ogury : croître avec JavaScript en 2019

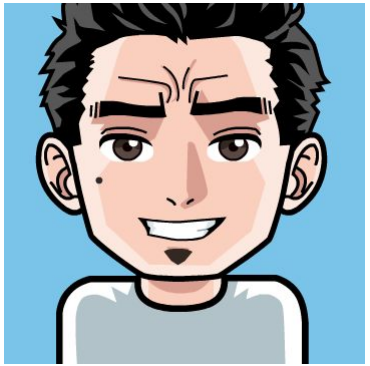
Qui sommes nous ?

David Babel

@DavidLearnJS

ogury

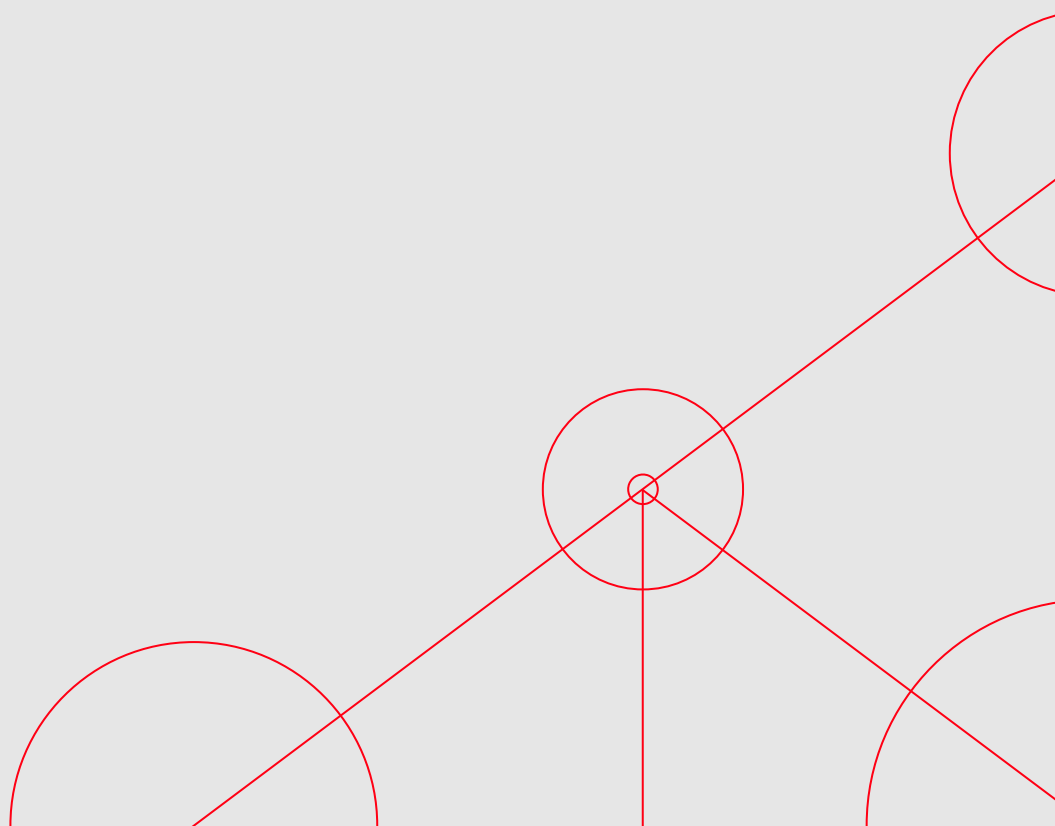
Mobile Journey Marketing



Carles Sistare

@CarlesSistare

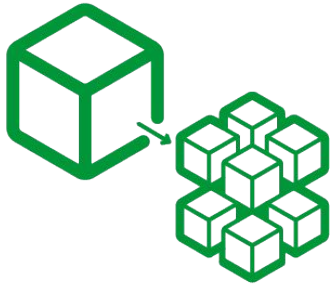
Pourquoi sommes nous là ?



Plan de la présentation

- Croissance d’Ogury
- Notre Approche micro services
- Bonne pratiques techniques pour éviter la dette technique

