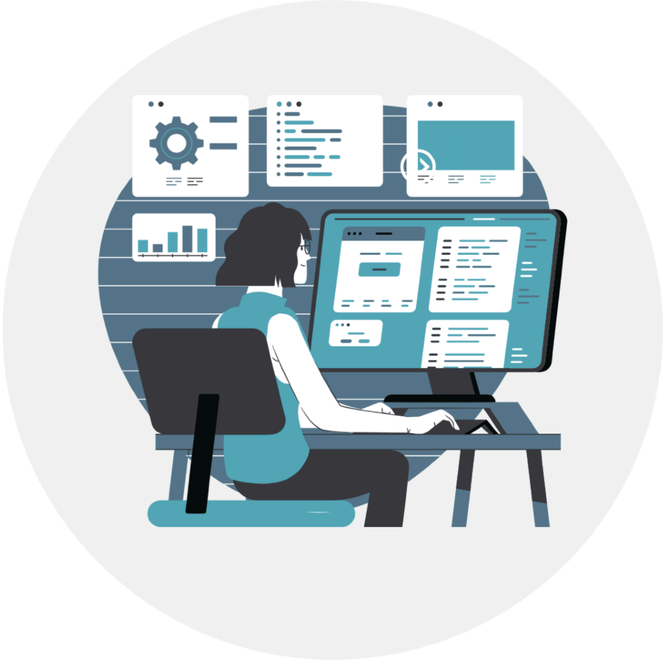
ITERACODE

Studio d’innovation digitale, en mode Agile



Comment on a testé nos tests qui testent ?

Comment on a testé la Nightcode ?

SLIDES

https://github.com/Charlynux/presentations

QUI SUIS-JE ?

Charles Fourdrignier

Lead Dev chez Iteracode

ITERACODE ?

Studio d'innovation digitale
Depuis Août 2015

Développement sur mesure

Formations - http://academie.iteracode.fr/

AVERTISSEMENT

Nous présentons ici ce qu'Iteracode fait pour préparer la Nightcode.

Il ne s'agit pas de ce que nous attendons de la part des étudiants.

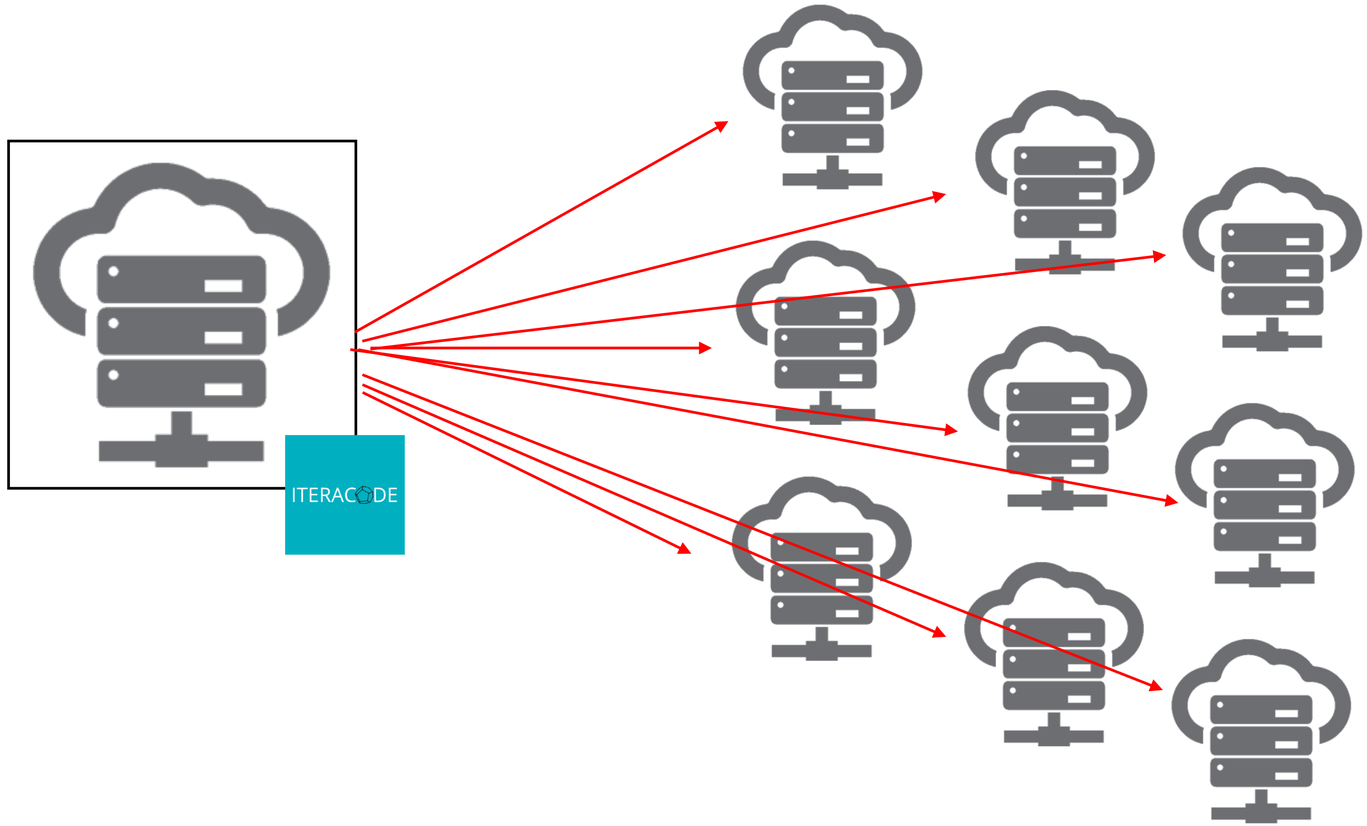
NIGHTCODE ?

