopstate - executa quando ocorre alteração no histórico da janela.
onredo- executa quando é acionado o comando de repetir.
onresize - executa quando a janela tem alteração de tamanho.
onstorage - executa quando um documento é carregado.
onundo - executa quando é acionado o comando de desfazer.
onunload - executa quando o usuário sai do documento.

HTML 5 - Novos Atributos e Eventos

Formulários:
oncontextmenu - executa quando um menu de contexto é acionado.
onformchange - executa quando ocorre alterações no formulário.
onforminput - executa quando o usuário dá entrada no formulário.
oninput - executa quando um elemento dá entrada do usuário no formulário.
oninvalid - executa quando um elemento não é válido.

Mouse:
ondrag - executa quando um elemento é arrastado.
ondragend - executa ao fim de uma operação de arrastar um elemento.
ondragenter - executa quando um elemento é arrastado e solto em seu destino.
ondragleave - executa quando um elemento é solto em um destino válido.
ondragover - executa quando elemento é arrastado e solto ao longo de um destino.
ondragstart - executa quando se inicia uma operação de arrastar.
ondrop - executa quando o elemento arrastado está sendo descartado.
onmousewheel - executa quando o scroll do mouse é girado.
onscroll - executa quando as barras de rolagem de um elemento está sendo rolada.

HTML 5 - Novos Atributos e Eventos

Multimídia:
oncanplay - executa quando uma mídia está sendo iniciada a tocar.
onclanplaythrought - executa quando a mídia está sendo tocada até o fim.
ondurationchange - executa quando o comprimento da mídia é alterado.
onemptied - executado quando um elemento de recursos de mídia torna-se vazio.
onended - executa quando a mídia chega ao fim.
onerror - executa quando ocorre um erro de carregamento de um elemento.
onloadeddata - executa quando os dados de mídia são carregados.
onloadedmetadata - executa quando a duração de um elemento de mídia está sendo carregado.
onloadstart - executa quando o navegador começa a carregar os dados de mídia.
onpause - executa quando a mídia de dados está em pausa.
onplay - executa quando a mídia de dados for começar a tocar.
onplaying - executa quando a mídia começa a tocar.
onprogress - executa quando o navegador está buscando os dados de mídia.
onratechange - executa quando altera a faixa de mídia .
onreadystatechange - executa quando ocorre uma mudança de estado.
onseeked - executa quando o atributo de busca de um elemento não é verdadeiro.
onseeking - executa quando o atributo de busca de um elemento é verdadeiro.
onstalled - executa quando há um erro na busca de dados de mídia.
onsuspend - executa quando o navegador para de buscar os dados da mídia.
ontimeupdate - executa quando a posição da mídia é alterada.
onvolumechange - executar quando a mídia muda de volume e, também, quando o volume fica mudo.
onwaiting - executar quando a mídia para de tocar.

HTML 5 - Tags Descontinuadas:

<acronym> Define siglas em HTML 4.01. (Desenvolvedores preferem utilizar a tag <abbr>);
<applet> Define um miniaplicativo incorporado. (Ficou obsoleto em função da tag <object>);
<basefont> Define as propriedades da font padrão para todo o texto do documento. (Apenas efeito visual);
<big> Usado para tornar o texto maior. (Apenas efeito visual);
 Usado para alinhar texto e conteúdo no centro. (Apenas efeito visual);
<dir> Define a lista do diretório. (Ficou obsoleto em função da Tag <ul>);
<font> Especifica o tipo de fonte, tamanho, e cor do texto. (Apenas efeito visual);
<frame> Define uma janela particular dentro de um conjunto de "frames". (Fere princípios de usabilidade e acessibilidade);
<frameset> Define um conjunto de frames organizado por múltiplas janelas.(Fere princípios de usabilidade e acessibilidade);
<noframes> Texto exibido para navegadores que não lidam com "frames". (Fere princípios de usabilidade e acessibilidade);
<strike> Exibe texto rasurado. (Apenas efeito visual);
<tt> Define teletipo de texto. (Apenas efeito visual);
<u> Define sublinhado. (Apenas efeito visual);
<xmp> Define texto pré-formatado. (Ficou obsoleto em função da tag <pre>);

Javascript querySelectorAll()

Referência prática da função querySelectorAll()
var elem = document.querySelectorAll('seletor');
Este é o método definitivo de seleção de elemento: trata-se de uma técnica muito poderosa por meio da qual os programas JavaScript do lado do cliente podem selecionar os elementos do documento que vão manipular.

As folhas de estilos CSS possuem o que chamamos de seletores, um mecanismo para descrever elementos ou conjuntos de elementos dentro de um documento. Eis alguns poucos exemplos:

```
/* todos os parágrafos*/
p {}

/* todos os parágrafos e todos os títulos */
p, h1 {}

/* qualquer elemento com 'warning' em seu atributo 'class' */
.warning {}

/* um elemento com id="nav"*/
#nav {}
```

Javascript

/* qualquer elemento span com 'fatal' e 'error' em sua classe */
span.fatal.error {}
A funcao querySelectorAll() permite que os seletores CSS sejam usados no JavaScript, veja:
var elem = document.querySelectorAll('p');
var elem = document.querySelectorAll('p, h1');
var elem = document.querySelectorAll('.warning');
var elem = document.querySelectorAll('#nav');
var elem = document.querySelectorAll('span.fatal.error ‘);

A funcao querySelectorAll() recebe um argumento de string contendo um seletor CSS e retorna um objeto NodeList representando os elementos do documento que correspondem ao seletor.
Se nenhum elemento coincide, a função retorna um objeto NodeList vazio.
Se a string do seletor é invpalida, querySelectorAll() lançará um exceção.
O método querySelectorAll() faz parte tanto do objeto Document como do objeto Element.

Não é possível utilizar pseudoelementos como por exemplo: :first-line e first-letter.

Além de querySelectorAll(), temos a semelhante querySelector().
Esta última, retorna somente o primeiro (na ordem do documento) elemento coincidente ou null, caso não haja elementos correspondentes.

Todos os navegadores atuais suportam ambos os métodos,
Você poderá utilizar lançando mão de uma biblioteca como a jQuery, por exemplo.
Ela (a biblioteca) usa esse tipo de consulta baseada em seletor CSS como principal paradigma de programação. Os aplicativos Web baseados na jQuery utilizam um equivalente
de querySelectorAll() portável e independente de navegador chamado $().

A confusão: Javascript vs ECMAScript, ES6 e ES2015

ECMAScript é uma especificação de uma linguagem. JavaScript é uma implementação dessa linguagem.

Ex: A especificação da linguagem lida com conceitos abstratos, como "[[GetPrototypeOf]] slot interno", enquanto o JavaScript possui um método getPrototypeOf concreto.

É importante ter um entendimento claro do ECMAScript para que possamos desenvolver sites e aplicativos da Web independentes do navegador.

A confusão: Javascript vs ECMAScript, ES6 e ES2015

Existe uma enorme confusão na comunidade de desenvolvedores, se JavaScript e ECMAScript são os nomes diferentes para a mesma linguagem de script ou não. Alguns pensam que o JavaScript é realmente um subconjunto do ECMAScript, enquanto outros têm uma opinião diferente.

A verdade é que o JavaScript é uma linguagem de script criada para manter a especificação do ECMAScript em seu núcleo. O ECMAScript nada mais é que um padrão ou especificação definida para criar diferentes linguagens de script e uma delas é o JavaScript. Foi criado com o único objetivo de compatibilidade entre navegadores.

Importância da diferença entre JavaScript e ECMAScript - Quase todos os navegadores da web no mundo executam as linguagens de script baseadas no ECMAScript. Como linguagens de script como JavaScript não são baseadas apenas no ECMAScript e possuem algumas de suas próprias propriedades que não são definidas pela especificação do ECMAScript, sites ou aplicativos da Web criados com esses recursos extras podem funcionar mal dependendo da compatibilidade dos navegadores da Web com o JavaScript e seus recursos exclusivos . Em resumo, é necessário entender a diferença entre JavaScript e ECMAScript para executar um site confortavelmente em todos os navegadores.

A confusão: Javascript vs ECMAScript, ES6 e ES2015

A especificação não menciona nem o ES6 nem o ES2015, embora sejam abreviações úteis.

