
A
14.007
-3
-2
-1
+1
+2
+3
+4
+5
N
7
2 -5
gular
Uppesitt
39.948
Ar
18
2 -8 -8
kväll


ES6

HARMONY

ECMASCRIPT VERSION 6

ES2015

HISTORY OF ECMASCRIPT

- 1997: ES1
- 1998: ES2
- 1999: ES3
- 2007: ES4
- 2009: ES5
- 2015: ES6
- 2016: ES7
- 2017: ES8

CONST - NOT IMMUTABLE

```
const arr = [1, 2];
arr.push(3);
arr.someRandomProp = 'Jepp Jepp';

const a = 1;
a = 2; // TypeError: Assignment to constant variable
```

CONST - NOT IMMUTABLE

```
const arr = Object.freeze([1, 2]);
arr.push(3);                              // TypeError
arr.someRandomProp = 'Jepp Jepp';         // TypeError
```

ARROW FUNCTIONS

```
const add = function(a, b) { return a + b; }
```

ARROW FUNCTIONS

```
const add = (a, b) => a + b;
```

ARROW FUNCTIONS

```
const A = () => {};
```

- `A.prototype === undefined`
- `A.bind`, `A.call` **and** `A.apply` **are defined but can not change function scope**
- **Lexical scope:** `arguments` **and** `this` **are defined by host**
- `new A(); // TypeError: () => {} is not A constructor`

FOR OF

```
const arr = [3, 5, 7];
a.note = 'LA FAMILIA ES TODO';

for (let key in arr) {
   console.log(key); // 0, 1, 2, note
}

for (let value of arr) {
   console.log(value); // 3, 5, 7
}

arr.forEach(v => console.log(v)); // 3, 5, 7
```

FOR OF

```
const arr = undefined;

for (let key in arr) { // OK
   console.log(key);
}

for (let value of arr) { // Cannot read property 'Symbol(Symbol.iterator)' of undefined
   console.log(value);
}

arr.forEach(v => console.log(v)); // Cannot read property 'forEach' of undefined
```

PROMISES

```
const myImage = document.querySelector('img');

fetch('unicorns.jpg')
    .then(response => response.blob())
    .then(blob => URL.createObjectURL(blob))
    .then(blobURL => myImage.src = blobURL)
    .catch(error => console.error(error));
```

```
function findShowById(show, cb) {
    repo.getShowById(show.ids.id, function(error, _show) {
        if (error) {
            cb(error);
        } else if (_show) {
            cb(undefined, _show);
        } else {
            repo.getShowByTheTvDbId(show.ids.theTvDb, function(error, _show) {
                if (error) {
                    cb(error);
                } else if (_show) {
                    cb(undefined, _show);
                } else {
                    repo.getShowByImdbId(show.ids.imdb, function(error, _show) {
                        if (error) {
                            cb(error);
                        } else if (_show) {
                            cb(undefined, _show);
                        } else {
                            cb(new MissingShowError('Can not find show:' + error));
                        }
                    });
                }
            });
        }
    });
}
```

```
function findShowById(show) {
    return repo.getShowById(show.ids.id)
        .catch(() => repo.getShowByTheTvDbId(show.ids.theTvDb))
        .catch(() => repo.getShowByImdbId(show.ids.imdb))
        .catch(error => {
            throw new MissingShowError('Can not find show', error);
        })
        .then(result => console.log('Woop Woop! We have a result', result));
}
```

PROMISES - THE BAD PART

- No lazy execution
- No abort
- No retry
- Single value

REACTIVE PROGRAMMING (RXJS)

