INFINUM

Server–side rendering in React and Angular

FILIP VOSKA
IVICA BATINIĆ

IVICA BATINIĆ

Lead Engineer

FILIP VOSKA

Lead Engineer

AGENDA

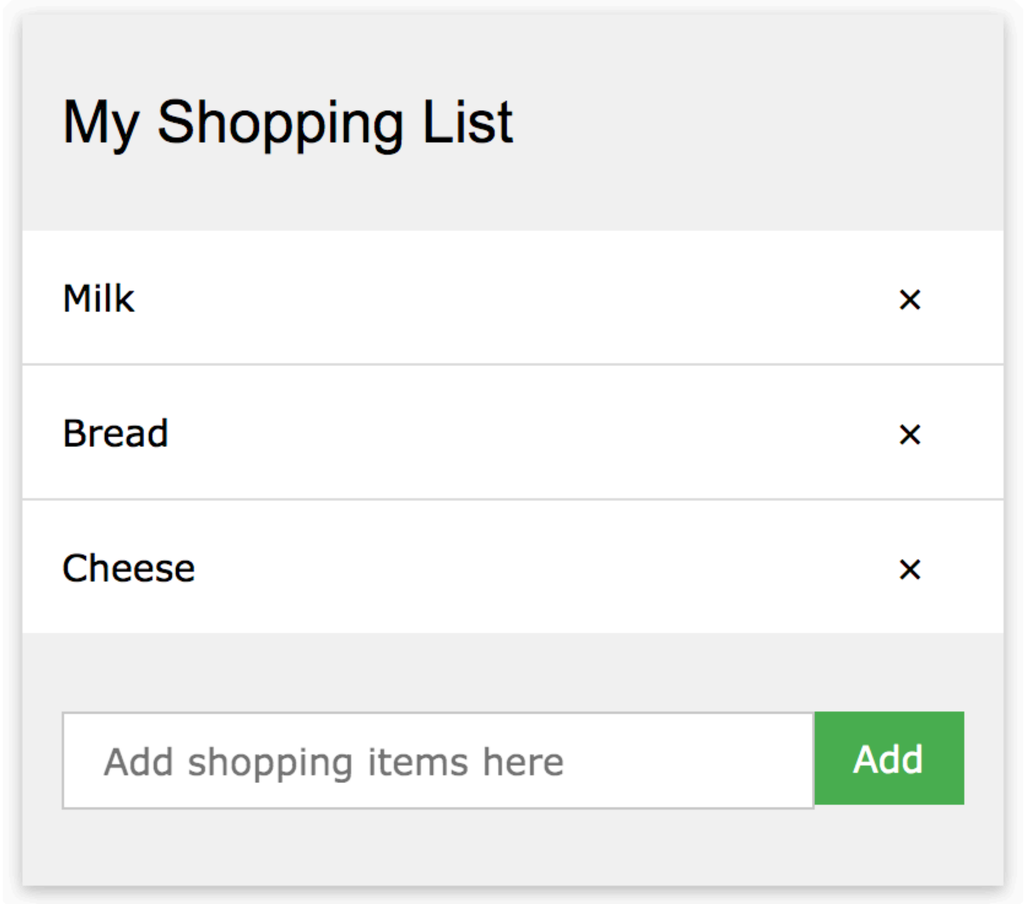
1. Why SSR
2. SSR Challenges
3. React
4. Angular

01

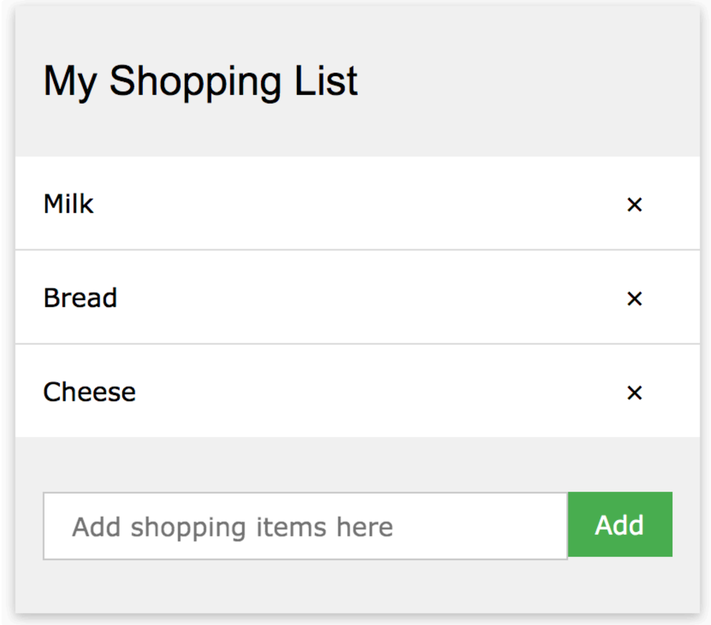
WHY SERVER-SIDE RENDERING?

WHAT DO FRONTEND LIBRARIES DO?