Flex Box

```
<div class="flex-container">
  <div style="flex-grow: 1">aula 1</div>
  <div style="flex: 1">aula 2</div>
  <div style="flex: 1">aula 3</div>
</div>
```

```
.flex-container {
  display: flex;
  background-color: grey;
  flex-direction: row;
}


.flex-container > div {
  background-color: white;
  margin: 10px;
  padding: 10px;
  font-size: 30px;
}
```

https://css-tricks.com/snippets/css/a-guide-to-flexbox/

Flex Box

As propriedades do contêiner flexível são:
flex-direction
flex-wrap
flex-flow
justify-content
align-items
align-content

As propriedades flex item são:
order
flex-grow
flex-shrink
flex-basis
flex
align-self

Flex Box

flex-wrap: wrap; wrap-reverse/ nowrap O valor de wrap especifica que os itens flexíveis serão quebrados, se necessário:

justify-content é usada para alinhar os itens flexíveis ex: justify-content: center;/flex-start/flex-end;
space-around; O valor de espaço ao redor exibe os itens flexíveis com espaço antes, entre e depois das linhas:

align-items é usada para alinhar os itens flexíveis verticalmente. Possui as mesmas propriedades do justify-content, porem
tem o stretch; O valor de estende os itens flexíveis para preencher o contêiner (padrão):
 align-items: baseline;
}

justify-content: space-between;O valor space-between exibe os itens flexíveis com espaço entre as linhas:

align-content A propriedade é usada para alinhar as linhas flexíveis.

https://codepen.io/cassioseffrin/pen/QWbqRjq

https://css-tricks.com/snippets/css/a-guide-to-flexbox/

Flex Box

A maioria das propriedades Flex:
display - Especifica o tipo de caixa usada para um elemento HTML
flex-direction - Especifica a direção dos itens flexíveis dentro de um contêiner flexível
justify-content Alinha horizontalmente os itens flexíveis quando os itens não usam todo o espaço disponível no eixo principal
align-items Alinha verticalmente os itens flexíveis quando os itens não usam todo o espaço disponível no eixo transversal
flex-wrap Especifica se os itens flexíveis devem ser agrupados ou não, se não houver espaço suficiente para eles em uma linha flexível
align-content Modifica o comportamento da propriedade flex-wrap. É semelhante a alinhar itens, mas em vez de alinhar itens flexíveis, alinha linhas flexíveis
flex-flow Uma propriedade abreviada para flex-direction e flex-wrap
order Especifica a ordem de um item flexível em relação ao restante dos itens flex dentro do mesmo contêiner.
align-self Usado em itens flexíveis. Substitui a propriedade de itens de alinhamento do contêiner
flex Uma propriedade abreviada para as propriedades flex-grow, flex-shrink e flex-base

https://css-tricks.com/snippets/css/a-guide-to-flexbox/

CSS e Sass

Sass é uma linguagem de script que é interpretada ou compilada em Cascading Style Sheets (CSS). Consiste em duas sintaxes. A sintaxe original, chamada de "sintaxe indentada", A sintaxe mais recente, "SCSS", usa formatação de bloco, como a de CSS. Esta usa chaves para designar blocos de código e ponto-e-vírgula para separar linhas dentro de um bloco.

Os arquivos com sintaxe de indentação e SCSS são tradicionalmente dados as extensões .sass e .scss.

A implementação oficial da Sass é open-source e codificada em Ruby; no entanto, existem outras implementações, incluindo PHP, e uma implementação de alto-desempenho em C chamada libSass. Há também uma implementação em Java

https://zurb.com/playground/motion-ui

Metodologias

Vanilla CSS

Tudo por conta própria
Media Queries
estilos, layouts

Component Frameworks

Foundation
Start here, build everywhere.

Bootstrap 4

componentes pre estilizados,
utilitarios / classes e

Utility Frameworks

Tailwind CSS

Faça seu próprio estilo e layout com
ajuda de utilitários e classes

Metodologias
Comparações

CSS Puro / Vanilla CSS

- Controle total
- Sem códigos desnecessários Nome Classes como você gosta
- Crie tudo, desde o zero
- pouco controle
- Perigo de "código incorreto"

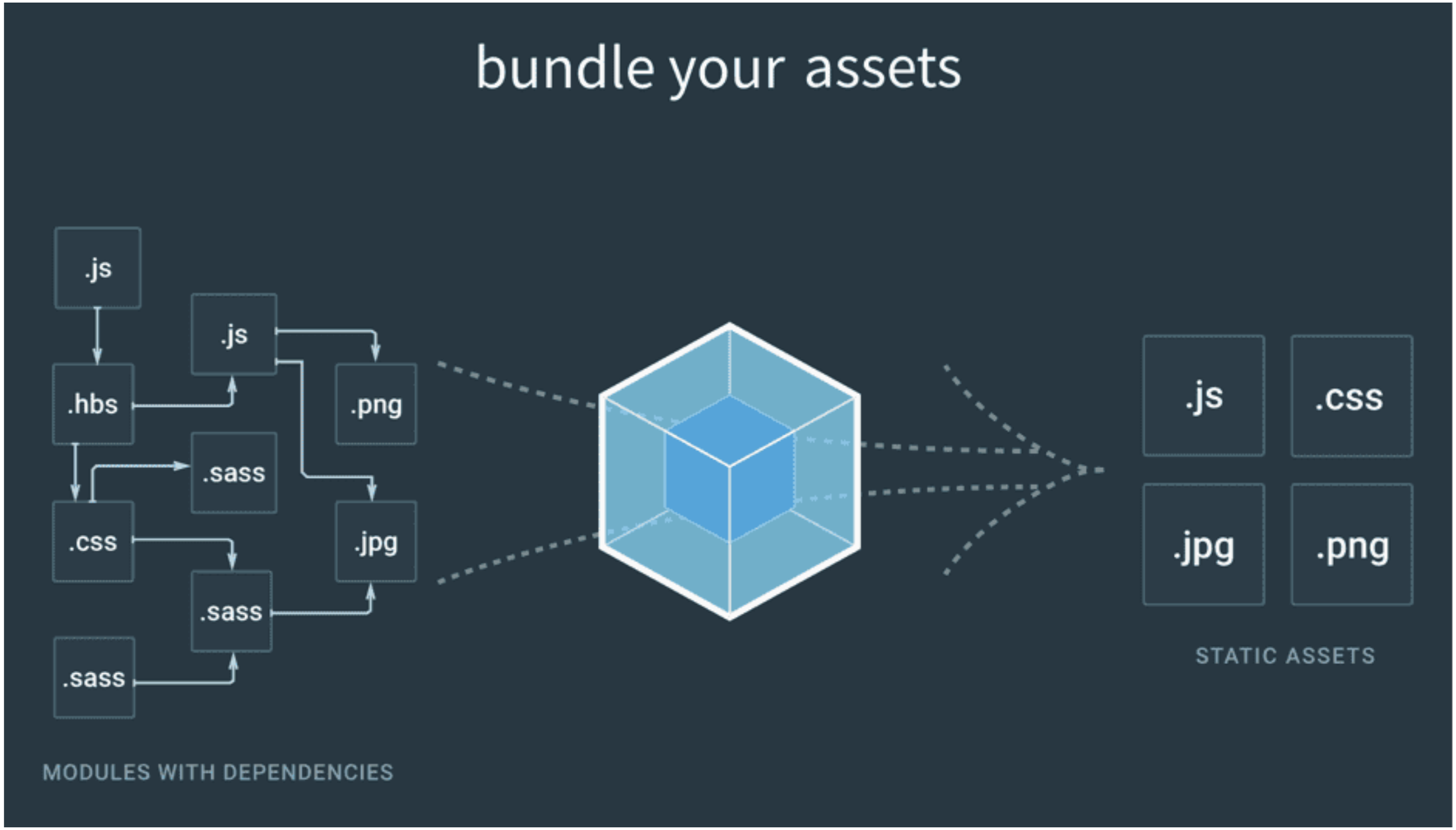
Bootstrap

- Desenvolvimento rápido
- Seguir práticas recomendadas
- Nenhum conhecimento especializado necessário
- Código indireto desnecessário
- Todos os sites têm a mesma aparência

Tailwind CSS

- Desenvolvimento rápido
- Seguir práticas recomendadas
- Não precisa ser um especialista
- Código indireto desnecessário

Web Pack - https://webpack.js.org/



Web Pack - https://webpack.js.org/

mkdir nomeprojeto

npm init -y

npm i rxjs webpack webpack-dev-server typescript ts-loader

npm i webpack-cli --dev

adicionar no package.json:

"scripts": {

    "start" : "webpack-dev-server --mode development",

criar :

webpack.config.js

tsconfig.json

index.html !

npm start

Web Pack - https://webpack.js.org/

npm install --save esm

O Node (CommonJS) ainda não suporta importações do ES6 (ES Modules).

const express = require('express')

const utils = require('./utils')

A solução chama-se esm, o carregador de módulo JavaScript brilhantemente simples, sem babel e sem pacote.

Instale o esm em seu projeto: npm install --save esm

Execute seu código com esm: node -r esm arquivo.js

E deve funcionar sem nenhuma modificação de código.

Também é possível usando o babel.

Web Pack - https://webpack.js.org/

3 artigos sobre webpack:

https://www.webdevdrops.com/webpack-sem-medo-introducao-af889eb659e7/

Programação Reativa

Um pouco de contexto…

Programação reativa no contexto de desenvolvimento de software foi citado pela primeira vez por Gérard Berry no artigo “Real time programming : special purpose or general purpose languages”. https://hal.inria.fr/inria-00075494/document

Rx Microsoft foi lançado em novembro de 2009 para .NET & Silverlight. Hoje, porém, já existem Rx para várias linguagens como: Java, Javascript, Ruby, Kotlin e mais…

Programação Reativa

"Paradigma baseada em fluxo de dados assíncronos e na propagação de mudanças nesse fluxo.”

