Integrated Web Stack with

Angular

with Minko Gechev

github.com/mgechev
twitter.com/mgechev
blog.mgechev.com

Minko Gechev

mgechev

1.6k	94	3
Followers	Starred	Following

Popular repositories

angularjs-style-guide — 4,186 ★
Community-driven set of best practices for Ang...

javascript-algorithms — 1,744 ★
JavaScript implementation of different compute...

angularjs-in-patterns — 1,551 ★
AngularJS in patterns - this repository provides...

angular2-seed — 1,370 ★
Modular seed project for Angular 2 apps with s...

angular2-style-guide — 1,061 ★
Community-driven set of best practices and sty...

codelyzer — 81 ★
Static code analyzer for Angular 2 applications

aspect.js — 48 ★
Library for aspect-oriented programming with J...

Community Experience Distilled

Switching to Angular 2

Build SEO-friendly, high-performance single-page applications with Angular 2

Foreword by Miško Hevery, creator of AngularJS and Angular 2

Minko Gechev

[PACKT] open source*

agenda

What is

Angular 2

One framework.
Mobile and desktop.

GET STARTED

One framework.
Mobile and desktop.

GET STARTED

Mobile

- Web apps
- Ionic 2
- NativeScript
- React Native

One framework.
Mobile and desktop.

GET STARTED

Desktop

- Web apps
- Electron

Angular 2 is

platform agnostic


core
(zones, di, abstractions, etc.)

core

(zones, di, abstractions, etc.)

renderer

core

(zones, di, abstractions, etc.)

compiler

renderer


core
(zones, di, abstractions, etc.)
http
compiler
renderer


forms
core
(zones, di, abstractions, etc.)
http
compiler
renderer

router

forms

compiler

core

(zones, di, abstractions, etc.)

http

renderer

router

forms

core

(zones, di, abstractions, etc.)

http

i18n

compiler

renderer

router

forms

core

(zones, di, abstractions, etc.)

http

i18n

ngUpgrade

compiler

renderer


router
forms
core
(zones, di, abstractions, etc.)
http
i18n
ngUpgrade
compiler
renderer
animations

router

forms

core

(zones, di, abstractions, etc.)

http

i18n

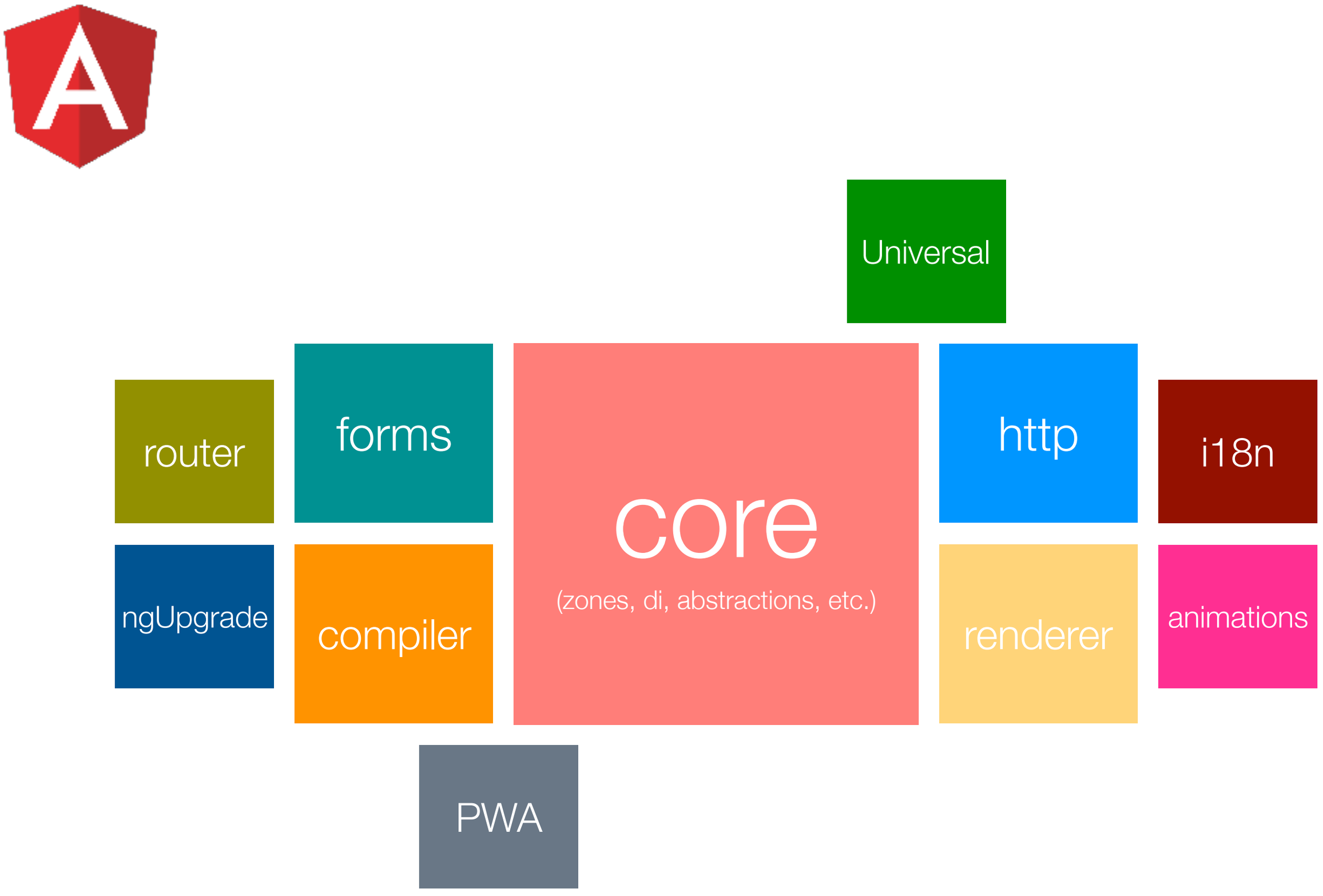
ngUpgrade

compiler

renderer

animations

PWA


Universal
router
forms
core
(zones, di, abstractions, etc.)
http
i18n
ngUpgrade
compiler
renderer
animations
PWA

Universal

router

forms

core

(zones, di, abstractions, etc.)

http

i18n

ngUpgrade

compiler

renderer

animations

PWA

material 2

Universal

router

forms

core

(zones, di, abstractions, etc.)

http

i18n

ngUpgrade

compiler

renderer

animations

PWA

CLI

material 2

language services

Universal

router

forms

core

(zones, di, abstractions, etc.)

http

i18n

ngUpgrade

compiler

renderer

animations

PWA

CLI

material 2

angularfire2

language services

Universal

router

forms

core

(zones, di, abstractions, etc.)

http

i18n

ngUpgrade

compiler

renderer

animations

PWA

CLI

material 2

angularfire2

language services

Universal

router

forms

core

(zones, di, abstractions, etc.)

http

i18n

ngUpgrade

compiler

renderer

animations

PWA

CLI

material 2

Angular 2's **Core**

Core

- Zone.js
- Ultrafast Change Detection
- Dependency Injection
- View Encapsulation

"A Zone is an execution context that persists across async tasks. You can think of it as thread-local storage for JavaScript VMs."

"A Zone is an execution context that persists across async tasks. You can think of it as thread-local storage for JavaScript VMs."

no-zones.js

```
@Component(...)
class SimpleComponent {
  getData() {
    fetch('https://example.com/data.json')
      .then(response => response.json())
      .then(data => {
        this.users = data;
        this.cd.detectChanges();
      });
  }
}
```

no-zones.js

```
@Component(...)
class SimpleComponent {
  getData() {
    fetch('https://example.com/data.json')
      .then(response => response.json())
      .then(data => {
        this.users = data;
        this.cd.detectChanges();
      });
  }
}
```

with-zones.js

```
@Component(...)
class SimpleComponent {
  getData() {
    fetch('https://example.com/data.json')
      .then(response => response.json())
      .then(data => this.users = data);
  }
}
```

in-zones.js

```
const old = Promise.prototype.then;
Promise.prototype.then = function () {
  old.apply(Promise.prototype, arguments);
  this.rootCD.detectChanges();
};
```

mhevery / **Zone.md**

Last active 2 days ago

Star 11 | Fork 0

Code | Revisions 3 | Stars 11

Embed <script src="https://gist | Download ZIP

TC39 Zone Proposal

example.md — Raw

Examples

Logging Zone

```
class LoggingZoneSpec {
  constructor() {
    this.name = 'logging';
    this.prefix = '';
  }

  onInvoke(parentZoneDelegate, currentZone, targetZone, callback, applyThis, applyArg) {
    console.log(this.prefix + 'Enter Zone:', targetZone.name);
    this.prefix += '  ';
    try {
```