UNE NUIT DE CODE

18h -> 6h

À QUOI ÇA RESSEMBLE ?

EN PRATIQUE ?

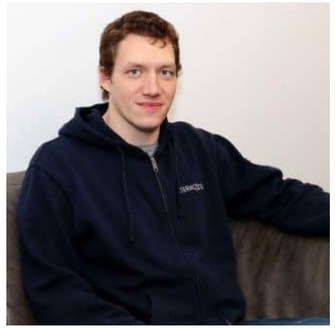


https://blog.xebia.fr/2013/07/15/techevent-code-elevator-un-concours-de-programmation/

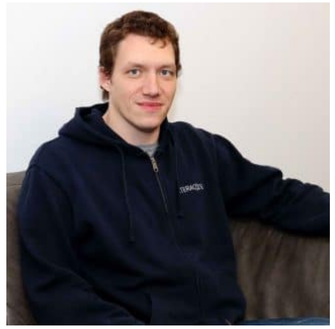
INSPIRATIONS

HackerRank

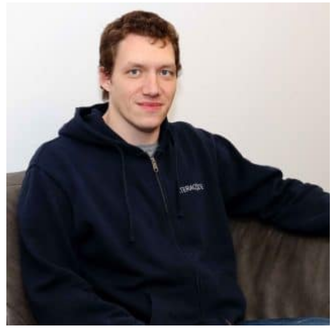
L’ÉQUIPE