ogury

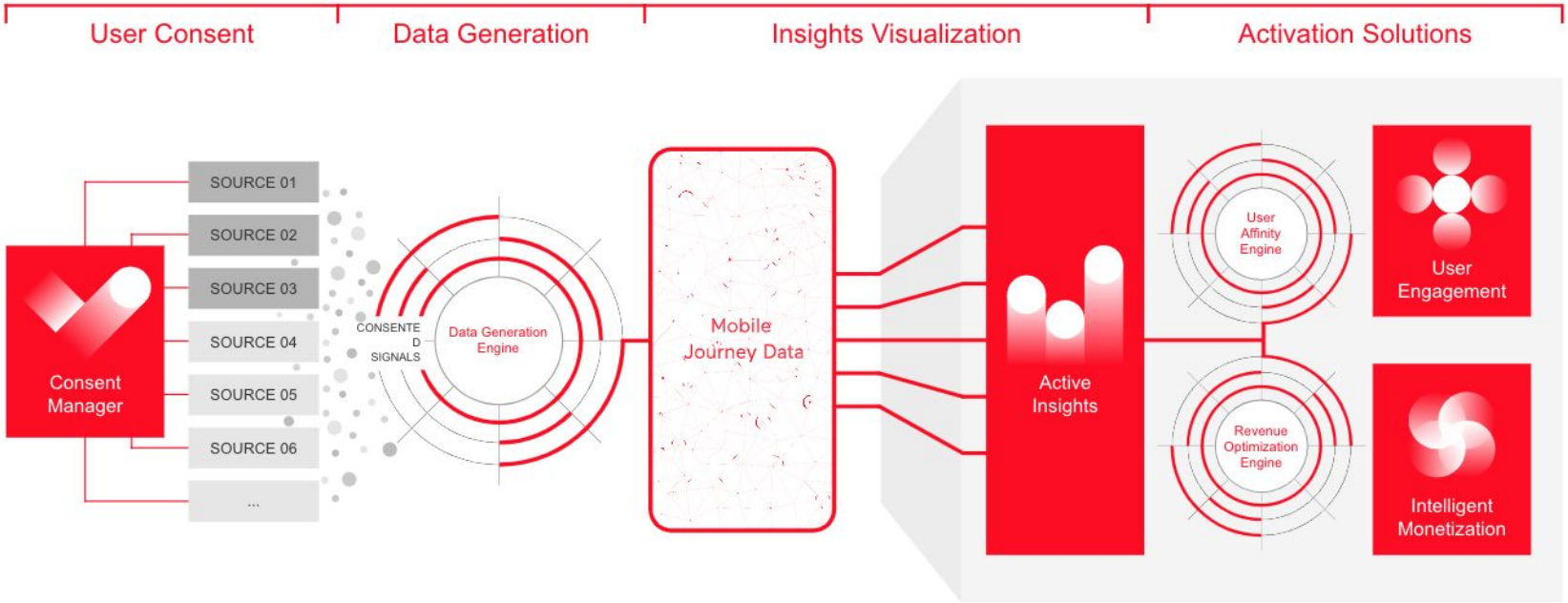


Micro services

Partie 1

La croissance d'ogury

Comprendre les contraintes



Contraintes

Forte charge

- 100k QPS
- 2To par Jour

01

Temps de réponse limité

10ms

02

Archi complexe

- 50 MServ
- 60 Jobs BigData

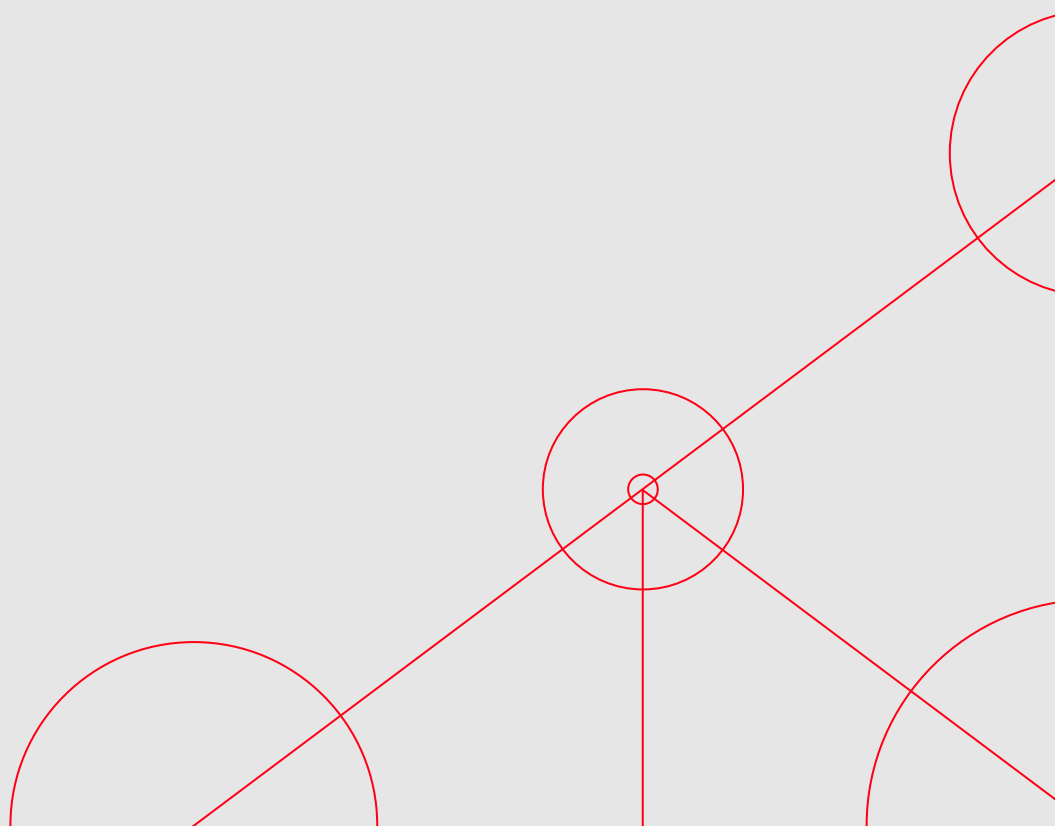
03

Réactivité

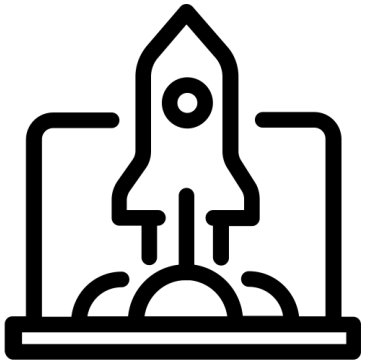
300 Releases par mois

04

De Start-up à Scale-up



De Start-up à Scale-up

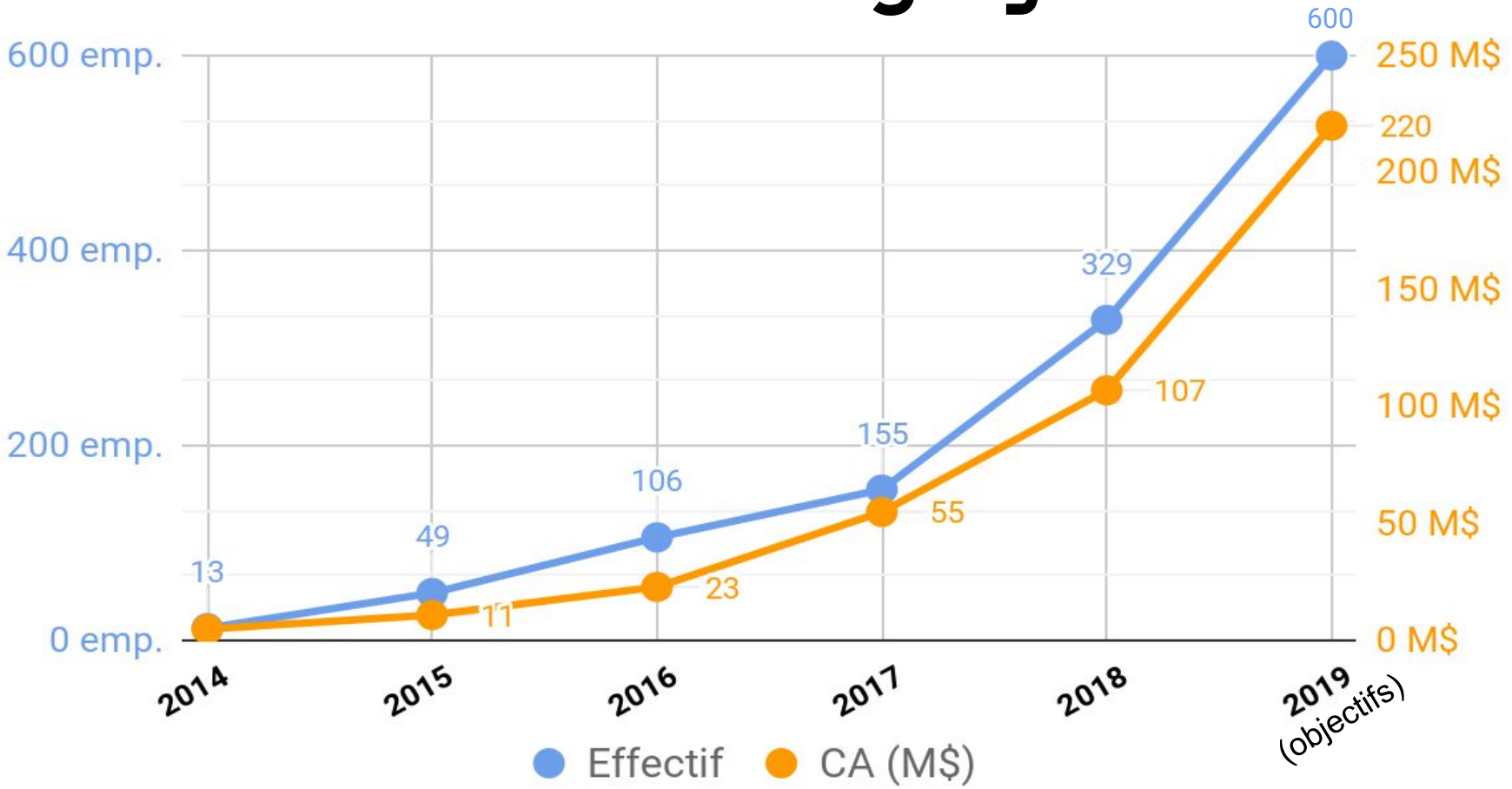


2014

20 équipes

400+ dépôts

Croissance d'ogury



Le Top 100 des Start-up 2017 : un dynamisme confirmé

RANG	RAISON SOCIALE	ACTIVITÉ	ANNÉE DE CREATION	CA 2016 (EN K€)	EFFECTIF 2016	TOTAL DES FONDS LEVÉS (EN K €)
1	WIKO	Constructeur de smartphones	2011	331 561	215	.
2	REMADE GROUP	Reconditionnement de smartphones	2011	131 137	382	22 000
3	INOP'S	Conseil digital	2009	35 608	27	700
4	CROSSCALL SA	Constructeur de smartphones et mobiles outdoor	2009	29 789	84	4 500
5	ARKOLIA ENERGIES	Energies renouvelables	2009	23 000	49	.
6	MOUSTACHE BIKES	Fabrication de cycles électriques	2011	20 466	35	.
7	OGURY	Mobile advertising, Mobile Analytics	**2014**	18 350	105	18 512

Source: https://lentreprise.lexpress.fr/creation-entreprise/le-top-100-des-start-up-2017-un-dynamisme-confirme_1942988.html

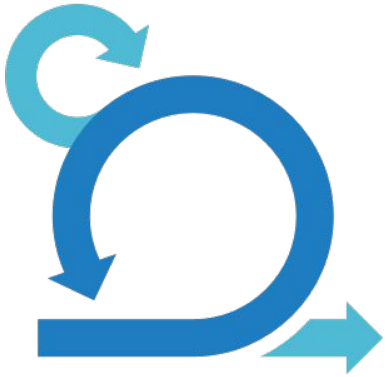
Scale-Up Awards 2018
Ogury sacrée scale-up de l'année

Publié par Mickaël Deneux le 6 avr. 2018 - mis à jour à 17:11

Source: https://www.chefdentreprise.com/Thematique/profession-1056/Breves/OGURY-sacre-Scale-annee-2018-329610.htm

Ogury at Scale

Cloud

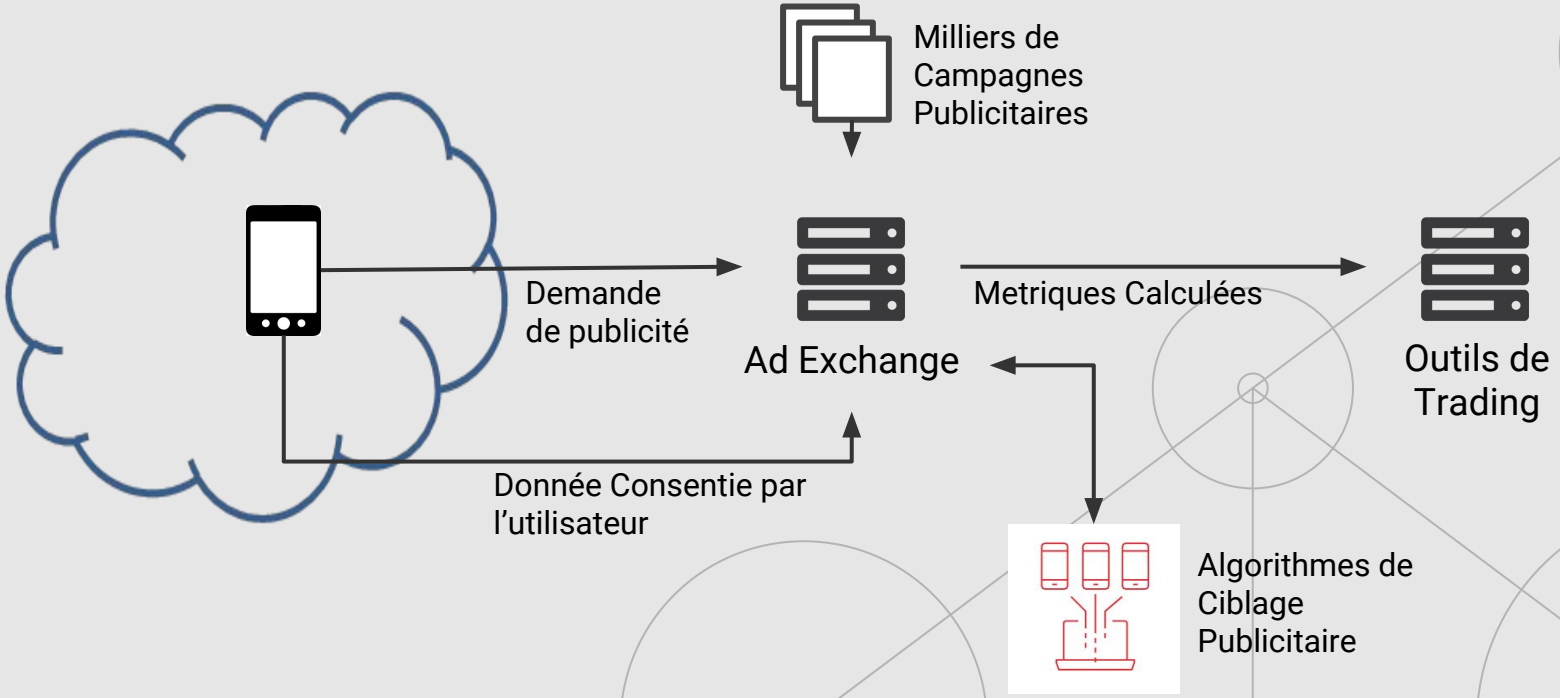


Agile

JavaScript

Devenir une Scale-up avec les Microservices

Ad Exchange = Plateforme de Trading Publicitaire



Contraintes du Scaling

Manque de fluidité d’une Architecture Microservices

01

Manque de scalabilité des tests d’intégration

02

Manque d' homogénéité d’une archi microservices

03

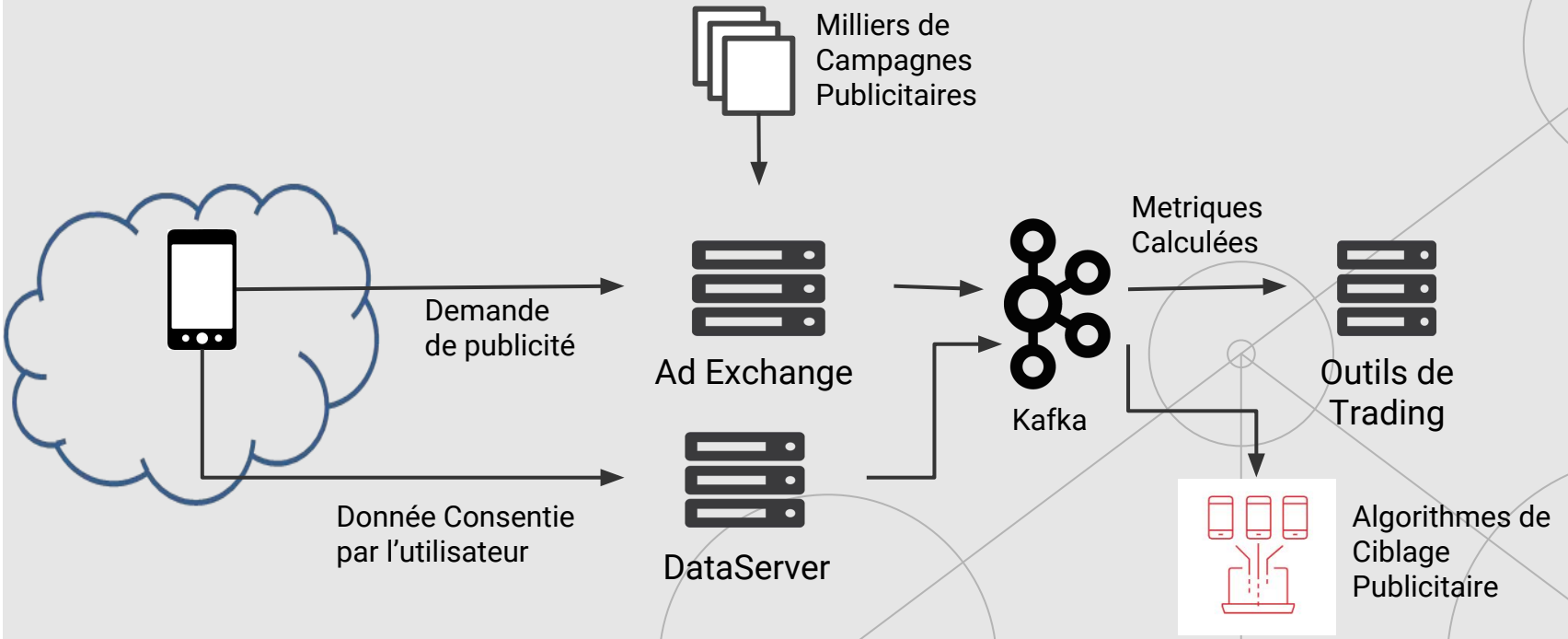
1. Manque de fluidité d'une architecture Microservices

Temps de réponse hors contrôle

Difficultés pour automatiser les tests

Interconnections complèxes

Ne pas reporter à demain ce que nous pouvons faire … avec Kafka



1. Manque de fluidité d'une architecture Microservices

Temps de réponse hors contrôle

Difficultés pour automatiser les tests

Interconnections complèxes

2. Manque de scalabilité des tests d’intégration

On peut pas automatiser les tests End To End

Tests d’intégration isolés avec une cohérence assurée

Evangélisation des Best Practices de nos Services

Chaque service offre ses propres connecteurs

Service de Health Check Intégré

01

Contrats entre Services (*.proto, *.yml)

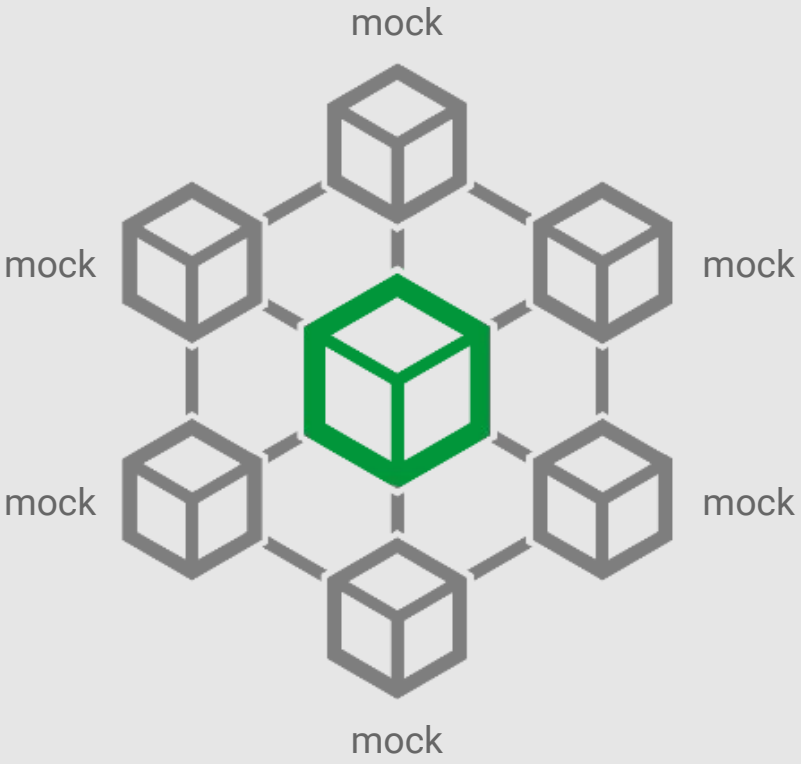
02

Strategie de Recovery

03

Service de MOCK

04


mock
mock
mock
mock
mock
mock

Exemple de mock: COUCHBASE

Mock Testing

As part of the core library, we additionally include a powerful Mock version of the library which supports a significant set of the features that are available when connected to a real Couchbase Server.