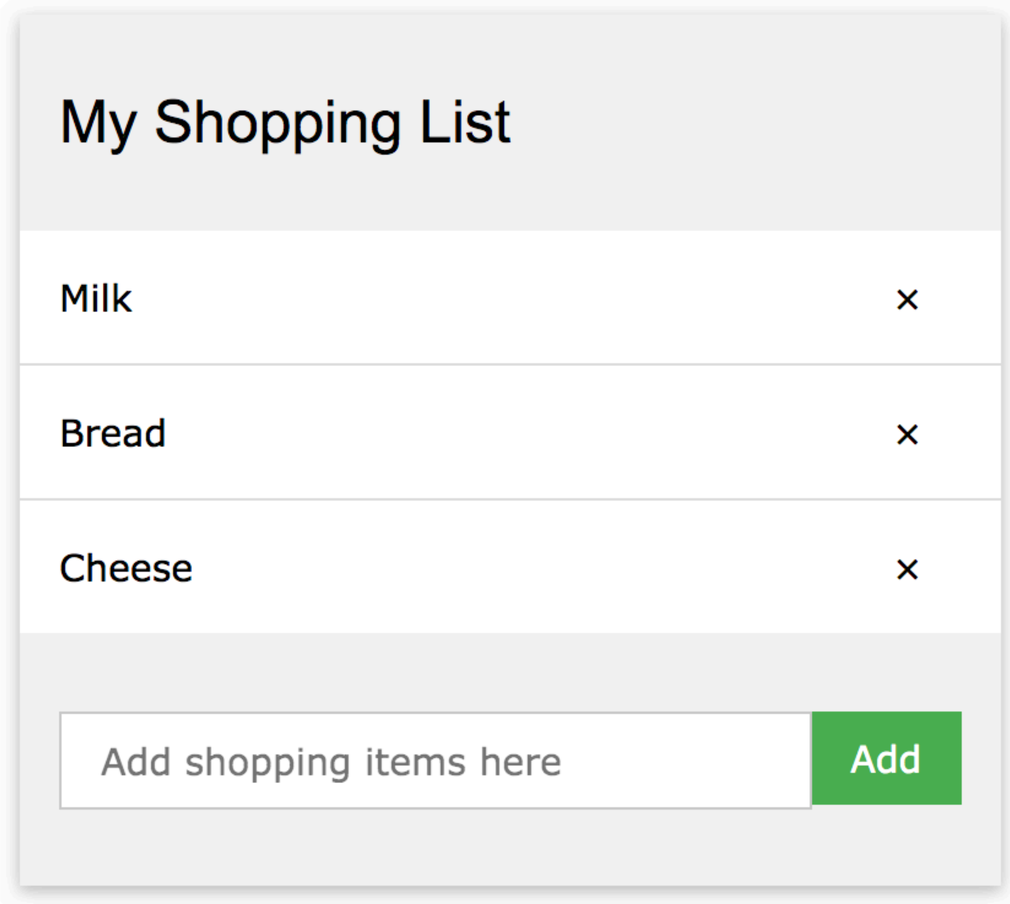
ANGULAR.JS



https://www.w3schools.com/angular/angular_application.asp



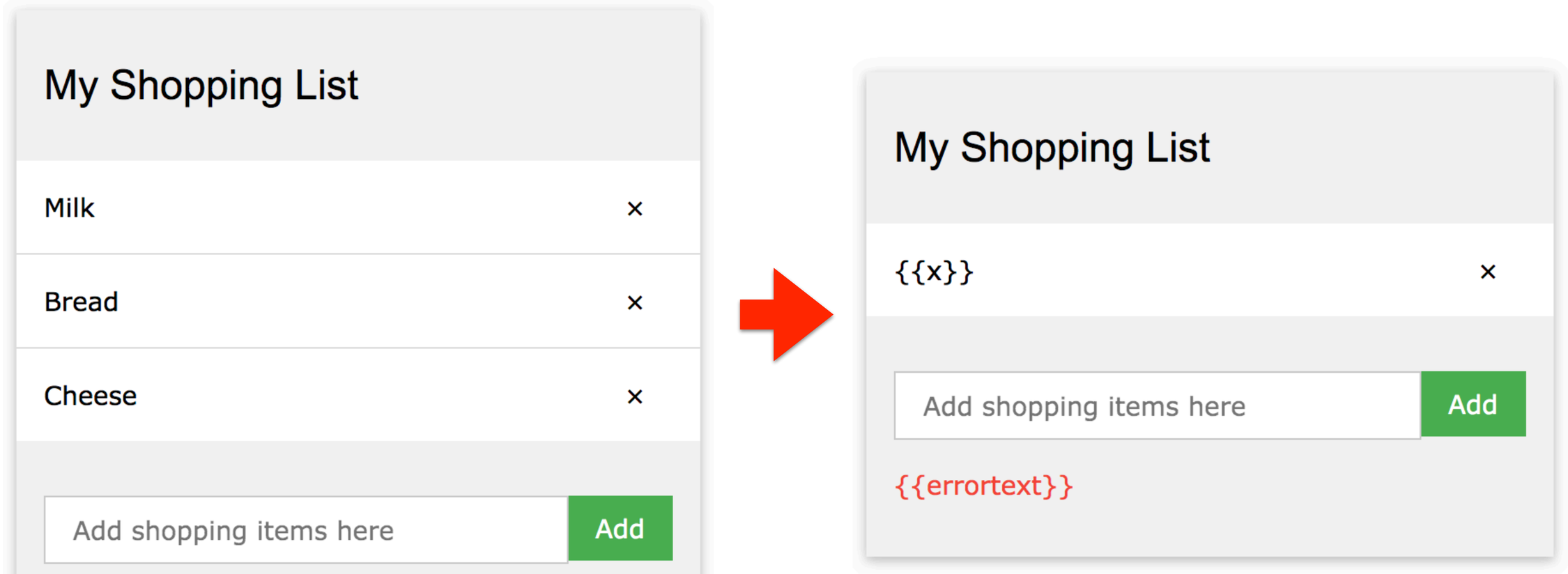

```
<script>
  var app = angular.module("myShoppingList", []);
  app.controller("myCtrl", function($scope) {
    $scope.products = ["Milk", "Bread", "Cheese"];
  });
</script>

<div ng-app="myShoppingList" ng-controller="myCtrl">
  <ul>
    <li ng-repeat="x in products">{{x}}</li>
  </ul>
</div>
```



Debugger

Disable JavaScript

Disable async stack traces


My Shopping List
Milk
Bread
Cheese
Add shopping items here
Add
My Shopping List
{{x}}
Add shopping items here
Add
{{errortext}}

```
1 <ul>
2   <li ng-repeat="x in products">{{x}}</li>
3 </ul>
```

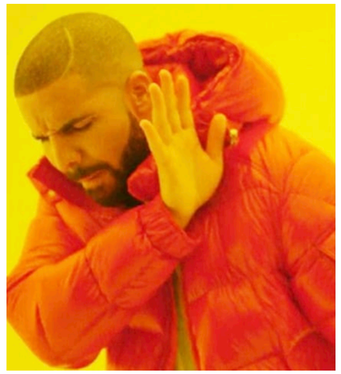
My Shopping List

{{x}} ×

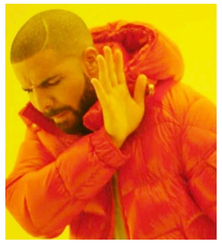
Add shopping items here | Add

{{errortext}}

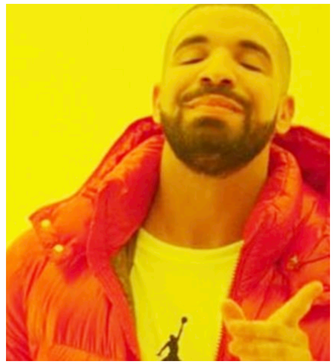
OK, BUT THAT WAS *ANGULAR.JS*...



OK, BUT THAT WAS *ANGULAR.JS*...



WHAT DO **MODERN** FRAMEWORKS DO?