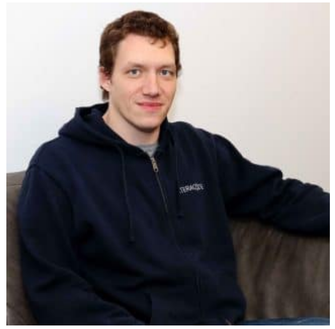
L'ÉQUIPE



L'ÉQUIPE



L'ÉQUIPE



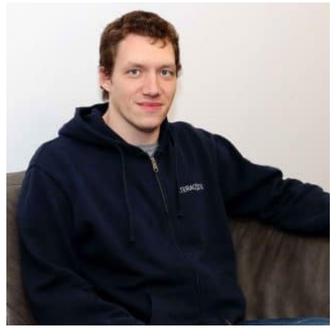
🔧

🍕

☁️

✍️

L'ÉQUIPE



🔧

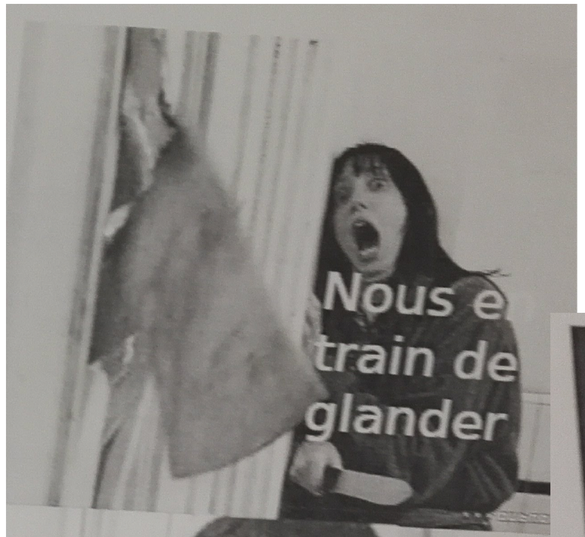
🍕

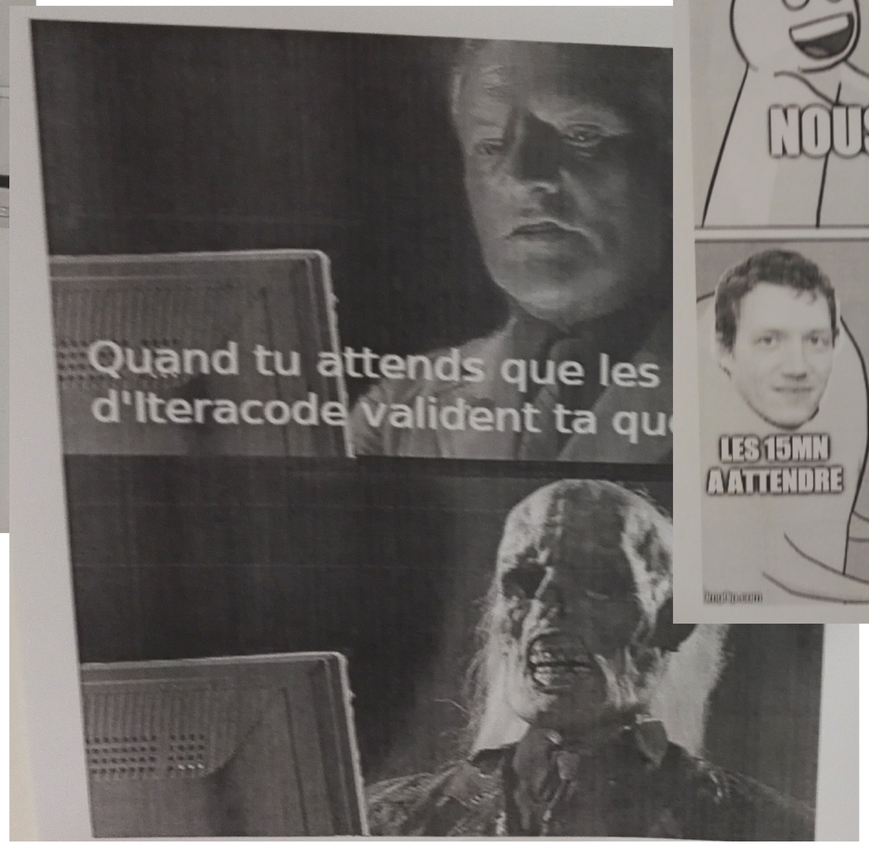
☁️

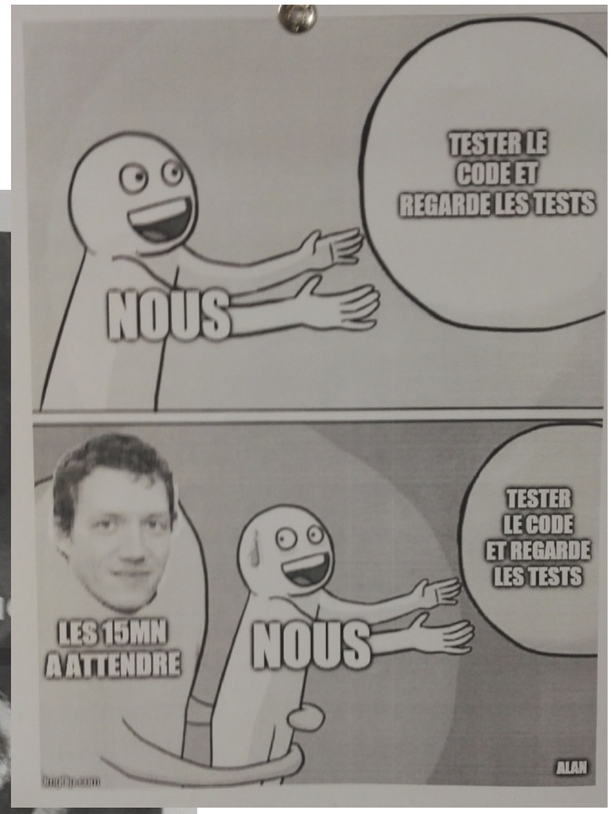
✍️

🎨

BONNE AMBIANCE







TESTER ?

