



Por que angular?

Da linguagem protótipo aos frameworks SPA



Por que angular?

Quem sou eu?

- 15 anos de experiência em desenvolvimento de aplicações.
- Graduado na UFF - Ciência da Computação
- 13 anos na indústria de óleo/gás
- Arquiteto de soluções
- Desenvolvedor full-stack
- Principais linguagens: C#, javascript, PL/SQL
- Angular 7 (projeto atual)
- Pós-graduando em IA.
- Casado, pai de dois filhos



victor-hugo-07570789



Por que angular?

O objetivo é entender os conceitos do angular a partir da evolução do mundo front-end.

- O surgimento do javascript (o protótipo)
- Applets Java (kkkkkk)
- Tecnologias component based
- XMLHttpRequest
- AJAX e RIA (já morreu)
- Bibliotecas javascript
- Frameworks SPA (ou bibliotecas?)



Por que angular? – O protótipo

JavaScript ~~é um lixo~~ é uma linguagem de programação que permite a você criar conteúdo que se atualiza dinamicamente, controlar mídias, imagens animadas, e tudo o mais que há de interessante. Ok, não tudo, mas é maravilhoso o que você pode efetuar com algumas linhas de código JavaScript.

Fonte: <https://developer.mozilla.org/pt-BR/>



Por que angular? – O protótipo

- O JavaScript foi criado na década de 90 por Brendan Eich a serviço da Netscape.
- O protótipo feito em 10 dias é colocado em produção.
- É aquela linguagem para validar formulários?
- O principal browser era o NCSA.






Por que angular? – O protótipo

- A Microsoft cria sua própria implementação no IE 3. (Jscript)
- *European Computer Manufacturers Association?*
- *Technical Committee 39*
- *ECMA-262*





Standards

Contact Ecma
Rue du Rhône 114 CH-1204 Geneva
T: +41 22 849 6000 F: +41 22 849 6001

[SITE MAP](#)

[What is Ecma](#)[Activities](#)[News](#)[Standards](#)

[Standards Index](#)[Standards List](#)[Withdrawn Standards](#)[Tech. Reports Index](#)[Tech. Reports List](#)[Withdrawn Tech. Reports](#)[Mementos](#)

[Printer Friendly Version](#)[Back](#)

Standard ECMA-262

ECMAScript® 2019 Language Specification

10th edition (June 2019)

This Standard defines the ECMAScript 2019 general-purpose programming language.

The following files can be freely downloaded:

File name	Size (Bytes)	Content
ECMA-262 edition 10		Browsable HTML
ECMA-262.pdf	14 423 437	Acrobat (r) PDF file

Kindly note that **the normative copy is the HTML version**; the PDF version has been produced to generate a printable document.



Por que angular? – O protótipo

- ECMAScript é uma especificação de linguagem de scripts padronizada pelo ECMA-262.
- Javascript é ECMAScript porque segue suas especificações.



Por que angular? – O protótipo

Histórico das versões

- **ECMAScript 1 (Junho de 1997)**

O protótipo de 10 dias.

- **ECMAScript 2 (Agosto de 1998)**

- **ECMAScript 3 (Dezembro de 1999)**

- **ECMAScript 4 (abandonada em Julho de 2008)**

- **ECMAScript 5 (Dezembro de 2009)**

Suporte a JSON entre outras coisas. Essa é a atualização incremental acordada no fechamento da ECMAScript 4.

- **ECMAScript 5.1 (Junho de 2011)**

- **ECMAScript 6 (Junho de 2015)**

Também conhecida como ECMAScript 2015, é a primeira fase da versão Harmony. Inclui sintaxe muito mais enxuta e funcionalidades como arrow functions, binary data, arrays tipados, coleções (maps, sets e weak maps), promises, melhorias em numerais e matematica, reflection, e proxies.

- **ECMAScript 7 (Junho de 2016)**

Também conhecida como ECMAScript 2016, é a ultima fase da versão Harmony. Inclui features como operadores exponenciais e o método `Array.prototype.includes`.

Alguns browsers ainda não suportam totalmente a versão 6 e 7 da ECMAScript. Porém é possível **transpilar** para ECMAScript 5 através de bibliotecas como o **Babel** ou **Polyfills**.



Por que angular? – O protótipo

MAS.....



Por que angular? – O protótipo

Artigo Medium

- <https://medium.com/@trungluongquang/why-javascript-is-popular-despite-being-a-crappy-illogical-language-a8be98b20779>

```
9999999999999999 == 10000000000000000 → true  
0.1 + 0.2 === 0.3 → false
```

```
x = 1.0000000000000001  
x === 1 → true
```

```
typeof NaN == 'number' → true  
NaN != NaN → true
```

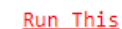


Sites legais para “aprender” Javascript

Erros bizarros de Javascript

<https://wtfjs.com/>

<http://www.jsfuck.com/>





Por que angular? – O protótipo

Os melhores livros que li.





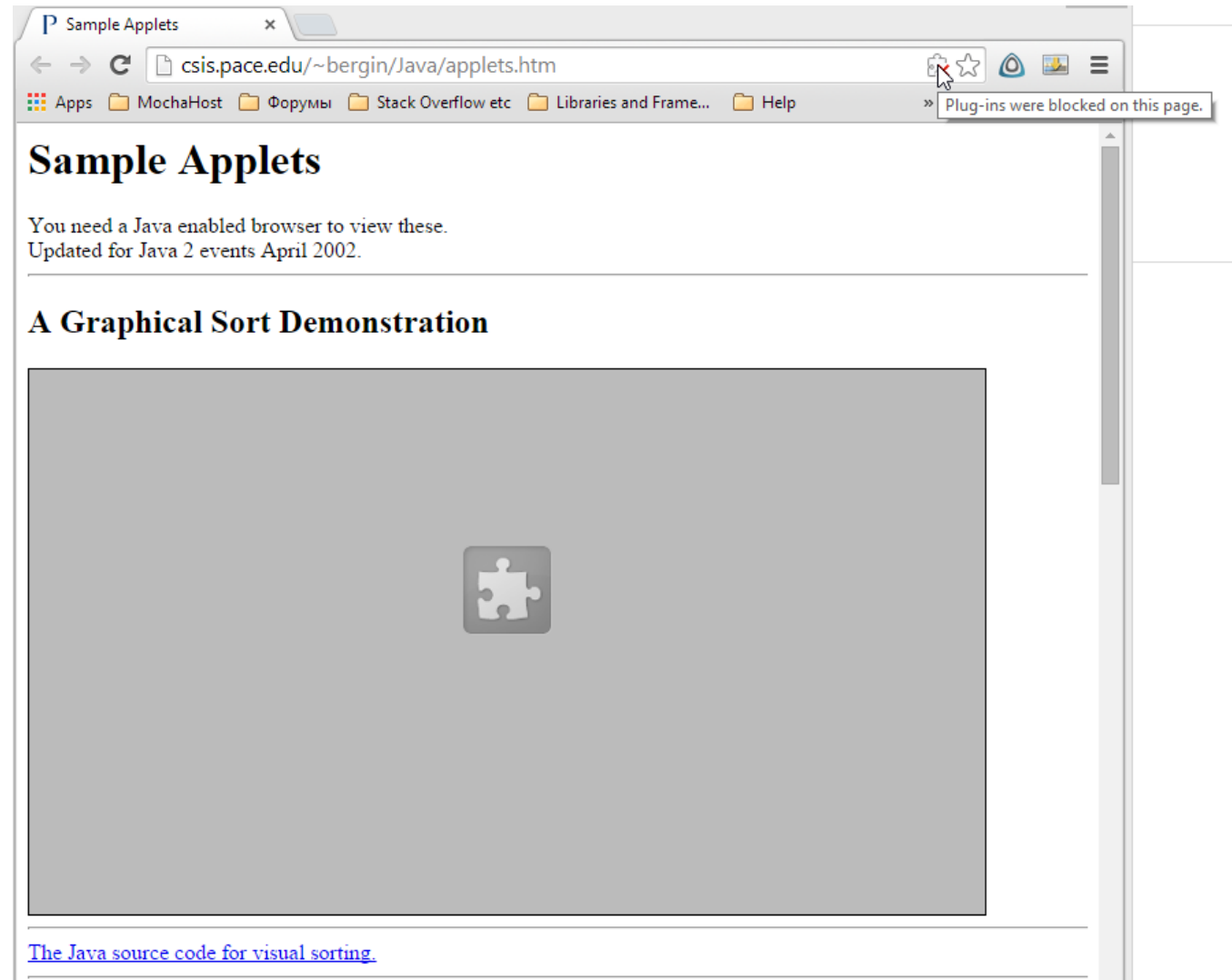
Por que angular? – Applets Java

- Write once, run everywhere! #sqn
- Grande vantagem é não depender do JS
- ~~• Principal desvantagem é depender do Java~~
- Ter a JRE instalada era uma grande desvantagem
- Conexões lentas na época