mhevery / **Zone.md**

Last active 2 days ago

Star 11 | Fork 0

Code | Revisions 3 | Stars 11

Embed <script src="https://gist | Download ZIP

TC39 Zone Proposal

example.md | Raw

Examples

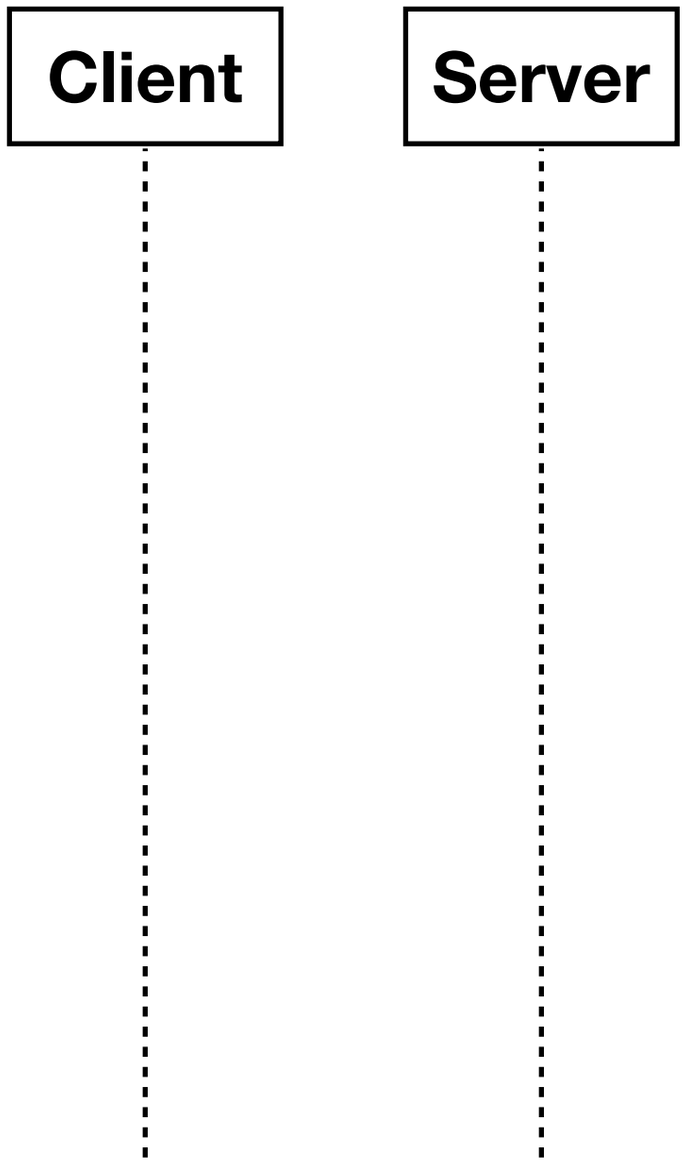
Logging Zone

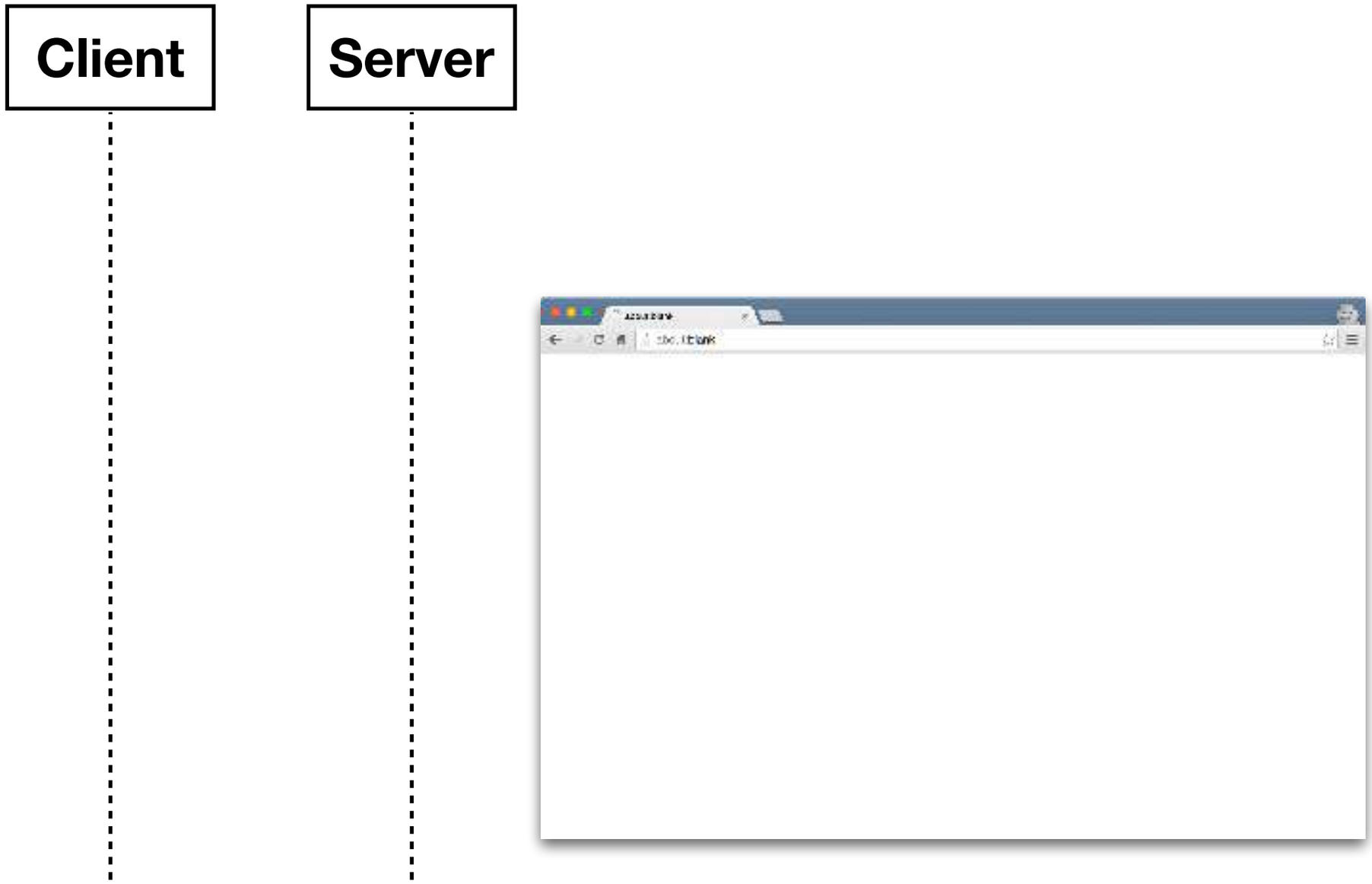
```
class LoggingZoneSpec {
  constructor() {
    this.name = 'logging';
    this.prefix = '';
  }

  onInvoke(parentZoneDelegate, currentZone, targetZone, callback, applyThis, applyArg) {
    console.log(this.prefix + 'Enter Zone:', targetZone.name);
    this.prefix += '  ';
    try {
```

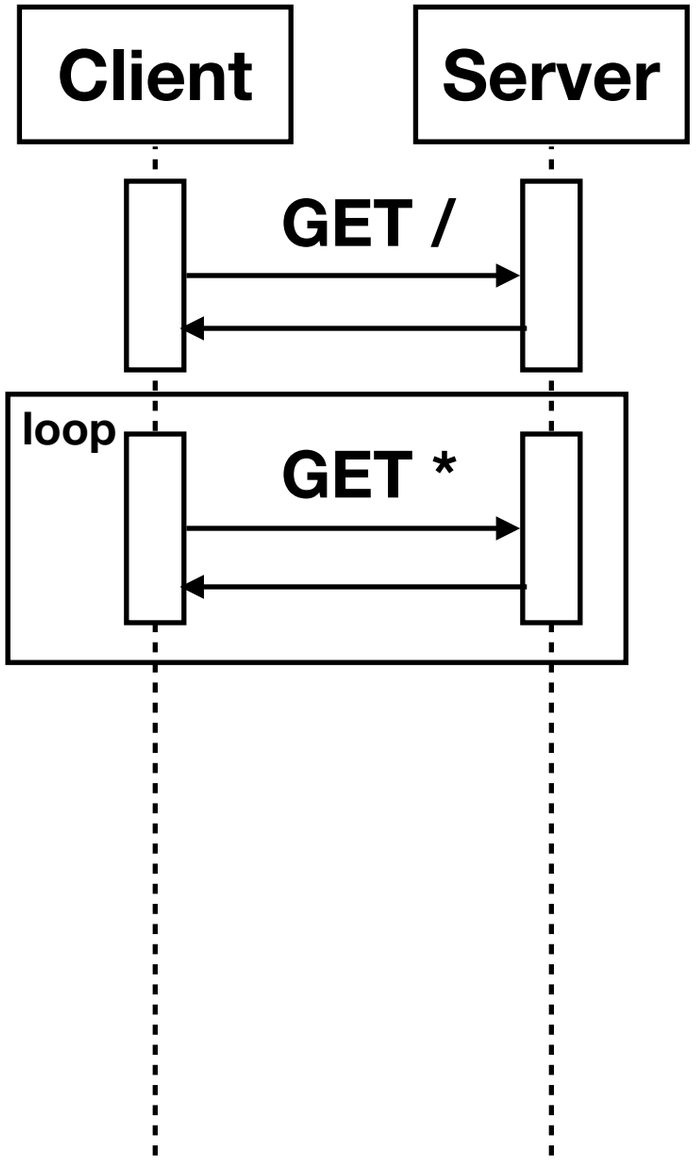
Renderer

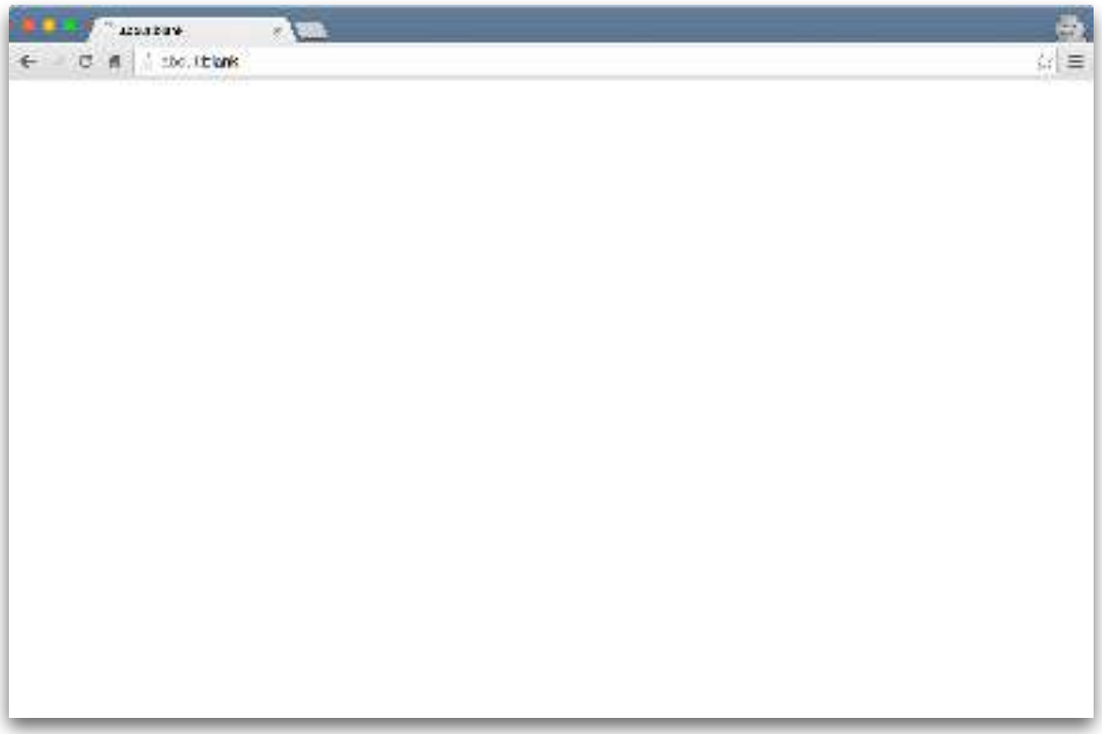
"A set of abstract operations which can be specialized for a specific platform"

"A set of abstract operations which can be specialized for a specific platform"