COMMENT ON TESTE ?

System under test

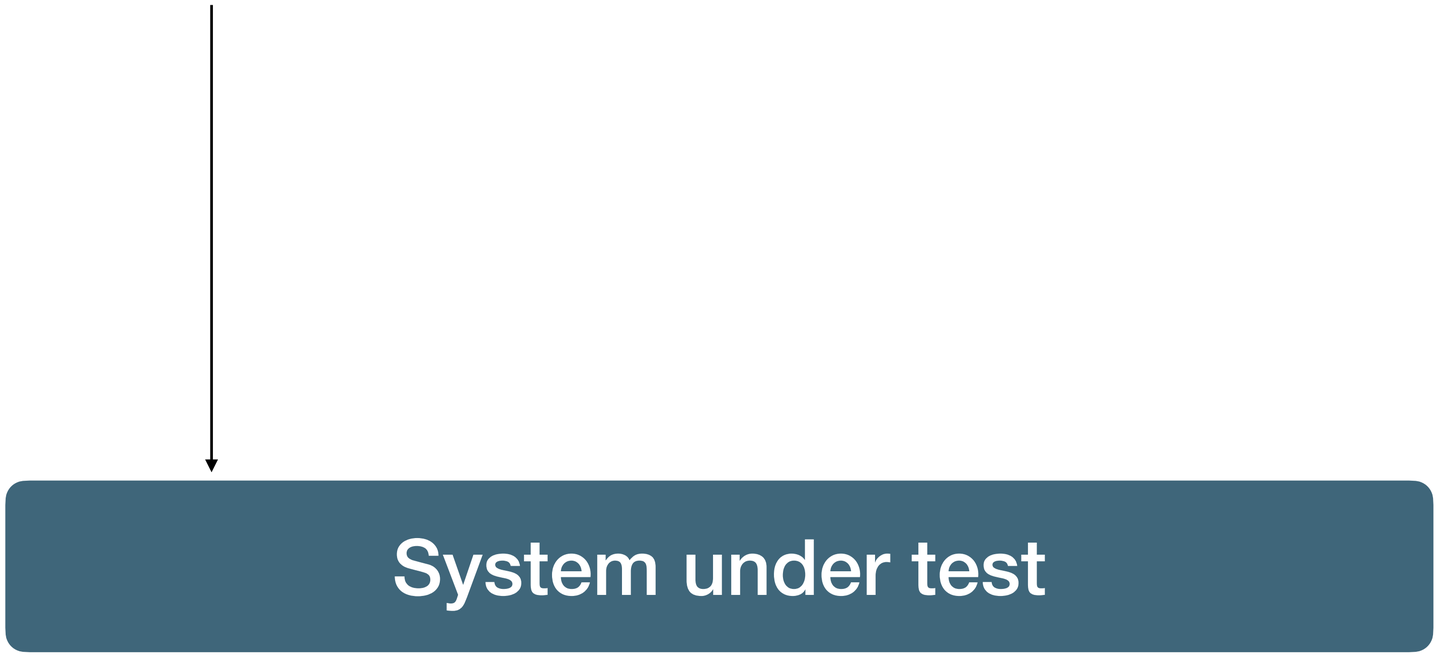
COMMENT ON TESTE ?

[Inputs Expectations]