Por que angular? – Applets Java

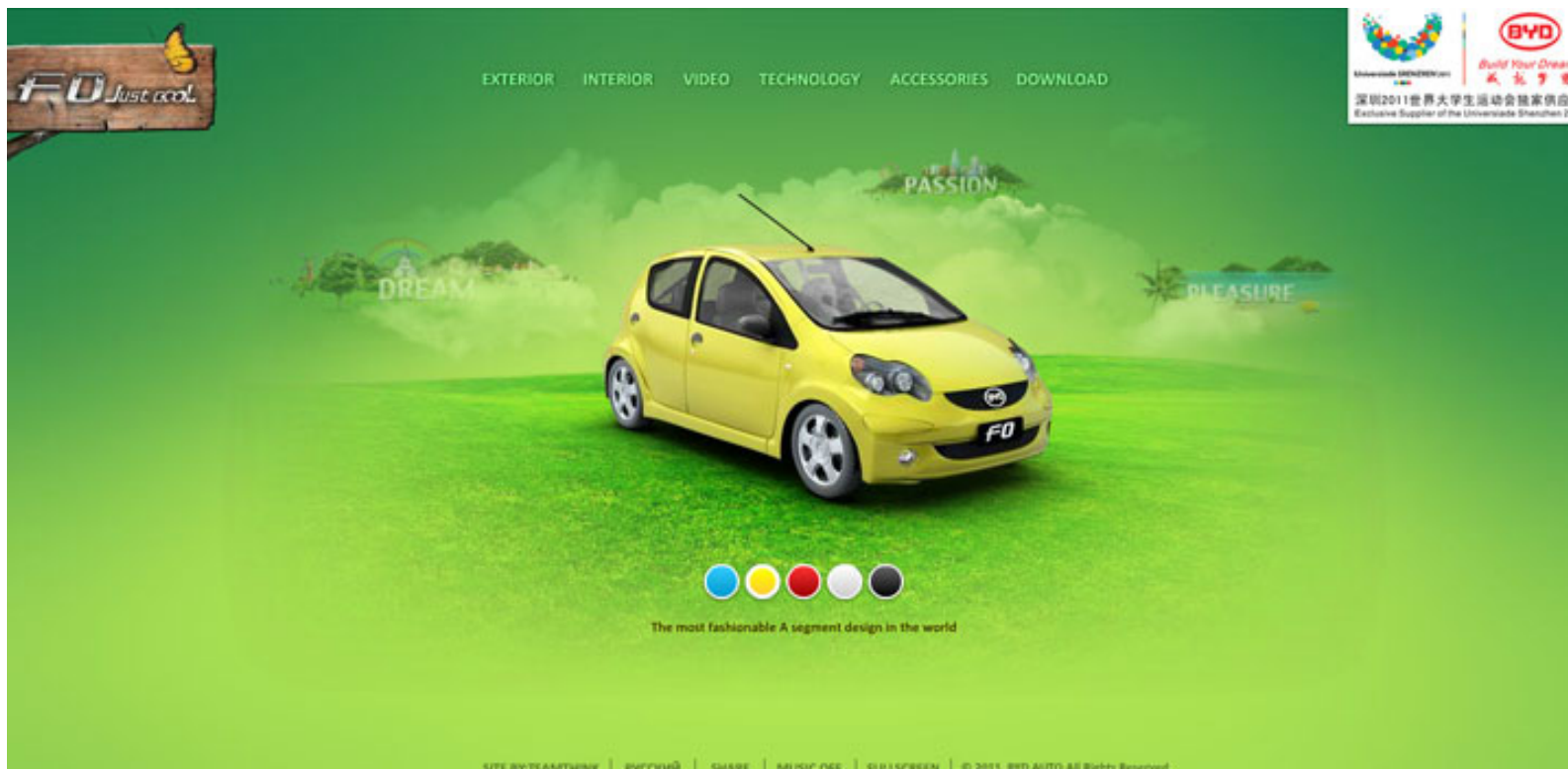
- Exemplo de Applet Java rodando.





Por que angular? – Applets Java

- Surge o Flash Player e domina o mercado de aplicações ricas na web.





Por que angular? – Cenário

- Microsoft acuada no segmento Desktop (Delphi x VB6)
- Java crescia na web com servlets e jsp.
- ASP e CGI/PHP
- Web ainda era server-side rendering (SSR)



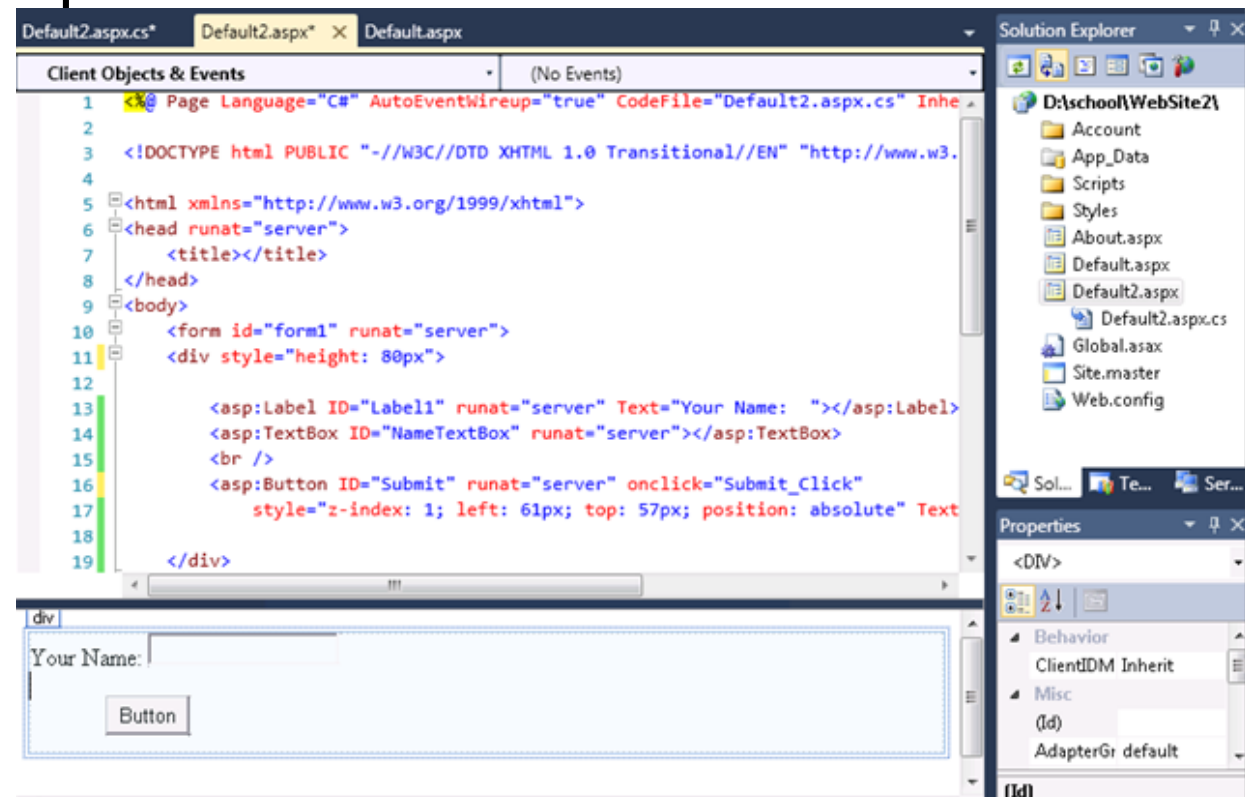
Por que angular? – Tecnologias component based

- Bill Gates contrata Anders Hejlsberg para criar o .NET framework e C#
- Surge WebForms ASP.NET
- Tornar desenvolvimento web tão fácil quanto Desktop.
- JSF surge no mundo Java (Sim, uma idéia ruim também é copiada...)