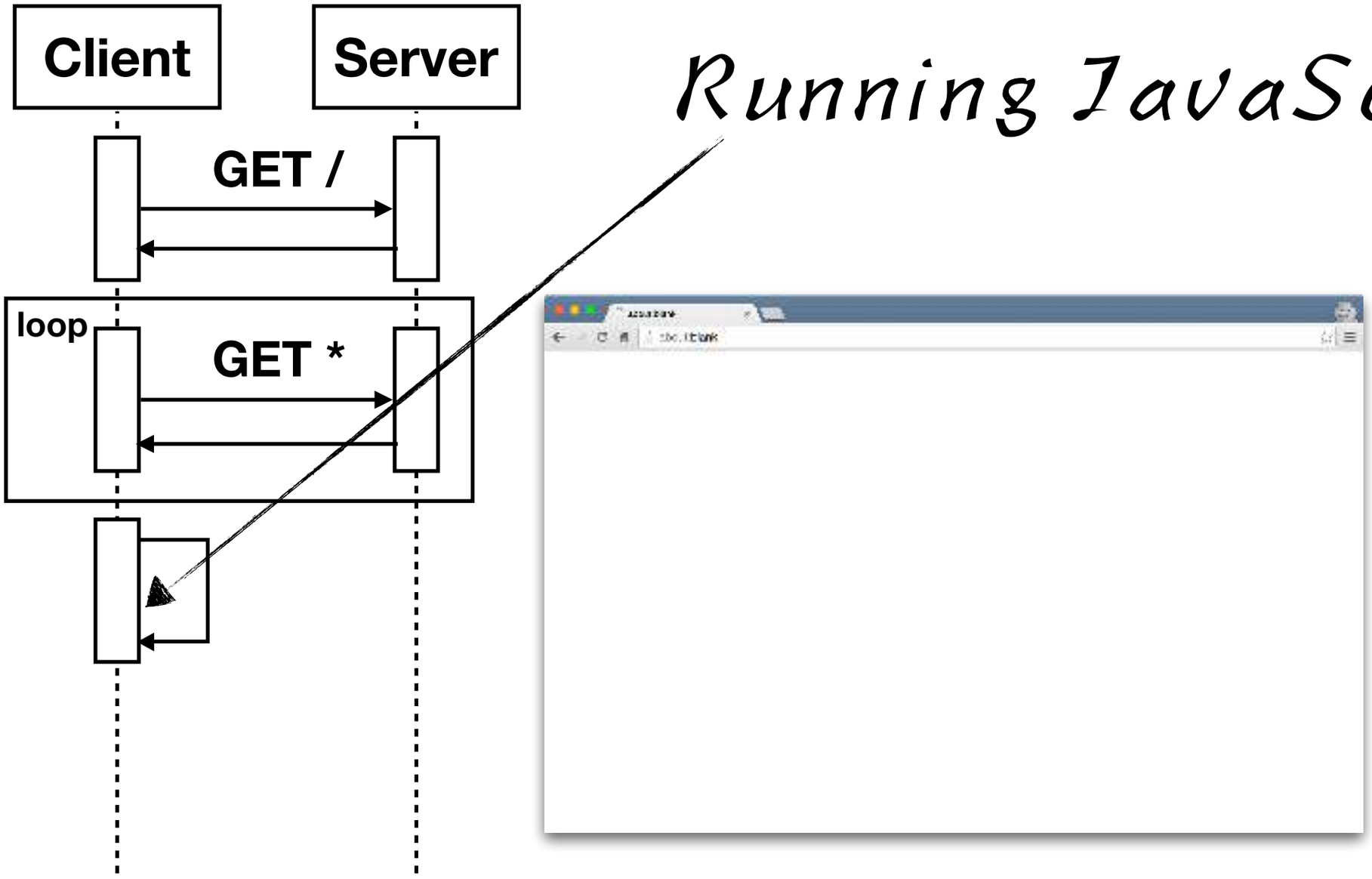

Client
Server

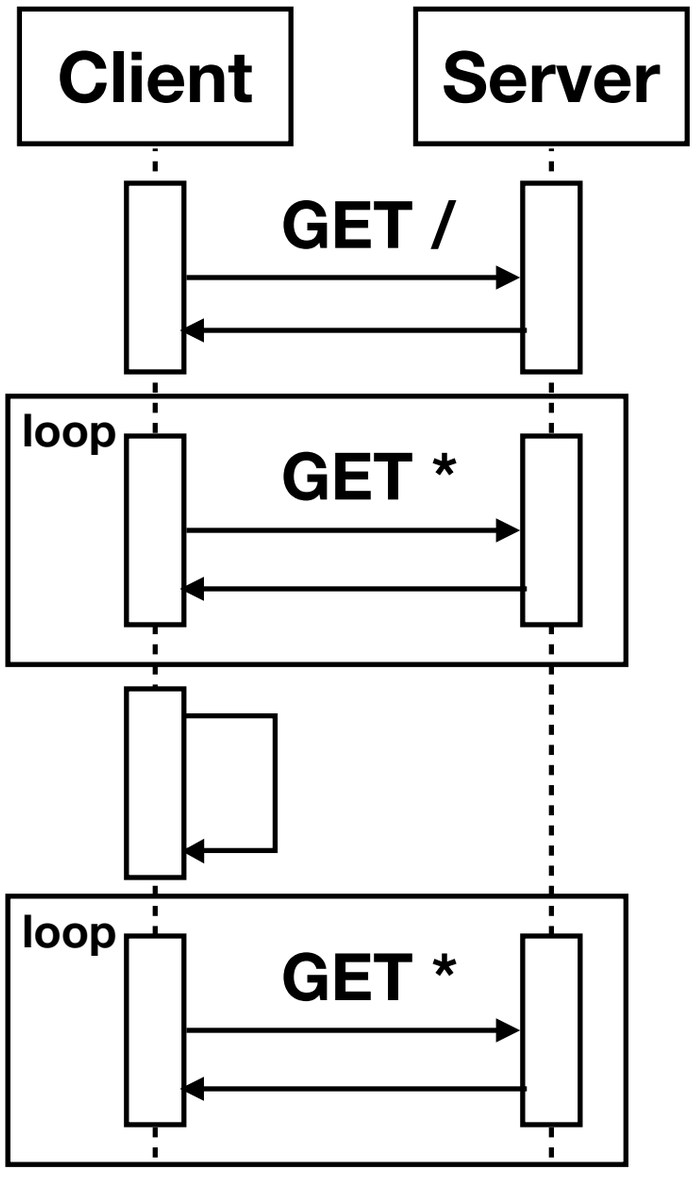

Client
Server

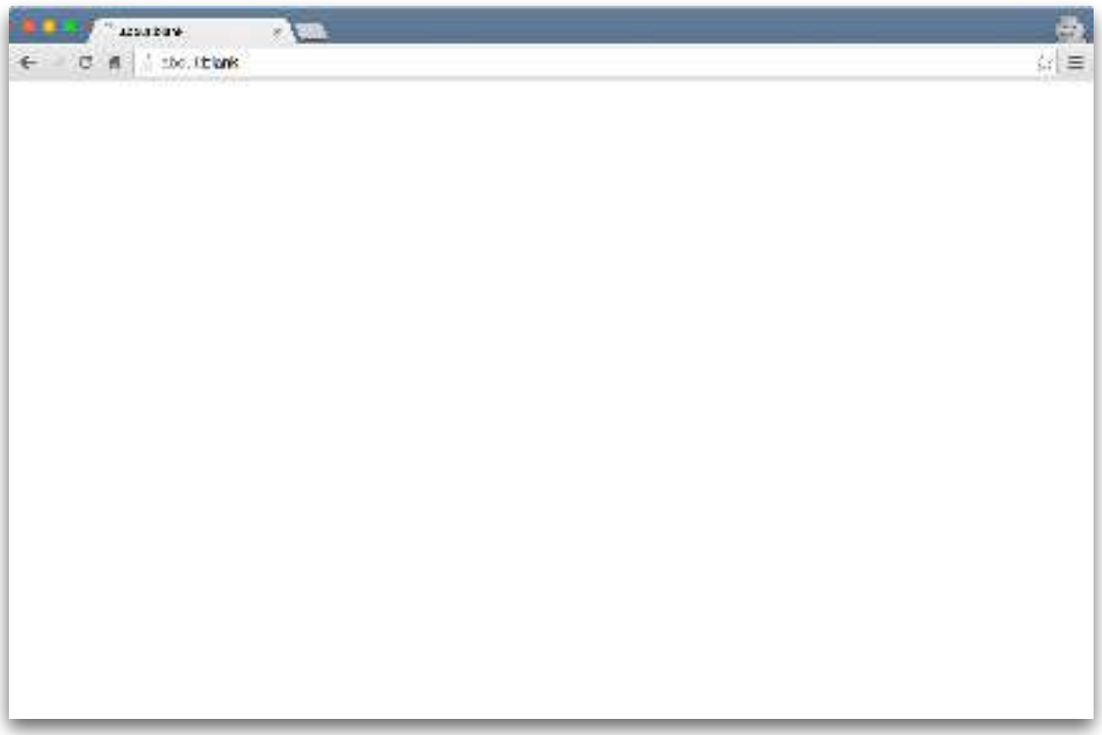

Client
Server
GET /
loop
GET *

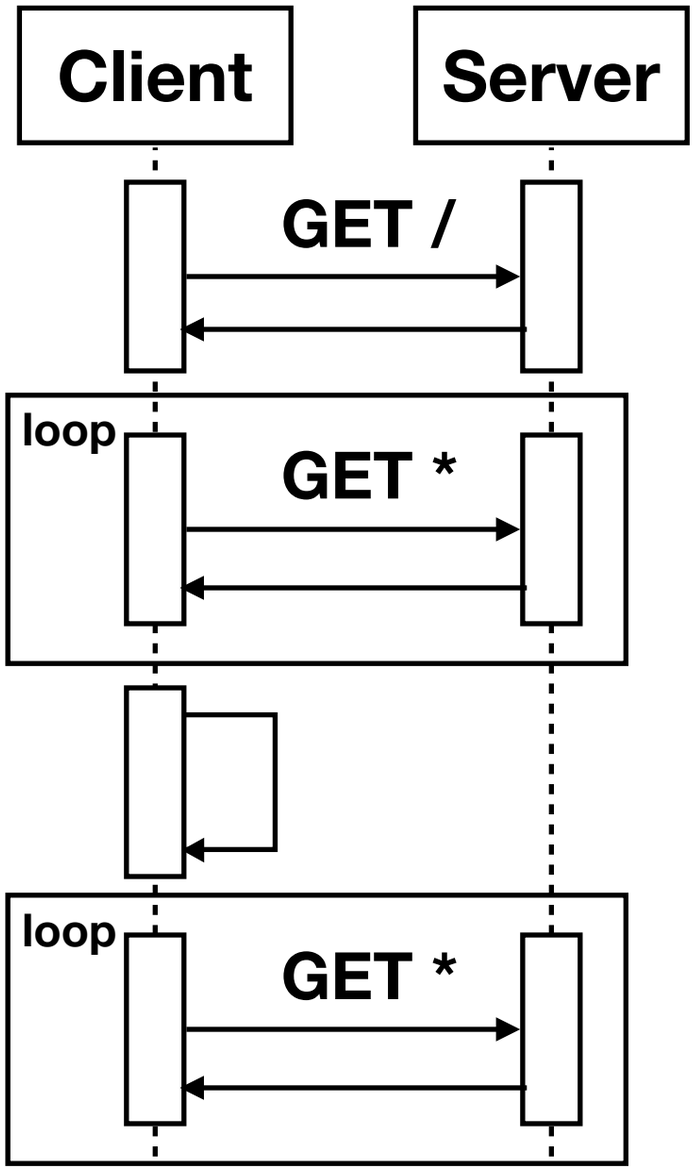


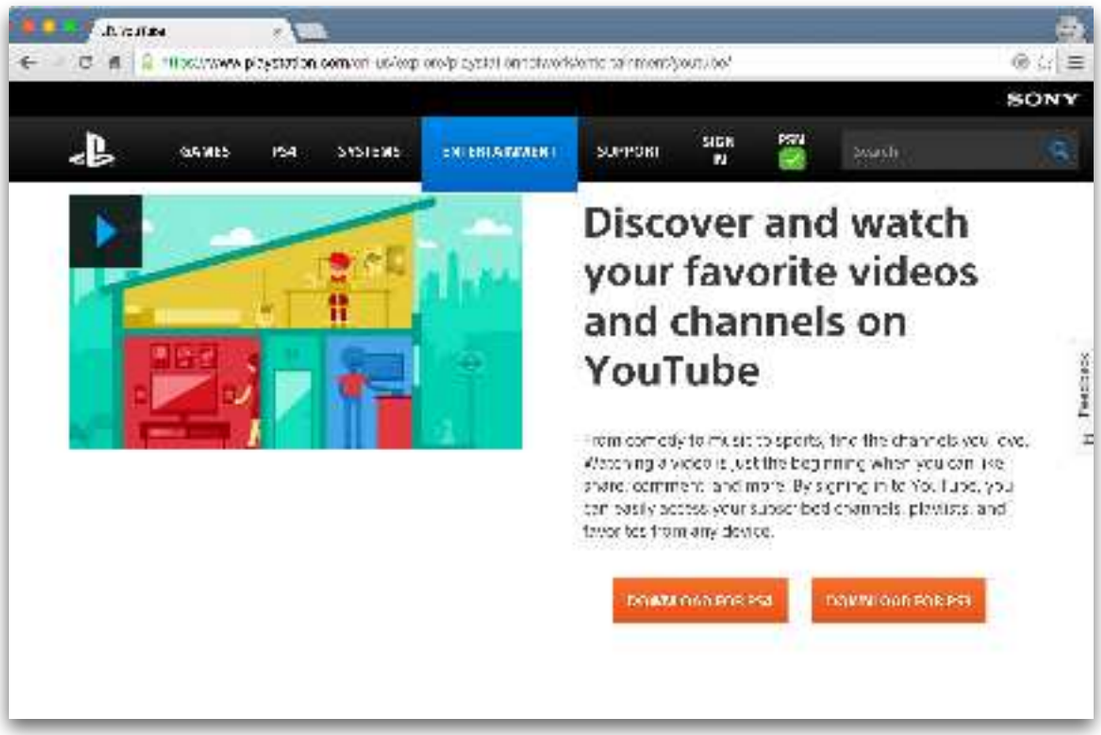
Running JavaScript



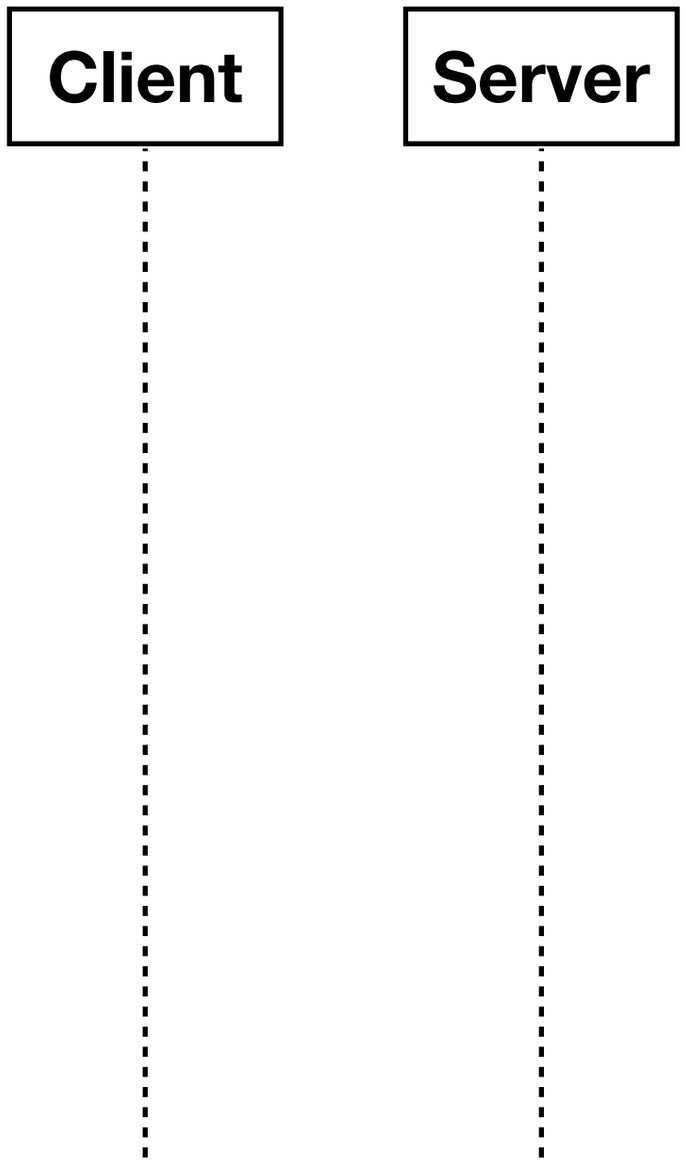

Client
Server
GET /
loop
GET *
loop
GET *

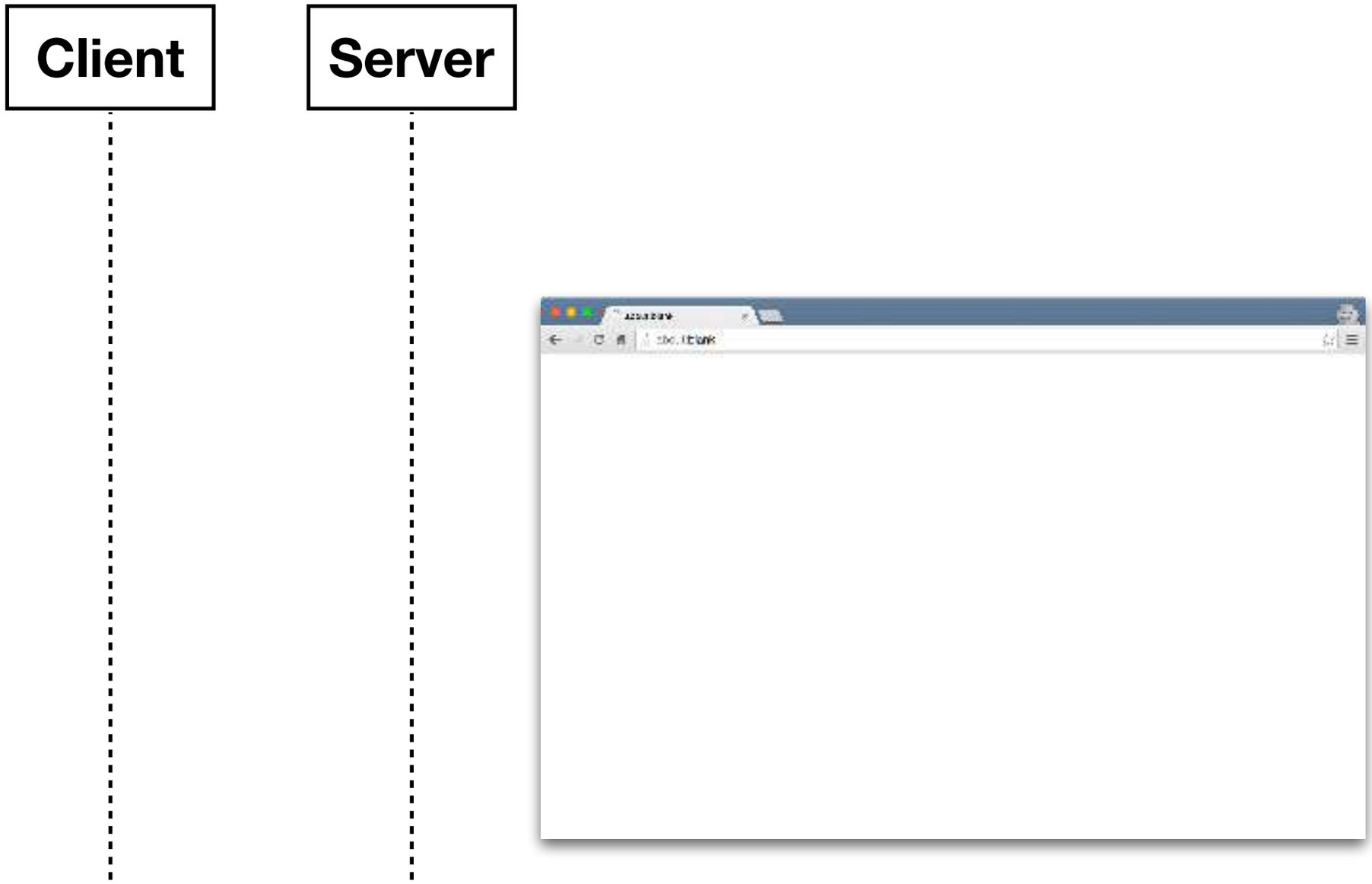


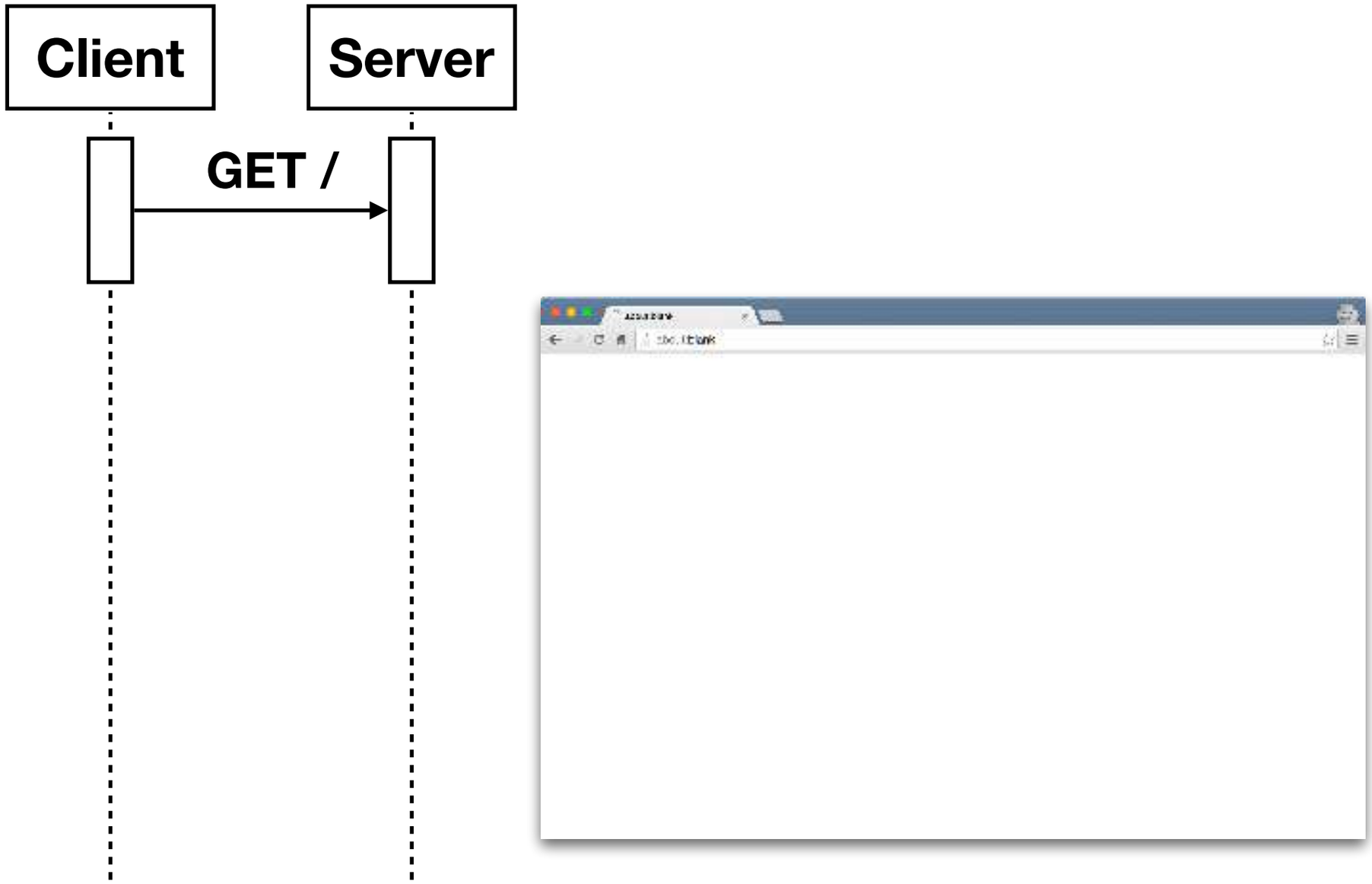

Client
Server
GET /
loop
GET *