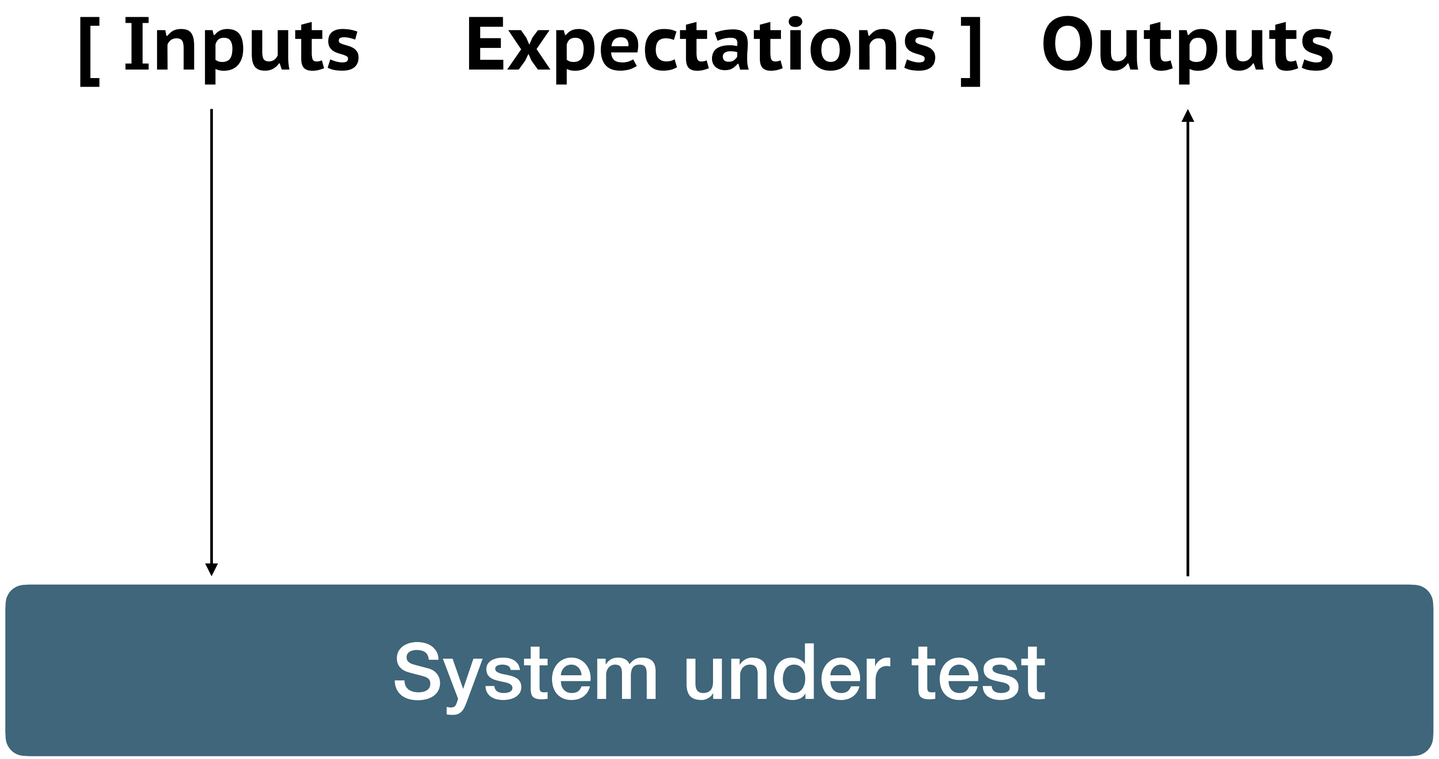
System under test

COMMENT ON TESTE ?