Por que angular? – Tecnologias component based

- Sem controle do html gerado
- Ciclo de vida lento
- Refresh em toda página a cada clique
- Stateful





Por que angular? – XmlHttpRequest

- Necessidade de trazer poucos bytes do servidor
- Pequenos trechos de página carregados
- Manipular o DOM era complexo

```
function Xhr(){ /* returns cross-browser XMLHttpRequest, or null if unable */
  try {
    return new XMLHttpRequest();
  }catch(e){}
  try {
    return new ActiveXObject("Msxml3.XMLHTTP");
  }catch(e){}
  try {
    return new ActiveXObject("Msxml2.XMLHTTP.6.0");
  }catch(e){}
  try {
    return new ActiveXObject("Msxml2.XMLHTTP.3.0");
  }catch(e){}
  try {
    return new ActiveXObject("Msxml2.XMLHTTP");
  }catch(e){}
  try {
    return new ActiveXObject("Microsoft.XMLHTTP");
  }catch(e){}
  return null;
}
```



Por que angular? – AJAX e RIA

- Cansado de escrever javascript para vários browsers, John Resig cria o JQuery.
- Padronização de Request assíncrono.
- QuerySelector torna-se padrão graças ao JQuery.
- Bundle do JQuery cai com o tempo.

```
$.get( "test.php", function( data ) {  
    alert( "Data Loaded: " + data );  
});
```



Por que angular? – AJAX e RIA

- Processo de manipular DOM é custoso.
- Testes são difíceis de implementar.

```
$.get( "https://final-project-recording.firebaseio.com/lessons.json",  
      function( data ) {  
        var lessons = Object.values(data);  
  
        var html = "<table class='table lessons-list'>" + " +  
                  "<thead>" +  
                    "<th>Description</th>" +  
                  "</thead>" +  
                  "<tbody>";  
  
        lessons.forEach(function(lesson) {  
          html += '<tr>' + +  
                '<td>' + lesson.description + '</td>' +  
                '</tr>';  
        });  
  
        html += '</tbody></table>';  
  
        $("#lessons").html(html);  
  
      }  
    );
```



Por que angular? – AJAX e RIA

- Surge Silverlight, Adobe Flex...
- Todos precisavam de plugin no cliente



E então....



Por que angular? – AJAX e RIA

- FLASH NÃO!!!!!!!!!!!!!!!!!!!!





Por que angular? – Bibliotecas Javascript

- Desafio continuava: criar aplicações robustas com javascript

```
<!doctype html>
<html>
  <head>
    <title>Video Page</title>
  </head>
  <body>
    <!-- Server rendered video page content goes here ... ->
    <script src="/js/jquery.js"><script>
    <script src="/js/underscore.js"></script>
    <script src="/js/react.js"></script>
    <script src="/js/sidebar.js"></script>
    <script src="/js/video-player.js"></script>
    <script src="/js/discussion.js"></script>
  </body>
</html>
```




Por que angular? – Bibliotecas Javascript

- Modularização com ES5

```
// Expose module as global variable
var singleton = function(){

    // Inner logic
    function sayHello(){
        console.log('Hello');
    }

    // Expose API
    return {
        sayHello: sayHello
    }
}()
```



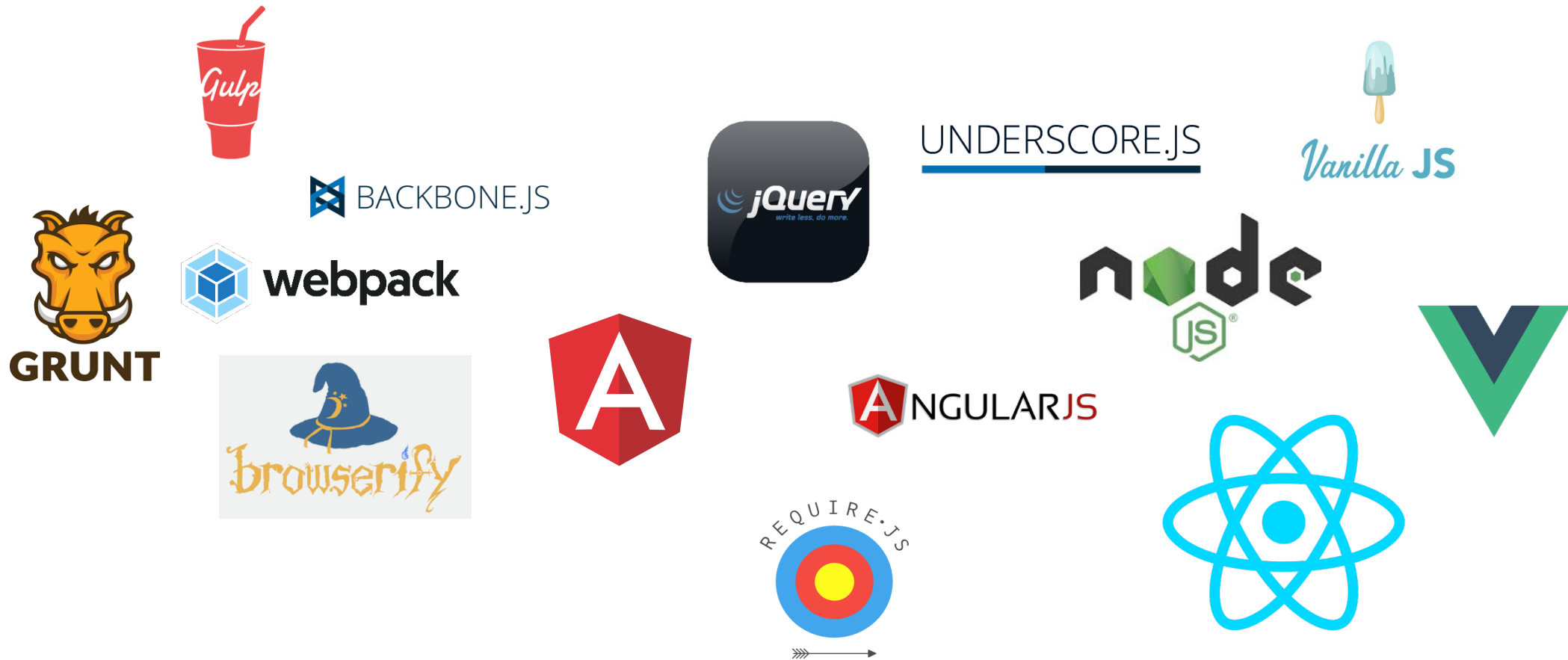
Por que angular? – Bibliotecas Javascript

- Desafios:
- Performance de rede (Network performance)
- Cache
- Tamanho dos arquivos(bundle size, minification)
- Download apenas do que é necessário
- Modularização

E para resolver todos esses problemas...

Por que angular? – Bibliotecas Javascript

O que foi que fizemos????