ANGULAR 2+

Tour of Heroes

Dashboard Heroes

Top Heroes

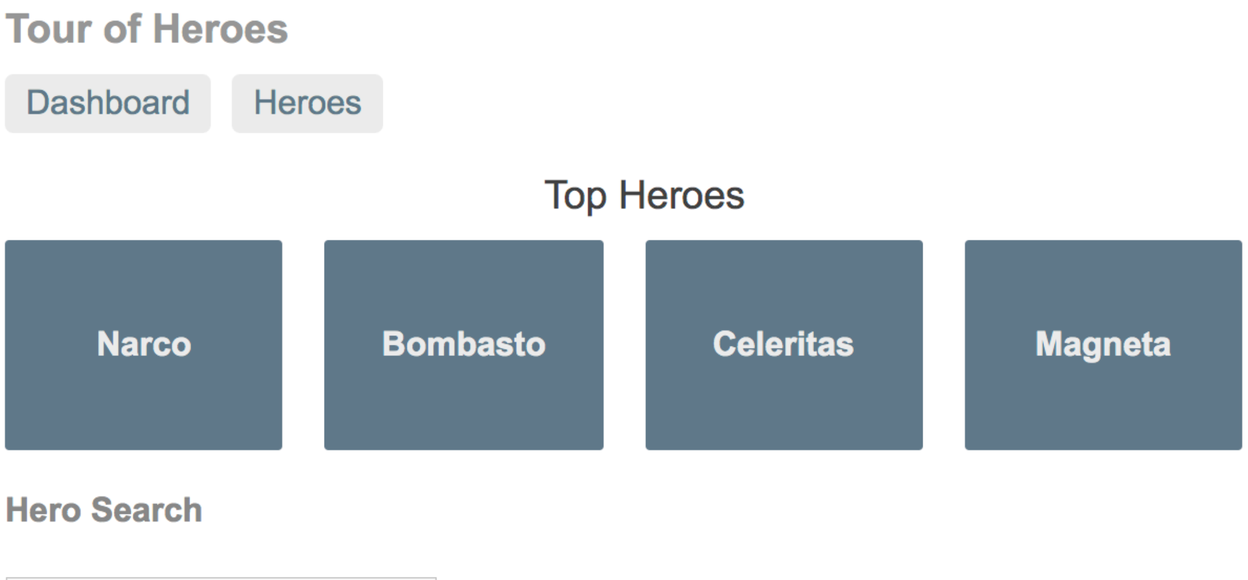
Narco Bombasto Celeritas Magneta

Hero Search

Messages

clear

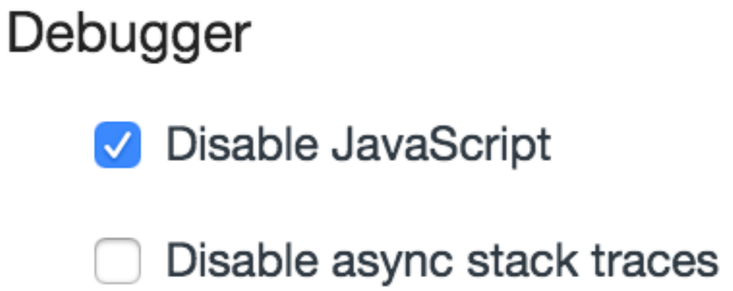
HeroService: fetched heroes




```
<body>
  <app-root>Loading ... </app-root>
</body>
```

Loading...

```
<body>
  <app-root>Loading ...</app-root>
</body>
```


Debugger
Disable JavaScript
Disable async stack traces

Loading...

Debugger

- [x] Disable JavaScript
- [ ] Disable async stack traces

Loading...

```
1 <body>
2   <app-root>Loading ... </app-root>
3 </body>
```

Debugger

- [x] Disable JavaScript
- [ ] Disable async stack traces

THE PROBLEM

Content rendering relies completely on JavaScript.

No JavaScript, no bueno

BUT GOOGLE SHOWS MY PAGE CORRECTLY?

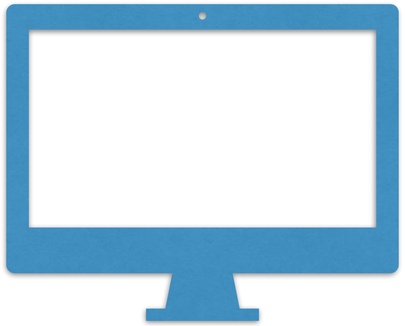
BUT GOOGLE SHOWS MY PAGE CORRECTLY?