USE CASE - AUTOCOMPLETE SEARCH

AUTOCOMPLETE SEARCH - PROBLEMS

- Debounce
- Min search string length
- HTTP response order

```
const source = Rx.Observable.fromEvent($input, 'keyup');
```

```
const source = Rx.Observable.fromEvent($input, 'keyup')
    .map(event => event.target.value);
```

```
const source = Rx.Observable.fromEvent($input, 'keyup')
    .map(event => event.target.value)
    .filter(text => text.length > 2);
```

```
const source = Rx.Observable.fromEvent($input, 'keyup')
    .map(event => event.target.value)
    .filter(text => text.length > 3)
    .debounceTime(200);
```

```
const source = Rx.Observable.fromEvent($input, 'keyup')
    .map(event => event.target.value)
    .filter(text => text.length > 2)
    .debounceTime(200)
    .switchMap(text => {
        return this.http(`${url}?q=${text}`);
    });
```

```
const source = Rx.Observable.fromEvent($input, 'keyup')
    .map(event => event.target.value)
    .filter(text => text.length > 3)
    .debounceTime(400)
    .flatMapLatest(text => {
        return this.http(`${url}?q=${text}`);
    });

source.subscribe(
            result => updateView(result),
            error => console.error('Oh my god!!', error));
```

ES.NEXT

DECORATORS

```
class Heisenberg {

    constructor() {
        ...
    }

}
```

DECORATORS

```
function canCook(target) {
    target.canCook = true;
}

@canCook
class Heisenberg {

    constructor() {
        ...
    }

}

const heisenberg = new Heisenberg();
heisenberg.canCook // true
```

DECORATORS

```
@template(`
    <button>Click me</button>
`)
class ButtonController {

}
```

DECORATORS

```
import {autoInject, dependencyInjection} from 'autoinject';

class Db {
    connectionString: 'user:pass@local';
}

@autoInject
class MyClass {

    db: Db;

    constructor(db: Db) {
        this.db = db;
    }
}

const myClass = dependencyInjection(MyClass);
myClass.db.connectionString // 'user:pass@local'
```

DECORATORS

```
class MrWhite {

    @readOnly name = 'Heisenberg';

    knocking() {
        return 'I am the one who knocks.';
    }

}

function readOnly(target, name, descriptor) {
    descriptor.writable = false;
}
```

DECORATORS

```
class MyClass {

    @memorizeFor(1000)
    fetchFromServer(delta) {
        ...
    }

    @trace()
    complexFunction() {
        ...
    }

}
```

OTHER ES.NEXT FEATURES

- System global: https://github.com/tc39/proposal-global
- Call constructor: https://github.com/tc39/ecma262/blob/master/workingdocs/callconstructor.md
- observable: https://github.com/zenparsing/es-observable
- decorators: https://github.com/wycats/javascript-decorators/blob/master/README.md
- async/await https://github.com/tc39/ecmascript-asyncawait
- SIMD: http://tc39.github.io/ecmascript_simd/
- This binding: https://github.com/zenparsing/es-function-bind

TYPESCRIPT

TYPESCRIPT

- Boolean
- Null
- Undefined
- Number
- String
- Symbol
- Object

TYPESCRIPT

- `typeof Array() === 'object'`
- `typeof ( new Map() ) === 'object'`
- `typeof class {} === 'function'`
- `typeof NaN === 'number'`
- `typeof document.all === 'undefined'`
- `typeof null === 'object'`

TYPESCRIPT

```
function area(r) {
    return Math.PI * r * r;
}

console.log(area('O'));
```

TYPESCRIPT

```
function area(r: number): number {
    return Math.PI * r * r;
}
```