loop
GET *

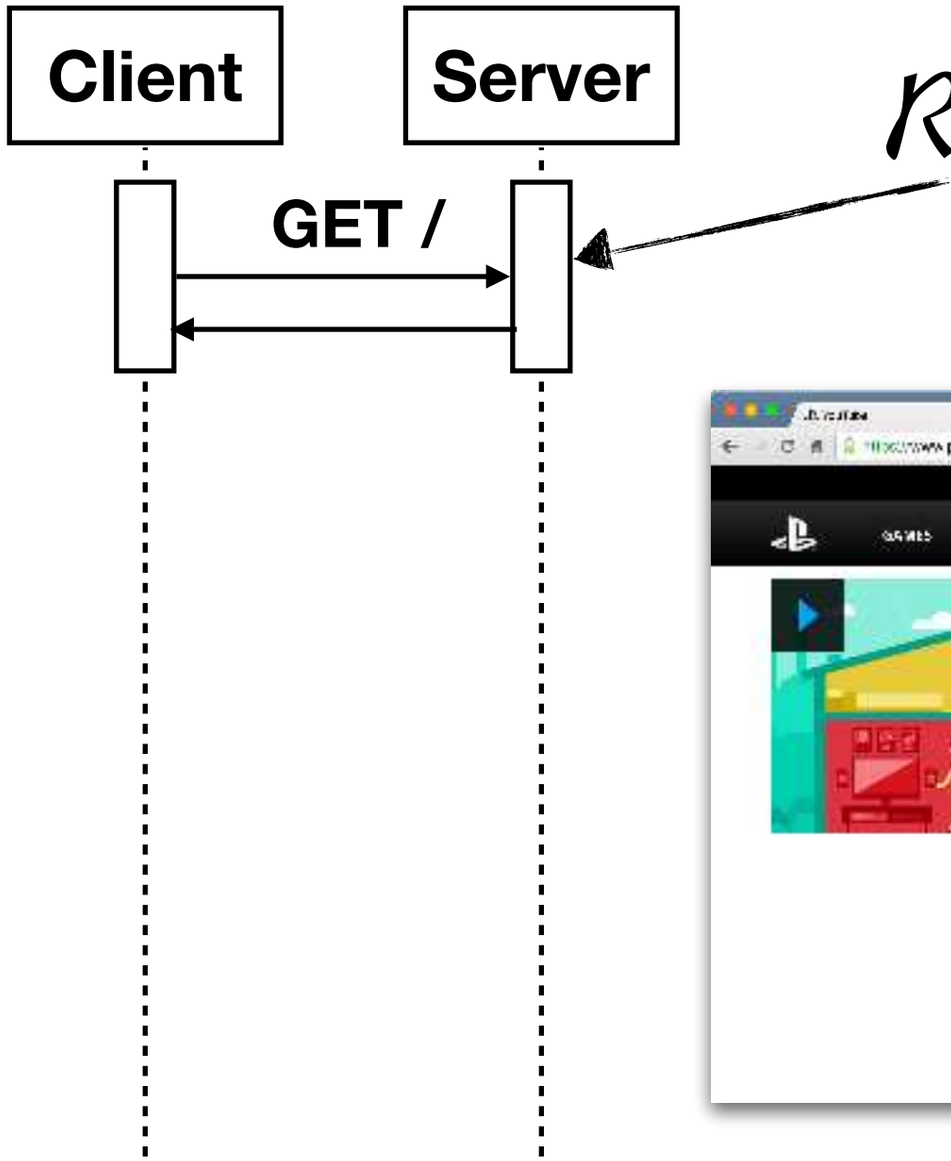

SONY
Discover and watch your favorite videos and channels on YouTube

server-side rendering

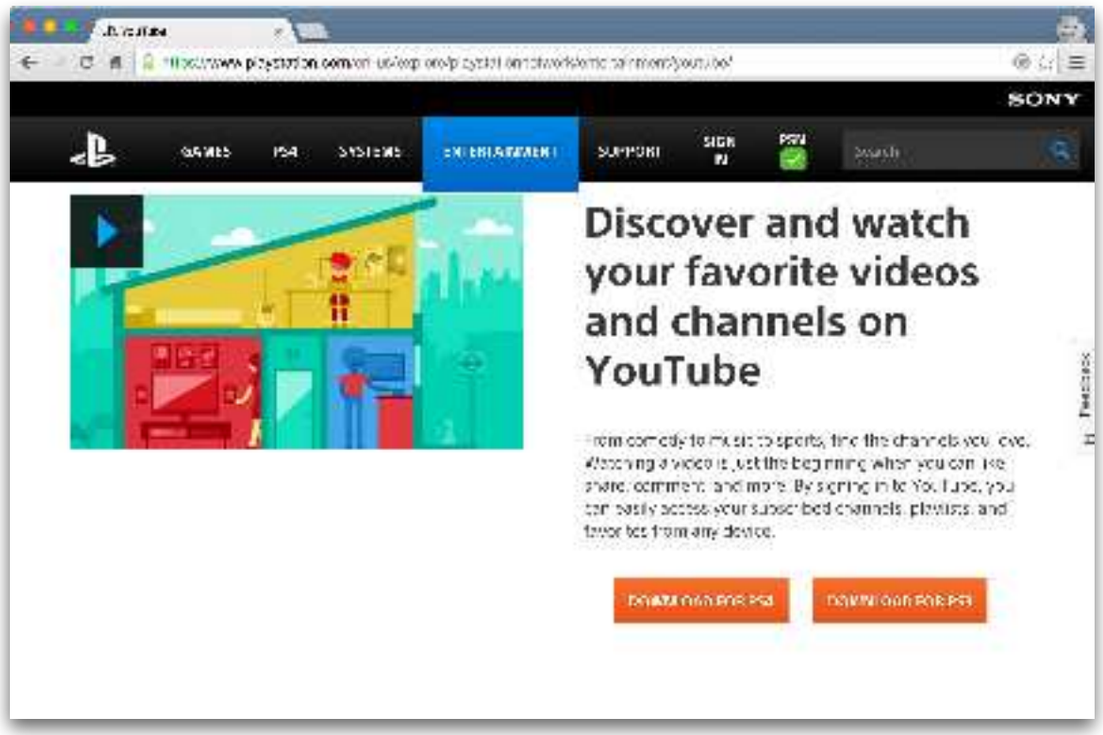

Client
Server

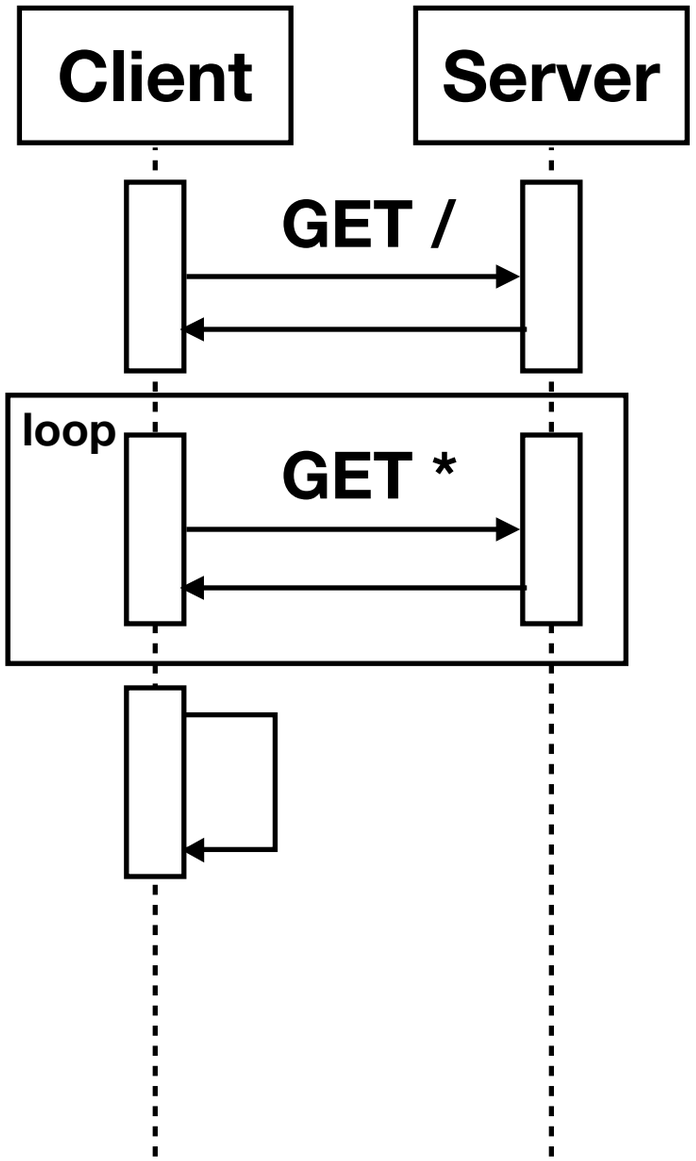

Client
Server

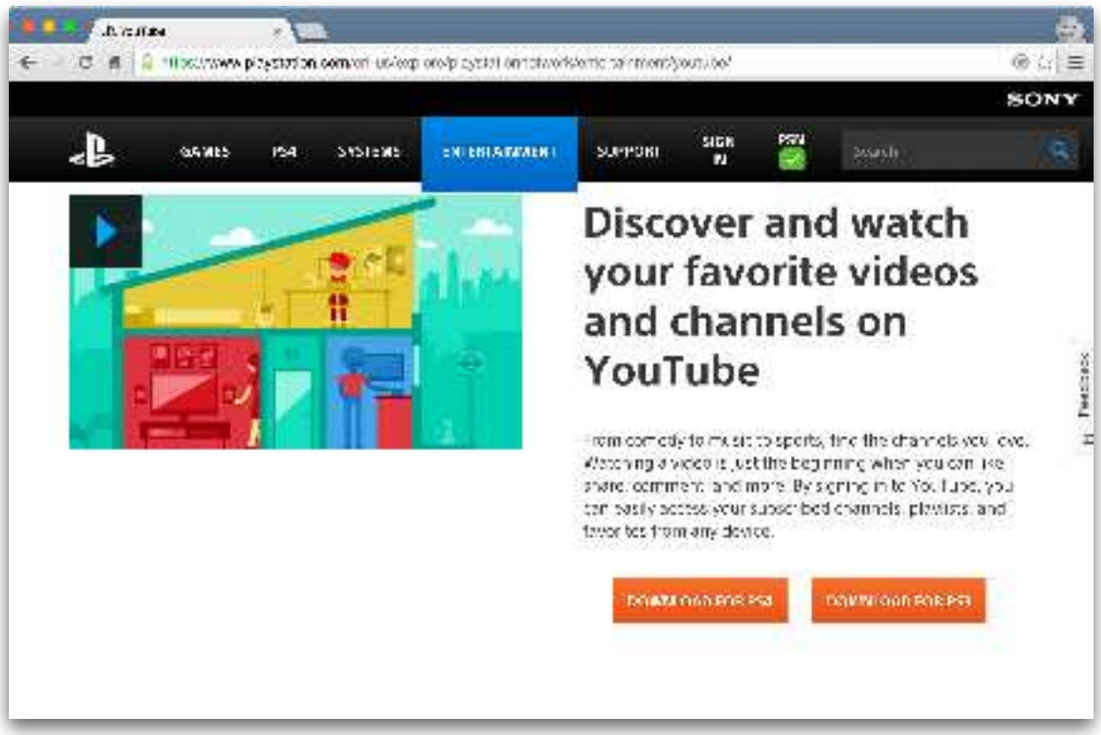

Client
Server
GET /



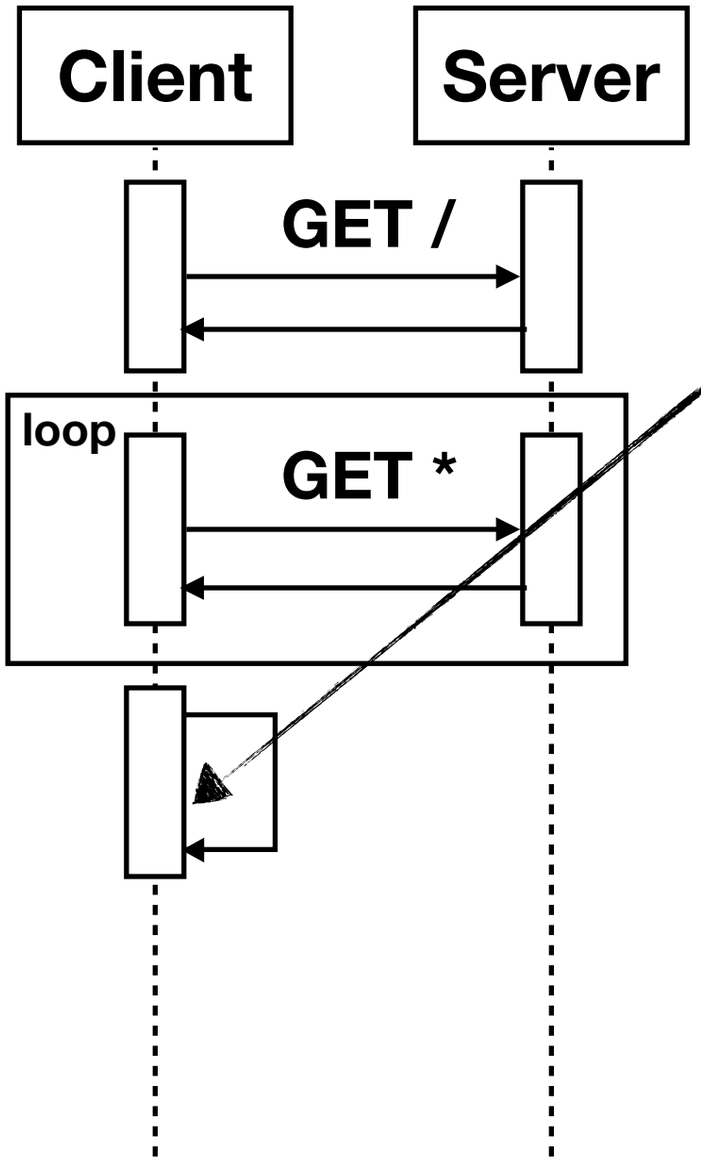
Running JavaScript

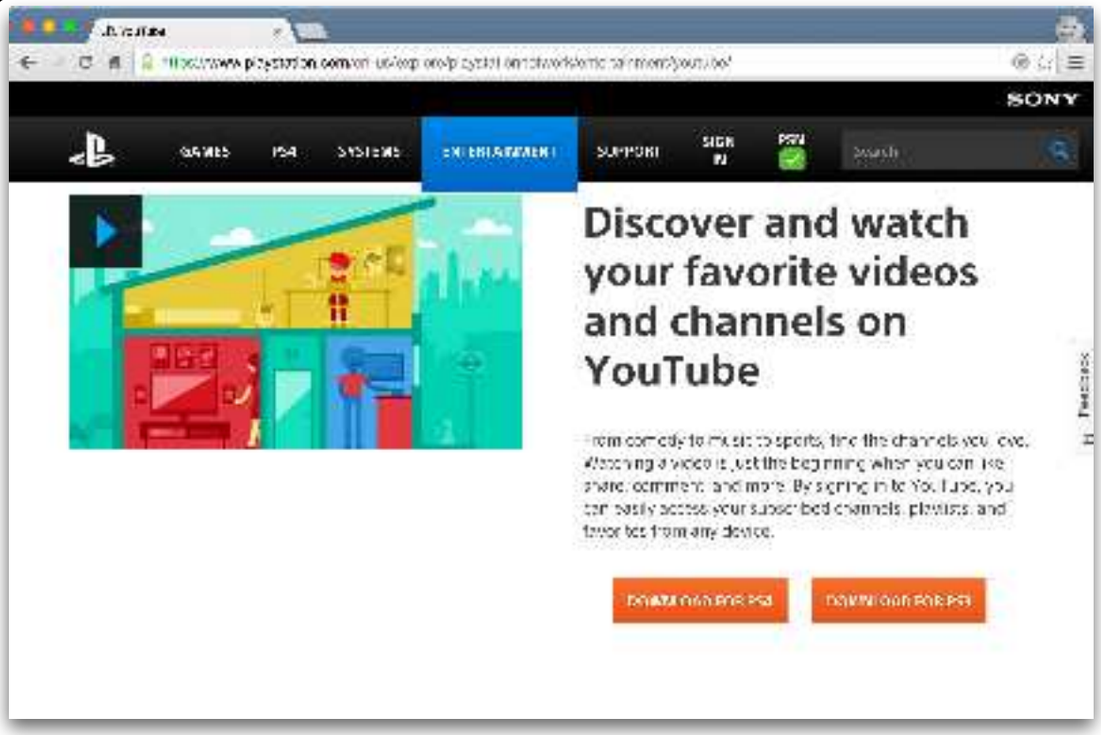


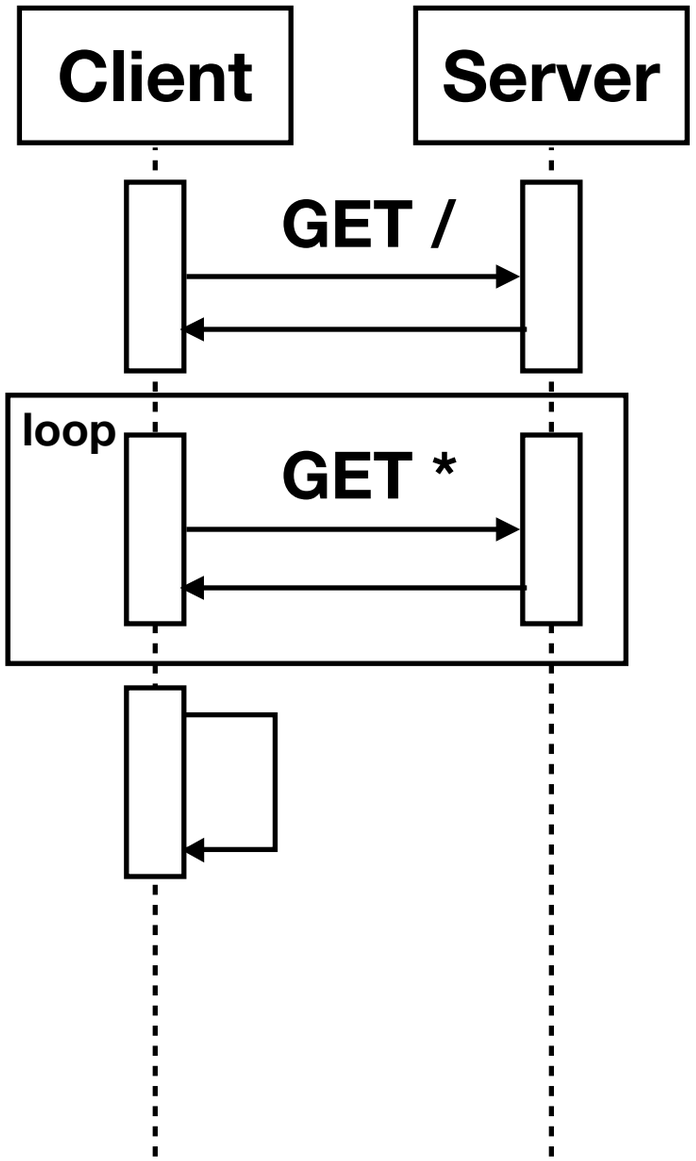

Client
Server
GET /