As part of this library, we include a mock version of the client that supports nearly the exact same feature set as the library itself, but which does not require that a server be configured. Note that these Mock connections currently are per-instance, if another connection is instantiated, none of the data will be shared.

Using the Mock is as simple as this:

```
var couchbase = require('couchbase').Mock;
var cluster = new couchbase.Cluster();
var bucket = cluster.openBucket();

bucket.upsert('testdoc', {name:'Frank'}, function(err, result) {
  if (err) throw err;

  bucket.get('testdoc', function(err, result) {
    if (err) throw err;

    console.log(result.value);
    // {name: Frank}
  });
```

https://github.com/couchbase/couchnode

Exemple de mock: AWS

Use in your Tests

```
var AWS = require('aws-sdk-mock');

AWS.mock('DynamoDB', 'putItem', function (params, callback){
  callback(null, "successfully put item in database");
});

AWS.mock('SNS', 'publish', 'test-message');

// S3 getObject mock - return a Buffer object with file data
awsMock.mock("S3", "getObject", Buffer.from(require("fs").readFileSync("tes


/**
    TESTS
**/

AWS.restore('SNS', 'publish');
AWS.restore('DynamoDB');
AWS.restore('S3');
// or AWS.restore(); this will restore all the methods and services
```

https://github.com/dwyl/aws-sdk-mock

3. Manque d'homogénéité d'une archi Microservices

Incohérence des contrats parmis les microservices

Incohérence des protocols de communication

Perte de temps de définition de synchro entre services

3. Manque de cohérence d’une archi Microservices

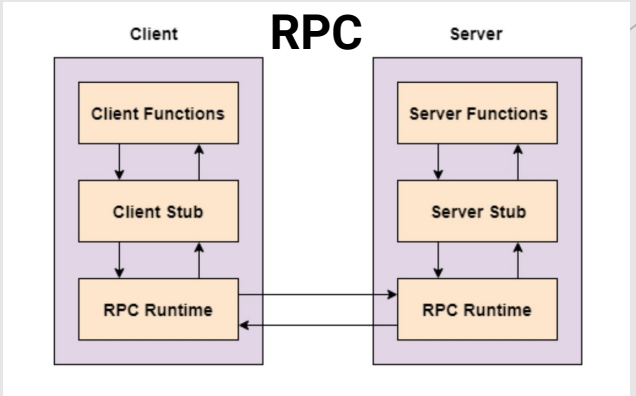
- Temps de réponse élevés en fonction des technos utilisés
- Problèmes de typage sur des champs JSON
- Complexité des connectivités entre services
- Incoherence des contrats entre services