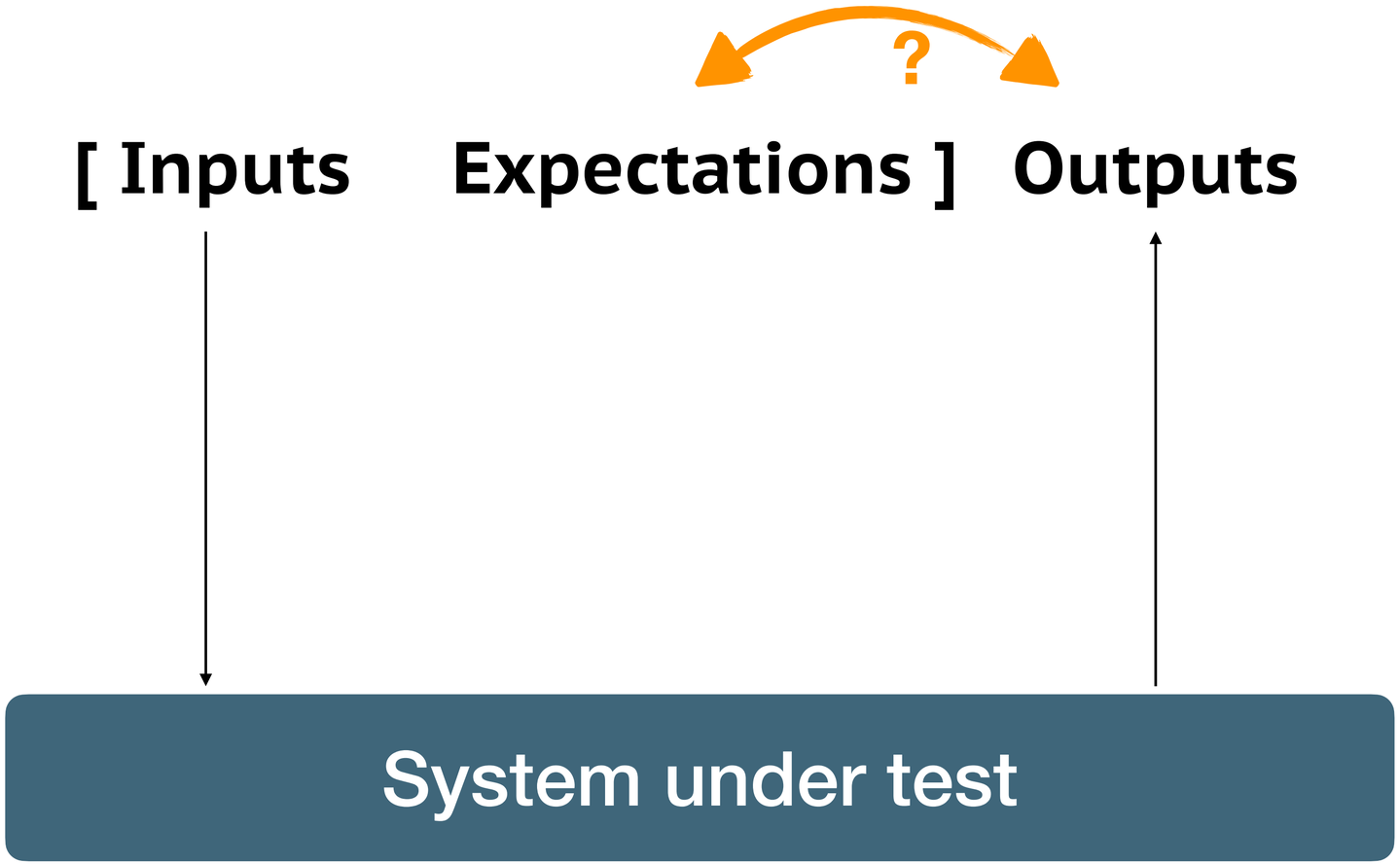
[Inputs Expectations]



COMMENT ON TESTE ?



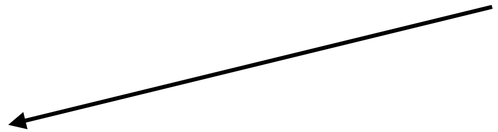
COMMENT ON TESTE ?



COMMENT ON TESTE ?

```
{
  method: "POST",
  url: "/api/sum",
  body: "{\"payload\":[1, 2, 3, 4, 5]}"
}
```

Serveur

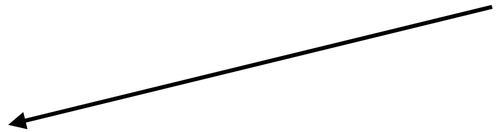


```
{
  status: 200,
  body: "{\"value\" : 15}"
}
```

COMMENT ON TESTE ?

```
{
  method: "POST",
  url: "/api/sum",
  body: "{\"payload\":[1, 2, 3, 4, 5]}"
}
```

Serveur

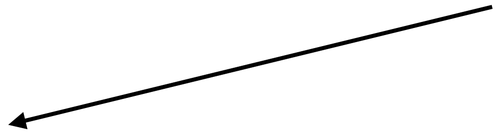


```
{
  status: 200,
  body: "{\"value\" : 15}"
}
```

COMMENT ON TESTE ?

```
{
  method: "POST",
  url: "/api/sum",
  body: "{\"payload\":[1, 2, 3, 4, 5]}"
}
```

Serveur

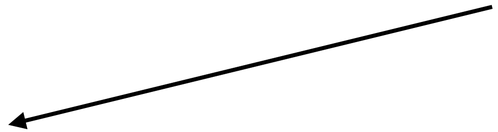


```
{
  status: 200,
  body: "{\"value\" : 15}"
}
```

COMMENT ON TESTE ?