Princípios da programação reativa:

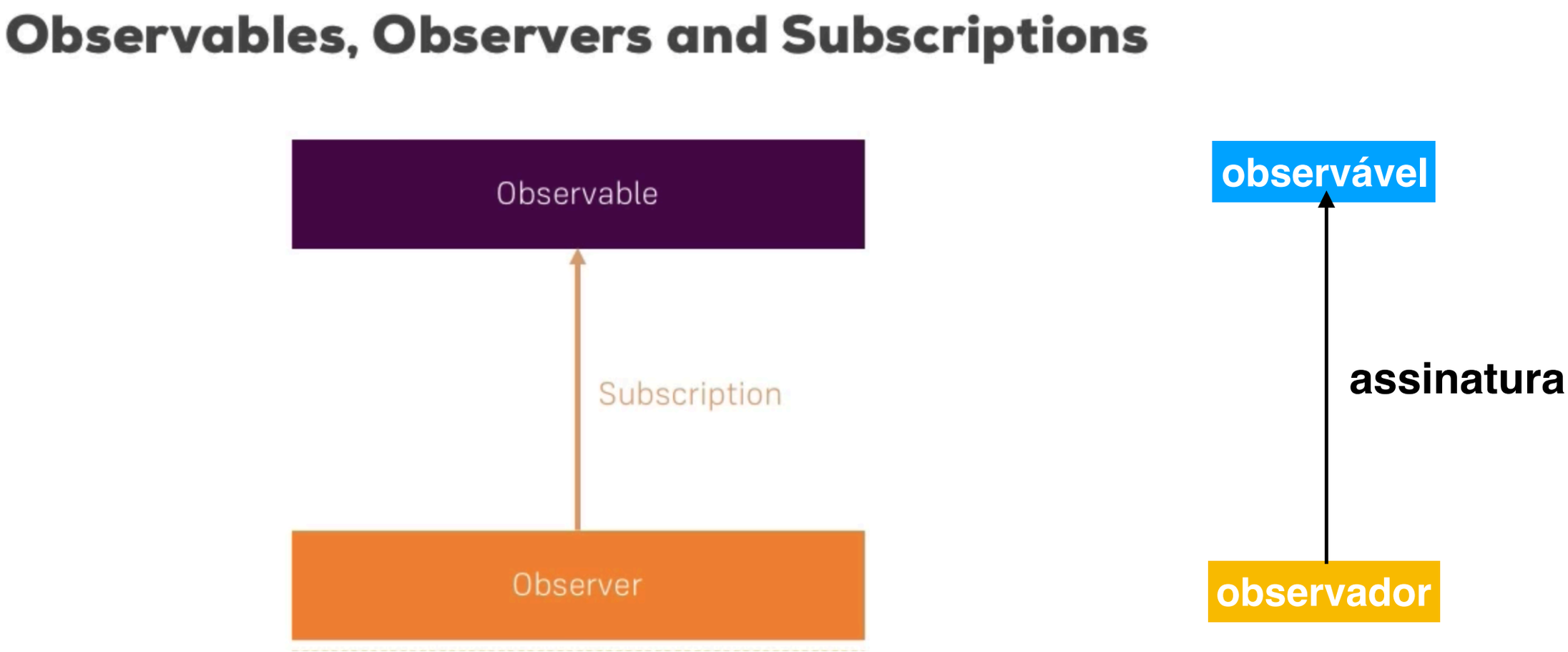
- Elástica: Reage à demanda/carga: aplicações podem fazer uso de múltiplos núcleos e múltiplos servidores;

- Resiliente: Reage às falhas; aplicações reagem e se recuperam de falhas de software, hardware e de conectividade;

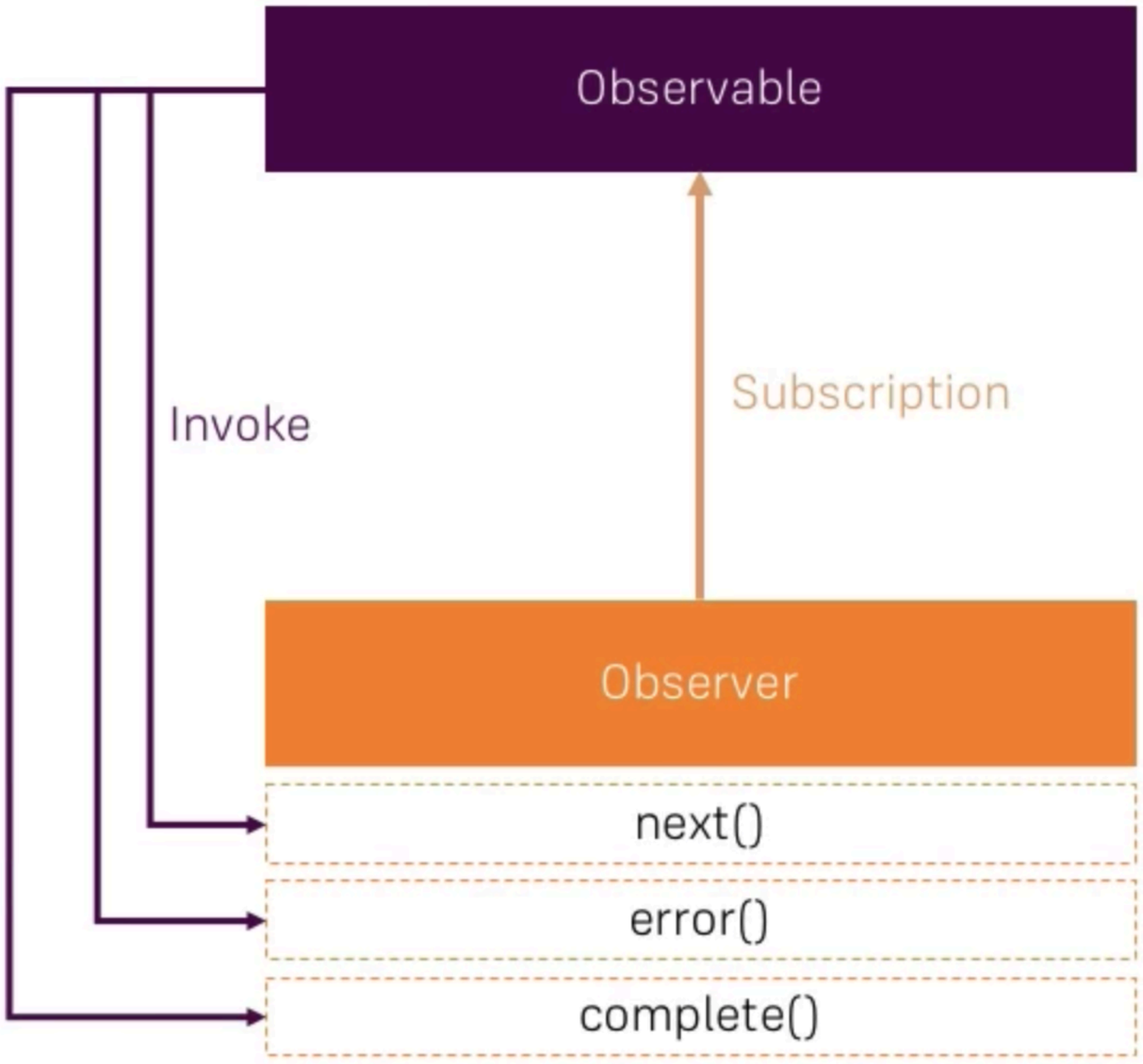
- Message Driven: Reage aos eventos (event driven): em vez de compor aplicações por múltiplas threads síncronas, sistemas são compostos de gerenciadores de eventos assíncronos e não bloqueantes;

- Responsiva: Reage aos usuários: aplicações que oferecem interações ricas e "tempo real" com usuários.