gRPC

protobuf
Protocol Buffers

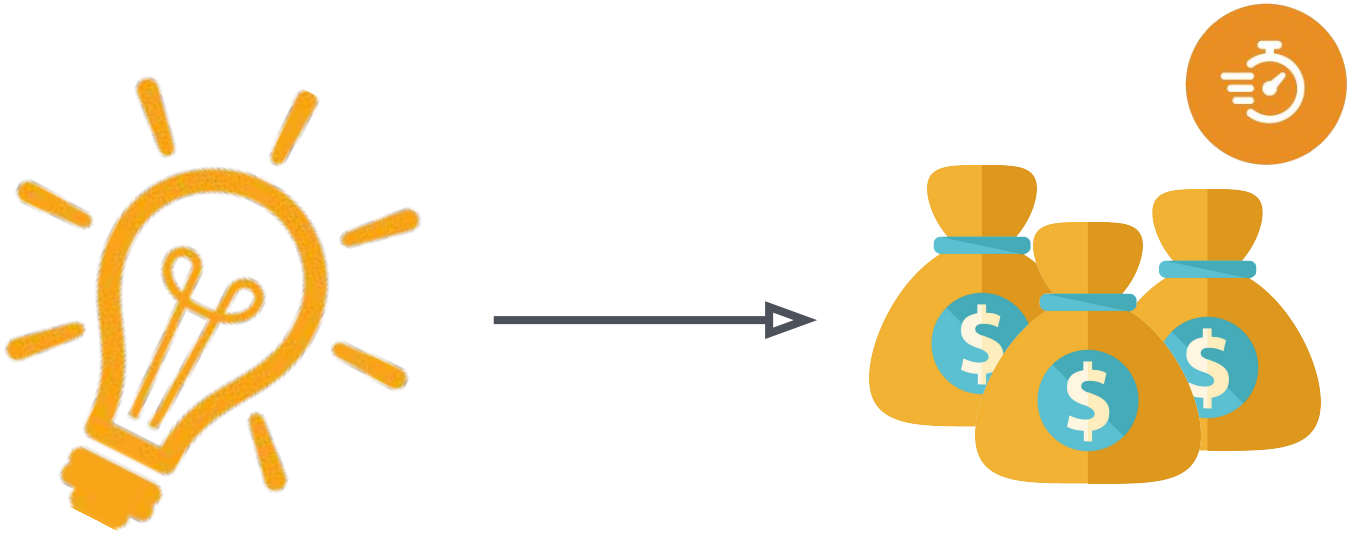
HTTP/2

http://bit.ly/practical-grpc

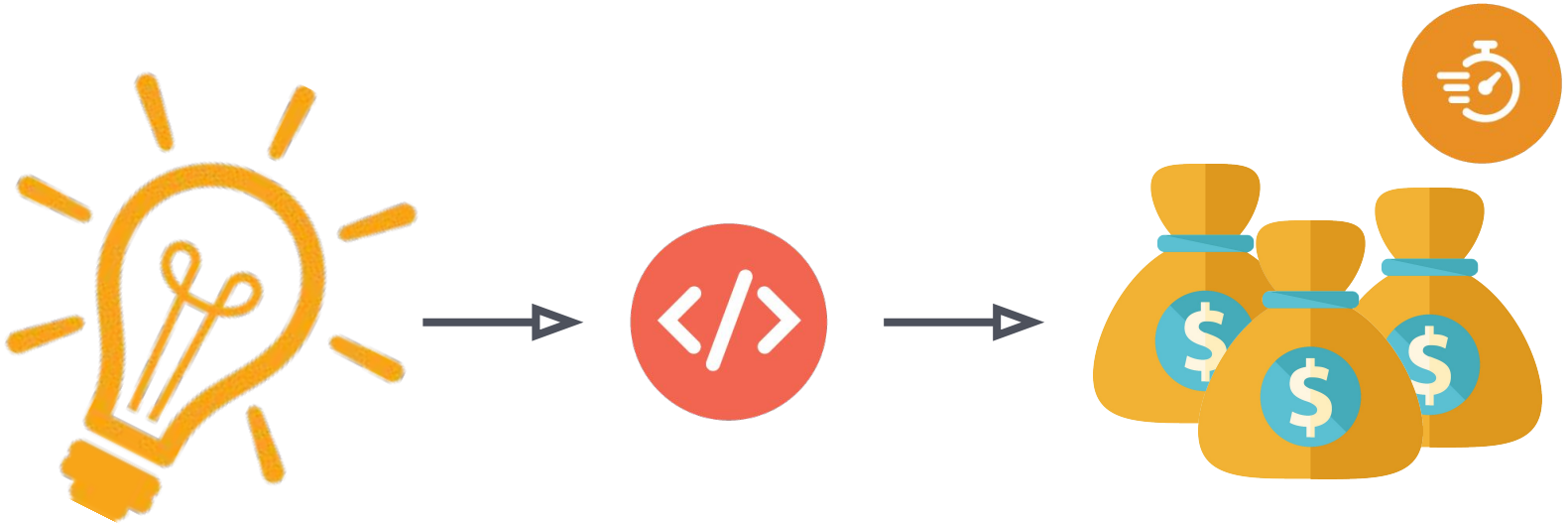


Introduction aux Bonnes pratiques

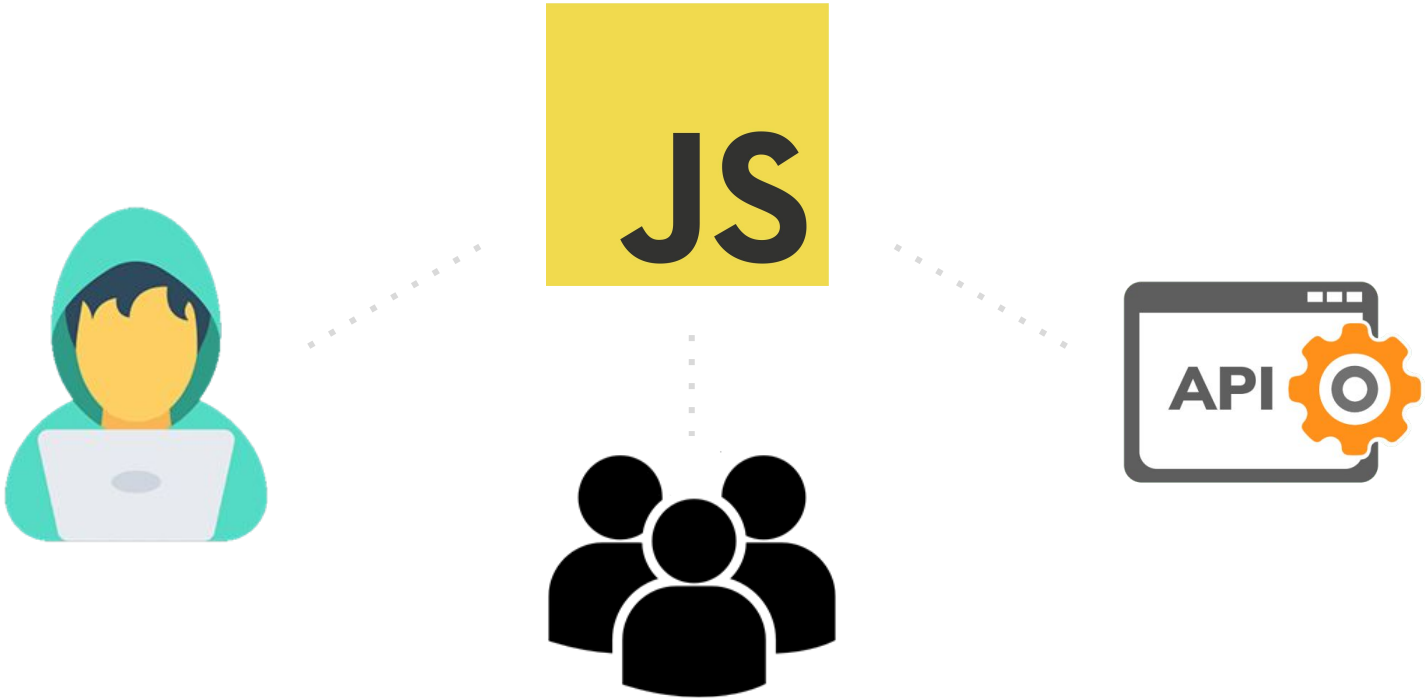
Startup Nation



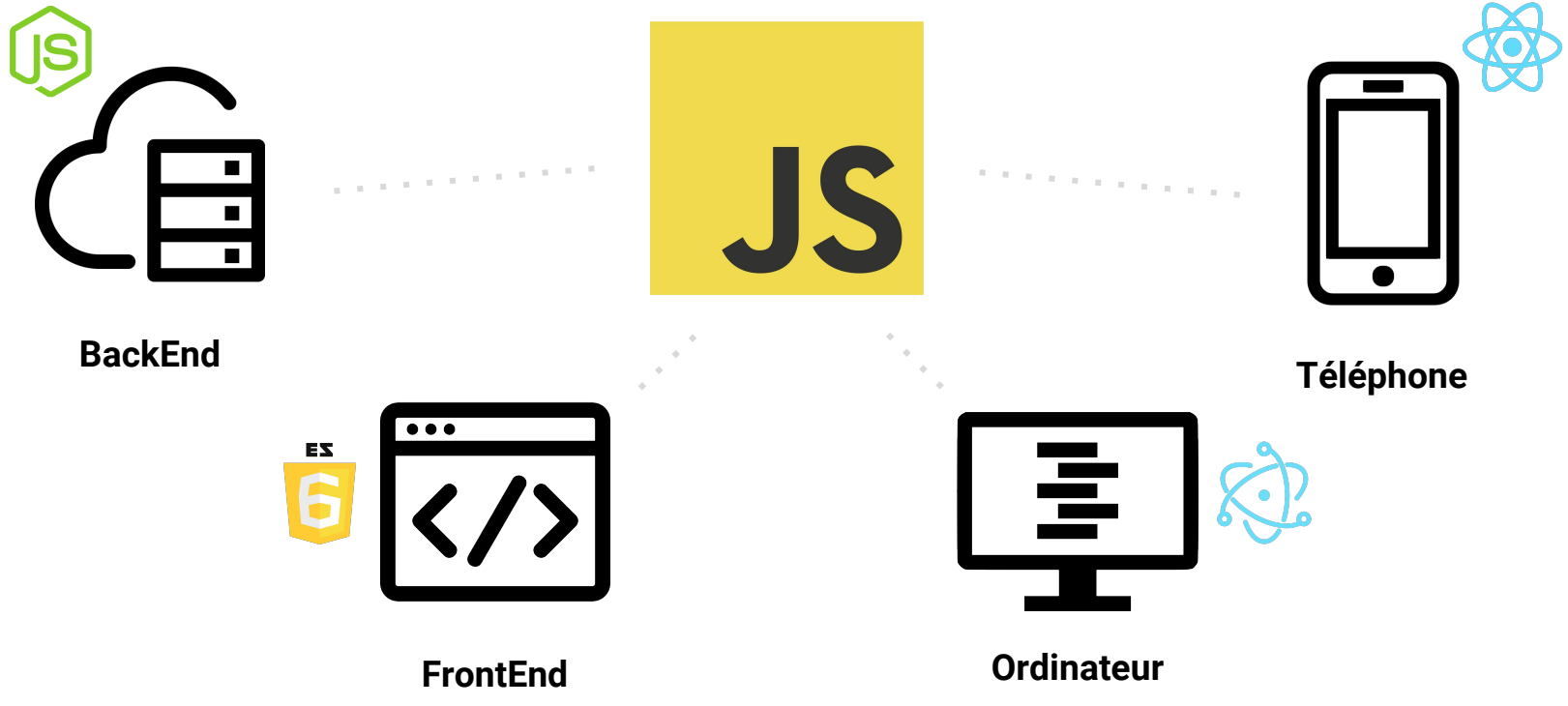
Startup Nation



JavaScript: un allié puissant



JavaScript: un allié puissant



JavaScript: un allié puissant, mais ...

Source: https://tc39.github.io/ecma262/

JavaScript fatigue et Dette technique

La JavaScript Fatigue

Saul: "How's it going?"
Me: "Fatigued."
Saul: "Family?"
Me: "No, Javascript."

Eric Clemmons @ericclemmons
Creator of React Resolver

Source: https://medium.com/@ericclemmons/javascript-fatigue-48d4011b6fc4

La JavaScript Fatigue

Traduit de l'anglais

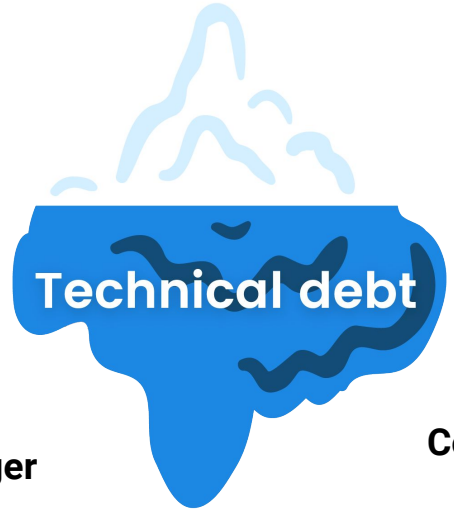
Traduit de l'anglais

Source: https://medium.com/@ericclemmons/javascript-fatigue-48d4011b6fc4

L’ennemi: la dette technique

Toute connaissance à de la valeur

Suis-je bon pour apprendre et solutionner des problèmes ?



Répète après moi :
Les choses vont continuer à changer

Connaître ce que je ne connais pas

Source: Gabriela Motroc : 4 tips to resist JavaScript fatigue (see)

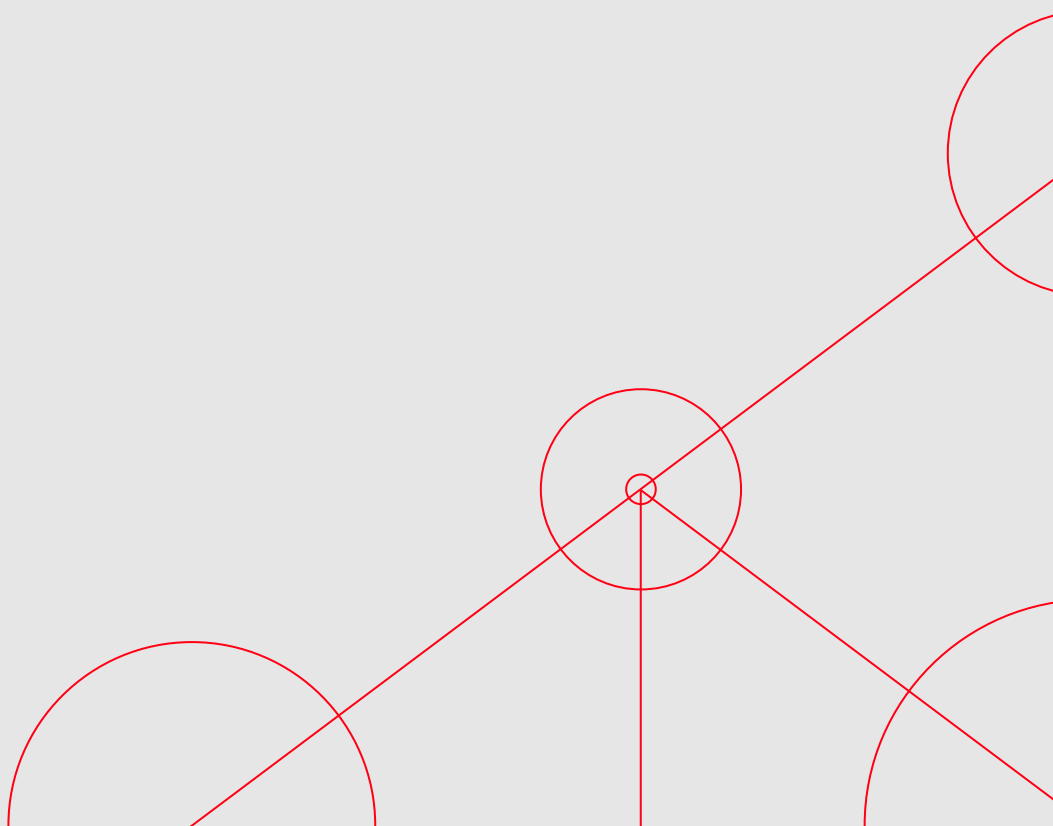
La dette technique

jQuery

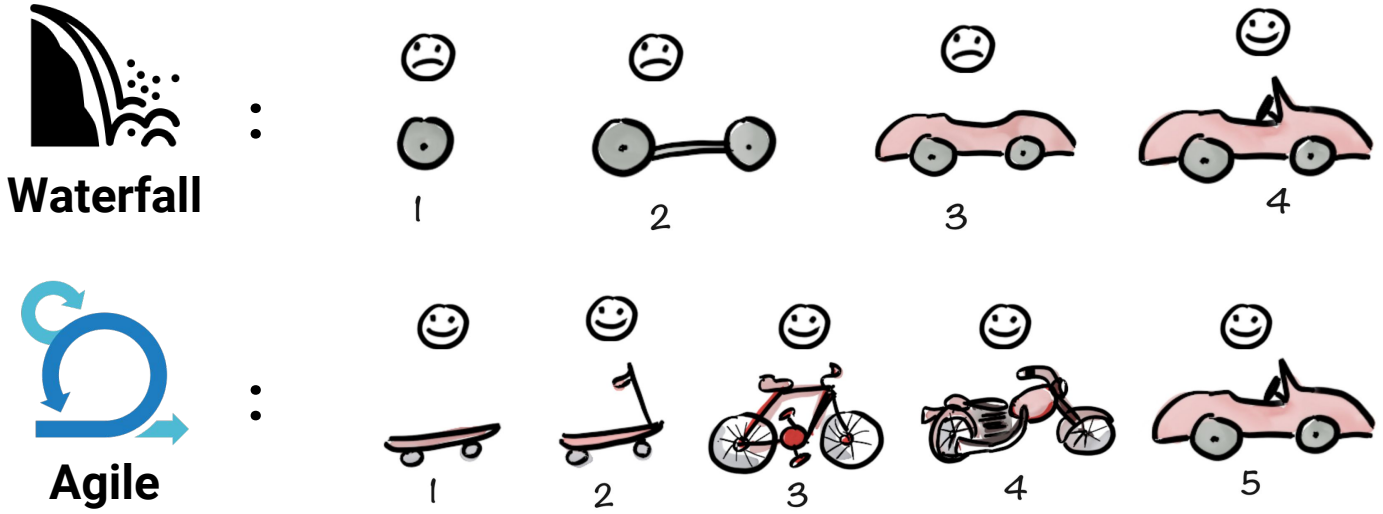
Lo

CoffeeScript

Au fait, c’est quoi une bonne pratique ?

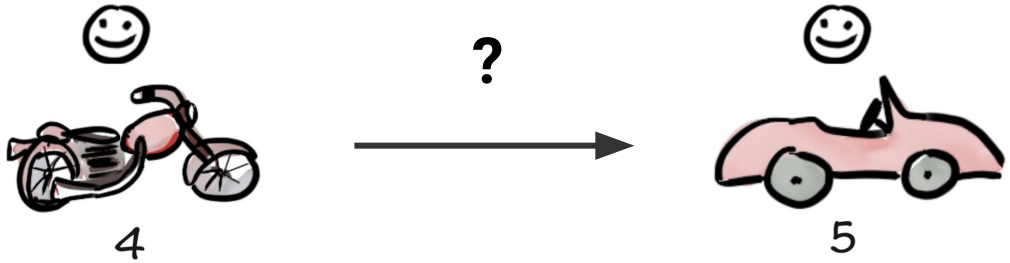


Bonnes pratiques ?



Source: Henrik Kniberg "What is Agile"

Bonnes pratiques ?



Source: Henrik Kniberg “What is Agile”

Bonnes pratiques ?

Transpileur ?

Bundler ?

Bonnes pratiques ?

Transpileur ?

Bundler ?

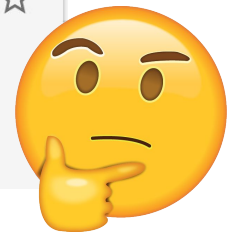
Bonnes pratiques ?

WIKIPEDIA
The Free Encyclopedia

Coding best practices are a set of informal rules that the software development community has learned over time which can help improve the quality of software.[1]

FRENCH ENGLISH SPANISH

Les meilleures pratiques de codage constituent un ensemble de règles informelles que la communauté des développeurs de logiciels a apprises au fil du temps et qui peuvent aider à améliorer la qualité des logiciels.



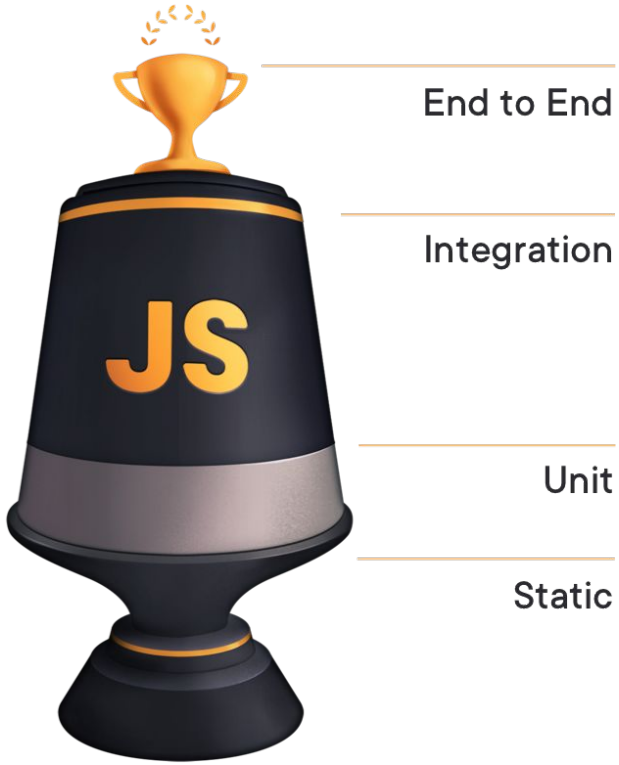
Source: https://en.wikipedia.org/wiki/Best_coding_practices

Définition d’une bonne pratique

- Créer un minimum de dette technique
- Reposer sur des outils stables et répandus
- Etre simple pour être facile à s’approprier