Por que angular? – Bibliotecas Javascript

- Backbone



```
1 <!DOCTYPE html>
2 <html>
3   <head>
4     <meta charset="UTF-8" />
5     <title>Hello World</title>
6     <script src="../lib/jquery-min.js"></script>
7     <script src="../lib/underscore-min.js"></script>
8     <script src="../lib/backbone-min.js"></script>
9     <script>
10       $(document).ready(function() {
11         var HelloView = Backbone.View.extend({
12           el: $('body'),
13           initialize: function() {
14             this.render();
15           },
16           render: function() {
17             $(this.el).append("<h1>Hello World</h1>");
18           }
19         });
20         var helloView = new HelloView();
21       });
22     </script>
23   </head>
24   <body></body>
```



Por que angular? – Bibliotecas Javascript

- Requirejs – Padrão de modularização AMD



```
//my/shirt.js now has some dependencies, a cart and inventory  
//module in the same directory as shirt.js  
define(["./cart", "./inventory"], function(cart, inventory) {  
    //return an object to define the "my/shirt" module.  
    return {  
        color: "blue",  
        size: "large",  
        addToCart: function() {  
            inventory.decrement(this);  
            cart.add(this);  
        }  
    }  
}  
);
```



Por que angular? – Bibliotecas Javascript

- Surge o nodejs!
- Javascript no lado do servidor.
- Padrão de modularização CommonJS



```
var dep1 = require('./dep1');  
var dep2 = require('./dep2');  
  
module.exports = function(){  
    // ...  
}
```


Por que angular? – Bibliotecas Javascript

- E mais ferramentas!!
- Modularização e automatização de tarefas





Por que angular? – Bibliotecas Javascript

 npm trends

gulp vs grunt vs webpack vs browserify

Enter an npm package...