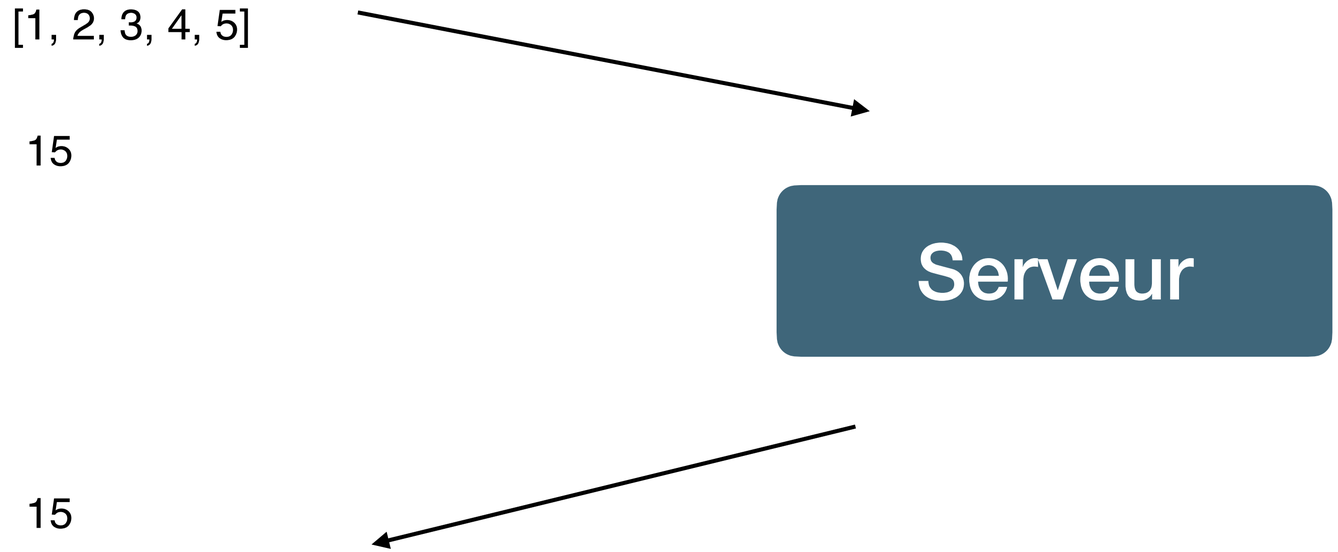
```
{
  method: "POST",
  url: "/api/sum",
  body: "{\"payload\":[1, 2, 3, 4, 5]}"
}
```

Serveur

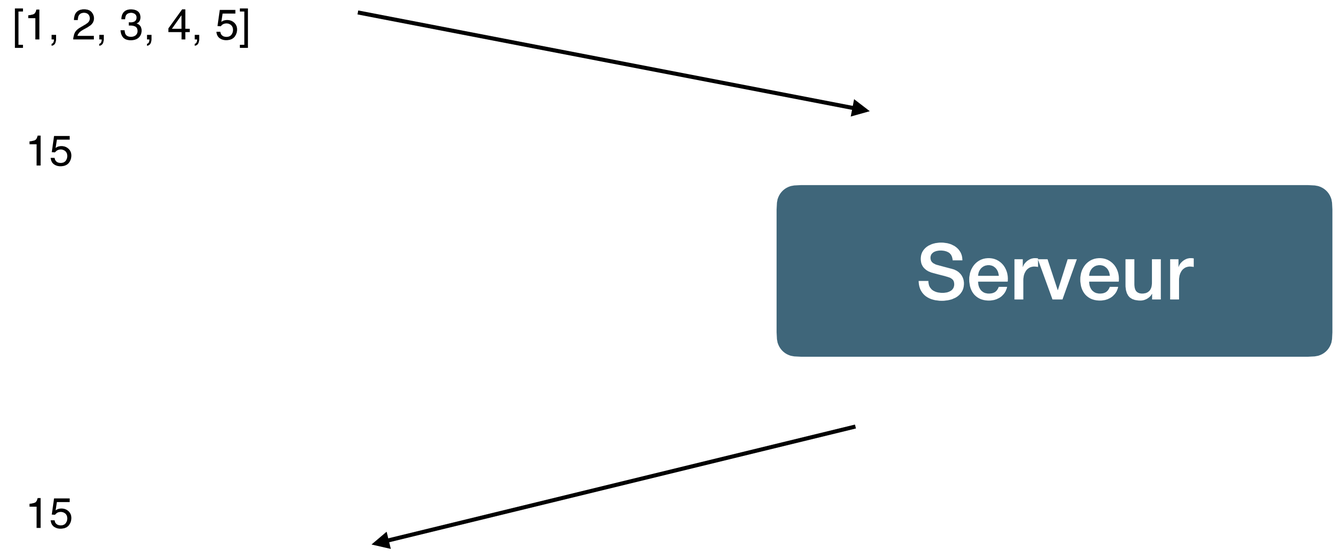


```
{
  status: 200,
  body: "{\"value\" : 15}"
}
```

COMMENT ON TESTE ?