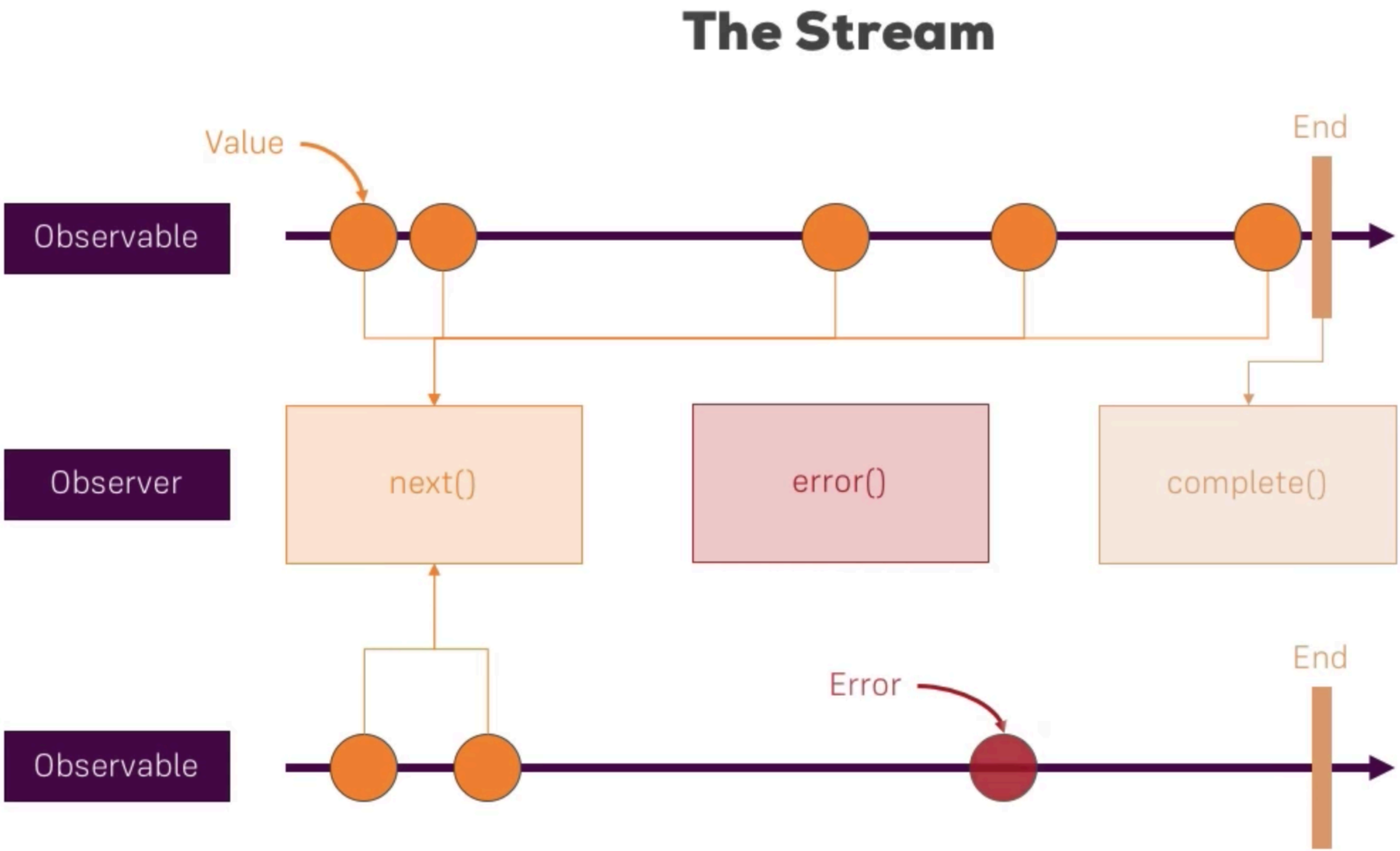
Programação Reativa



Programação Reativa



Programação Reativa



O Node ainda não suporta importações modulares do ES6.

esm

The brilliantly simple, babel-less, bundle-less ECMAScript module loader.

> `esm` is the world's most advanced ECMAScript module loader. This fast, production ready, zero dependency loader is all you need to support ECMAScript modules in Node 6+. See the release **post** and **video** for details!

A solução chama-se esm, o carregador de módulo JavaScript brilhantemente simples, sem babel e sem pacote.

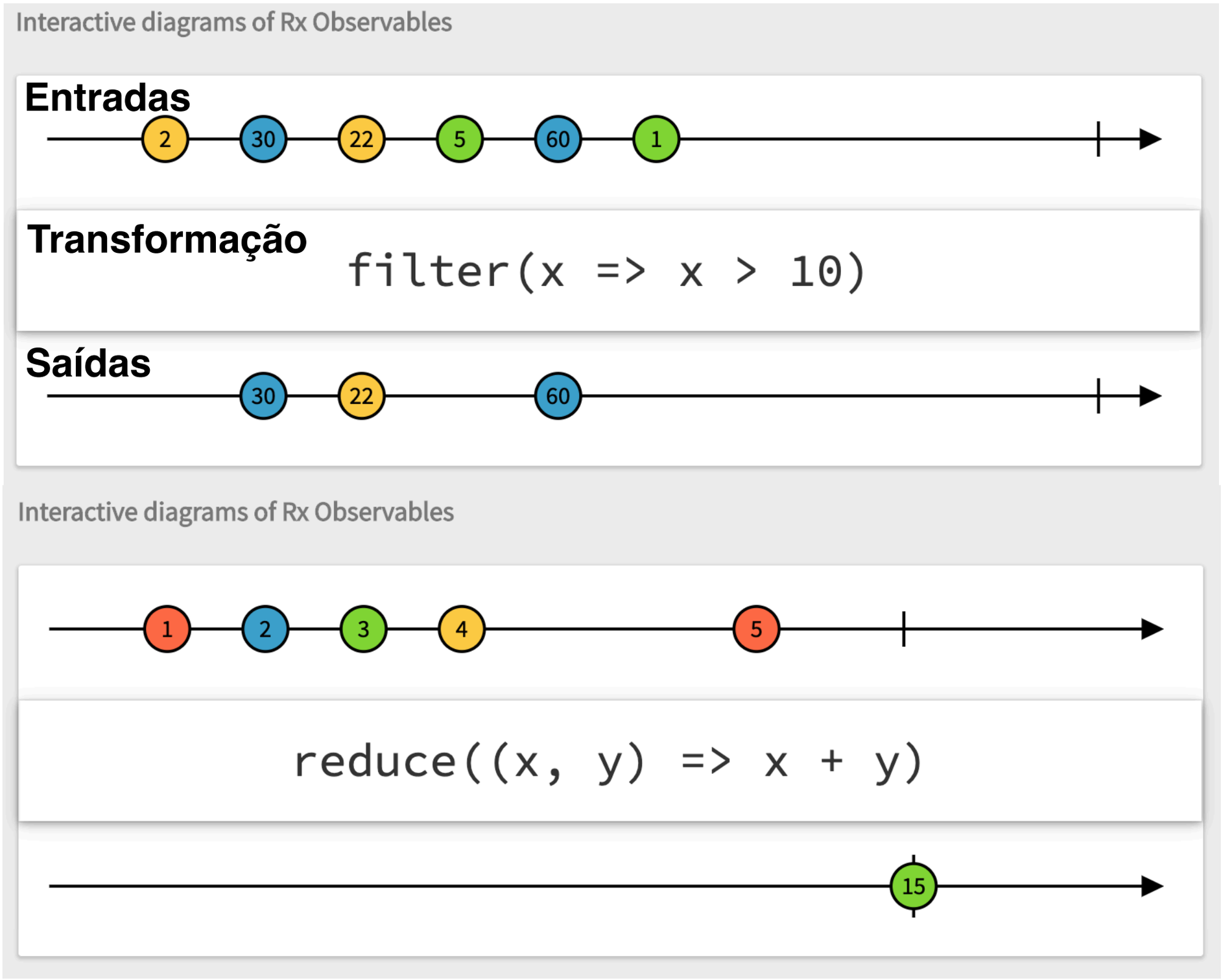
Instale o esm em seu projeto: npm install --save esm

Execute seu código com esm: node -r esm arquivo.js

Irá funcionar sem nenhuma modificação de código.