gulp x

grunt x

webpack x

browserify x

+ yarn

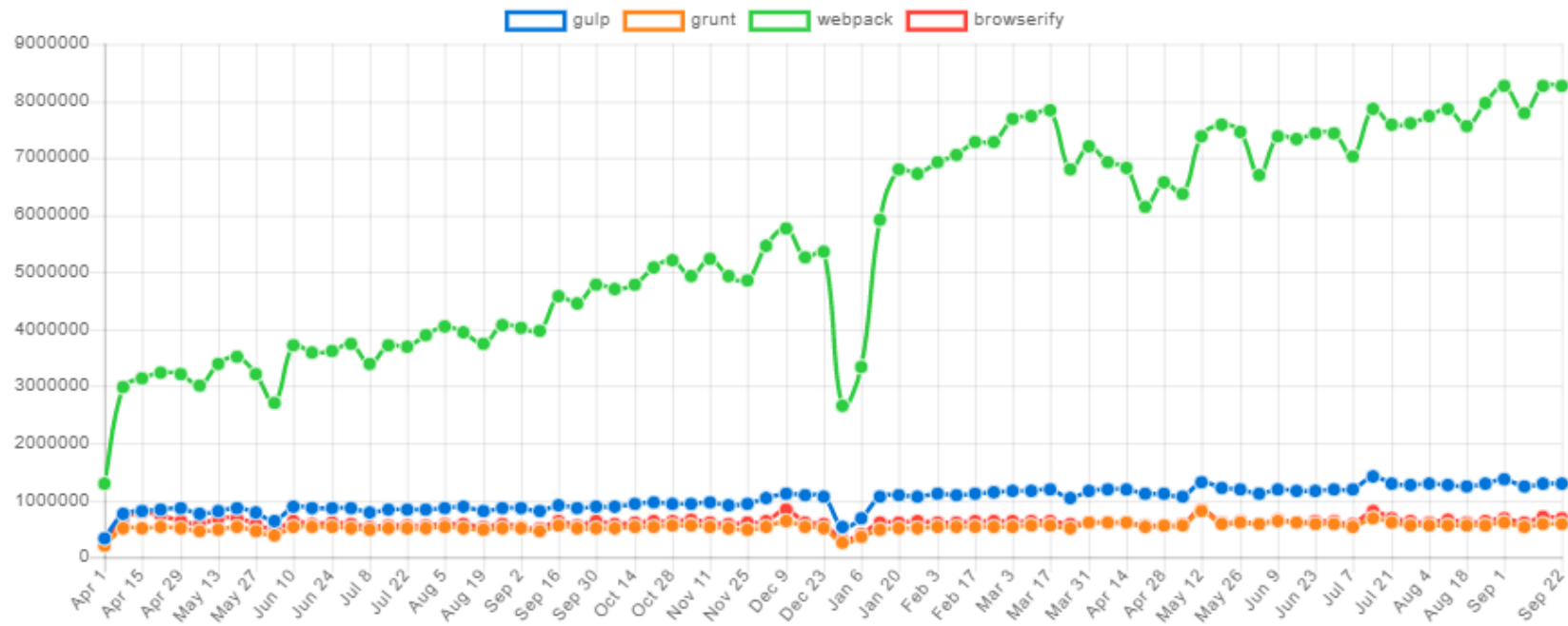
+ parcel

+ bower

+ rollup

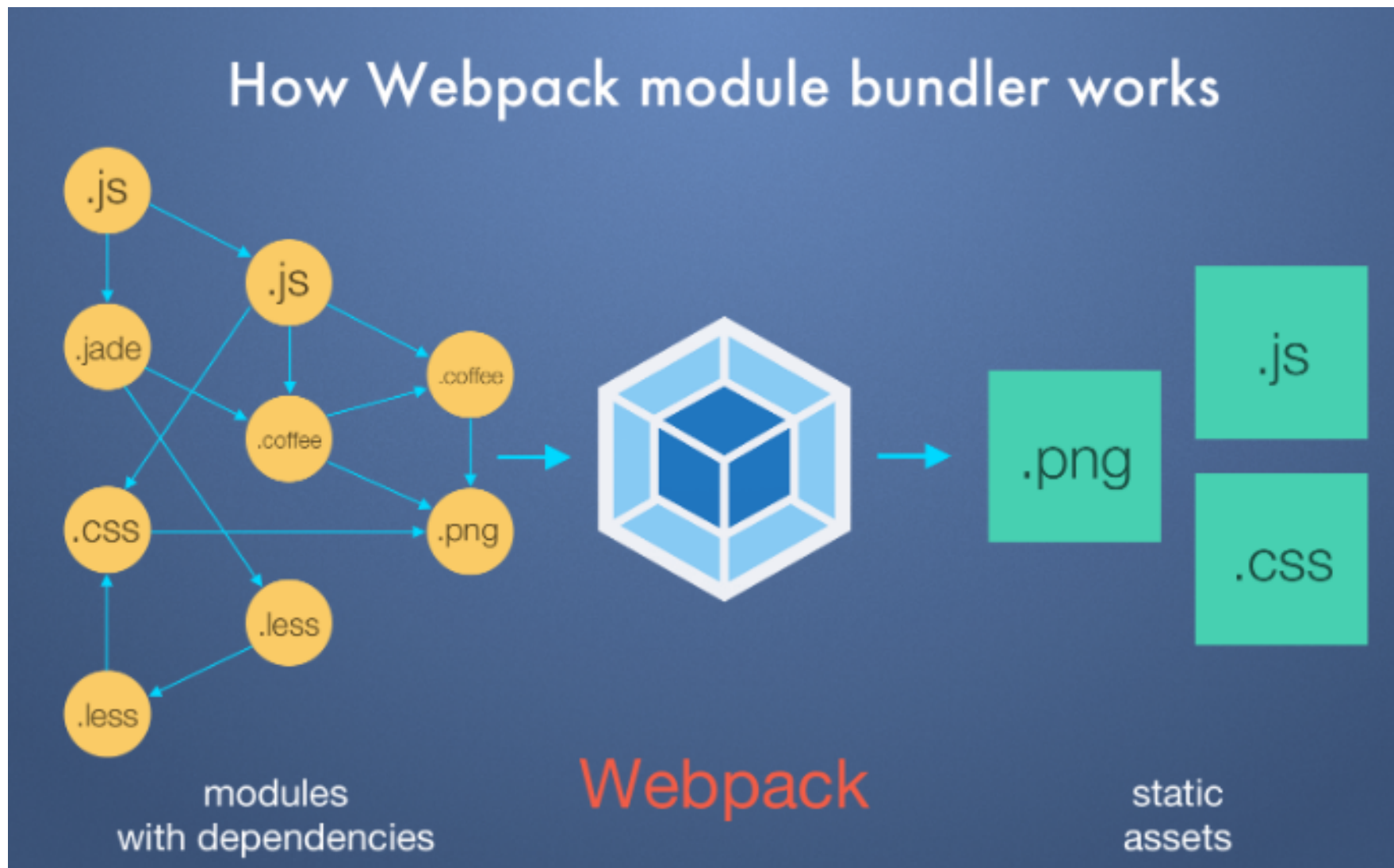
+ vue

Downloads in past 2 Years ▾





Por que angular? – Webpack





Por que angular? – Bibliotecas Javascript



webpack

- Resolve dependências em uma aplicação modularizada.
- Não precisamos de uma biblioteca para cada etapa.
- Inicia um servidor web.
- Monitora alterações nos arquivos durante o desenvolvimento
- Extensível com plugins



Por que angular? – SPA

Vantagens

- Roteamento de páginas é feito no cliente.
- Renderização no cliente
- Melhor experiência do usuário
- Servidor troca apenas dados com a aplicação (json)

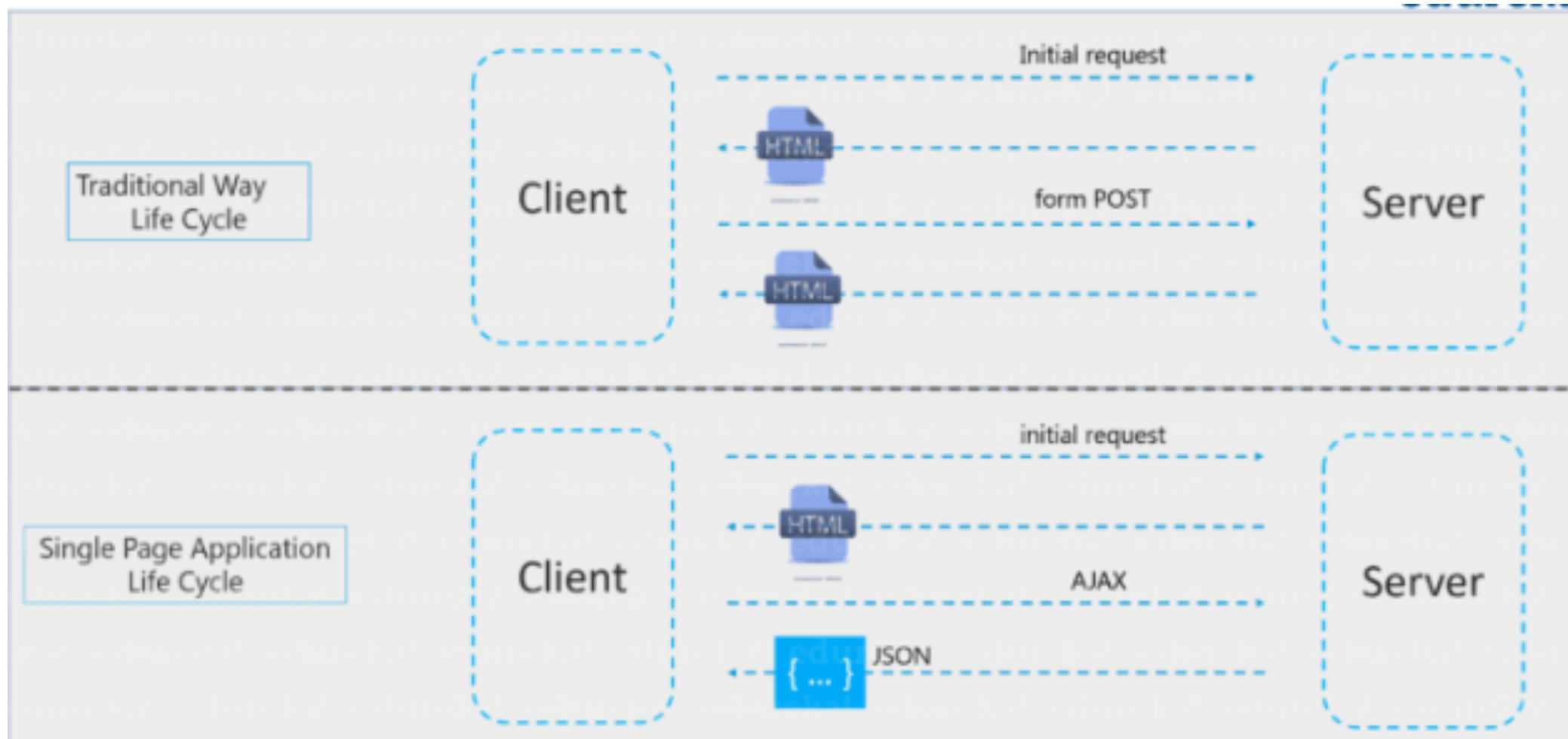
Desvantagens

- Indexadores de busca não pegam js
- Problemas com memory leak



Por que angular? – SPA

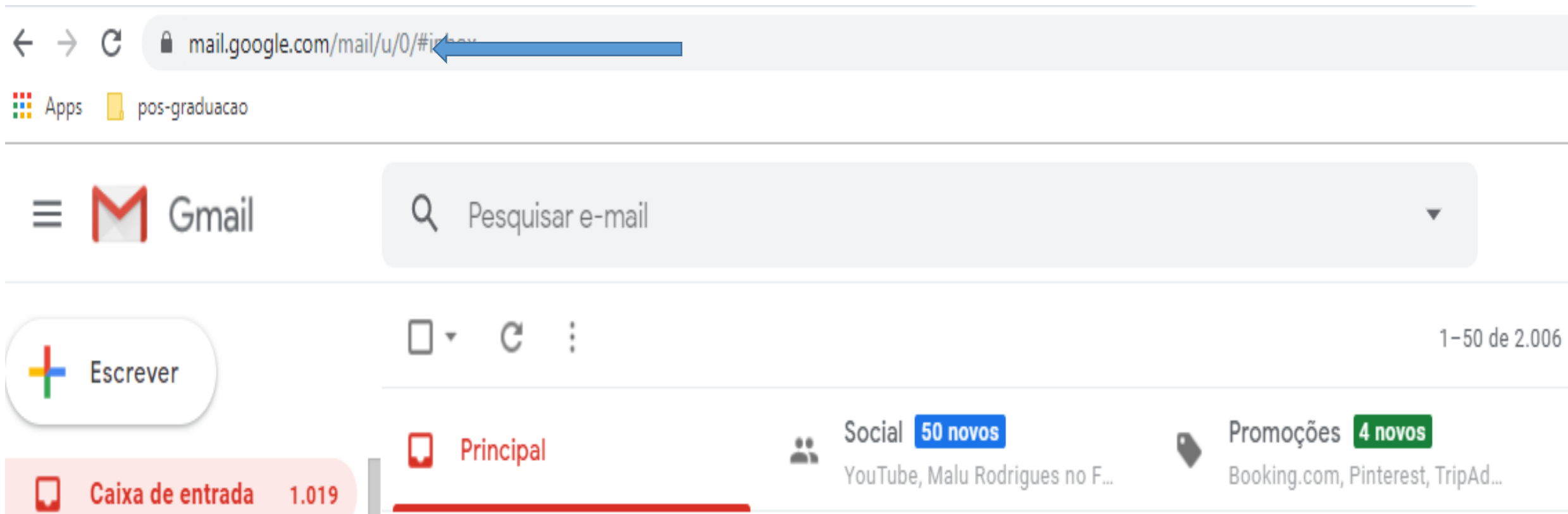
Ciclo da aplicação





Por que angular? – SPA

Não precisa ter a “#”