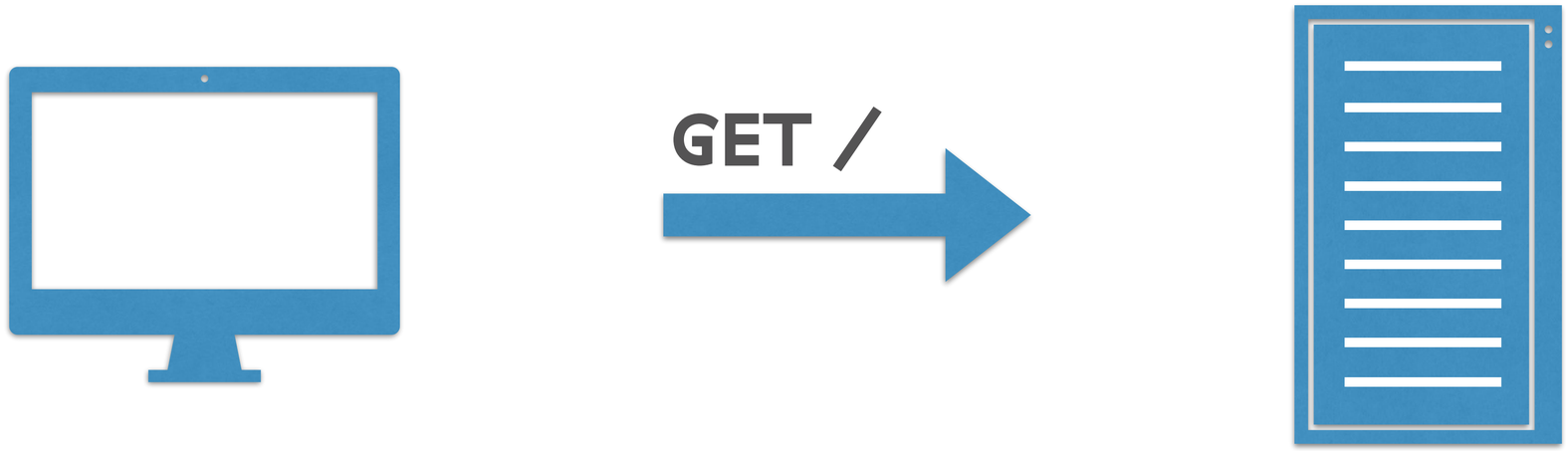
Yes, because they run JavaScript

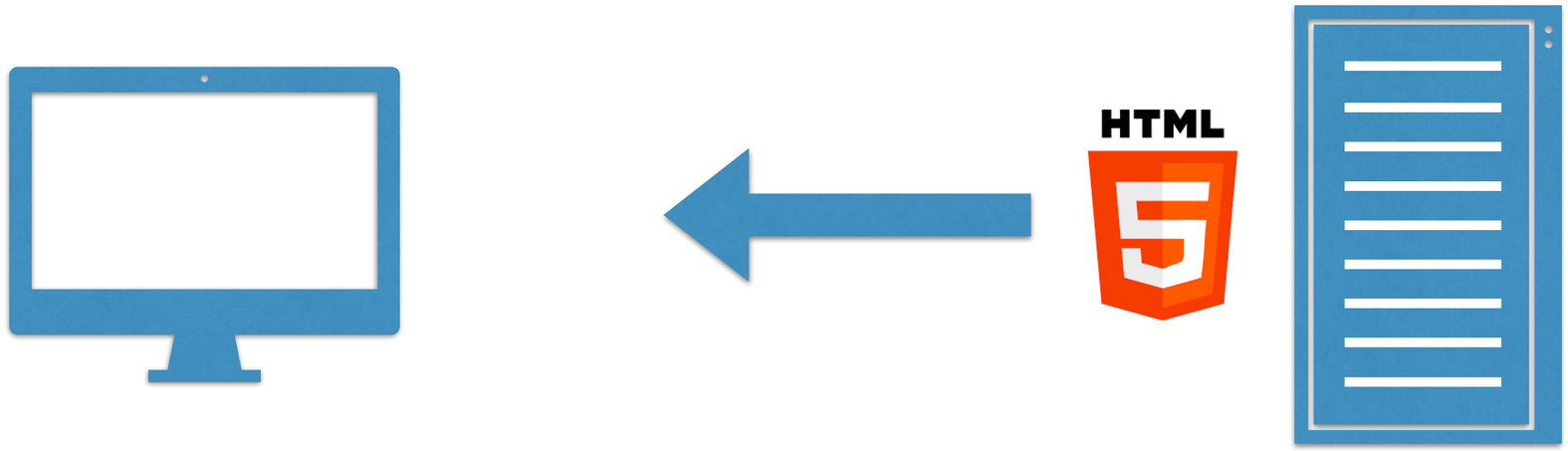
PROBLEMS WITH “VIEW” FRAMEWORKS

- Rendering can be heavy
- Search Engine / Social Media Optimisation
- Time to first contentful paint

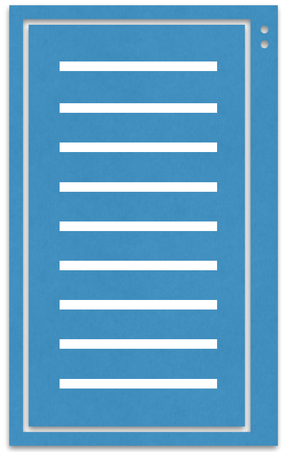
HOW ARE APPS DELIVERED?