loop
GET *

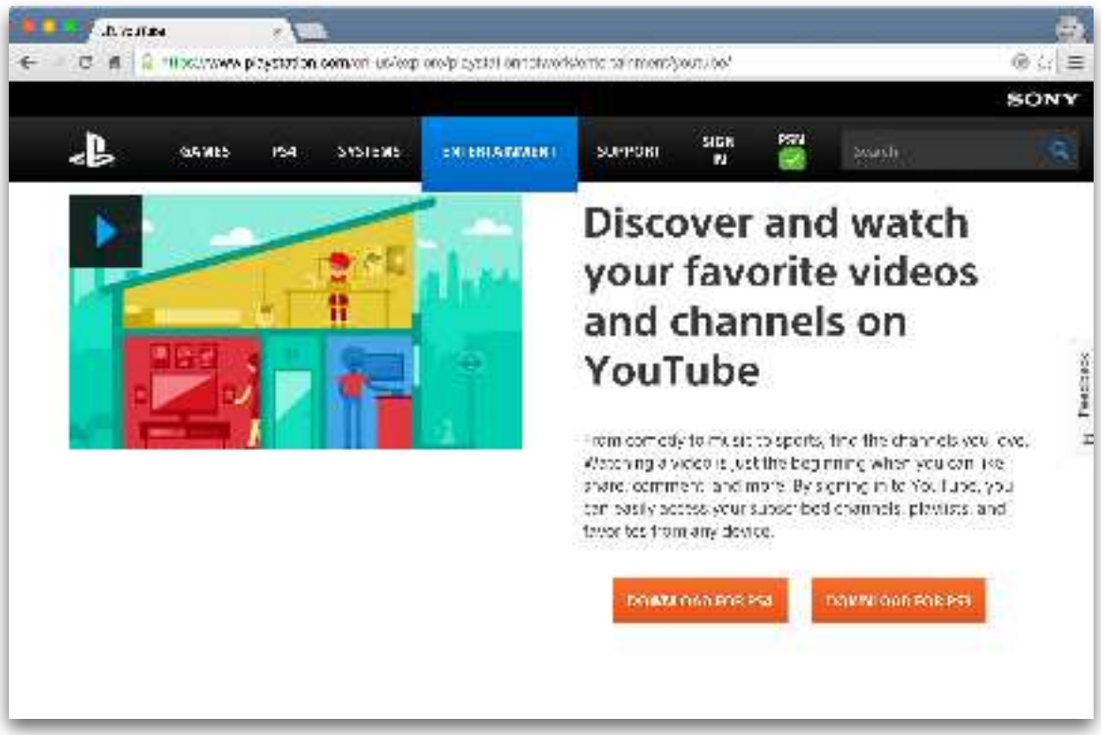

SONY
Discover and watch your favorite videos and channels on YouTube

Running JavaScript






Client
Server
GET /
loop
GET *


SONY
Discover and watch your favorite videos and channels on YouTube

NativeScript

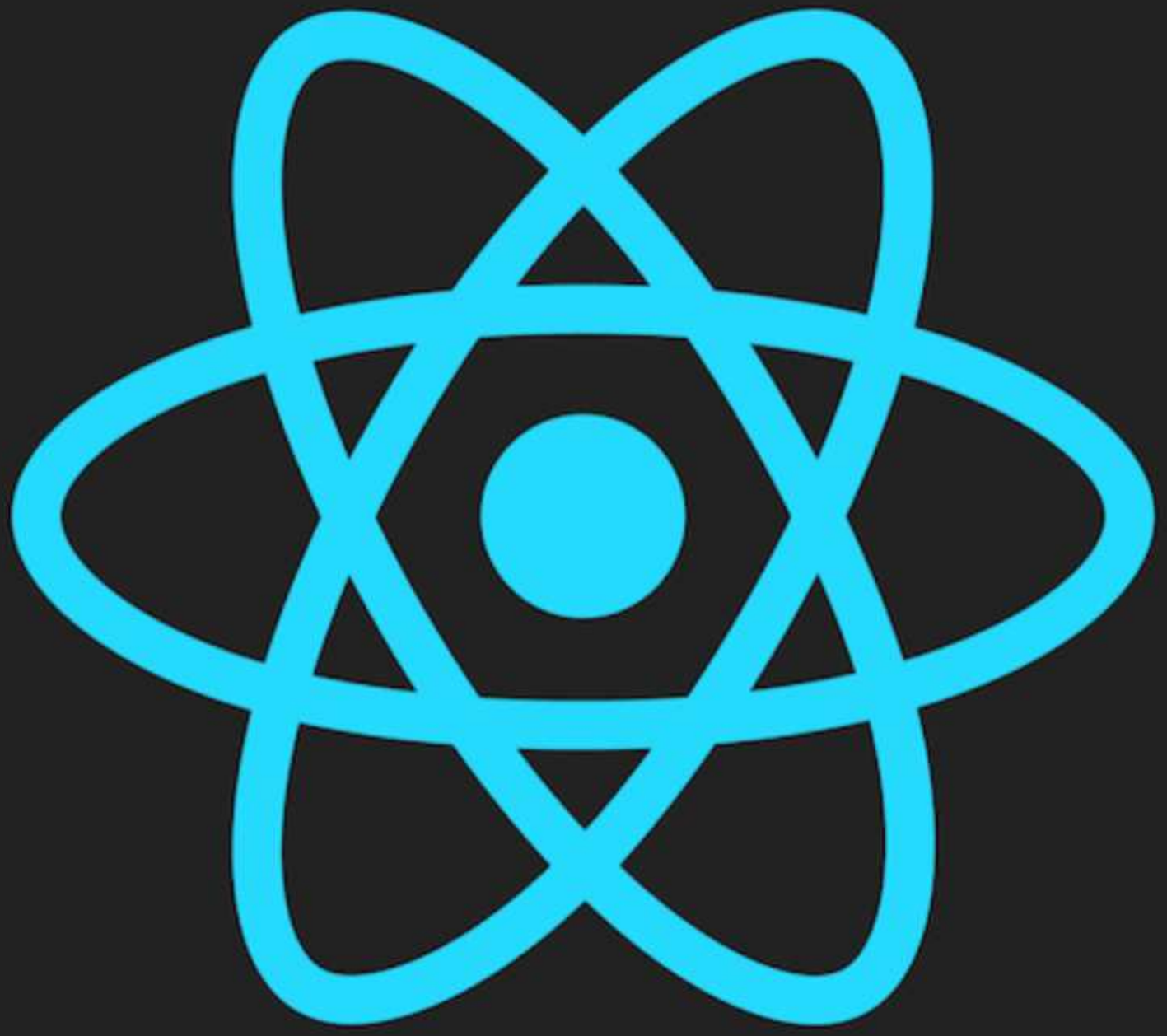
"NativeScript 2.0

truly native mobile apps"

"NativeScript 2.0

truly native mobile apps"

why not...



React Native

Compiler

"Compiles your templates to JavaScript calls to the renderer"