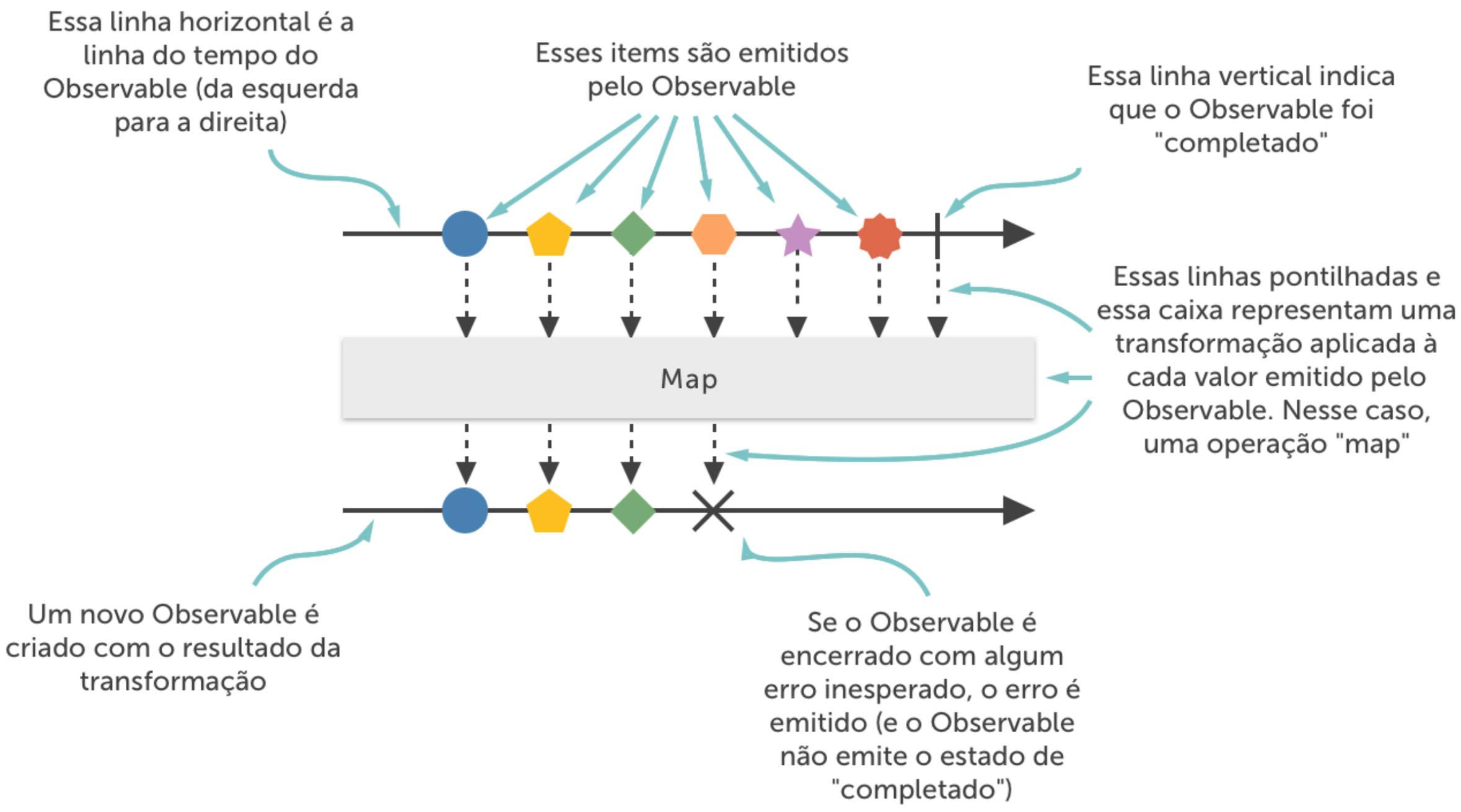
Programação Reativa

linha do tempo

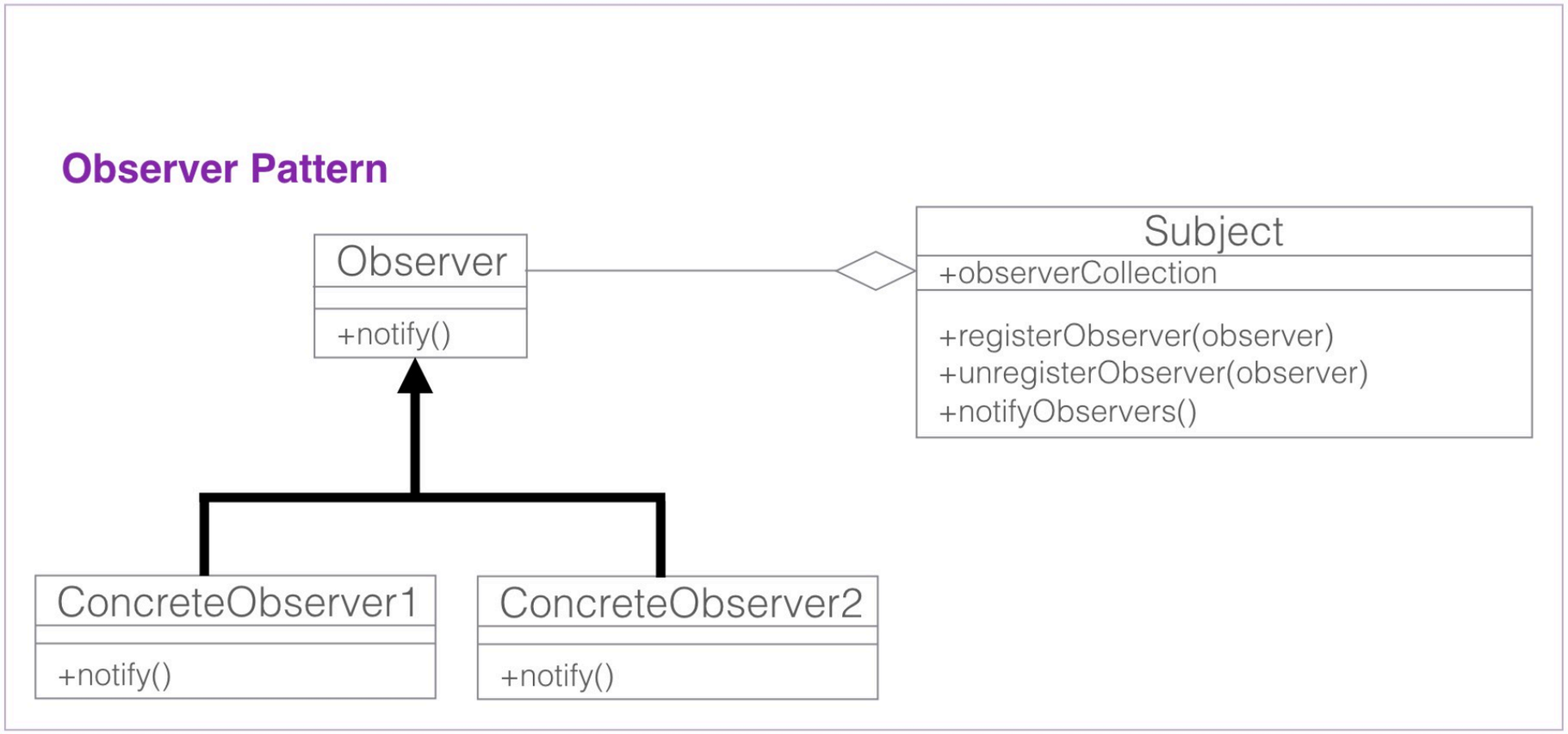


fonte: https://rxmarbles.com

Programação Reativa



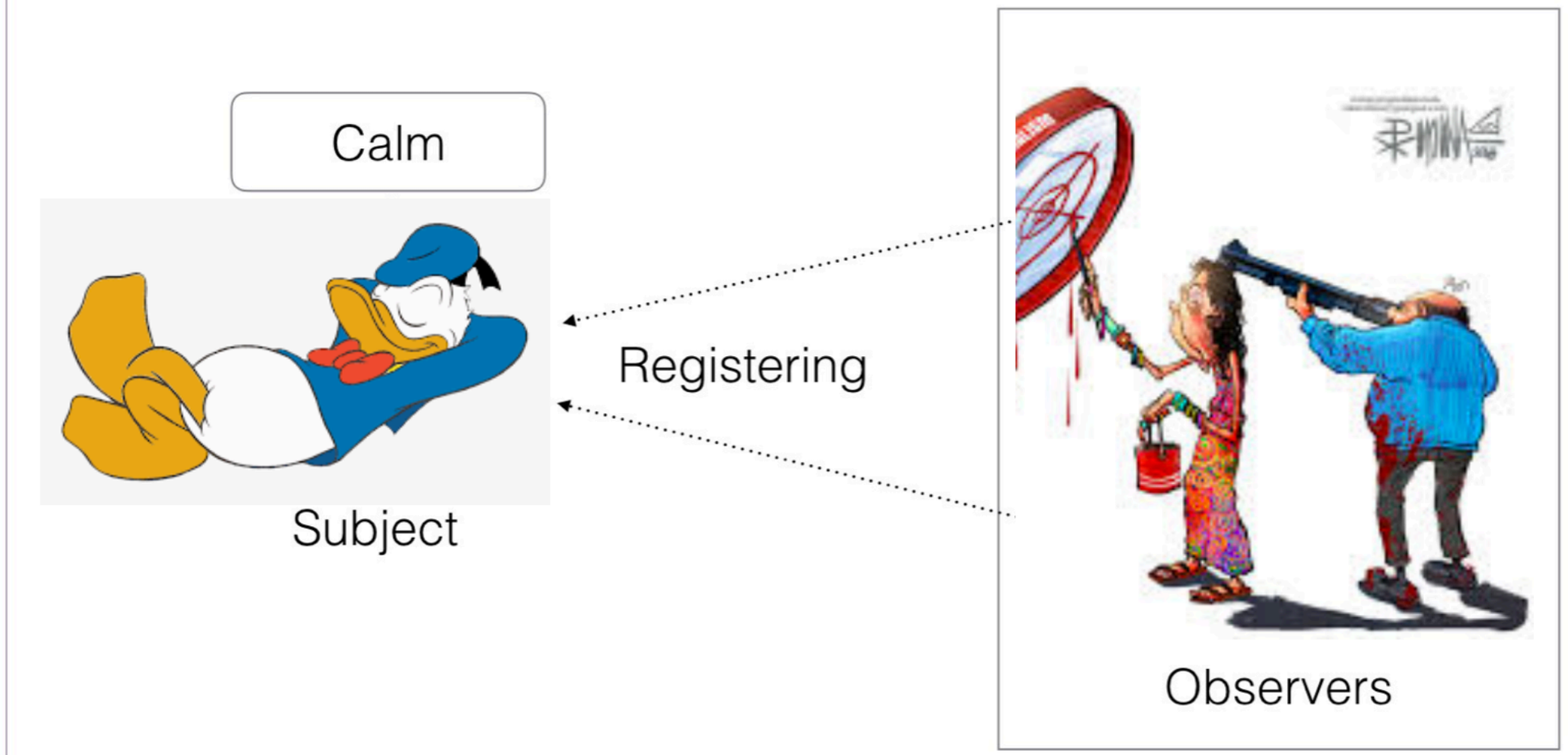
Programação Reativa



Propagação de mudança remete a um padrão muito conhecido: O padrão do Observador.