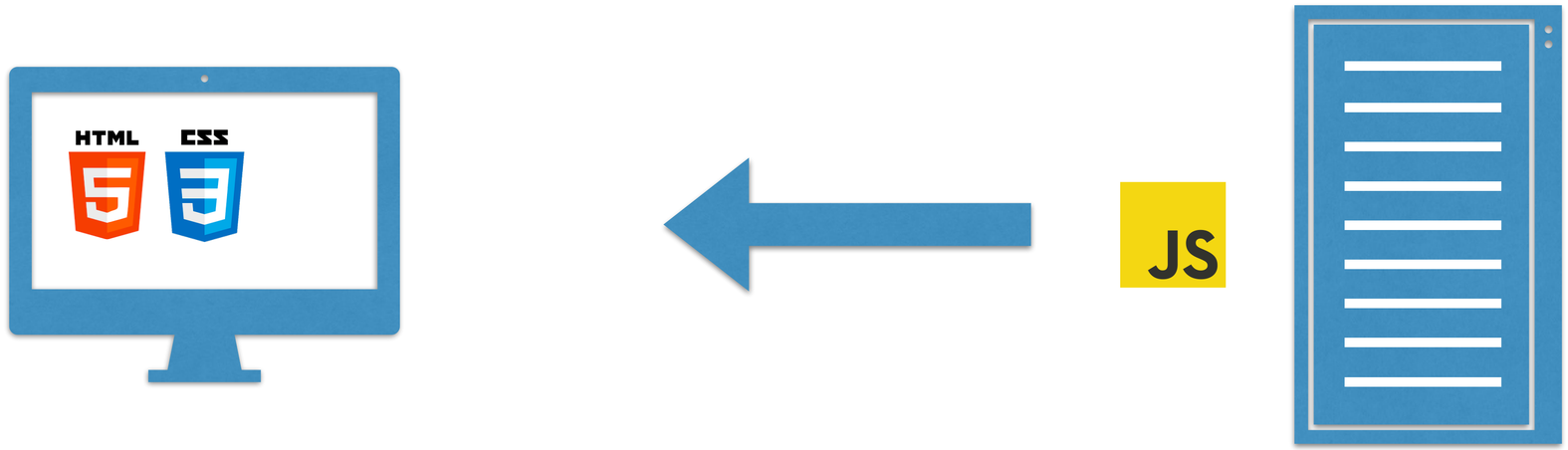

GET /


HTML

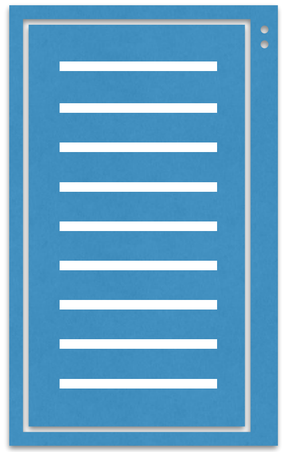
GET /STYLE.CSS



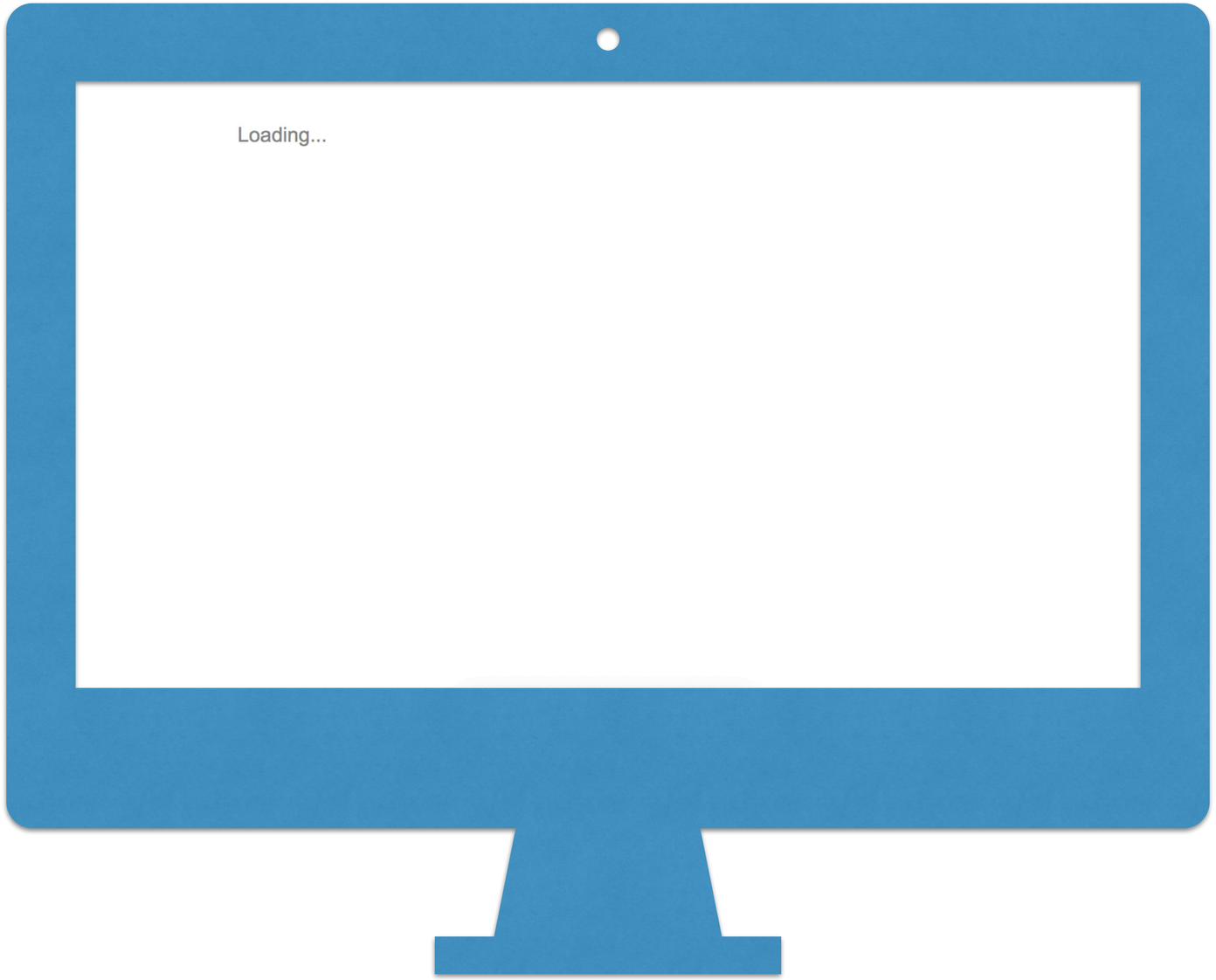

HTML
CSS
GET /MAIN.JS


HTML
CSS
JS


HTML
CSS
JS




HTML
CSS
JS


Loading...

SOME TERMINOLOGY

- TTFB: Time to First Byte
- FP: First Paint
- FCP: First Contentful Paint
- TTI: Time to Interactive

CLIENT-SIDE RENDERING

- TTFB: Time to First Byte
- FP: First Paint
- FCP: First Contentful Paint
- TTI: Time to Interactive

CLIENT-SIDE RENDERING

- TTFB: Time to First Byte 
- FP: First Paint
- FCP: First Contentful Paint
- TTI: Time to Interactive

CLIENT-SIDE RENDERING

- TTFB: Time to First Byte 
- FP: First Paint 
- FCP: First Contentful Paint
- TTI: Time to Interactive

CLIENT-SIDE RENDERING

- TTFB: Time to First Byte 
- FP: First Paint 
- FCP: First Contentful Paint 
- TTI: Time to Interactive

CLIENT-SIDE RENDERING

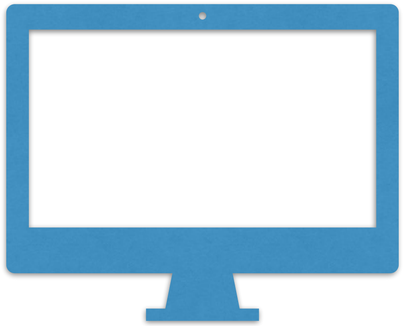
- TTFB: Time to First Byte 
- FP: First Paint 
- FCP: First Contentful Paint 
- TTI: Time to Interactive 