template.html

```
<p>
  Howdy! Here's a list
  <form>
    ...
  </form>
</p>
```

compiled.js

```
...
var parentRenderNode = renderer
    .createViewRoot(declarationAppElement.nativeElement);
_el_0 = renderer
    .createElement(parentRenderNode,'p',debug(0,0,0));
_text_1 = renderer
    .createText(_el_0,'\n  Howdy! Here\'s a list',debug(1,0,3));
_text_2 = renderer
    .createText(parentRenderNode,'\n\n',debug(2,3,4));
_el_3 = renderer
    .createElement(parentRenderNode,'form',debug(3,5,0));
...
```

Compiler works run-time and build time

Single instance of the renderer

Optimizations

- Payload reduction (tree-shaking)
- Pre-compilation
- Ultra-fast change detection
- WebWorkers

Rob Eisenberg
@EisenbergEffect

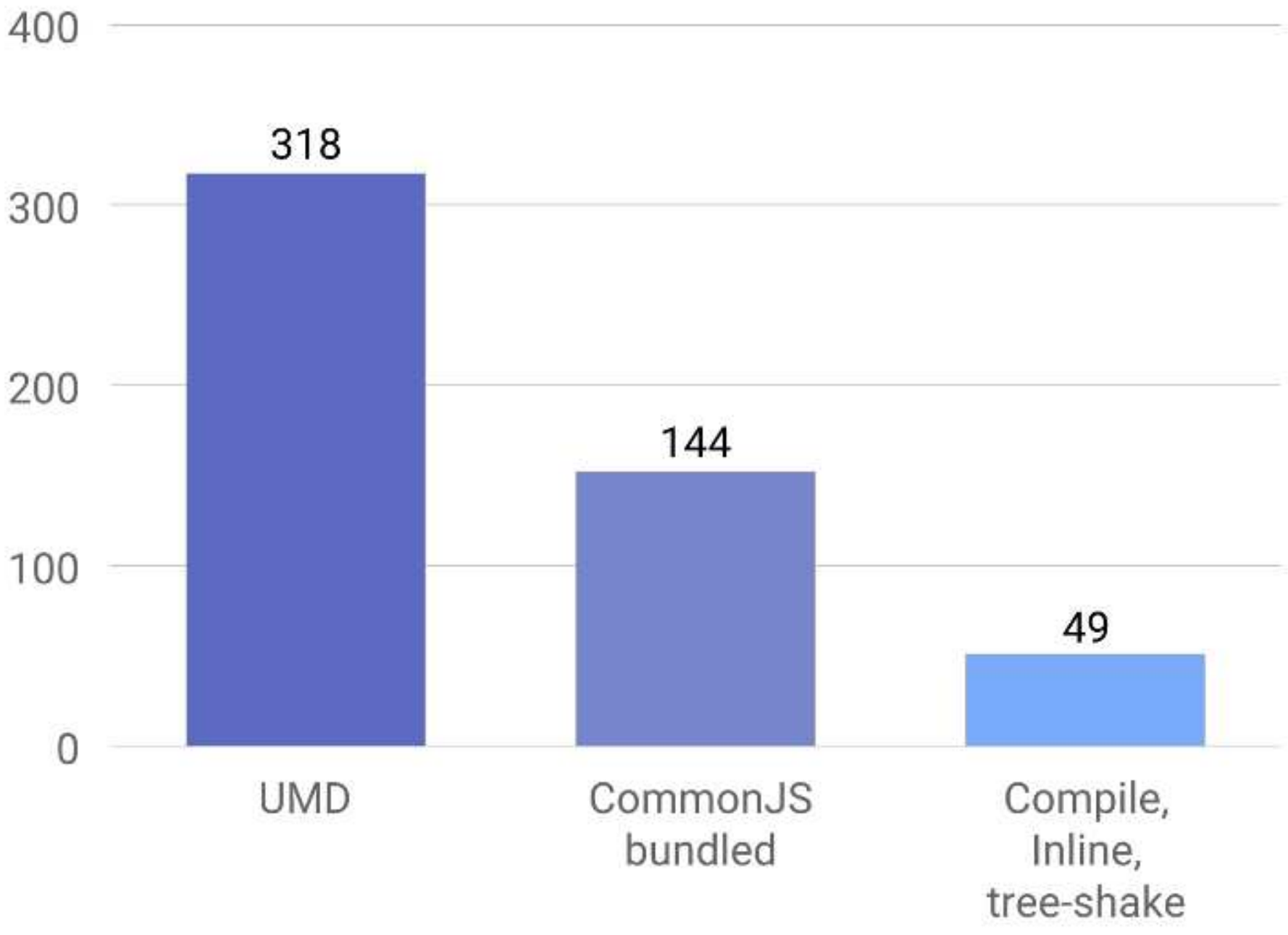
Following

If your SPA framework is larger than 500k after minification, you might want to reconsider. That's probably not the best choice.

RETWEETS 23 LIKES 19

10:11 PM - 9 Mar 2016

Size (in KB, gzipped)