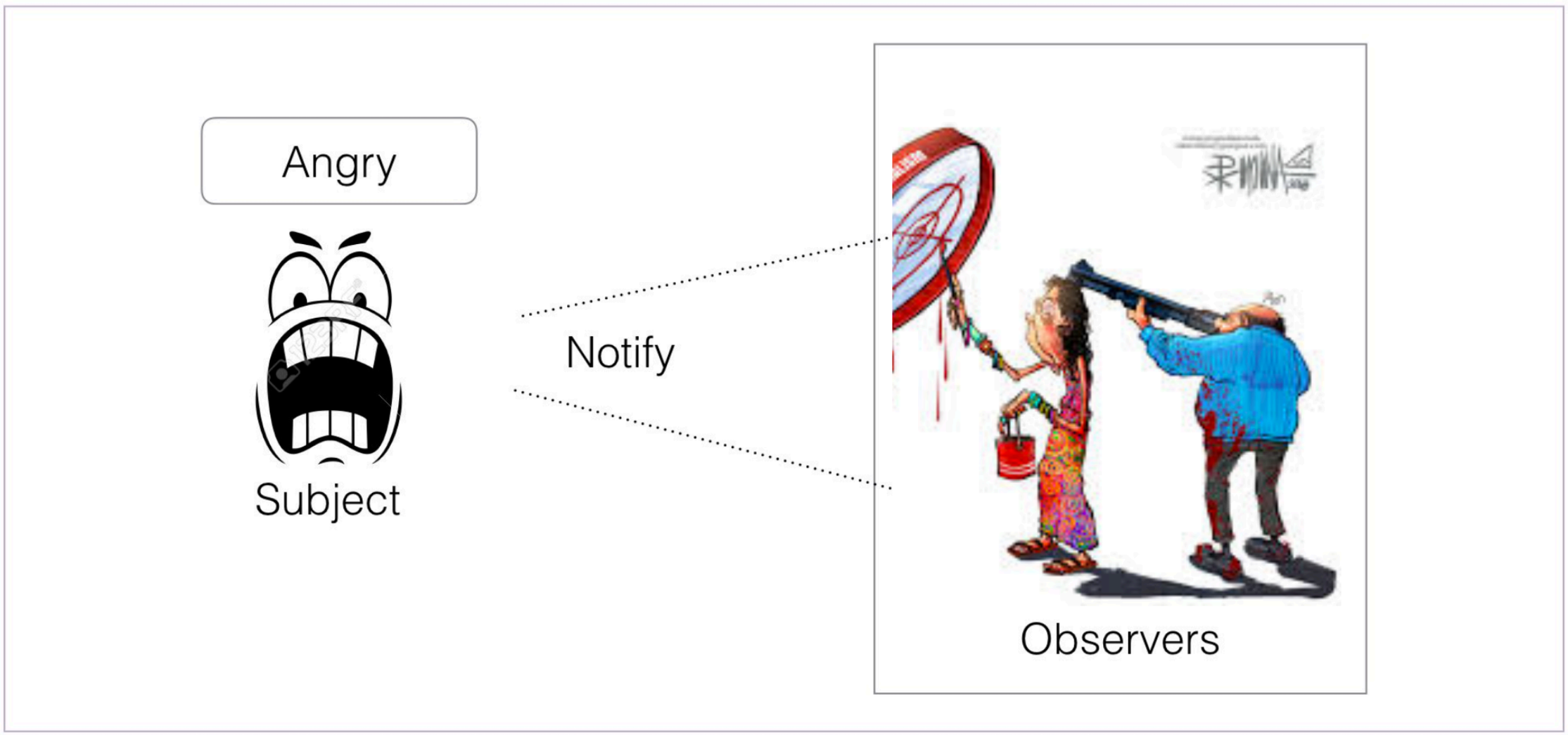
Esse padrão define uma dependência de um pra muitos, onde você tem um Sujeito e uma interface Observador, a qual pode ser implementada por muitos.

Programação Reativa



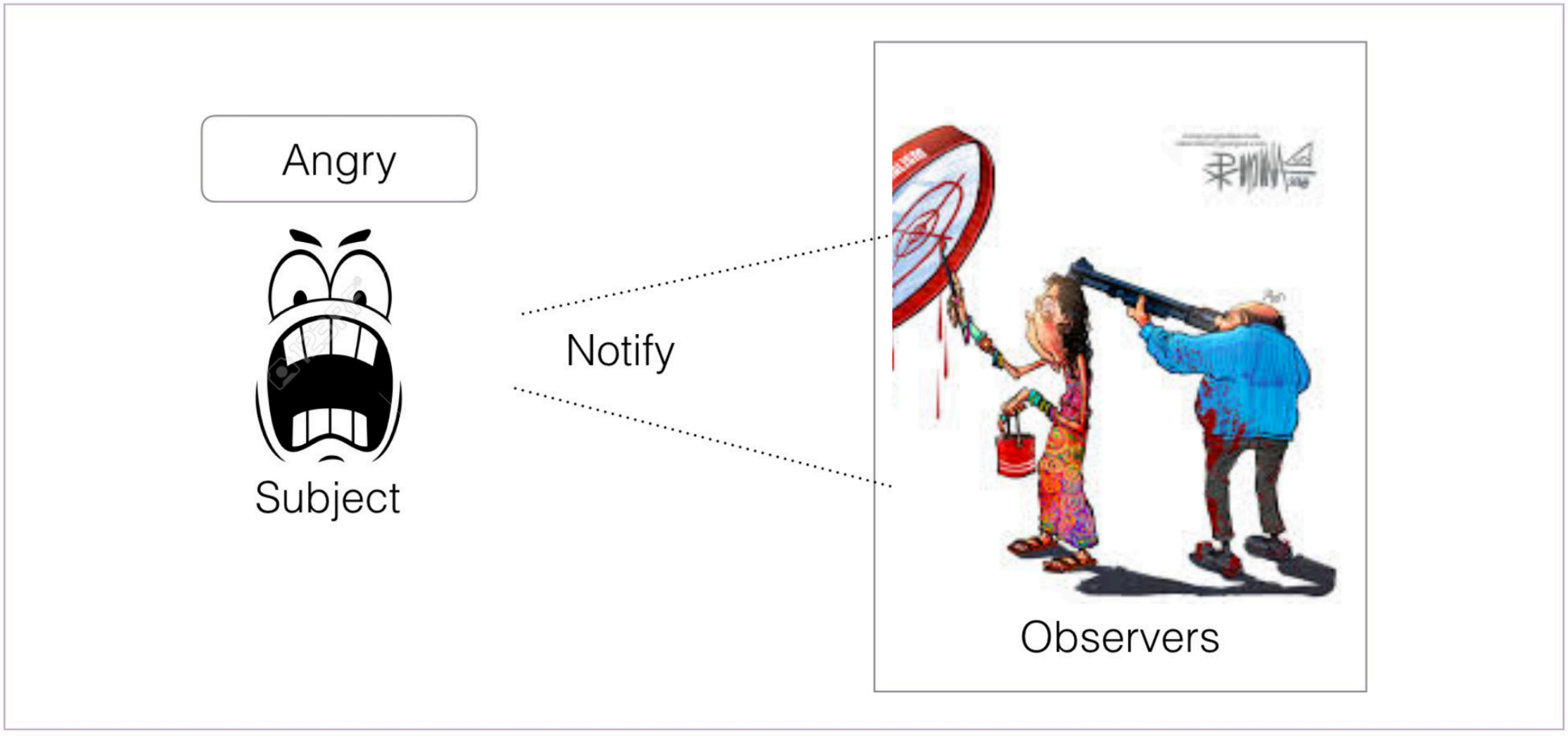
Nesse exemplo temos um Sujeito, o qual possui um estado ou informação que os Observadores estão interessados. Eles então se registram para ouvir mudanças de estado ou notificações de um Sujeito.

Programação Reativa



O Sujeito podem então notificar Observadores registrados quando necessário, como, por exemplo, na ocorrência de uma mudança de estado.

Programação Reativa



Os Observadores podem, então, agir de acordo com as mudanças que ocorreram da maneira como foram programados.

O Sujeito também pode ser chamado Observável (Observable) e pode ser definido como um recipiente que emite sinais para observadores ao longo do tempo.

Apache2, Wordpress e WebView

* criamos uma instancia na amazon free tier.->https://aws.amazon.com/pt/ec2/
* alternativas aws : godady, MS Azure, google
* criamos um subdomínio e apanhamos para IP da AWS

Wordpress:

- mysql -p -u root mysql> create database wordpress;
- mysql> create user 'wordpress'@'%' identified by ‘Senha@123';
- mysql> grant all on wordpress.* to ‘wordpress'@'%';

login wordpress cassio
senha admin: Pets@Concordia

Programação Reativa

RX eventos

https://codepen.io/cassioseffrin/pen/mdJxrNz

Os sinais emitidos pelos Observáveis podem ser considerados um **fluxo de dados** (Data Stream).