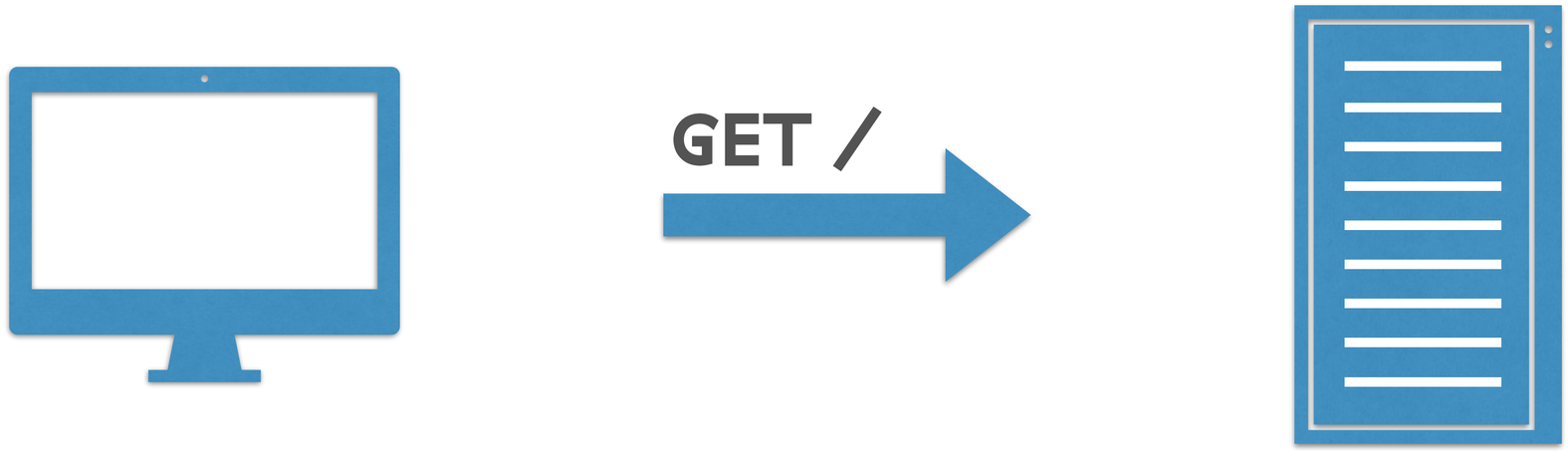
CLIENT-SIDE RENDERING

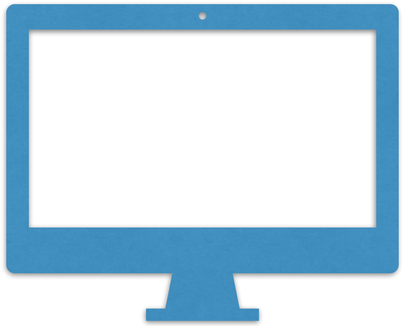
- TTFB: Time to First Byte 
- FP: First Paint 
- FCP: First Contentful Paint 
- TTI: Time to Interactive 

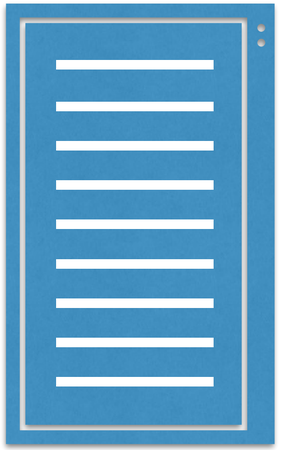
Enter...