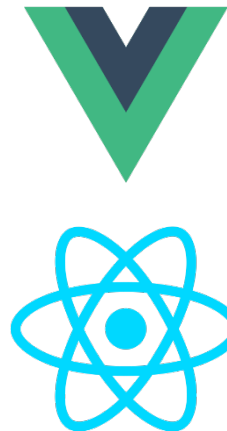
Por que angular? – SPA

Frameworks/Bibliotecas mais utilizados



Framework

X



Bibliotecas



Por que angular? – SPA

Angular

Vantagens:

- Adoção do Typescript facilita migração de back-end developers
- Typescript facilita uso de OO no cliente-side
- Solução mais completa e robusta facilita implementação de regras de negócio complexas

Desvantagens:

- Muitos pacotes desnecessários criam bundle maior
- DirtyChecking pode levar a problemas de performance se mal utilizado
- Curva de aprendizagem acentuada



Por que angular? – SPA

React

Vantagens:

- Não é framework, o bundle é menor
- Maior liberdade é vantagem onde performance é crônica
- Algoritmo de Virtual DOM

Desvantagens:

- Precisa ser combinado com outras bibliotecas para ter “stack completo”
- Necessidade de aprender mais bibliotecas
- Equipe com alta rotatividade pode ser ruim



Por que angular? – SPA

Vue

Vantagens:

- É o mais leve de todos
- Une o melhor dos dois mundos
- Algoritmo de Virtual DOM


Desvantagens:

- Precisa ser combinado com outras bibliotecas para ter “stack completo”
- Necessidade de aprender mais bibliotecas
- Equipe com alta rotatividade pode ser ruim
- Não tem uma grande empresa dando suporte



Por que angular? – SPA

Frameworks/Bibliotecas mais utilizados

 npm trends

react vs vue vs @angular/core vs react-router

Enter an npm package...

react x

vue x

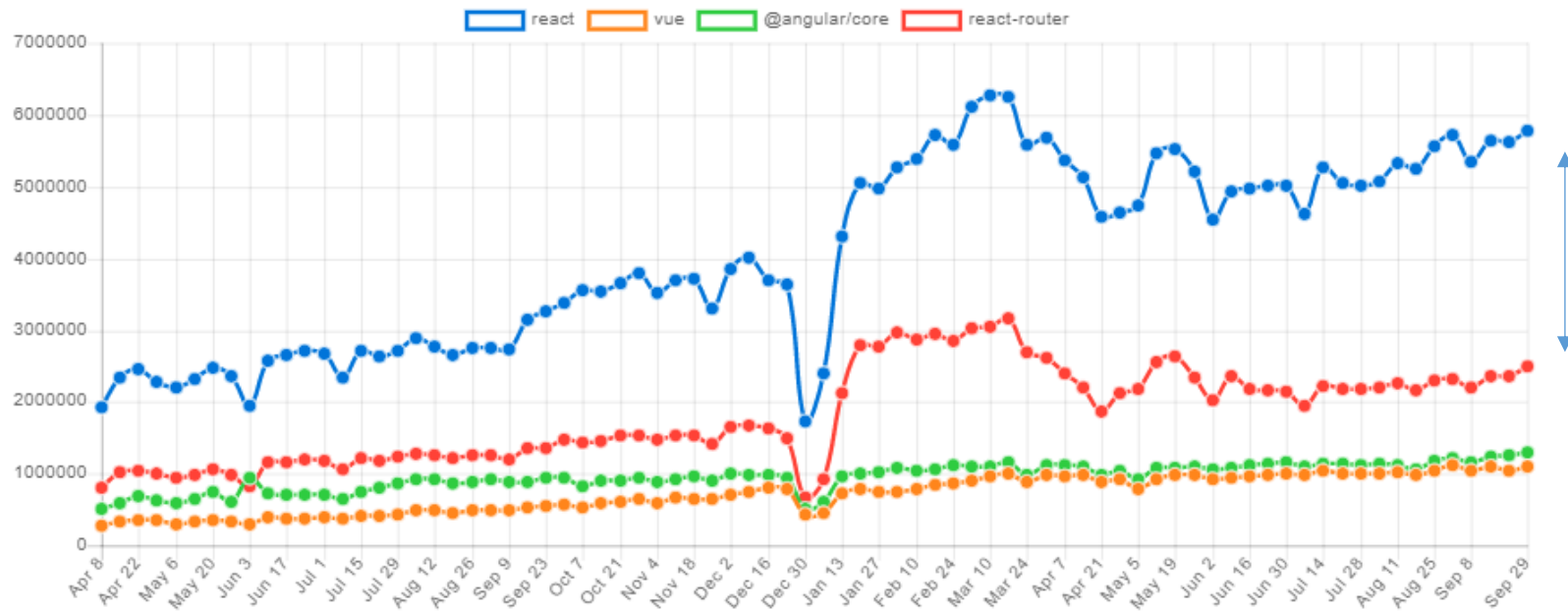
@angular/core x

react-router x

+ angular

+ ember-source

Downloads in past 2 Years v





Por que angular? – Angular finalmente

- Parceria Google e Microsoft
- Open Source
- Quebra da versão 1 (angularjs)
- Uso do angular CLI (webpack)
- Uso de Web Components
- Injeção de dependências
- É um framework SPA completo



Por que angular? – Angular CLI

- Ferramenta que encapsula o webpack(plug-ins pré-configurados)
- Necessário baixar o nodejs
- Necessário baixar o angular/cli

```
npm install -g @angular/cli
```

```
ng new NomeProjeto
```



Por que angular? – Angular CLI

- Compilar para produção

```
ng build --prod
```

```
Generating ES5 bundles for differential loading...
```

```
index.html x
dist > serviveworker > index.html
1  <!doctype html>
2  <html lang="en">
3  <head>
4    <meta charset="utf-8">
5    <title>Serviveworker</title>
6    <base href="/">
7    <meta name="viewport" content="width=device-width, initial-scale=1">
8    <link rel="icon" type="image/x-icon" href="favicon.ico">
9    <link rel="manifest" href="manifest.webmanifest">
10   <meta name="theme-color" content="#1976d2">
11   <link rel="stylesheet" href="styles.3ff695c00d717f2d2a11.css"></head>
12   <body>
13     <app-root></app-root>
14     <noscript>Please enable JavaScript to continue using this application.
15   <script src="polyfills-es5.3aa54d3e5134f5b5b842.js" nomodule defer></scr
16   </html>
```

```
dist
├── serviveworker
│   ├── assets
│   │   ├── 3rdpartylicenses.txt
│   │   ├── 5-es5.dfe5513c46e1d8bd4086.js
│   │   ├── 5-es2015.dfe5513c46e1d8bd4086.js
│   │   ├── favicon.ico
│   │   └── index.html
│   ├── main-es5.284fee2a9f7e62581e8e.js
│   ├── main-es2015.284fee2a9f7e62581e8e.js
│   ├── manifest.webmanifest
│   ├── ngsw-worker.js
│   ├── ngsw.json
│   ├── polyfills-es5.3aa54d3e5134f5b5b842.js
│   ├── polyfills-es2015.fd917e7c3ed57f282ee5.js
│   ├── runtime-es5.77fc8190150cd1c2b4f9.js
│   ├── runtime-es2015.77fc8190150cd1c2b4f9.js
│   ├── safety-worker.js
│   ├── styles.3ff695c00d717f2d2a11.css
│   └── worker-basic.min.js
```



Por que angular? – Typescript

- TypeScript mantido pela Microsoft
- Tipagem de javascript
- Transpilador que converte código para outra spec (Babel)

```
interface IComponent{  
    getId() : string;  
}  
  
class Button implements IComponent{  
    id:string;  
    getId():string{  
        return this.id;  
    }  
}
```

```
var Button = (function () {  
    function Button() {  
    }  
    Button.prototype.getId = function () {  
        return this.id;  
    };  
    return Button;  
})();
```