Outros exemplos de fluxo de dados são, por exemplo: partes de um arquivo sendo baixado, posições do cursor do mouse, um campo de texto sendo editado, etc

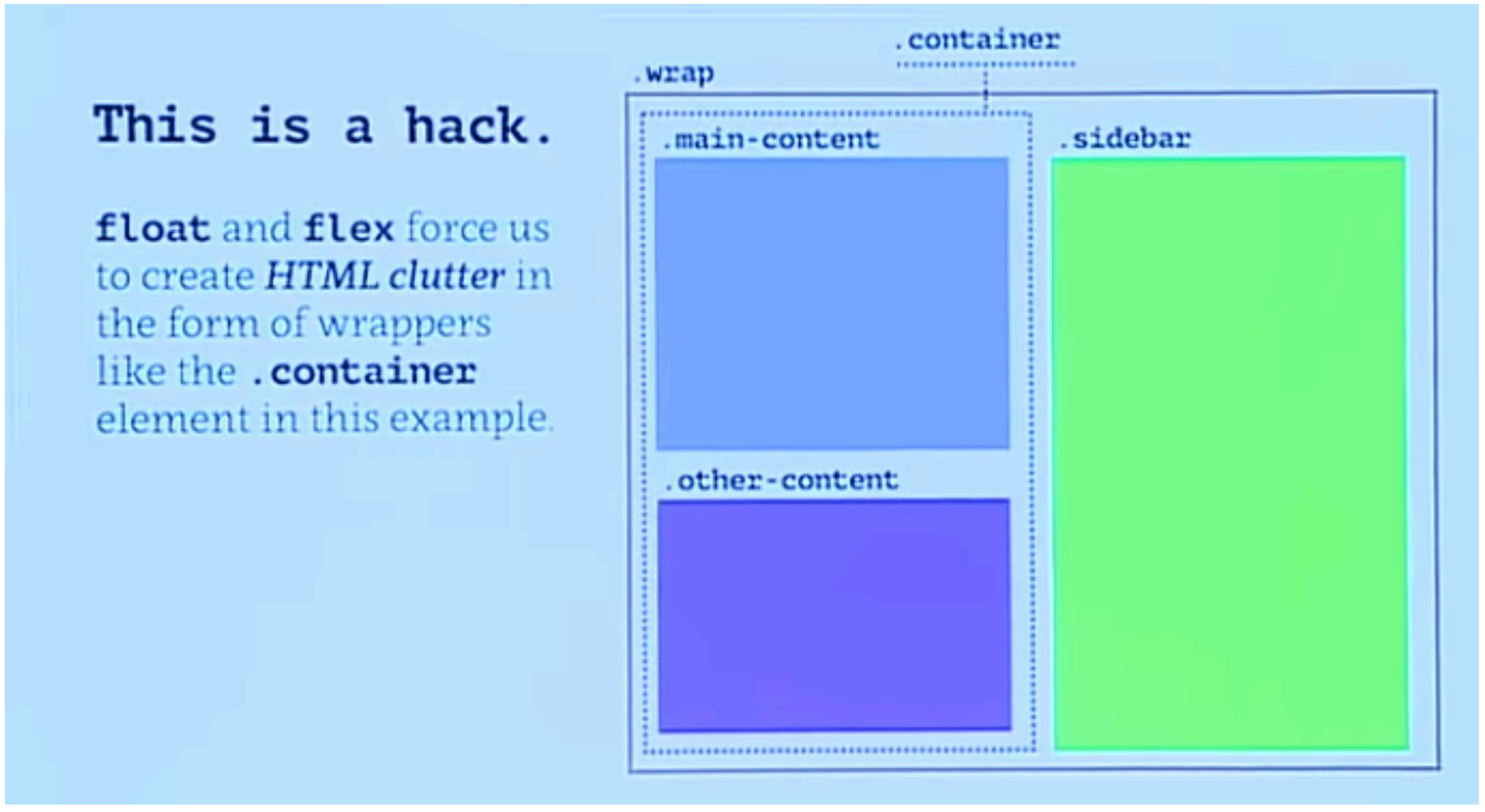
Programação Reativa

- Observables ok
- Subjects ok
- ReplaySubject ok
- BehaviorSubject ok

alguns métodos do RX

- pluck ok
- pipe ok
- distinctUntilChanged ok

CSS Grid


This is a hack.
float and flex force us to create HTML clutter in the form of wrappers like the .container element in this example.
.container
.wrap
.main-content
.sidebar
.other-content

Apache2, Wordpress e WebView

Webview:

- android:
- https://tiagoaguiar.co/webview-android-tutorial
- ios:
- http://agenciadac.com.br/blog-ios/criando-um-web-view-no-iphone/

Tecnologias backend

Parse Server/Dashboard (open)
vs
Firebase (Google)
vs
Amplify (Amazon)
vs
Criar API do zero

Parse Server

1. criar a VM (usar uma VM/VPS ou cloud AWS)

apt install postgresql-10
passwd postgres
su - postgres
psql
create database apirest
psql -c "ALTER USER postgres WITH PASSWORD 'apirest';"
Node.js v12.x:
curl -sL https://deb.nodesource.com/setup_12.x | sudo -E bash -
sudo apt-get install -y nodejs
apt install npm (Node Package Manager)

Parse Server

npm init
npm install express
npm install parse-server
npm install parse-dashboard
node iniciaParseServer.sh

criar os certificados e montar o script js
https://stackoverflow.com/questions/58193944/parse-server-running-with-postgres-parse-error-error-column-rperm-does-not?rq=1

Referencias:
https://github.com/parse-community/parse-server
https://github.com/parse-community/parse-dashboard

https://parseplatform.org/

https://www.youtube.com/watch?v=o522ovITvW4
https://www.youtube.com/watch?v=Em5grOlQNFQ

Parse Server / Parse Dashboard

apt install postgresql-10
psql
create database apirest

passwd postgres
su - postgres
psql -c "ALTER USER postgres WITH PASSWORD 'apirest';"

Node.js v12.x:
curl -sL https://deb.nodesource.com/setup_12.x | sudo -E bash -
sudo apt-get install -y nodejs

Node.js v13.x:

curl -sL https://deb.nodesource.com/setup_13.x | sudo -E bash -
sudo apt-get install -y nodejs

apt install npm
criar os certificados e montar o script js
https://stackoverflow.com/questions/58193944/parse-server-running-with-postgres-parse-error-error-column-rperm-does-not?rq=1

Parse Server / Parse Dashboard / GraphQL

npm init
npm install express
npm install parse-server
npm install parse-dashboard
node iniciaParseServer.sh

Referencias:
https://github.com/parse-community/parse-server
https://github.com/parse-community/parse-dashboard
https://parseplatform.org/
https://www.youtube.com/watch?v=o522ovITvW4
https://www.youtube.com/watch?v=Em5grOlQNFQ

```
#apt install -y mongodb
#mongo
#db.createUser(
  {
    user: "Cassio",
    pwd: "123",
    roles: [ { role: "userAdminAnyDatabase", db: "admin" } ]
  }
)
```

Parse Server / Parse Dashboard / GraphQL

npm install -g parse-server mongodb-runner parse-dashboard express
mongodb-runner start

Script inicialização no github

Exemplo de query do GraphQL

```
query{
  objects{
    findCliente (where: { idade: {_gte:20}}){
      count
      results {nome}
    }
  }
}
```

##link para estudar o graphql
http://apis.guru/graphql-apis/

Testando…
Agora que você configurou seu ambiente GraphQL, é hora de executar sua primeira consulta. Execute o seguinte código no seu GraphQL Playground para verificar a saúde da sua API:

```
query healthy {
  health
}
```

resposta:

```
{
  "data": {
    "health": true
  }
}
```

GraphQL - Classes

4 Vantagens em relação API REST:

1 : Resposta dinâmica, ou seja, o trafego de informação é somente a necessária para o frontend
2 : Toda API GraphQL fica exposta em um único endpoint.
3 : Facilidade de mesclar dados, ou seja, vc pode buscar dados de cliente e faturamento na mesma query.
4 : Desde de que o usuário tenha permissões, você pode criar classes

GraphQL - Classes

Como seu aplicativo ainda não possui um esquema, é possível usar a mutação createClass para criar sua primeira classe por meio da API GraphQL.

```
mutation createFornecedorClass {
  createClass(
    input: {
      clientMutationId: "criacaoFornecedor"
      name: "Fornecedor"
      schemaFields: {
        addStrings: [{ name: "razaoSocial"}, {name:"cnpj" }, {name:"endereco"}]
        addNumbers: [{ name: "pontuacao" }]
        addBooleans: [{ name: "ativo" }]
      }
    }
  ) {
    clientMutationId
    class {
      name
      schemaFields {
        name
        __typename
      }
    }
  }
}
```

O Parse Server aprendeu com a primeira classe que você criou e agora você tem a classe Cliente no seu esquema. Você pode começar a usar as operações geradas!

GraphQL - Classes


```
{
  "data": {
    "createClass": {
      "clientMutationId": "criacaoFornecedor",
      "class": {
        "name": "Fornecedor",
        "schemaFields": [
          {
            "name": "objectId",
            "__typename": "SchemaStringField"
          },
          {
            "name": "createdAt",
            "__typename": "SchemaDateField"
          },
          {
            "name": "updatedAt",
            "__typename": "SchemaDateField"
          },
          {
            "name": "razaoSocial",
            "__typename": "SchemaStringField"
          },
          {
            "name": "cnpj",
            "__typename": "SchemaStringField"
          },
          {
            "name": "endereco",
```

GraphQL - Criação de Objetos

Para cada classe no esquema do seu aplicativo, o Parse Server gera automaticamente uma mutação personalizada para criar os objetos dessa classe por meio da API GraphQL.

Ex, se você tiver uma classe chamada Clinte no esquema, o Parse Server gerará automaticamente uma nova mutação chamada createGameScore, e você poderá executar o código abaixo no seu GraphQL:

GraphQL - Create - Fornecedor

query:

```
mutation criarFonecedor {
  createFonecedor(
    input: {
      clientMutationId: “1"
      fields: {
        razaoSocial: “Lojas Maneco",
        cnpj: “1234.567-890/1234",
        endereco: “Rua do Comercio, 42",
        pontuacao: 10,
        ativo: true
      }
    }
  ) {
    clientMutationId
    fornecedor {
      id
      razaoSocial
      cnpj
      endereco
      ACL {
        public {
          write
          read
        }
      }
    }
  }
}
```

resposta:

```
{
  "data": {
    "createFornecedor": {
      "clientMutationId": "sldjf1",
      "fornecedor": {
        "id":
"Rm9ybmVjZWRvcjpyc0kyRkRET05S",
        "updatedAt":
"2020-05-08T23:06:05.762Z",
        "createdAt": "2020-05-08T23:06:05.762Z",
        "razaoSocial": "Lojas Maneco",
        "cnpj": "1234.567-890/1234",
        "endereco": "Rua do Comercio, 42",
        "pontuacao": 10,
        "ativo": true,
        "ACL": {
          "public": {
            "write": true,
            "read": true
          }
        }
      }
    }
  }
}
```

Nota: o Relay Global Object Identification,
não é um Parse objectId.
Na maioria das vezes, Relay Node Id é um
Base64 do ParseClass e o objectId.