JAVASCRIPT VS TYPESCRIPT

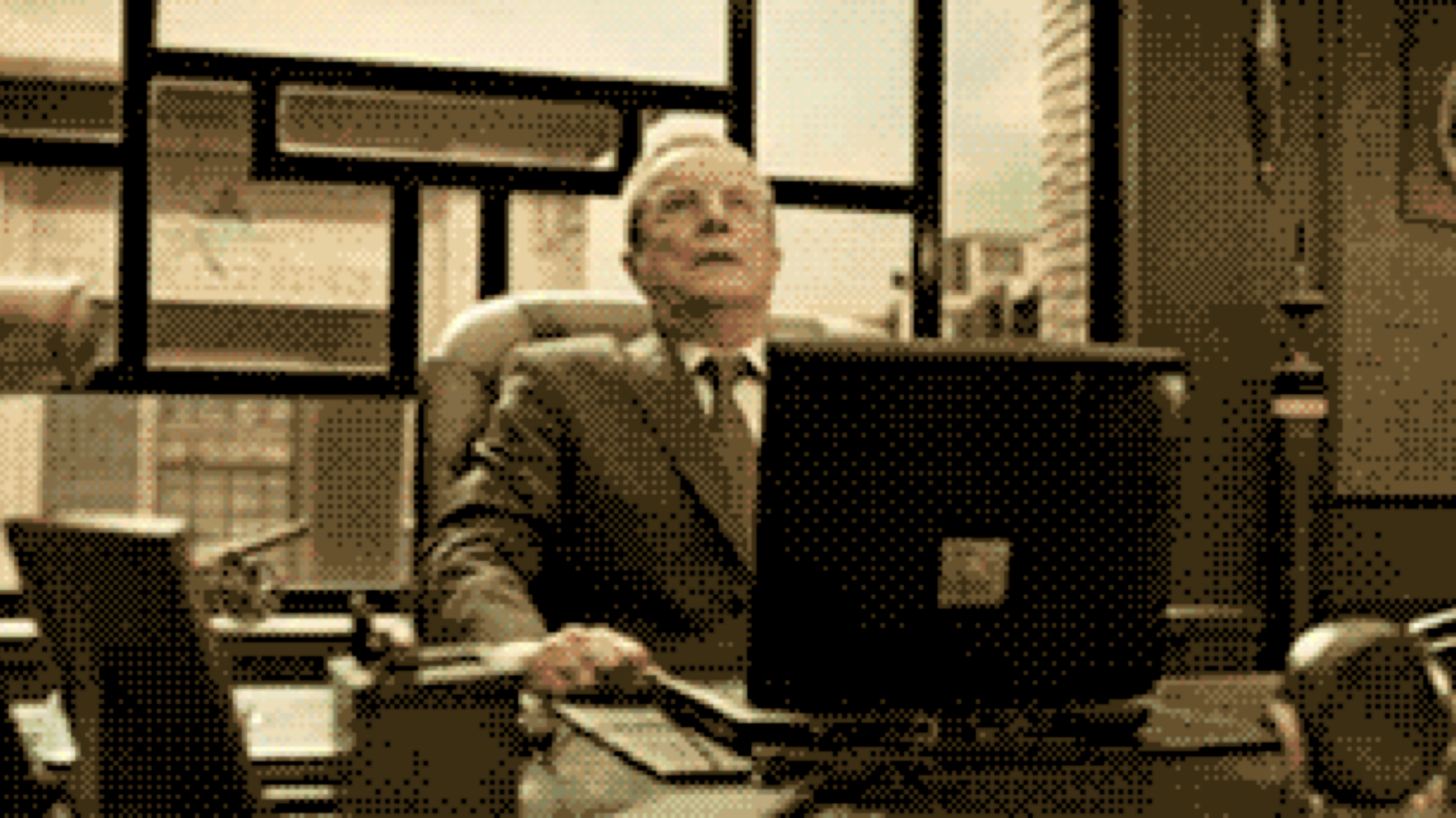
```
const obj = {
    firstNumber: 1,
    secondNumber: 2,
    thirdNumber: 3
};

function add(a, b, c) {
    return a + b + c;
}

const result = add(obj.firstNumber, obj.secondNumber, obj.thirdNunber);
console.log(result);
```

JAVASCRIPT VS TYPESCRIPT

```
const obj = {
    firstNumber: 1,
    secondNumber: 2,
    thirdNumber: 3
};

function add(a, b, c) {
    return a + b + c;
}

const result = add(obj.firstNumber, obj.secondNumber, obj.thirdNunber);
console.log(result); // NaN
```

TYPESCRIPT

```
interface tvShow {
    title: string;
    runtime: number;
    airDate: Date;
    genre: string[];
    episode: episode[];
}

function fetchTvShowsFromServer(): tvShow {
    return http.get('breaking-bad');
}

const show = fetchTvShowsFromServer();
show.title // OK
show.name  // Property 'name' does not exist on type 'tvShow'.
```



```
const obj = {};
obj.name = 'Mr. White';
// Property 'name' does not exist on type {}.
```