Por que angular? – Typescript

- ES6(EC2015)

```
1  const sum = (x, y) => {  
2    return x + y  
3  }  
4  
5  sum(1, 2) // 3
```

ES5

```
1  var sum = function(x, y) {  
2    return x + y;  
3  };  
4  
5  sum(1, 2); // 3
```



Por que angular? – Typescript

- Tipagem de dados

```
add(name: string): void {  
  name = name.trim();  
  if (!name) { return; }  
  this.heroService.addHero({ name } as Hero)  
    .subscribe(hero => {  
    this.heroes.push(hero);  
  });  
}
```

```
add(name: any) {  
  name = name.trim();  
  if (!name) { return; }  
  this.heroService.addHero({ name } as Hero)  
    .subscribe(hero => {  
    this.heroes.push(hero);  
  });  
}
```




Por que angular? – Componentes

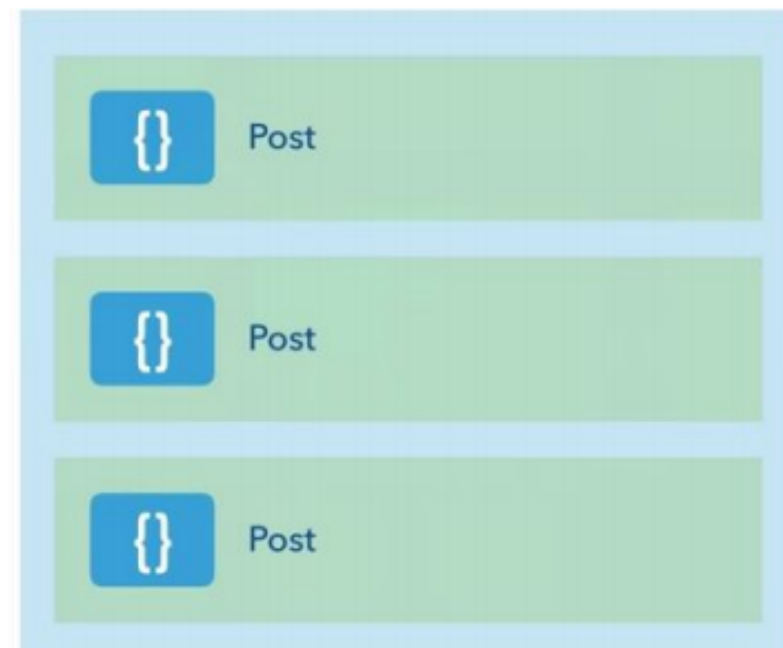
- Web componentes
- Encapsula padrões (shadow DOM)

```
<h1>{{title}}</h1>
<nav>
  <a routerLink="/dashboard">Dashboard</a>
  <a routerLink="/heroes">Heroes</a>
</nav>
<router-outlet></router-outlet>
<app-messages></app-messages>
```



Por que angular? – Componentes

- Cada parte é um componente.





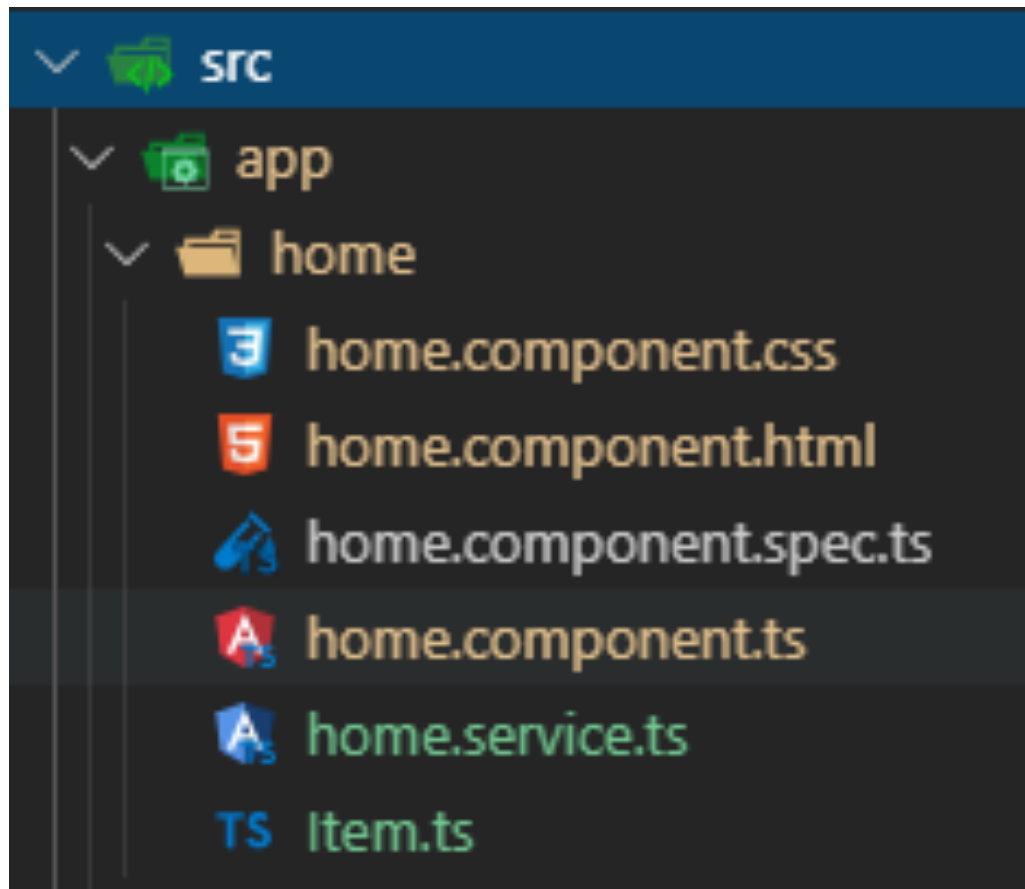
Por que angular? – Componentes

- Componente é composto por um template (html)
- Estilo (Css)
- Metadados
- Injeção de serviços



Por que angular? – Componentes

- Estrutura do componente





Por que angular? – Componentes

```
import { Component, OnInit } from '@angular/core';
import { HomeService } from '../home.service';
import { takeWhile } from 'rxjs/operators';
import { Item } from '../Item';
```

```
@Component({
  selector: 'app-home',
  templateUrl: '../home.component.html',
  styleUrls: ['../home.component.css']
})
```

```
export class HomeComponent implements OnInit {
  alive: boolean = true;
  list: Item[] = [];
  constructor(private home: HomeService) { }
```

```
  ngOnInit() {
  }

  ngOnDestroy() {
    this.alive = false;
  }
```

```
  trazerLista() {
    this.home.getListaMeetup().pipe(takeWhile(() => this.alive)).subscribe((item: Item) => {
      this.list.push(item);
    });
  }

  limparLista() {
    this.list = [];
  }
}
```

Área de Importação (ES2015)

Metadados

Injeção de dependência

Ciclo de vida do componente



Por que angular? – Componentes

```
import { Component, OnInit } from '@angular/core';
import { HomeService } from '../home.service';
import { takeWhile } from 'rxjs/operators';
import { Item } from '../Item';
```

```
@Component({
  selector: 'app-home',
  templateUrl: '../home.component.html',
  styleUrls: ['../home.component.css']
})
```

```
export class HomeComponent implements OnInit {
  alive: boolean = true;
  list: Item[] = [];
  constructor(private home: HomeService) { }
```

```
  ngOnInit() {
  }
```

```
  ngOnDestroy() {
    this.alive = false;
  }
```

```
  trazerLista() {
    this.home.getListaMeetup().pipe(takeWhile(() => this.alive)).subscribe((item: Item) => {
      this.list.push(item);
    });
  }
```

```
  limparLista() {
    this.list = [];
  }
}
```

```
<button type="button" (click)="trazerLista()" style="margin-right: 5px">Trazer Lista</button>
<button type="button" (click)="limparLista()">Limpar Lista</button>
<ul class="lista">
  <li *ngFor="let item of list">{{item.id}} - {{item.descricao}}</li>
</ul>
```

Todas as variáveis públicas são observadas pelo Angular!