GraphQL - Create

query:

```
mutation criarCliente {
  createCliente(
    input: {
      clientMutationId: "chaveUnica-234u5"
      fields: {
        nome: "Cassio",
        sexo: "Masculino",
        idade: 36,
        apto: true
      }
    }
  ) {
    clientMutationId
    cliente {
      id
      updatedAt
      createdAt
      nome
      sexo
      idade
      apto
      ACL {
        public {
          write
          read
        }
      }
    }
  }
}
```

resposta:

```
{
  "data": {
    "createCliente": {
      "clientMutationId": "chaveUnica-234u5",
      "cliente": {
        "id": "Q2xpZW50ZTpVVWQxaVBMTXZw",
        "updatedAt":
"2020-05-08T01:23:34.550Z",
        "createdAt": "2020-05-08T01:23:34.550Z",
        "nome": "Cassio",
        "sexo": "Masculino",
        "idade": 36,
        "apto": true,
        "ACL": {
          "public": {
            "write": true,
            "read": true
          }
        }
      }
    }
  }
}
```

Nota: o Relay Global Object Identification,
não é um Parse objectId.
Na maioria das vezes, Relay Node Id é um
Base64 do ParseClass e o objectId.

GraphQL - Atualização de Objetos

Para cada classe no esquema do seu aplicativo, o Parse Server gera automaticamente uma mutação personalizada para atualizar os objetos dessa classe por meio da API GraphQL.

Por exemplo, se você tiver uma classe chamada Cliente no esquema, o Parse Server gerará automaticamente uma nova mutação chamada **updateCliente**.

GraphQL - Update

query:

```
mutation updateCliente {
  updateCliente(
    input: {
      id: "Q2xpZW50ZTpVVWQxaVBMTXZw"
      fields: { nome: "Cassio Seffrin" }
    }
  ) {
    cliente {
      nome,
      sexo
    }
  }
}
```

resposta:

```
{
  "data": {
    "updateCliente": {
      "cliente": {
        "nome": "Cassio Seffrin",
        "sexo": "Masculino"
      }
    }
  }
}
```

GraphQL - Update Fornecedor

query:

```
mutation updateFornecedor {
  updateFornecedor(
    input: {
      id: "Rm9ybmVjZWRvcjpYczUwZmhqWjVD"
      fields: { endereco: "Rua Victor Sopelsa" }
    }
  ) {
    fornecedor {
      razaoSocial,
      endereco
    }
  }
}
```

resposta:

```
{
  "data": {
    "updateFornecedor": {
      "fornecedor": {
        "razaoSocial": "Casa da Cozinha",
        "endereco": "Rua Victor Sopelsa"
      }
    }
  }
}
```

GraphQL - Remoção de Objetos

Para cada classe no esquema do seu aplicativo, o Parse Server gera automaticamente uma mutação personalizada para excluir os objetos dessa classe por meio da API GraphQL.

Por exemplo, se você tiver uma classe chamada Cliente no esquema, o Parse Server gerará automaticamente uma nova mutação chamada deleteCliente

GraphQL - Delete

query:

```
mutation deleteCliente{
  deleteCliente(input: { id:
"Q2xpZW50ZTpVVWQxaVBMTXZw" }) {
    cliente {
      id
      nome
    }
  }
}
```

resposta:

```
{
  "data": {
    "deleteCliente": {
      "cliente": {
        "id": "Q2xpZW50ZTpVVWQxaVBMTXZw",
        "nome": "Cassio Seffrin"
      }
    }
  }
}
```

Nota: A API retorna o objeto excluído, o que pode permitir que você mostre mensagens como "O cliente Cassio Seffrin foi removido com sucesso" no front-end.

GraphQL - Buscas

Para cada classe no esquema do seu aplicativo, o Parse Server gera automaticamente uma consulta personalizada para encontrar os objetos dessa classe por meio da API GraphQL.

Por exemplo, se você tiver uma classe chamada Cliente no esquema, o Parse Server gerará automaticamente uma nova consulta chamada clientes, e você poderá executar o código abaixo no seu GraphQL:

GraphQL - Busca Cliente

query:

```
query getCliente {
  cliente(id: "Q2xpZW50ZTpHckRnMjluRk1J") {
    id
    nome
    idade
    sexo
    apto
    ACL {
      public {
        read
        write
      }
    }
  }
}
```

resposta:

```
{
  "data": {
    "cliente": {
      "id": "Q2xpZW50ZTpHckRnMjluRk1J",
      "nome": "Cassio",
      "idade": 36,
      "sexo": "Masculino",
      "apto": true,
      "ACL": {
        "public": {
          "read": true,
          "write": true
        }
      }
    }
  }
}
```

Nota: A API retorna o objeto excluído, o que pode permitir que você mostre mensagens como "O cliente Cassio Seffrin foi removido com sucesso" no front-end.

GraphQL - Busca Fornecedor

query:

```
query getFornecedor {
  fornecedor(id:
"Rm9ybmVjZWRvcjpYczUwZmhqWjVD") {
    id
    razaoSocial
  }
}
```

resposta:

```
{
  "data": {
    "fornecedor": {
      "id":
"Rm9ybmVjZWRvcjpYczUwZmhqWjVD",
      "razaoSocial": "Casa da Cozinha"
    }
  }
}
```

Nota: A API retorna o objeto excluído, o que pode permitir que você mostre mensagens como "O cliente Cassio Seffrin foi removido com sucesso" no front-end.

GraphQL - Query

query:

```
query getClientesMaiores10 {
  clientes(where: {
        idade: { greaterThan: 10 }
  }) {
    count
    edges {
      cursor
      node {
        id
        nome
      }
    }
  }
}
```

resposta:

```
{
  "data": {
    "clientes": {
      "count": 3,
      "edges": [
        {
          "cursor":
"YXJyYXIjb25uZWN0aW9uOjA=",
          "node": {
            "id": "Q2xpZW50ZTpHckRnMjluRk1J",
            "nome": "Cassio"
          }
        },
        {
          "cursor":
"YXJyYXIjb25uZWN0aW9uOjE=",
          "node": {
            "id":
"Q2xpZW50ZTpSRmJ5WU5TRDBS",
            "nome": "Jana"
          }
        }
      ]
    }
  }
}
```

GraphQL - Query

query com paginacao:

```
query getClientesPaginacao {
  clientes(after:
"YXJyYXIjb25uZWN0aW9uOjA") {
     pageInfo {
      hasNextPage
      hasPreviousPage
      startCursor
      endCursor
    }
    count
    edges {
      cursor
      node {
        nome
      }
    }
  }
}
```

resposta:

```
{
  "data": {
    "clientes": {
      "pageInfo": {
        "hasNextPage": false,
        "hasPreviousPage": false,
        "startCursor": "YXJyYXIjb25uZWN0aW9uOjE=",
        "endCursor": "YXJyYXIjb25uZWN0aW9uOjI="
      },
      "count": 0,
      "edges": [
        {
          "cursor": "YXJyYXIjb25uZWN0aW9uOjE=",
          "node": {
            "nome": "Jana"
          }
        },
        {
          "cursor": "YXJyYXIjb25uZWN0aW9uOjI=",
          "node": {
            "nome": "Maria"
          }
        }
      ]
    }
  }
}
```

GraphQL - Exemplos CURL

```
curl -k 'https://aws.magnani.ind.br:1337/graphql' -H 'Accept-Encoding: gzip,
deflate, br' -H 'Content-Type: application/json' -H 'Accept: application/json' -H
'Connection: keep-alive' -H 'DNT: 1' -H 'Origin: https://aws.magnani.ind.br:1337' -H
'X-Parse-Application-Id: aulaparse' -H 'X-Parse-Master-Key: aulaparse' --data-binary
'{"query":"\nquery getClientesPag {\n  clientes(after: \"YXJyYXljb25uZWN0aW9uOjA\")
{\n    pageInfo {\n      hasNextPage\n      hasPreviousPage\n      startCursor\n
endCursor\n    }\n    count\n    edges {\n      cursor\n      node {\n        nome\n
}\n    }\n  }\n}"}' --compressed
```

```
curl -k 'https://aws.magnani.ind.br:1337/graphql' -H 'Accept-Encoding: gzip,
deflate, br' -H 'Content-Type: application/json' -H 'Accept: application/json' -H
'Connection: keep-alive' -H 'DNT: 1' -H 'Origin: https://aws.magnani.ind.br:1337' -H
'X-Parse-Application-Id: aulaparse' -H 'X-Parse-Master-Key: aulaparse' --data-binary
'{"query":"query getFornecedor {\n  fornecedor(id: \"Rm9ybmVjZWRvcjpYczUwZmhqWjVD\")
{\n    id\n    razaoSocial\n  }\n}"}' -compressed
```

Web Assembly - MS Blazor