TYPESCRIPT

```
const obj: any = {};
obj.name = 'Mr. White';
```

TYPESCRIPT - THE BAD PART

- ≈12k loc / file
- non-module
- differ from ecma262*

FLOW TYPE + BABEL = <3

SONG SCRIPT

ANGULAR 2

```
// app.component.ts
import {Component} from 'angular2/core';

@Component({
    selector: 'my-app',
    template: `<h1>I'm the one who knocks</h1>`
})
class AppComponent { }

export {AppComponent};
```

```
import {bootstrap}    from 'angular2/platform/browser';
import {AppComponent} from './app.component';

bootstrap(AppComponent);
```

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

DATA BINDING

```
import {Component} from 'angular2/core';

@Component({
    selector: 'my-app',
    template: `
        <h1>{{title}}</h1>
        <h2>{{quote}}</h2>
    `
})
export class AppComponent {
  title = 'Breaking Bad';
  quote = `I'm not in the meth business. I'm in the empire business.`;
}
```

ITERATOR

```
import {Component} from 'angular2/core';

@Component({
    selector: 'my-app',
    template: `
        <h1>{{title}}</h1>
        <ul>
            <li *ngFor="#episodeName of episodes">
                {{ episodeName }}
            </li>
        </ul>
    `
})
export class AppComponent {
  title = 'Breaking Bad';
  episodes = [
    'Live Free or Die', 'Madrigal', 'Hazard Pay'
  ];
}
```

LOCAL DOM VARIABLE

```
import {Component} from 'angular2/core';

@Component({
    selector: 'search',
    template: `
        <input #searchInput (keyup)="undefined">
        <p>Value: {{searchInput.value}}</p>
    `
})
export class SearchComponent {}
```

http://plnkr.co/edit/7s5RJAKXdvFXPD31TdK8?p=preview

EVENTS

```
import {Component} from 'angular2/core';

@Component({
    selector: 'search',
    template: `
        <input (keyup)="onKeyUp($event)">
        <p>Value: {{value}}</p>
    `
})
export class SearchComponent {
    value;

    onKeyUp(event) {
        this.value = event.target.value;
    }
}
```

http://plnkr.co/edit/7s5RJAKXdvFXPD31TdK8?p=preview

PROPERTY BINDINGS

```
import {Component} from 'angular2/core';

@Component({
    selector: 'login',
    template: `
        <input [value]="username">
    `
})
export class LoginComponent {
    username = 'Jesse';
}
```

http://plnkr.co/edit/00Y7uBnSFUNEaJSKGf61?p=preview

PROPERTY BINDING

```
<input [value]="username">
```

EVENT BINDING

```
<input (keyup)="onKeyUp($event)">
```

2-WAY BINDING?

2-WAY BINDING

```
<input [value]="username" (input)="username=$event.target.value" >
```

2-WAY BINDING - ANGULAR WAY

```
<input [ngModel]="username" (ngModelChange)="username=$event" >
```

2-WAY BINDING - ANGULAR WAY

```
<input [(ngModel)]="username">
```

2-WAY BINDING - IMPLEMENTATION

```
@Directive({
  selector: '[ngModel]',
  host: {
    '[value]': 'ngModel',
    '(input)': 'ngModelChange.next($event.target.value)'
  }
})
class NgModel {
  @Input() ngModel;
  @Output() ngModelChange = new EventEmitter();
}
```


COMPONENT IN COMPONENT

```
import {Component, Input, Output, EventEmitter, Directive} from 'angular2/core';

@Component({
    selector: 'quote-generator',
    template: `<button (click)="onClick()">{{buttonText}}</button>`
})
class QuoteGeneratorComponent {
    @Input() buttonText = '';
    @Output() newQuote = new EventEmitter();
    quotes = ['Say my name', 'I am the one who knocks', 'Science bitch', 'Stay out of my territory'];

    onClick() {
      const random = Math.floor(Math.random() * this.quotes.length);
      this.newQuote.next(this.quotes[random]);
    }
}

@Component({
    selector: 'quote',
    template: `
        <quote-generator [buttonText]="buttonText" (newQuote)="updateQuote($event)"></quote-generator>
        {{quoteText}}
    `,
    directives: [QuoteGeneratorComponent]
})
export class SearchComponent {
    buttonText = 'Click me';
    quoteText = '';

    updateQuote(quote) {
      this.quoteText = quote;
    }
}
```