SERVER-SIDE RENDERING




GET /

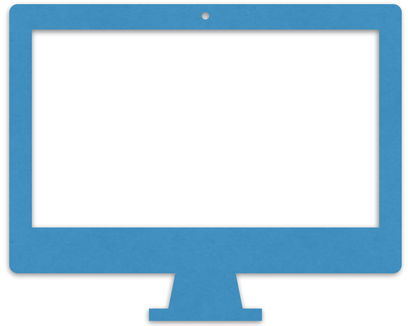


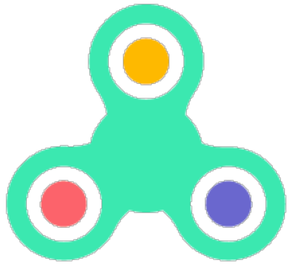


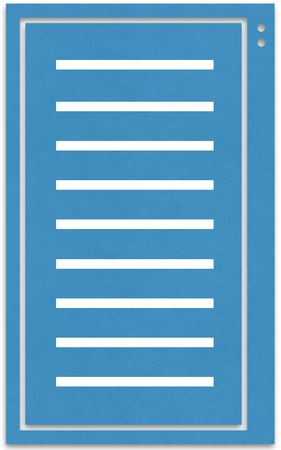

HTML

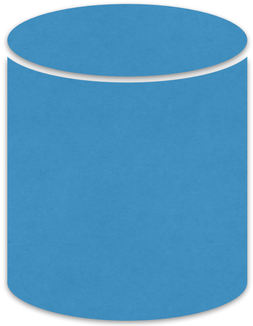

CSS


JS





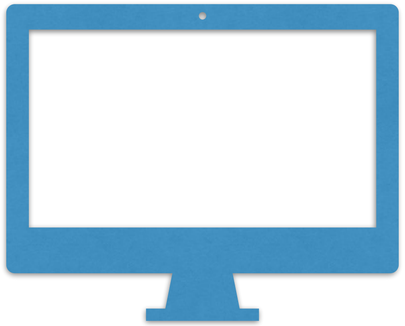


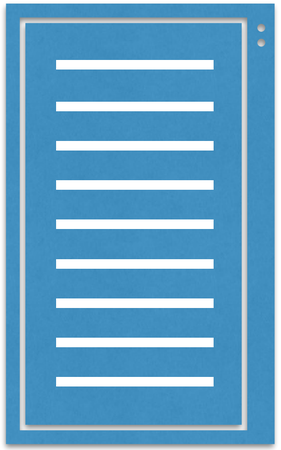



HTML


CSS


JS

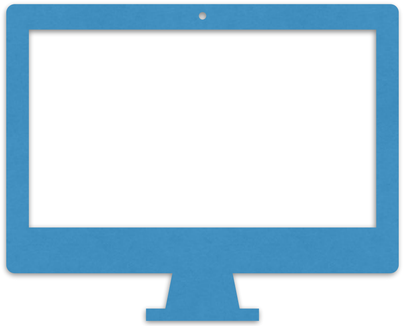


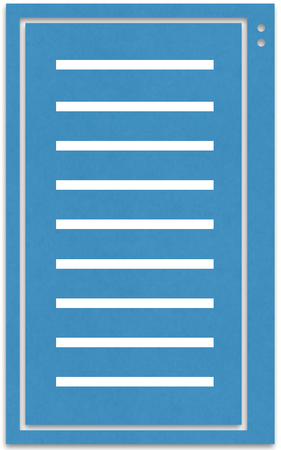


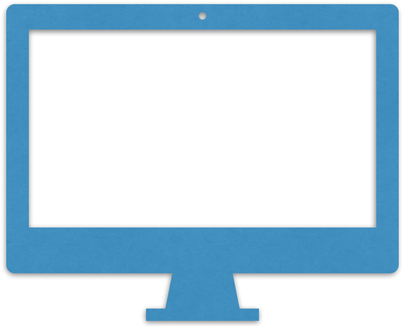

HTML

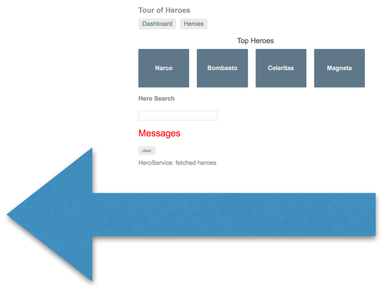

CSS

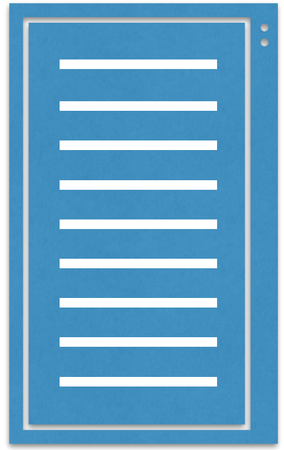

JS

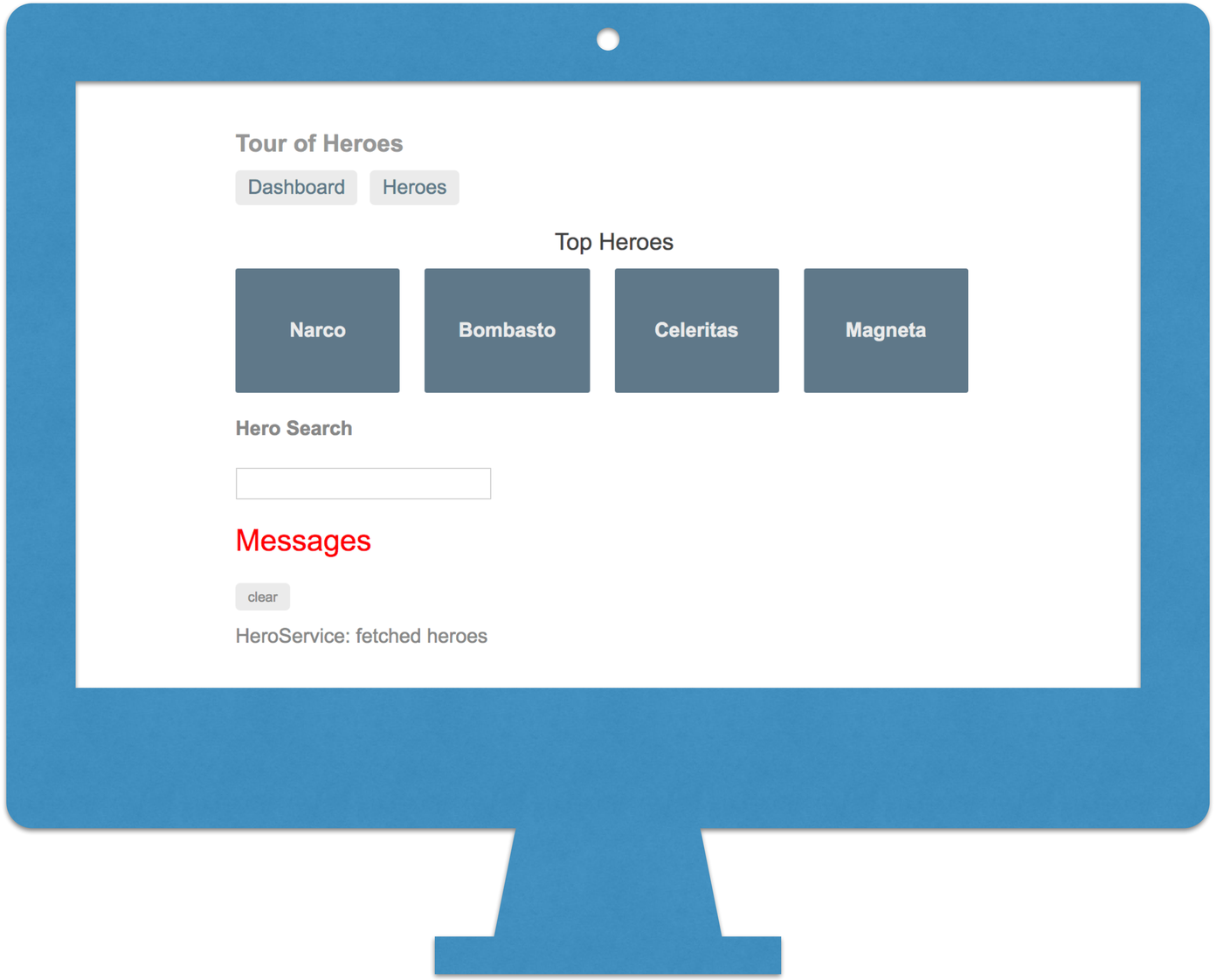


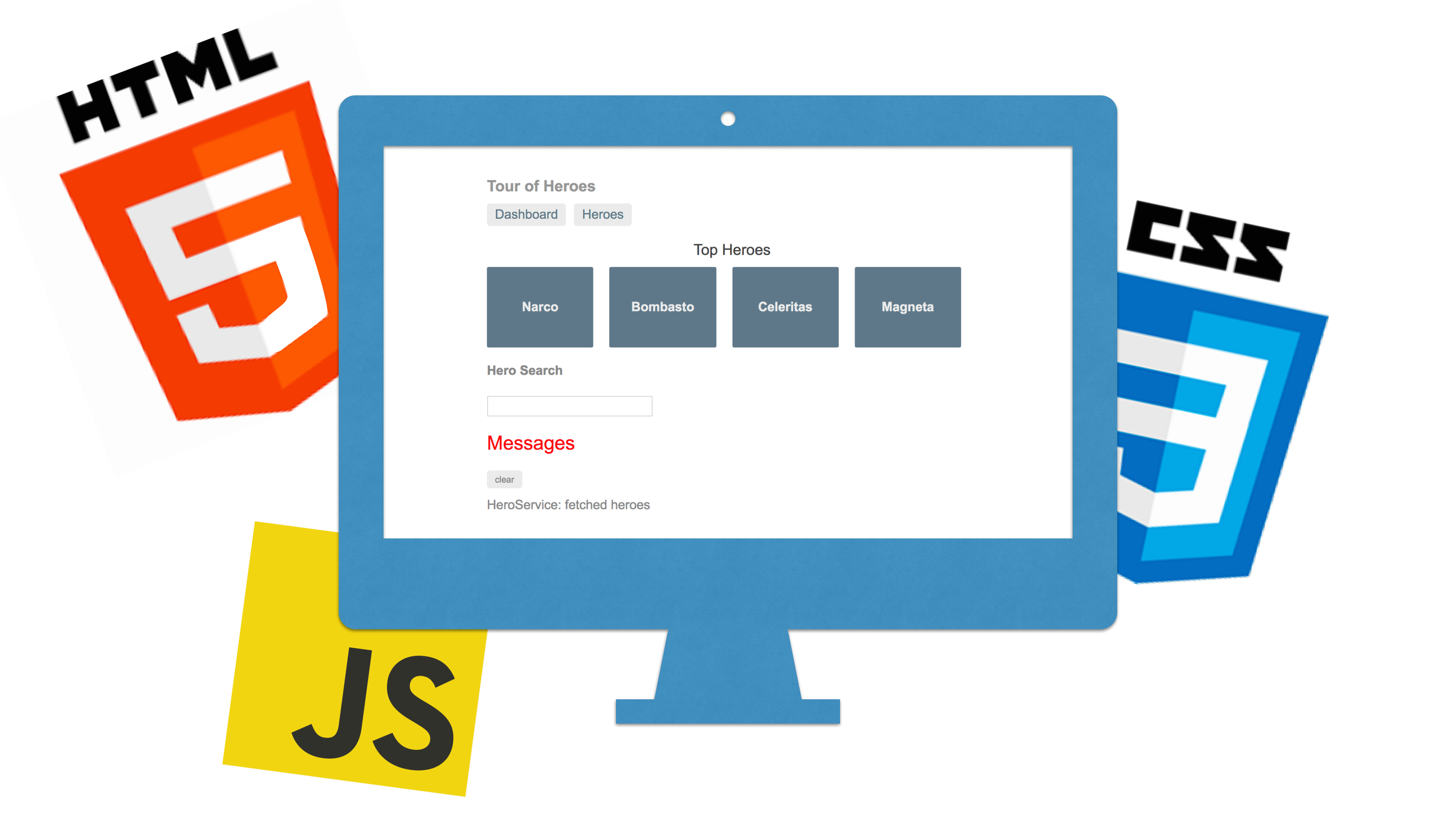





Tour of Heroes
Dashboard
Heroes
Top Heroes
Narco
Bombasto
Celeritas
Magneta
Hero Search
Messages
clear
HeroService: fetched heroes




Tour of Heroes
Dashboard
Heroes
Top Heroes
Narco
Bombasto
Celeritas
Magneta
Hero Search
Messages
clear
HeroService: fetched heroes


HTML
CSS
JS
Tour of Heroes
Dashboard
Heroes
Top Heroes
Narco
Bombasto
Celeritas
Magneta
Hero Search
Messages
clear
HeroService: fetched heroes


Tour of Heroes
Dashboard
Heroes
Top Heroes
Narco
Bombasto
Celeritas
Magneta
Hero Search
Messages
clear
HeroService: fetched heroes

SERVER-SIDE RENDERING

- TTFB: Time to First Byte
- FP: First Paint
- FCP: First Contentful Paint
- TTI: Time to Interactive

SERVER-SIDE RENDERING

- TTFB: Time to First Byte 
- FP: First Paint
- FCP: First Contentful Paint
- TTI: Time to Interactive

SERVER-SIDE RENDERING

- TTFB: Time to First Byte 
- FP: First Paint 
- FCP: First Contentful Paint
- TTI: Time to Interactive

SERVER-SIDE RENDERING

- TTFB: Time to First Byte 
- FP: First Paint 
- FCP: First Contentful Paint 
- TTI: Time to Interactive

SERVER-SIDE RENDERING

- TTFB: Time to First Byte 
- FP: First Paint 
- FCP: First Contentful Paint 
- TTI: Time to Interactive 

SERVER-SIDE RENDERING

- TTFB: Time to First Byte 
- FP: First Paint 
- FCP: First Contentful Paint 
- TTI: Time to Interactive 

CLIENT-SIDE RENDERING

- TTFB: Time to First Byte 
- FP: First Paint 
- FCP: First Contentful Paint 
- TTI: Time to Interactive 

GOOD READS ON SSR

- Rendering on the Web
  - https://developers.google.com/web/updates/2019/02/rendering-on-the-web

https://developers.google.com/web/updates/2019/02/rendering-on-the-web

	Server Rendering	"Static SSR"	SSR with (Re)hydration	CSR with Prerendering	Full CSR