Two way data-bind em ação!



Por que angular? – Componentes

```
import { Component, OnInit } from '@angular/core';
import { HomeService } from '../home.service';
import { takeWhile } from 'rxjs/operators';
import { Item } from '../Item';
```

```
@Component({
  selector: 'app-home',
  templateUrl: '../home.component.html',
  styleUrls: ['../home.component.css']
})
export class HomeComponent implements OnInit {
  alive: boolean = true;
  list: Item[] = [];
  constructor(private home: HomeService) { }
```

```
<button type="button" (click)="trazerLista()" style="margin-right: 5px">Trazer Lista</button>
<button type="button" (click)="limparLista()">Limpar Lista</button>
<ul class="lista">
  <li *ngFor="let item of list">{{item.id}} - {{item.descricao}}</li>
</ul>
```

Eventos disparados pelo componente

```
ngOnInit() {
}

ngOnDestroy() {
  this.alive = false;
}

trazerLista() {
  this.home.getListaMeetup().pipe(takeWhile(() => this.alive)).subscribe((item: Item) => {
    this.list.push(item);
  });
}

limparLista() {
  this.list = [];
}
}
```



Por que angular? – Module

```
import { UpdateService } from './update.service';  
import { HomeService } from './home/home.service';
```

```
@NgModule({  
  declarations: [  
    AppComponent,  
    HomeComponent  
  ],  
  imports: [  
    BrowserModule,  
    AppRoutingModule,  
    HttpClientModule,  
    ServiceWorkerModule.register('ngsw-worker.js', { enabled: environment.production })  
  ],  
  providers: [UpdateService, HomeService],  
  bootstrap: [AppComponent]  
})  
export class AppModule { }
```

Componentes e diretivas do módulo

Importa outros módulos(SharedModule)



Por que angular? – Module

```
import { UpdateService } from './update.service';
import { HomeService } from './home/home.service';

@NgModule({
  declarations: [
    AppComponent,
    HomeComponent
  ],
  imports: [
    BrowserModule,
    AppRoutingModule,
    HttpClientModule,
    ServiceWorkerModule.register('ngsw-worker.js', { enabled: environment.production })
  ],
  providers: [UpdateService, HomeService],
  bootstrap: [AppComponent]
})
export class AppModule { }
```

Injeção de dependência



Por que angular? – Roteamento

```
<div class="content" role="main">
  <a routerLink="/home">Home Component</a>
  <a routerLink="/lazy">Lazy Module</a>
  <router-outlet></router-outlet>
</div>
```

O componente da rota será carregado aqui

```
import { Routes, RouterModule } from '@angular/router';
import { HomeComponent } from '../home/home.component';

const routes: Routes = [
  {path: '', redirectTo: 'home', pathMatch: 'full'},
  {path: 'home', component: HomeComponent},
  {path: 'lazy', loadChildren: '../lazy/lazy.module#LazyModule'}
];

@NgModule({
  imports: [RouterModule.forRoot(routes, { useHash: true })],
  exports: [RouterModule]
})
export class AppRoutingModule { }
```

Lazy loading

Uso da “#”



Por que angular? – Roteamento

localhost:4200/#/lazy

Apps pos-graduacao



Welcome serviceworker teste

Home Component
Lazy Module

lazy load com mudanca no servidor hahahah

Arquivo carregado sob demanda

Elements Console Sources Network Performance Memory Application Security Audits			
Filter <input type="checkbox"/> Hide data URLs All XHR JS CSS Img Media Font Doc WS Manifest Other			
20 ms 40 ms 60 ms 80 ms 100 ms 120 ms 140 ms 160 ms 180 ms 200 ms			
Name	Status	Type	Initiator
lazy-lazy-module.js	304	script	bootstrap:149



Por que angular? – Comunicação entre componentes

- A comunicação do pai para o filho é feita via Input()
- A comunicação do filho para o pai é feita via Output()

```
<div>
  <span>{{fraseVaderPai}}</span>
  <span>{{respostaFilho}}</span>
  <app-luke-skywalker [Frase]="fraseVaderPai" (LukeListener)="OuveFilho($event)"></app-luke-skywalker>
</div>
```

```
export class LukeSkywalkerComponent implements OnInit {

  @Input() Frase: string;
  @Output() LukeListener: EventEmitter<string> = new EventEmitter<string>();

  constructor() { }

  ngOnInit() {

  }

  ngOnChanges(){
    this.LukeListener.emit("Nãoooooooooooooooo!");
  }
}
```

```
fraseVaderPai: string = "Luke, eu sou seu pai!";
respostaFilho: string;

constructor() { }

ngOnInit() {

}

OuveFilho(event) {
  this.respostaFilho = event;
}
```



Por que angular? – Comunicação entre componentes

- A comunicação do pai para o filho é feita via Input()
- A comunicação do filho para o pai é feita via Output()

```
<div>
  <span>{{fraseVaderPai}}</span>
  <span>{{respostaFilho}}</span>
  <app-luke-skywalker [Frase]="fraseVaderPai" (LukeListener)="OuveFilho($event)"></app-luke-skywalker>
</div>
```

```
export class LukeSkywalkerComponent implements OnInit {

  @Input() Frase: string;
  @Output() LukeListener: EventEmitter<string> = new EventEmitter<string>();

  constructor() { }

  ngOnInit() {

  }

  ngOnChanges(){
    this.LukeListener.emit("Nãoooooooooooooooo!");
  }
}
```

```
fraseVaderPai: string = "Luke, eu sou seu pai!";
respostaFilho: string;

constructor() { }

ngOnInit() {
}

OuveFilho(event) {
  this.respostaFilho = event;
}
```



Por que angular? – Diretivas

- Diretivas manipulam o DOM
- Componente sem template html

```
@Directive({
  selector: '[posicaoMouse]',
})
export class MouseOverDirective {

  @Output() novaposicao: EventEmitter<any> = new EventEmitter<any>();

  constructor(private el: ElementRef, private render: Renderer2) {
  }

  @HostListener('mousemove', ['$event']) onMouseMove(event) {
    this.novaposicao.emit({ X: event.clientX, Y: event.clientY });
    this.render.setStyle(this.el.nativeElement, "border-radius", "20px");
  }

  @HostListener('mouseout', ['$event']) onMouseOut(event) {
    this.novaposicao.emit({ X: event.clientX, Y: event.clientY });
    this.render.removeStyle(this.el.nativeElement, "border-radius");
  }
}
```

```
<span>X:{{posicaoX}}</span>
<span>Y:{{posicaoY}}</span>
<div posicaoMouse (novaposicao)="getnovaposicao($event)">

</div>
```



Por que angular? – Service worker

- Suporte a service worker

```
import { SwUpdate } from '@angular/service-worker';

@Injectable()
export class UpdateService {
  constructor(private swUpdate: SwUpdate) {
    this.swUpdate.available.subscribe(evt => {
      // uma atualização da aplicação está disponível...
      console.log(evt);
    });
  }
}
```



Por que angular? – Tópicos extras

- Angular Universal (SSR) – Server side rendering
- Disponível aos web crawlers
- Melhor desempenho
- Mostrar primeira página rapidamente(first contentful paint)



Por que angular? – RXJS

- Biblioteca para programação reativa
- Observables
- Exemplos do angular utilizam RxJS

<https://rxjs.dev/guide/overview>

```
document.addEventListener('click', () => console.log('Clicked!'));
```

```
import { fromEvent } from 'rxjs';
```

```
fromEvent(document, 'click').subscribe(() => console.log('Clicked!'));
```