http://plnkr.co/edit/8PDsCRsZa1dCLovTaUWY?p=preview

```
@Component({
    selector: 'quote-generator',
    template: `<button (click)="onClick()">{{buttonText}}</button>`
})
class QuoteGeneratorComponent {
    @Input() buttonText = '';
    @Output() newQuote = new EventEmitter();
    quotes = ['Say my name', 'I am the one who knocks', 'Science bitch', 'Stay out of my territory'];

    onClick() {
      const random = Math.floor(Math.random() * this.quotes.length);
      this.newQuote.next(this.quotes[random]);
    }
}
```

```
@Component({
    selector: 'quote',
    template: `
        <quote-generator
            [buttonText]="buttonText"
            (newQuote)="updateQuote($event)">
        </quote-generator>
        {{quoteText}}
    `,
    directives: [QuoteGeneratorComponent]
})
export class SearchComponent {
    buttonText = 'Click me';
    quoteText = '';

    updateQuote(quote) {
      this.quoteText = quote;
    }
}
```

DEPENDENCY INJECTION

```
import {Component} from 'angular2/core';

class QuoteService {
  quote = 'Test data';
}

@Component({
    selector: 'quote',
    template: `{{quoteText}}`
})
export class QuoteComponent {
  quoteText = '';
  constructor(service: QuoteService) {
    this.quoteText = service.quote;
  }
}
```

DI Exception
No provider for QuoteService! (QuoteComponent -> QuoteService)

1. BOOTSTRAP

```
import {bootstrap} from 'angular2/platform/browser';
import {QuoteComponent} from './quote.component';

bootstrap(QuoteComponent);
```

```
import {bootstrap} from 'angular2/platform/browser';
import {QuoteComponent} from './quote.component';
import {QuoteService} from './quote.service';

bootstrap(QuoteComponent, [
    QuoteService
]);
```

```
import {bootstrap} from 'angular2/platform/browser';
import {QuoteComponent} from './quote.component';
import {QuoteService} from './quote.service';

bootstrap(QuoteComponent, [
    provide(QuoteService, {useClass: QuoteService})
]);
```

2. COMPONENT PROVIDERS

```
import {Component} from 'angular2/core';

class QuoteService {
  quote = 'Test data';
}

@Component({
    selector: 'quote',
    template: `{{quoteText}}`,
    providers: [QuoteService]
})
export class QuoteComponent {
  quoteText = '';
  constructor(service: QuoteService) {
    this.quoteText = service.quote;
  }
}
```

ROUTER

```
@Component({
    template: `<router-outlet></router-outlet>`
})
@RouteConfig([
  {path:'/search', name: 'Search', component: SearchComponent},
  {path:'/shows', name: 'Shows', component: ShowsComponent},
  {path:'/show/:id', name: 'Show', component: ShowComponent}
])
class AppComponent { }
```

```
@Component(...)
@RouteConfig([
  ...
  {path:'/show/:id', name: 'Show', component: ShowComponent}
])
class AppComponent { }

@Component(...)
class ShowComponent {
  constructor(params: RouteParms) {
    this.showId = params.get('id');
  }
}
```

```
<h1>Amazing app</h1>
<nav>
    <a [routerLink]="['Shows']">Shows</a>
    <a [routerLink]="['Show', {id: '5'}]">Heroes</a>
</nav>
<router-outlet></router-outlet>
```

ZONE.JS

$scope.$apply()

ZONE.JS

THE GOOD, THE BAD AND THE UGLY

THE GOOD

```
setTimeout(() => {
    // This will trigger a DOM update
    this.viewData = 'New view data';
}, 100);
```

THE BAD

```
setInterval(() => {
    // Paint the Sunset with me
    window.requestAnimationFrame(sunset);
}, 10);
```

```
import { Component, Input, ChangeDetectionStrategy } from 'angular2/core';

@Component({
    selector: 'my-comp',
    template: `{{myData.id}}, {{myData.name}}`,
    changeDetection: ChangeDetectionStrategy.OnPush
})
class MyComponent {
    @Input() myData;

    ngOnChanges(inputChanges) {
        if (inputChanges.myData) { // We have a new model
            console.log(inputChanges.myData.currentValue);
        }
    }
}
```