爆笑gif图

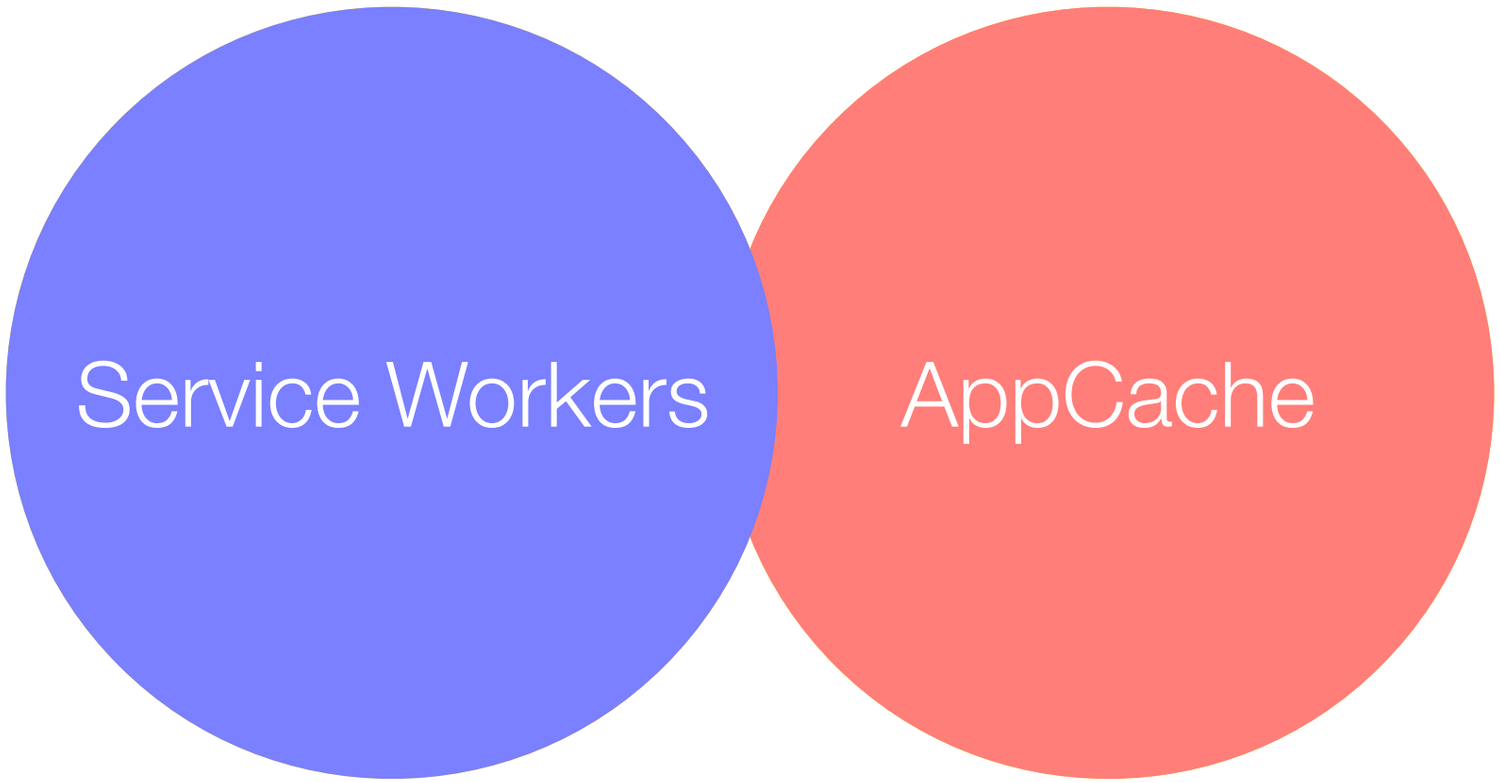
angular/**mobile-toolkit**

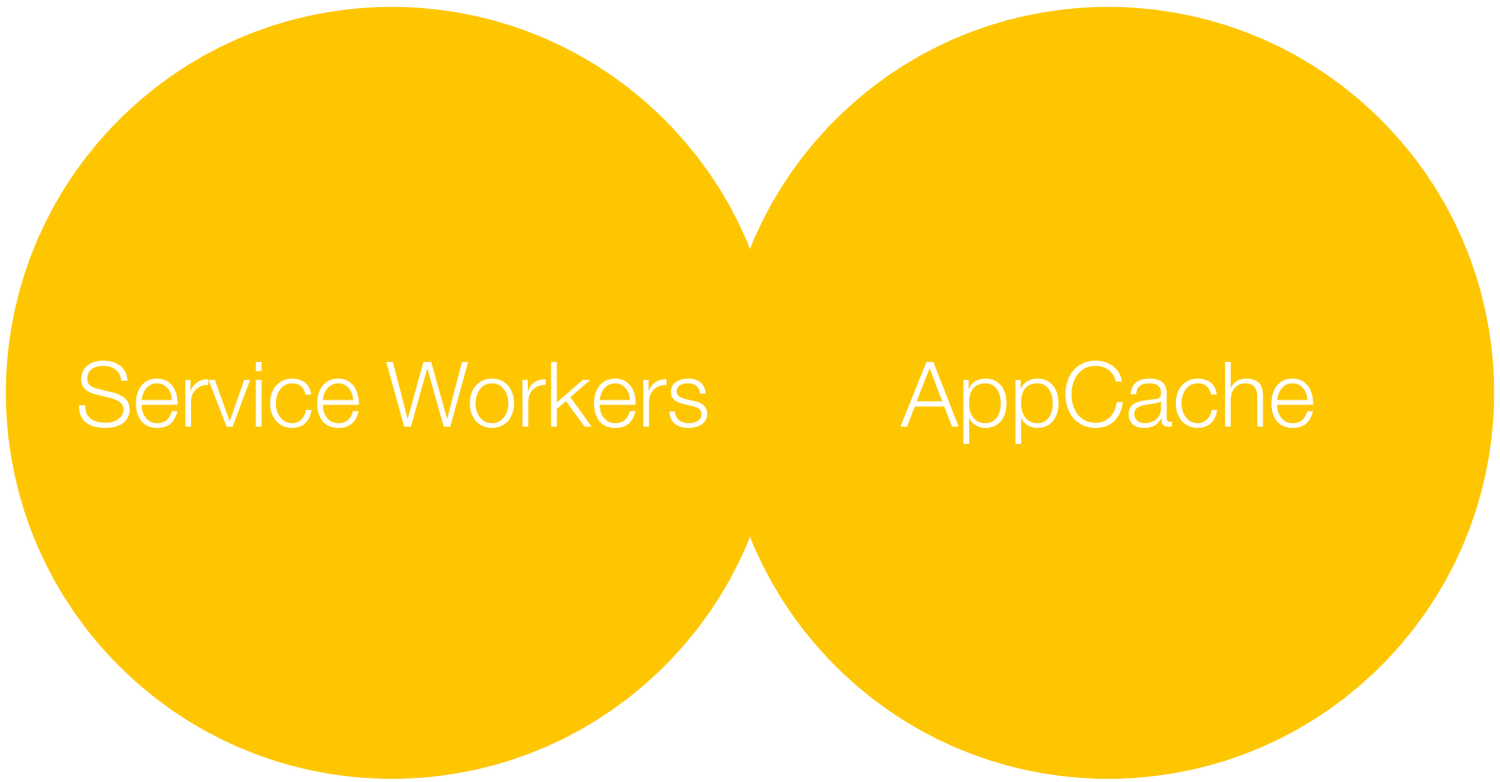
Progressive enhancement for web applications

mobile-toolkit

- Offline access to static content
- Notifications for new versions
- Efficient download of deltas
- Support of push notifications
- AppShell

Service Workers and AppCache


Service Workers
AppCache



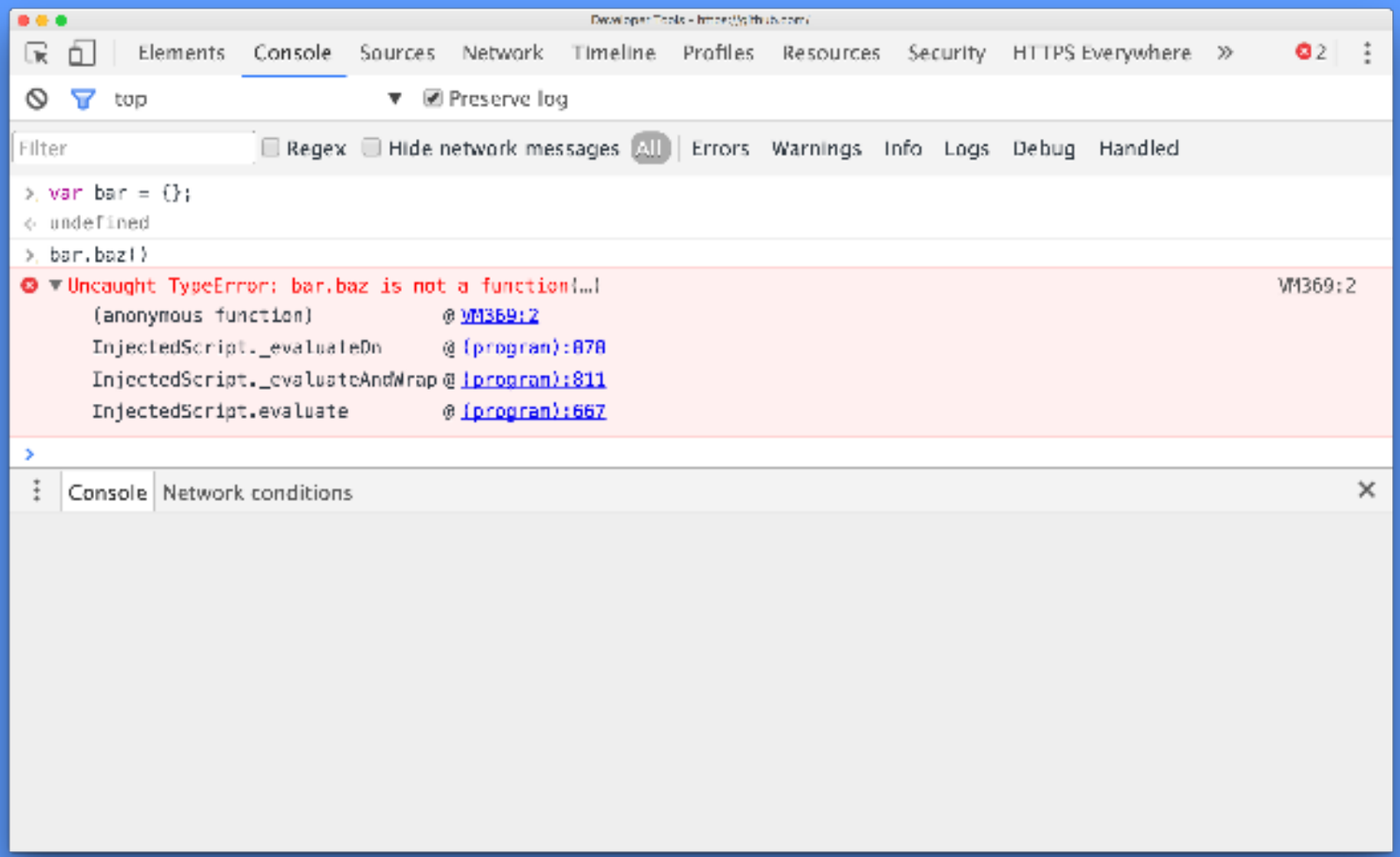
=

~91.8%

Development

experience

Typical JavaScript experience



Angular is...

- Built with TypeScript
  - Compile-time type checking
  - Great IDE/text editors support
- Analyzable templates

Type-checking and **IntelliSense** in templates

• app.ts

```
1  import {Component} from 'angular2/core'
2
3  @Component({
4      template: `<h1 [style]='font'></h1>
5      <div>{{person.address.street}}</div>
6      <span (click)='updatePerson("test")'></span>`
7  })
8  class Foo {
9      person = {
10         /** Full name */
11         name: "John Doe",
12         address: {
13             /** Street number and address */
14             street: "1 Happy St",
15             zip: 98118
16         }
17     };
18
19     font: string;
20     updatePerson(name: string){ /* TODO */}
21 }
22
```

Tracking: Language services provide intellisense for Angular 2 templates #7482

New Issue

Open **mhevery** opened this issue 15 days ago · 4 comments

mhevery commented 15 days ago — Angular member

We plan to use the extension API tracked by TypeScript issue #6508 to provide language services inside template strings of a `@Component` decorator as well as reuse these services in loose HTML files.

Features

The features we plan to support are,

- Syntax and semantic code highlighting
- Syntactic and semantic error reporting
- Completion lists for HTML built-in elements and referenced directives
- Completion list for data-binding expression
- Parameter lists for data-binding function calls
- Go-to definition on attributes, elements and binding expressions

A prototype of these features was already implemented by Microsoft as part of the work to validate the their extensibility and the referenced issue has an animated GIF that shows the prototype in action.

This work is to use the Angular 2's template parser, instead of a custom parser, as well as using the Angular 2 offline-compiler metadata to describe the components allowing libraries (such as Material) to be self-describing.

Labels

hotlist: A2 Blocking

Milestone

Angular 2 Release...

Assignee

chuckjaz

Notifications

Unsubscribe

You're receiving notifications because you're subscribed to this thread.

4 participants

Already supported by some IDEs

Still immature...

JavaScript Daily
@JavaScriptDaily

Follow

Community-Driven Best Practices and Style Guidelines for Angular 2 Application Development:

mgechev/angular2-style-guide

angular2-style-guide - Community-driven set of best practices and style guidelines for Angular 2 application development

github.com

RETWEETS 57

LIKES 107

7:10 PM - 12 Apr 2016

JavaScript Daily
@JavaScriptDaily

Follow

The Official Angular 2 Style Guide:
angular.io/styleguide

RETWEETS 90 LIKES 120

12:11 PM - 5 May 2016

Using style guide in **your project**

Enforcing common style

Enforcing common style

- Fork the official style guide
- *Modify the styles according to your needs*

Enforcing common style

- Fork the official style guide
  - *Modify the styles according to your needs*
- Introduce the style guide to your team

Enforcing common style

- Fork the official style guide
  - *Modify the styles according to your needs*
- Introduce the style guide to your team
- Verify that each individual code change follows it

Code review

process

HOW TO MAKE A GOOD CODE REVIEW

RULE 1: TRY TO FIND AT LEAST SOMETHING POSITIVE

manual

boring

error-prone

codelyzer

"codelyzer is a project which aims to enforce common style and verify correctness of your program"

Thanks for flying Vim (bash)

```
~/Projects/angular2-seed master v
```

Where to start?

angular-cli

```
ng new hello-world
```

angular-cli:

- Quick bootstrap of projects
- Generating:
  - Components
  - Services
  - Directives
  - Pipes
- Support for offline apps
- many others…

angular2-seed

angular2-seed

- TypeScript transpilation
- Statically typed dev and prod builds
- Out of the box unit-testing
- Test coverage
- e2e testing with protractor
- Static code analyzer

Resources:

- angular.io
- github.com/angular/angular
- github.com/angular/mobile-toolkit
- github.com/angular/angular-cli
- github.com/mgechev/angular2-seed

Thank you!

github.com/mgechev
twitter.com/mgechev
blog.mgechev.com