Nos bonnes pratiques

Les tests **automatisés**



Source: Kent C. Dodds : https://testingjavascript.com/

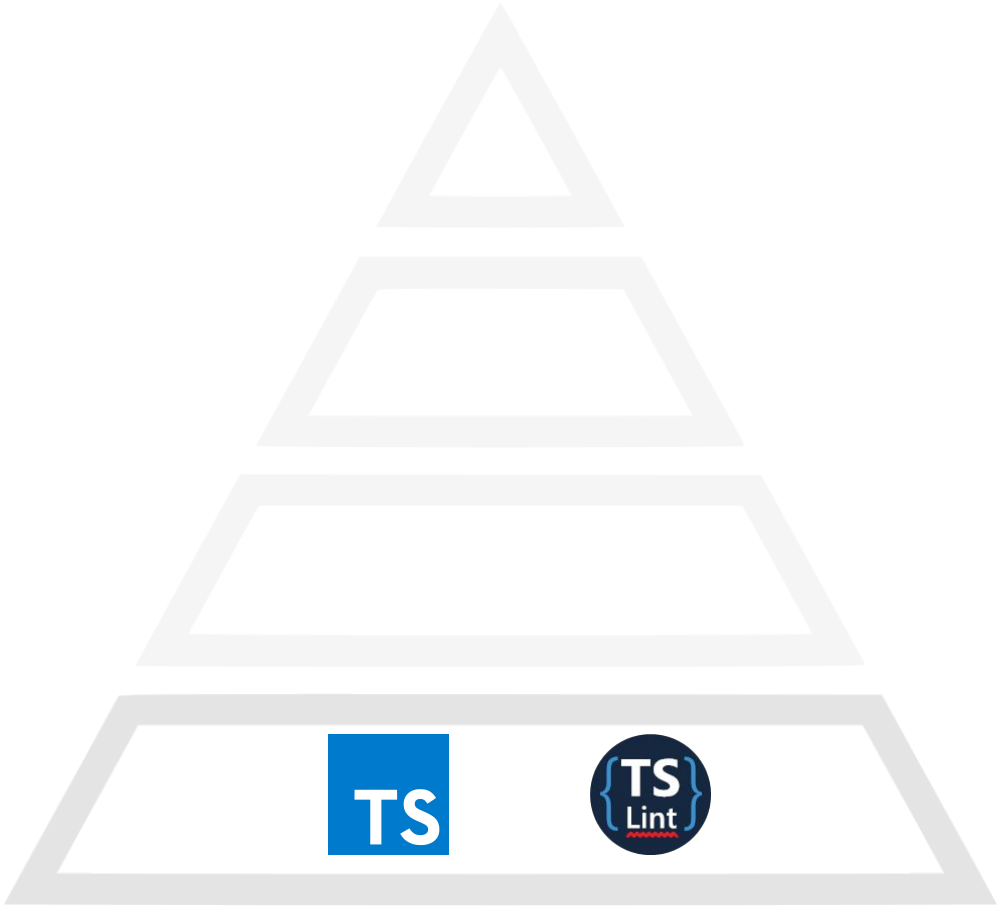
Les “tests” statics

End to End

Integration

Unit

Static



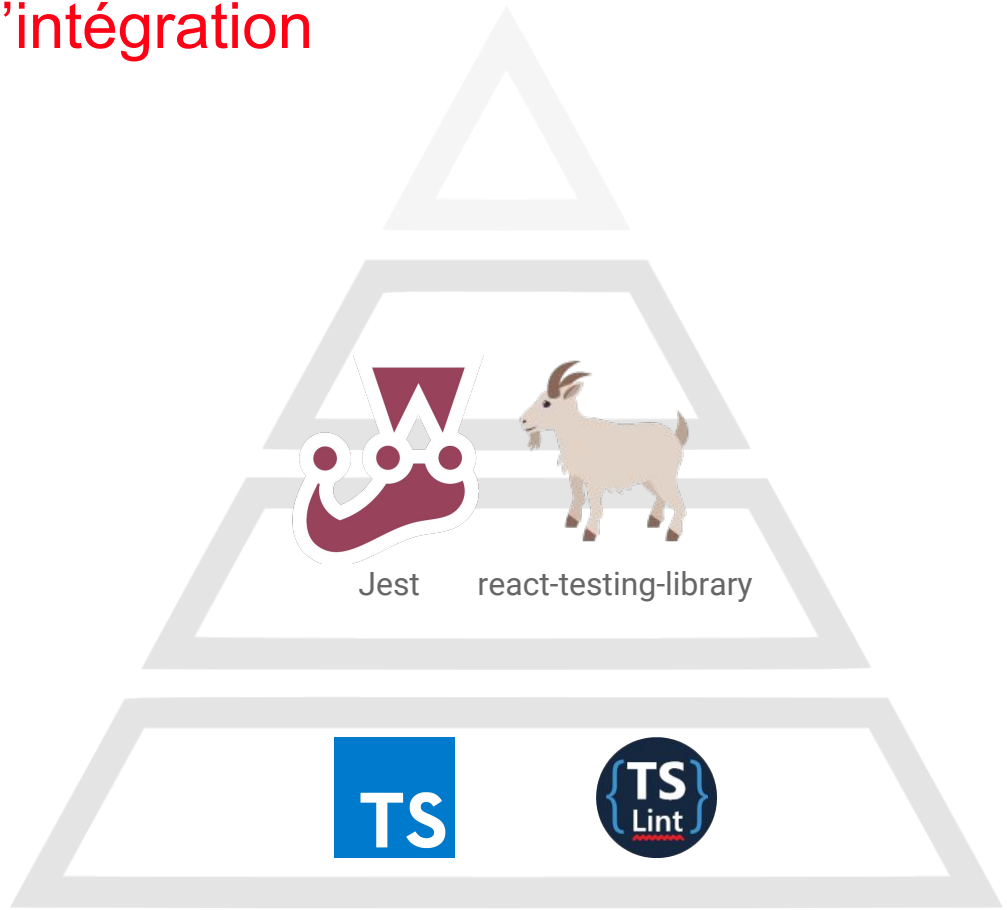
Les tests unitaires et d’intégration

End to End

Integration

Unit

Static



Les tests fonctionnels

End to End

Integration

Unit

Static

SuperTests
no logo

Puppeteer

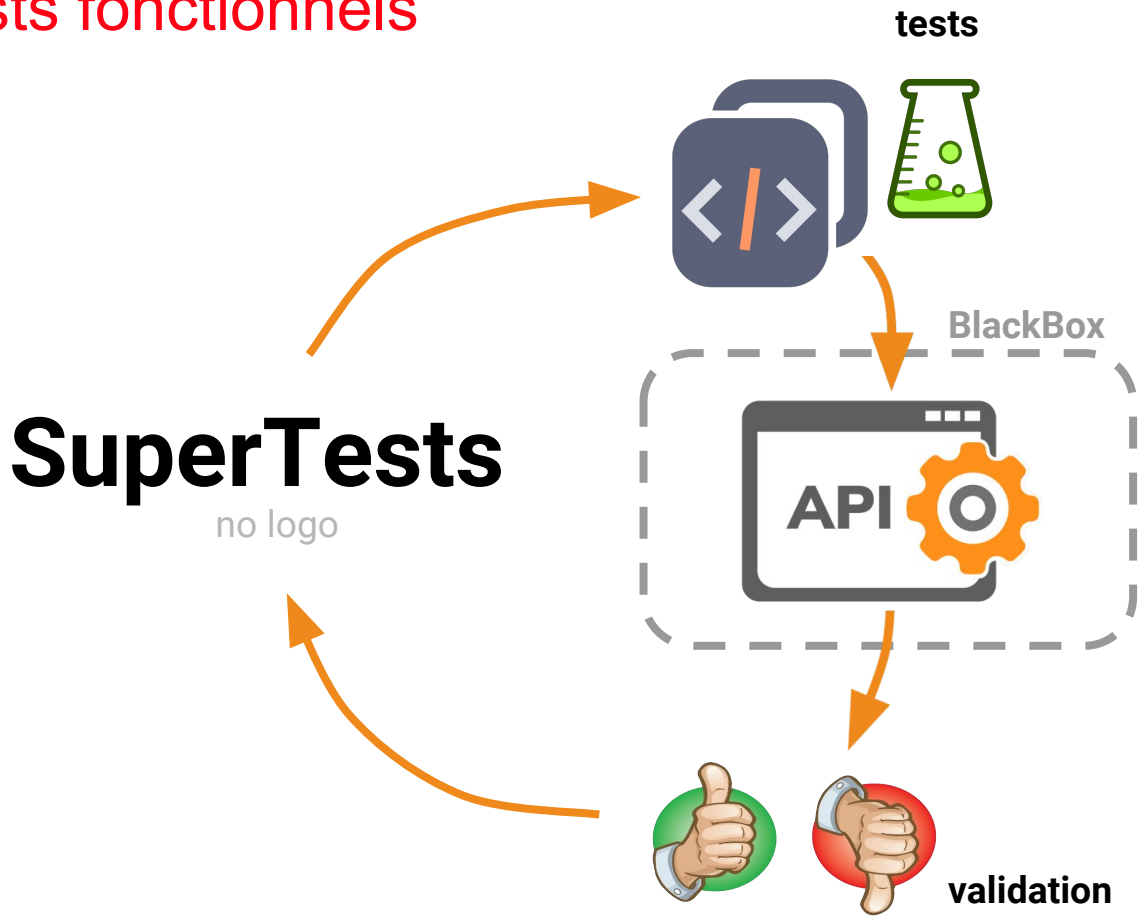
Jest

react-testing-library

TS

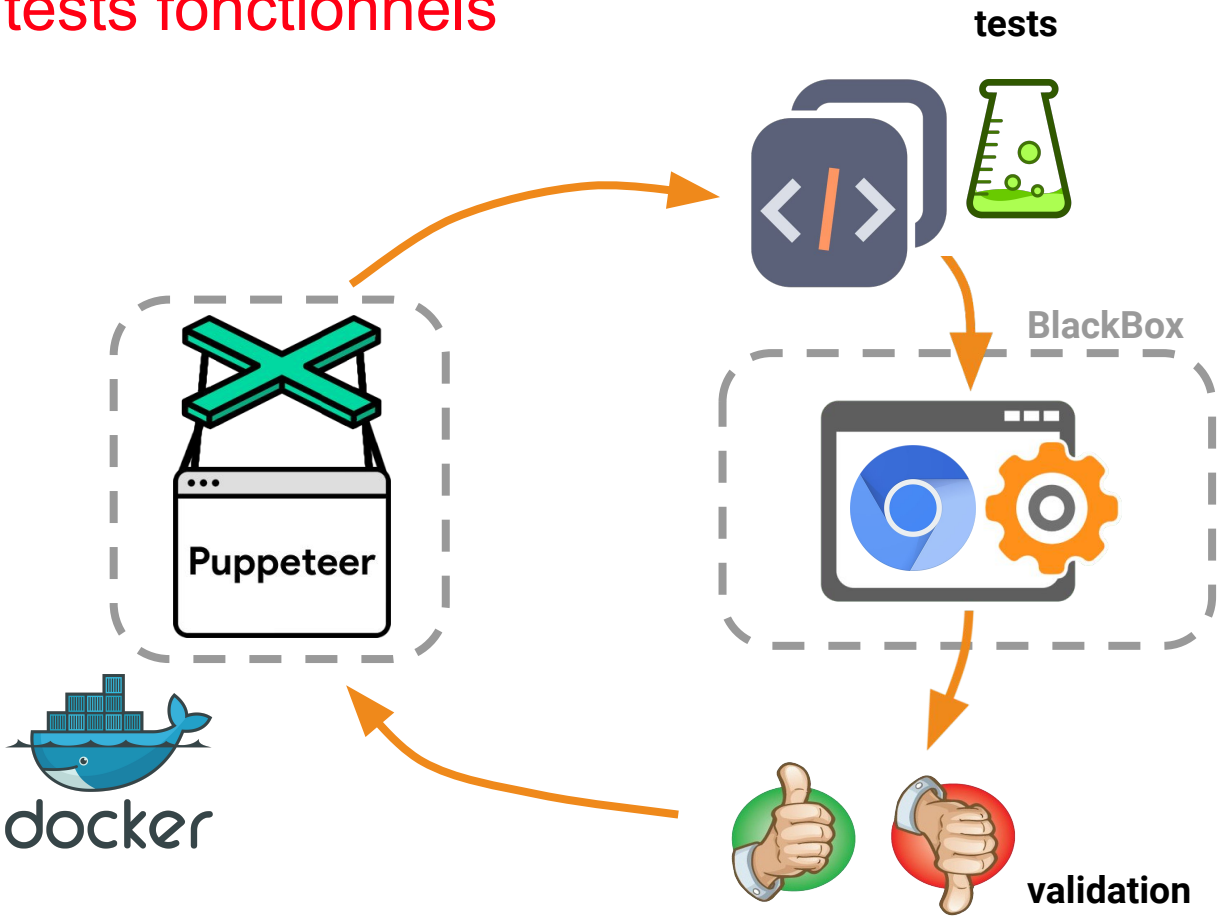
TS Lint

Les tests fonctionnels



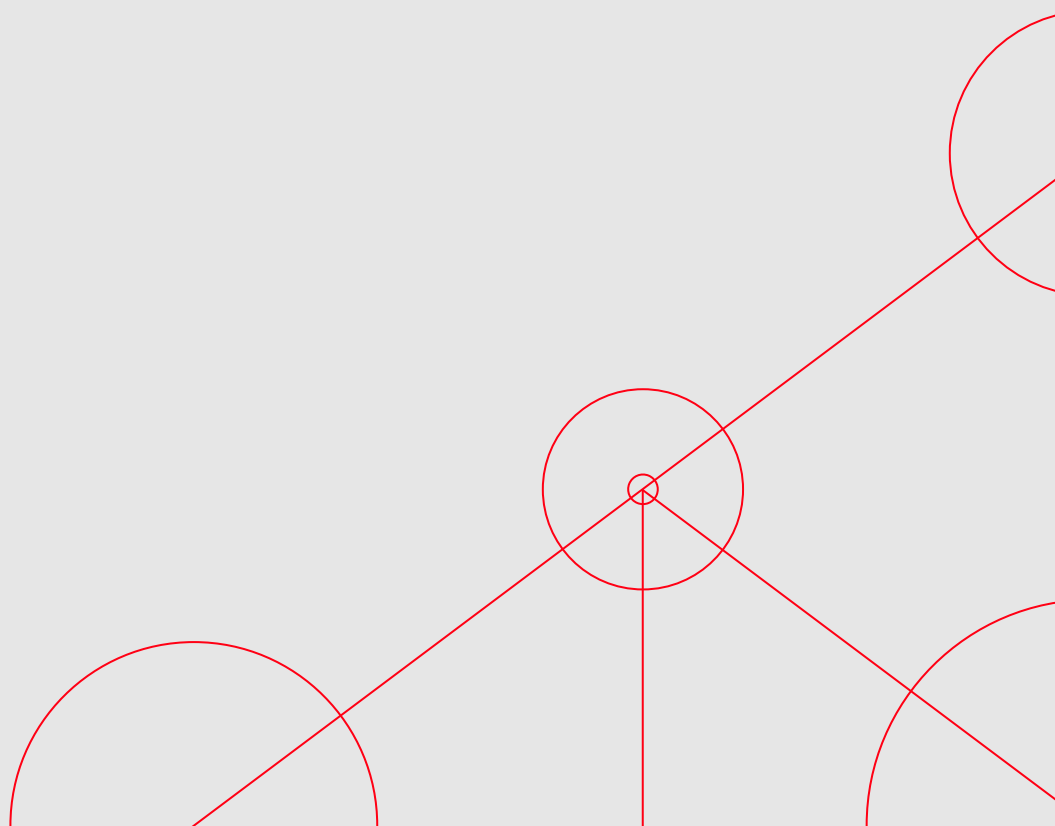
BackEnd

Les tests fonctionnels

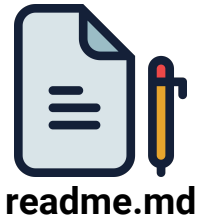


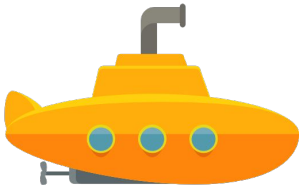
FrontEnd

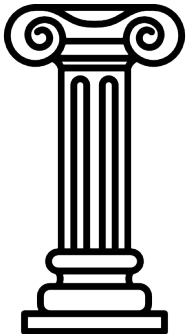
Les 4 piliers d’un bon projet

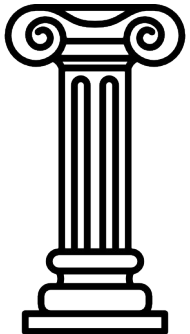


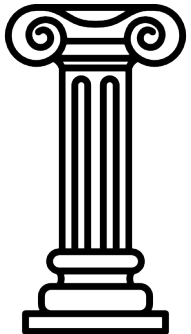
Les 4 piliers d’un bon projet

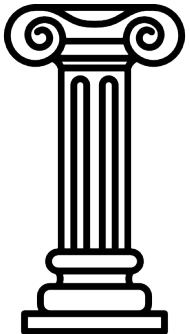








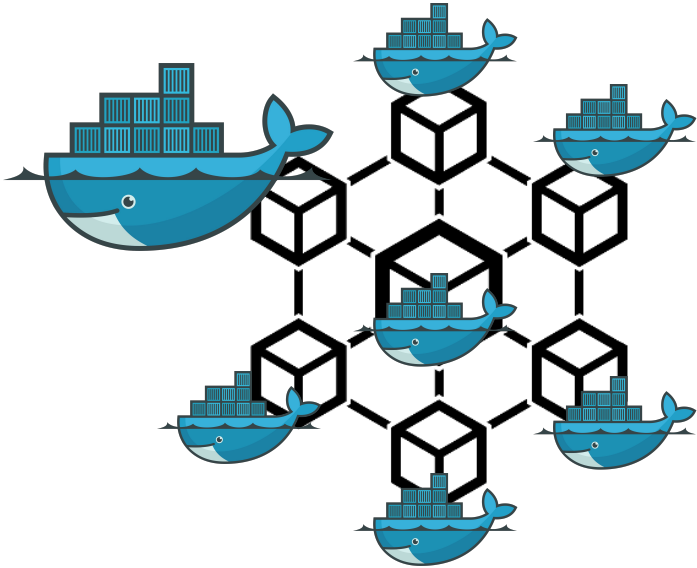




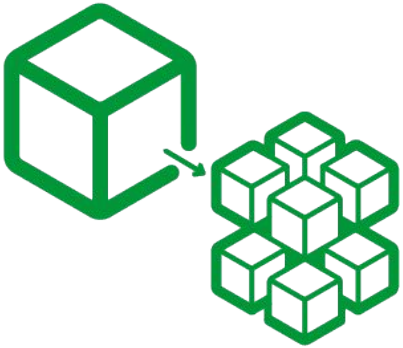
L’intégration continue

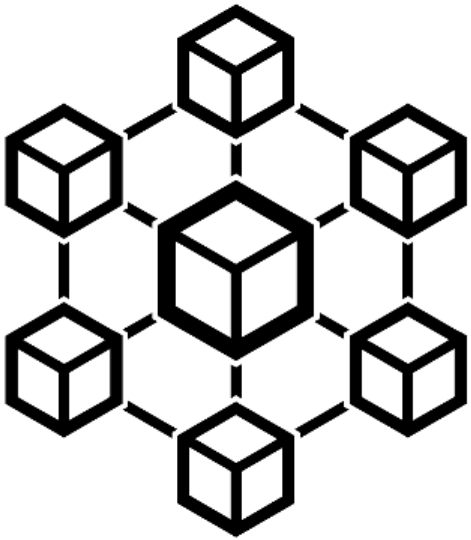
Jenkins

Travis CI

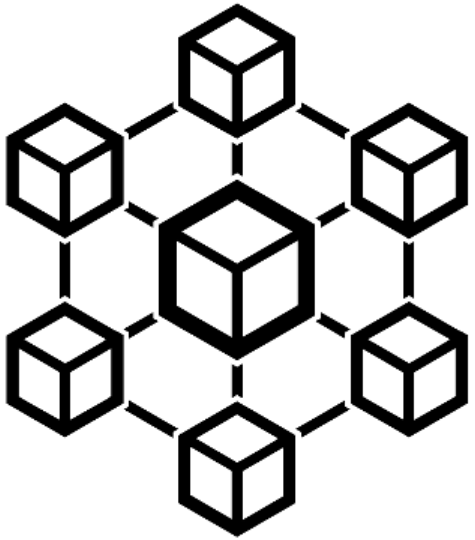


Micro Services et Micro Front-end





Micro Services et Micro Front-end



Express

BackEnd

REACT LOADABLE

ROUTE-CENTRIC CODE SPLITTING IS SHIT, COMPONENT-CENTRIC SPLITTING IS COOL AS SHIT

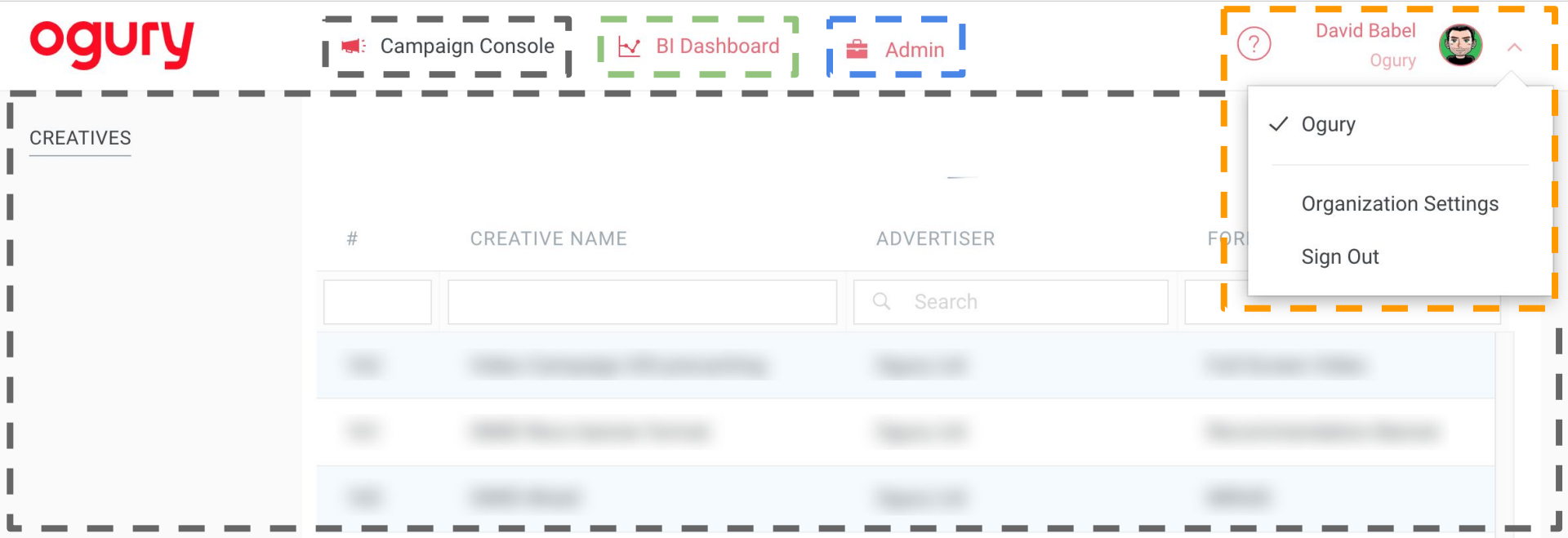
- Ken Wheeler... probably

FrontEnd

Source: https://micro-frontends.org/

REACT LOADABLE

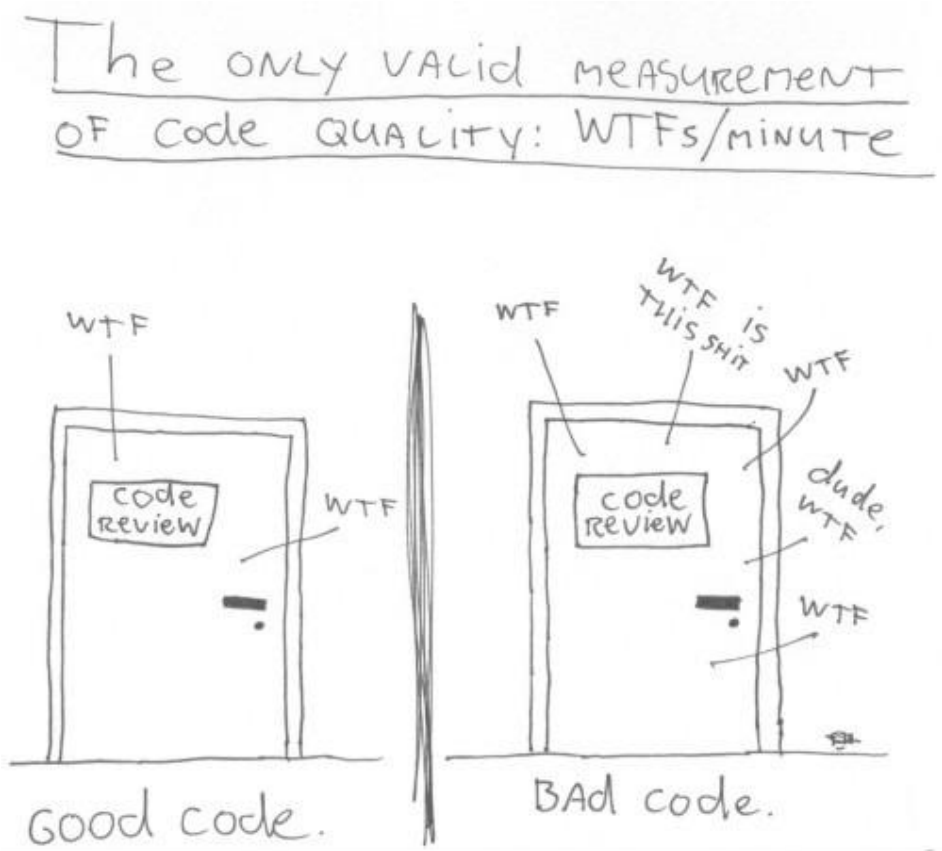
FrontEnd



Source: https://micro-frontends.org/

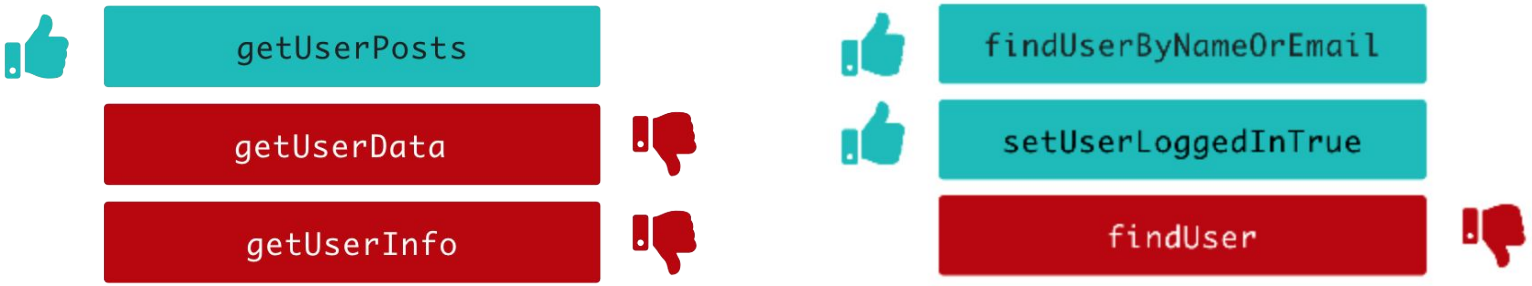
Qualité de code

Qualité de code



Source: Ryan McDermott : Clean Code Javascript (see)

Qualité de code



Source: Brandon Wozniewicz : JavaScript naming conventions: do’s and don’ts (see)

JS niveau-1.js

```
1
2  function execute(datas) {
3      const infos = [];
4      if (datas.g === 'f') {
5          infos.push('Mme.');
6      } else {
7          infos.push('M.');
8      }
9
10     infos.push(datas.f);