THE UGLY


MONKEY PATCH
ALL THE THINGS!
memegenerator.net

THE UGLY

```
const orgSetInterval = window.setInterval;
window.setInterval = function(...args) {
    return zone.run(orgSetInterval, args);
}
```

OTHER FEATURES

- Change detection
- Testing
- Pipes
- Lifetime hooks
- Directives

BEYOND ANGULAR

```
const angular = 1 * '🍎';
const react = 1 * '🍐';

angular > react   // false
angular < react   // false
angular === react // false
```

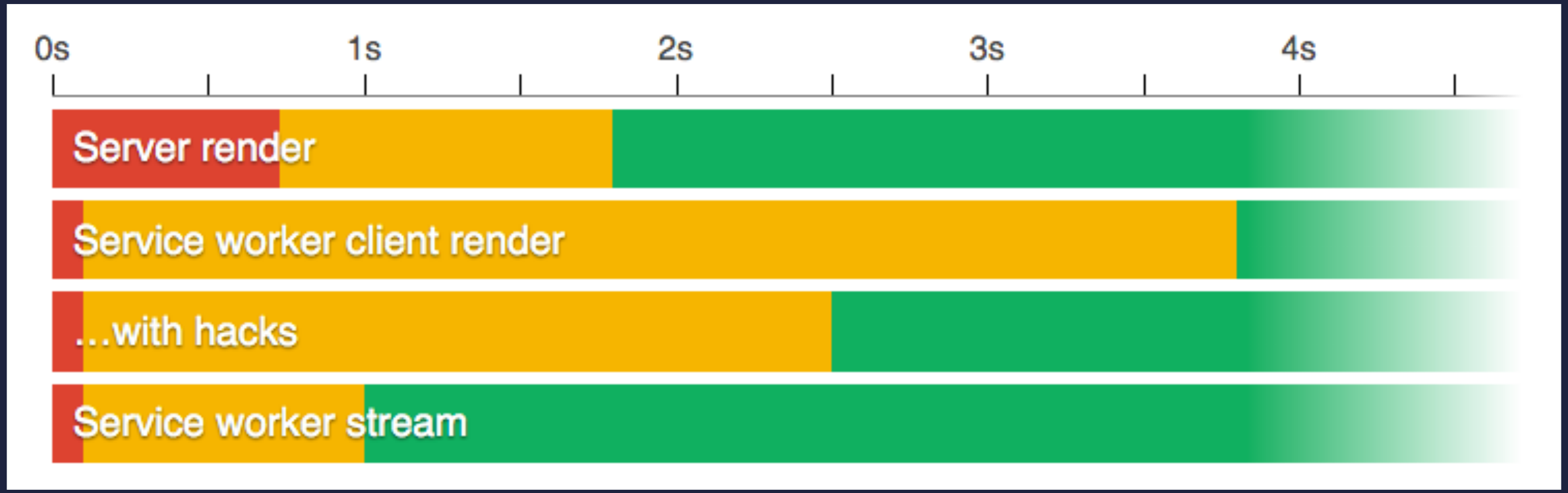
6 5TH CONSTRAINTS FOR PERFORMANCE

6[5TH] CONSTRAINTS FOR PERFORMANCE

- Under 60 kB css
- Under 60 kB html
- Under 60 kB javascript
- 60 fps
- .6 sec avg latency

https://developers.google.com/web/showcase/case-study/googleplus

- Size doesn't matter with service worker
- Use web workers
- Cache everything on local device



https://jakearchibald.com/2016/streams-ftw/

LABBA MED ANGULAR

LABBA MED ANGULAR

- https://github.com/tjoskar/angular2-labb
  - Uppdelat i 7 delar
  - Chapter 0 - Up and running
  - Chapter 1 - Bli varm i kläderna
  - Chapter 2 - Autocomplete search
  - Chapter 3 - Listning av sparad data
  - Chapter 4 - Testning
  - Chapter 5 - Change detection / View Encapsulation
  - Chapter 6 - Service worker and Web workers

TACK