Overview:	An application where input is navigation requests and the output is HTML in response to them.	Built as a Single Page App, but all pages prerendered to static HTML as a build step, and the JS is **removed**.	Built as a Single Page App. The server prerenders pages, but the full app is also booted on the client.	A Single Page App, where the initial shell/skeleton is prerendered to static HTML at build time.	A Single Page App. All logic, rendering and booting is done on the client. HTML is essentially just script & style tags.
Authoring:	Entirely server-side (request-response, HTML)	Built as if client-side (components, DOM*, fetch)	Built as client-side	Client-side	Client-side
Rendering:	Dynamic HTML	Static HTML	Dynamic HTML **and** JS/DOM	Partial static HTML, then JS/DOM	Entirely JS/DOM
Server role:	Controls all aspects. (thin client)	Delivers static HTML	Renders pages (navigation requests)	Delivers static HTML	Delivers static HTML
Pros:	👍 TTI = FCP 👍 Fully streaming	👍 Fast TTFB 👍 TTI = FCP 👍 Fully streaming	👍 Flexible	👍 Flexible 👍 Fast TTFB	👍 Flexible 👍 Fast TTFB
Cons:	👎 Slow TTFB 👎 Inflexible	👎 Inflexible 👎 Leads to hydration	👎 Slow TTFB 👎 TTI >>> FCP 👎 Usually buffered	👎 TTI > FCP 👎 Limited streaming	👎 TTI >>> FCP 👎 No streaming
Scales via:	Infra size / cost	build/deploy size	Infra size & JS size	JS size	JS size
Examples:	Gmail HTML, Hacker News	Docusaurus, Netflix*	Next.js, Razzle, etc	Gatsby, Vuepress, etc	Most apps

02

SSR CHALLENGES

THINGS WE WILL COVER

1. Detect where the code is running 🧭
2. Authentication 🔑
3. Handling secure routes 🚫
4. Setting correct response status code 🆗
5. Redirects ↩
6. State transfer (rehydration) 🌊
7. Using environment variables 🌍
8. Responsive design on server-side 🤔
9. Turning off JS in the browser 😱

1. DETECT WHERE THE CODE IS RUNNING 🧭

- Important because available APIs are not the same depending on where the code is running, examples:
  - On server-side we do not have `window`
  - On client-side we do not have `Request`

2. AUTHENTICATION 🔑

- How do we make sure that both client and server have correct tokens?

- Where do we store credentials?

3. HANDLING SECURE ROUTES 🚫

- What do we do when client navigates to a secure route?

4. SETTING CORRECT RESPONSE STATUS CODE 🆗

- How do we keep the semantics of HTTP responses correct?

5. REDIRECTS ↩

- Redirects on server and client are very different...

6. STATE TRANSFER (REHYDRATION) 🌊

- SSR instance will have state
  - Can we transfer that state to the client?
- Avoiding duplicate requests

7. USING ENVIRONMENT VARIABLES 🌍

- Awesome power of building once and deploying multiple times

8. RESPONSIVE DESIGN ON SERVER-SIDE 🤔

- Is there any way for server to know if rendered UI should be different depending on users' device?

9. TURNING OFF JS IN THE BROWSER 😱

- Are we crazy enough to turn off JS in our browsers and still be able to use the app?