11     infos.push(datas.m || datas.n);
12
13     console.log(...infos);
14 }
```

JS niveau-2.js

```
1
2  function execute(datas) {
3      const infos = [];
4      if (datas.gender === 'f') {
5          infos.push('Mme.');
6      } else {
7          infos.push('M.');
8      }
9
10     infos.push(datas.forname);
11     infos.push(datas.mariedName || datas.name);
12
13     console.log(...infos);
14 }
```

JS niveau-3.js

```
1
2  function printUserName(user) {
3      const infosToPrint = [];
4
5      if (user.gender === 'f') {
6          infosToPrint.push('Mme.');
7      } else {
8          infosToPrint.push('M.');
9      }
10
11     infosToPrint.push(user.forname);
12     infosToPrint.push(user.mariedName || user.name);
13
14     console.log(infosToPrint.join(' '));
15 }
```

TS niveau-4.ts

```ts

interface UserIdentity {
  forname: string;
  name: string;
  mariedName?: string;
  gender: 'm' | 'f';
}

function printUserName(user: UserIdentity): void {
  const infosToPrint: string[] = [];

  if (user.gender === 'f') {
    infosToPrint.push('Mme.');
  } else {
    infosToPrint.push('M.');
  }

  infosToPrint.push(user.forname);
  infosToPrint.push(user.mariedName || user.name);

  console.log(infosToPrint.join(' '));
}
```

TS niveau-4.ts

```ts
interface UserIdentity {
  forname: string;
  name: string;
  mariedName?: string;
  gender: 'm' | 'f';
}

function printUserName(user: UserIdentity): void {
  const infosToPrint: string[] = [];

  if (user.gender === 'f') {
    infosToPrint.push('Mme.');
  } else {
    infosToPrint.push('M.');
  }

  infosToPrint.push(user.forname);
  infosToPrint.push(user.mariedName || user.name);

  console.log(infosToPrint.join(' '));
}
```

Execution :

```ts
printUserName({
  forname: 'David',
  name: 'Babel',
  gender: 'm'
});
```

```
~/dev/devoxx
➜ ts-node niveau-4.ts
M. David Babel
```

JS niveau-1.js

```js
function execute(datas) {
  const infos = [];
  if (datas.g === 'f') {
    infos.push('Mme.');
  } else {
    infos.push('M.');
  }

  infos.push(datas.f);
  infos.push(datas.m || datas.n);

  console.log(...infos);
}
```

TS niveau-4.ts

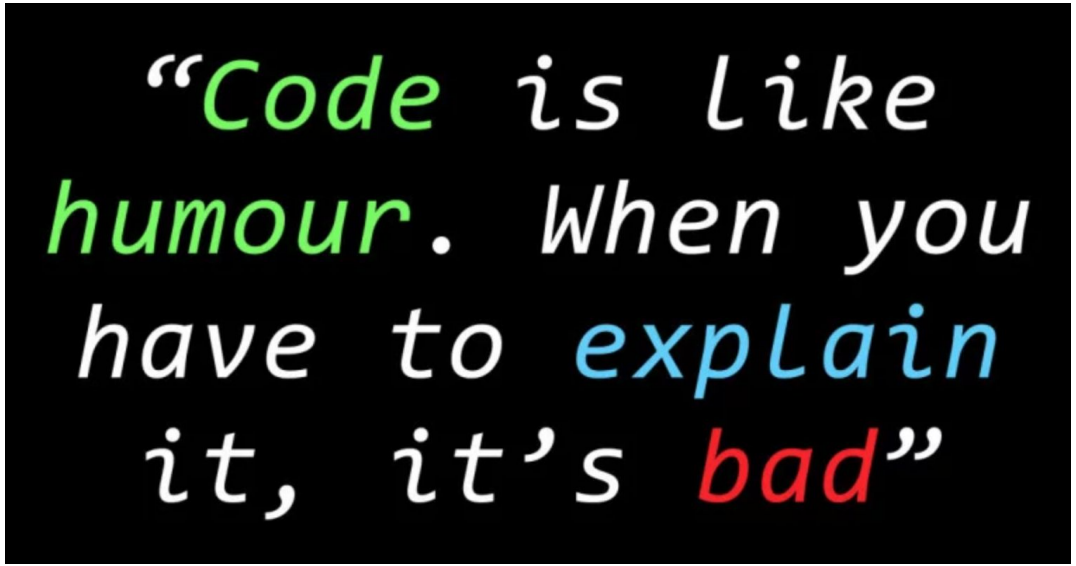
```ts
interface UserIdentity {
  forname: string;
  name: string;
  mariedName?: string;
  gender: 'm' | 'f';
}

function printUserName(user: UserIdentity): void {
  const infosToPrint: string[] = [];

  if (user.gender === 'f') {
    infosToPrint.push('Mme.');
  } else {
    infosToPrint.push('M.');
  }

  infosToPrint.push(user.forname);
  infosToPrint.push(user.mariedName || user.name);

  console.log(infosToPrint.join(' '));
}
```



Cory House @housecor
React evangelist

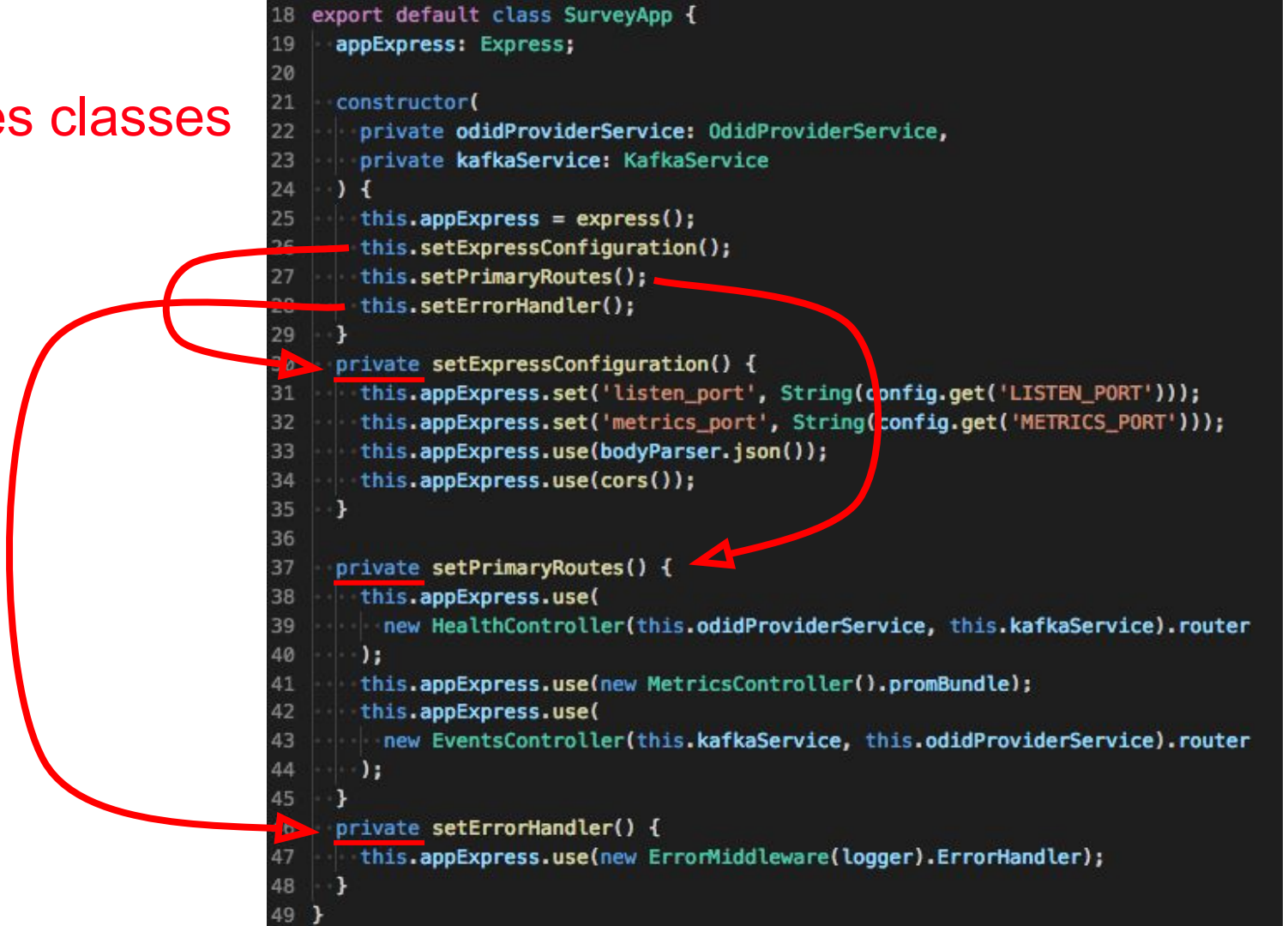
Comprendre JavaScript

Comprendre JavaScript

C'est un langage orienté objet à prototype, c'est-à-dire que les bases du langage et ses principales interfaces sont fournies par des objets qui ne sont pas des instances de classes

Source: https://fr.wikipedia.org/wiki/JavaScript

Les classes



```
18 export default class SurveyApp {
19   appExpress: Express;
20
21   constructor(
22     private odidProviderService: OdidProviderService,
23     private kafkaService: KafkaService
24   ) {
25     this.appExpress = express();
26     this.setExpressConfiguration();
27     this.setPrimaryRoutes();
28     this.setErrorHandler();
29   }
30   private setExpressConfiguration() {
31     this.appExpress.set('listen_port', String(config.get('LISTEN_PORT')));
32     this.appExpress.set('metrics_port', String(config.get('METRICS_PORT')));
33     this.appExpress.use(bodyParser.json());
34     this.appExpress.use(cors());
35   }
36
37   private setPrimaryRoutes() {
38     this.appExpress.use(
39       new HealthController(this.odidProviderService, this.kafkaService).router
40     );
41     this.appExpress.use(new MetricsController().promBundle);
42     this.appExpress.use(
43       new EventsController(this.kafkaService, this.odidProviderService).router
44     );
45   }
46   private setErrorHandler() {
47     this.appExpress.use(new ErrorMiddleware(logger).ErrorHandler);
48   }
49 }
```

Les classes

Classes are ...

Sophie Alpert @sophiebits
Facebook React team

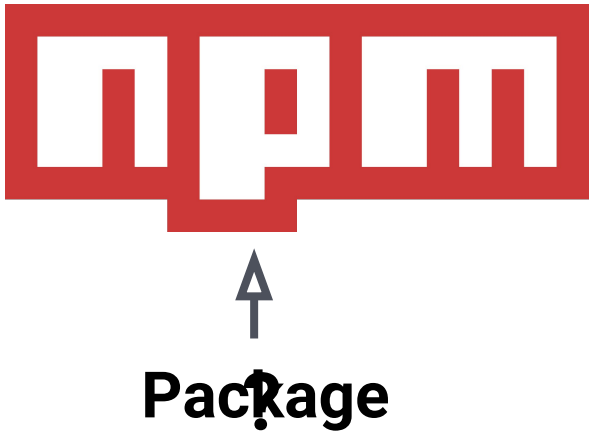
Source: Sophie Alpert: React Today and Tomorrow and 90% Cleaner React With Hooks (see at 10:00)

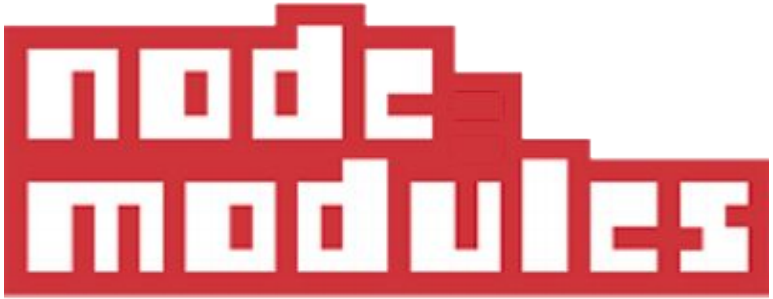
Les Modules

```
20 export const appExpress = express();
21
22 appExpress.set('listen_port', String(config.get('LISTEN_PORT')));
23 appExpress.set('metrics_port', String(config.get('METRICS_PORT')));
24 appExpress.use(bodyParser.json());
25 appExpress.use(cors());
26
27 appExpress.use(HealthController);
28 appExpress.use(MetricsController);
29 appExpress.use(EventsController);
30 appExpress.use(ErrorMiddleware);
```

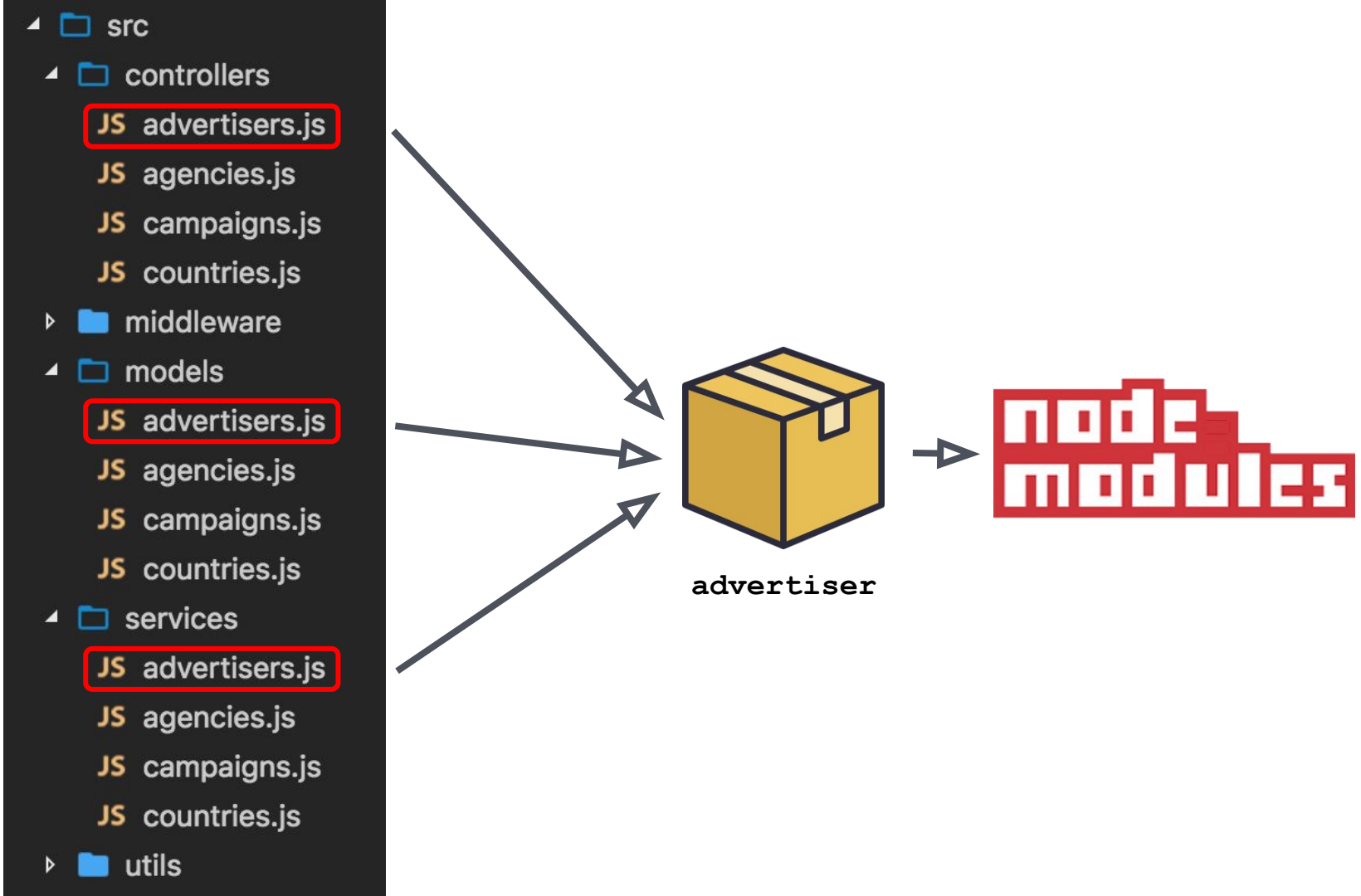
La même chose en 20 lignes de moins

Il y avait quelques indices ...





Source: Naren Yellavula : Everything I know about writing modular JavaScript applications (see)


src
controllers
advertisers.js
agencies.js
campaigns.js
countries.js
middleware
models
advertisers.js
agencies.js
campaigns.js
countries.js
services
advertisers.js
agencies.js
campaigns.js
countries.js
utils
advertiser
node modules


src
controllers
advertisers.js
agencies.js
campaigns.js
countries.js
middleware
models
advertisers.js
agencies.js
campaigns.js
countries.js
services
advertisers.js
agencies.js
campaigns.js
countries.js
utils
src
config
modules
advertisers
i18n
tests
advertisers-controller.ts
advertisers-model.ts
advertisers-service.ts
index.ts
agencies
campaigns
countries
index.ts
1 export * from './advertisers-controller';
2 export * from './advertisers-model';
3 export * from './advertisers-service';
4

Les modules

TS index.ts

```ts
export * from './advertisers-controller';
export * from './advertisers-model';
export * from './advertisers-service';
```

On peut importer depuis le chemin de notre fichier

```ts
import { controller, service, model } from './advertisers';
```

Ou créer des alias :

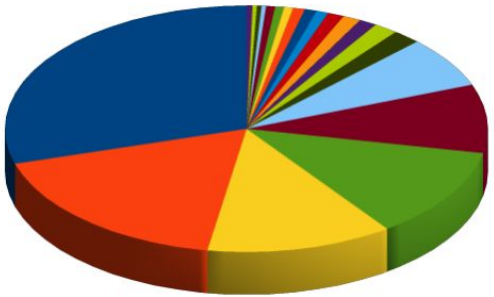
```ts
import { controller, service, model } from 'modules/advertisers';
```

Typage en JavaScript

TypeScript : un langage

Visual Studio Code

TypeScript : popularité



- Java (29,6%)
- JavaScript (17,8%)
- C# (12,5%)
- PHP (11,9%)
- Python (9,0%)
- C++ (6,7%)
- Perl (1,9%)
- TypeScript (1,6%)
- VBA (1,2%)
- Scala (1,2%)
- Delphi (1,1%)
- Ruby (0,8%)
- Matlab (0,8%)
- Swift (0,8%)
- Cobol (0,6%)
- VB.NET (0,6%)
- Windev (0,6%)
- Objective C (0,3%)
- C (0,3%)
- Go (0,3%)
- Autres

JS

JavaScript

17.8%

* dont potentiellements quelques annonces **TypeScript** cachées

TS

TypeScript

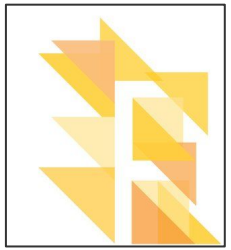
1.6%

Source: developpez.com : les langages les plus demandés et les mieux payés (see)

TypeScript : la promesse

- Types (vous avez atteint le niveau 4 !)
- Contrat entre front et back
- Transpileur
- Auto-linter efficace
- Tueur de la “JavaScript fatigue”

BABEL



npm

package.json

node_modules

500Mo

200Mo

TS

Tooling JavaScript efficace

Style de code et syntaxe

Prettier

+ **lint-staged** 🚫💩
+ **husky** (git hooks)

Exemple de formatage à la sauvegarde :

```
1
2 function aa ( test ) {console
3   .log({ objectKey1:
4     "value 1", objectKey2: 'value 2',
5 objectKey3: `value 3`
6   } )}
7
```

Gestion des dépendances

Vous voulez vous autoriser à forker un projet pour en commencer un nouveau ?

Execution de la commande

```
➜  yarn outdated
yarn outdated v1.13.0
info Color legend :
 "<red>"    : Major Update backward-incompatible updates
 "<yellow>" : Minor Update backward-compatible features
 "<green>"  : Patch Update backward-compatible bug fixes
Package                      Current  Wanted Latest Package Type
puppeteer                    1.13.0   1.14.0 1.14.0 devDependencies
react-hooks-testing-library 0.3.8    0.3.8  0.4.0  devDependencies
ts-jest                      24.0.0   24.0.1 24.0.1 devDependencies
typescript                   3.3.4000 3.4.1  3.4.1  devDependencies
```

Gestion des dépendances

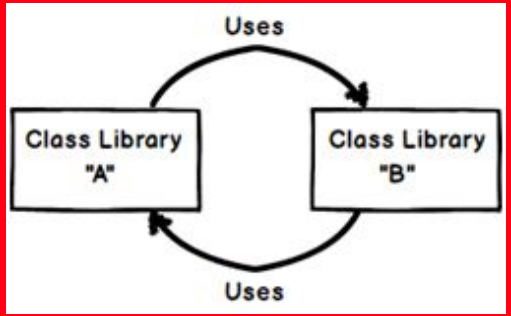
Vous voulez vous débarrasser de vos dépendances inutiles ?

depcheck

Execution de la commande

```
 ➜ yarn run depcheck
yarn run v1.13.0
$ depcheck . --ignores="@types/*,tslib,csstype"
Unused devDependencies
* jest-environment-puppeteer
* puppeteer
```

Dépendances circulaires



Protection contre les dépendances circulaires :

madge

Execution de la commande

```
➜  yarn run circular-dependencies
yarn run v1.13.0
$ madge --circular --exclude cache ./common ./formats --extensions js,jsx,ts,tsx
✖ Found 1 circular dependency!

1) common/js/config.ts > common/js/environment-utils.ts > common/js/utils.ts
```

Source: Dan Kelch : Eliminate Circular Dependencies from Your JavaScript Project (see)

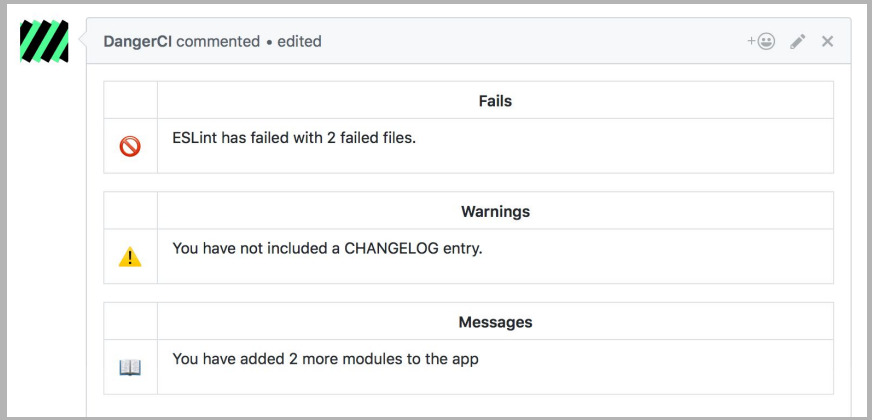
CI et code review

Automatisation de code review :

DANGER.js

⚠️ Stop saying "you forgot to ..." in code review

Aperçu dans github



2018 JavaScript Rising Stars

Une veille technique facile des librairies ?

https://**risingstars.js.org**

Conclusion

That’s all folks

Développeurs:

- Choisissez les bons outils pour tuer la dette technique
- Faites confiance à votre tooling

Startuppers :

- Faites confiance à vos développeurs

Merci de votre attention

Scan me

ogury

Carles Sistare

@CarlesSistare

David Babel

@DavidLearnJS