COMMENT ON TESTE ?



Example Based Testing

Property Based Testing

Générateur de données aléatoires

+

Propriétés

https://fsharpforfunandprofit.com/pbt/

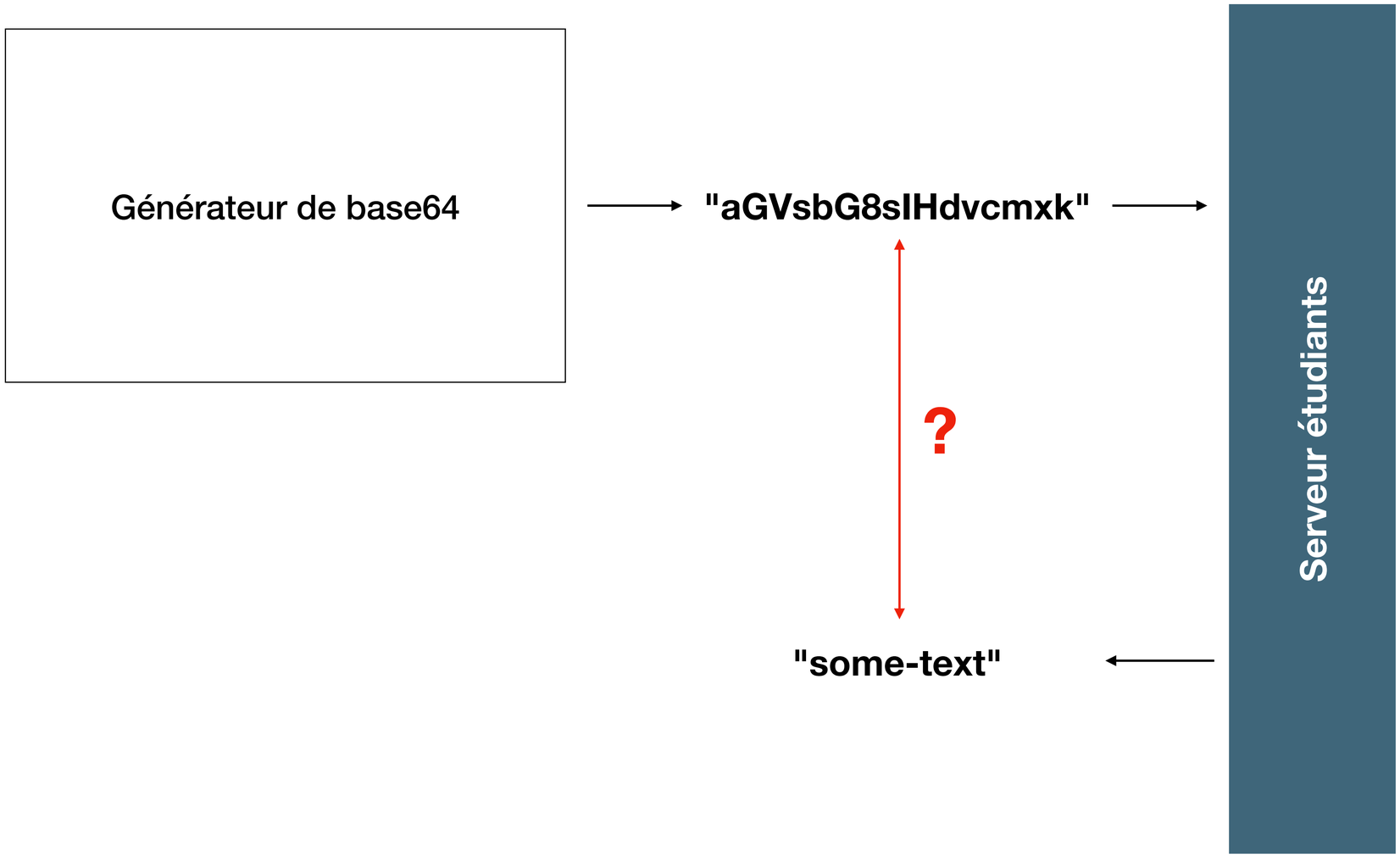
PROMENONS-NOUS DANS LES CAS

/api/secret-base64

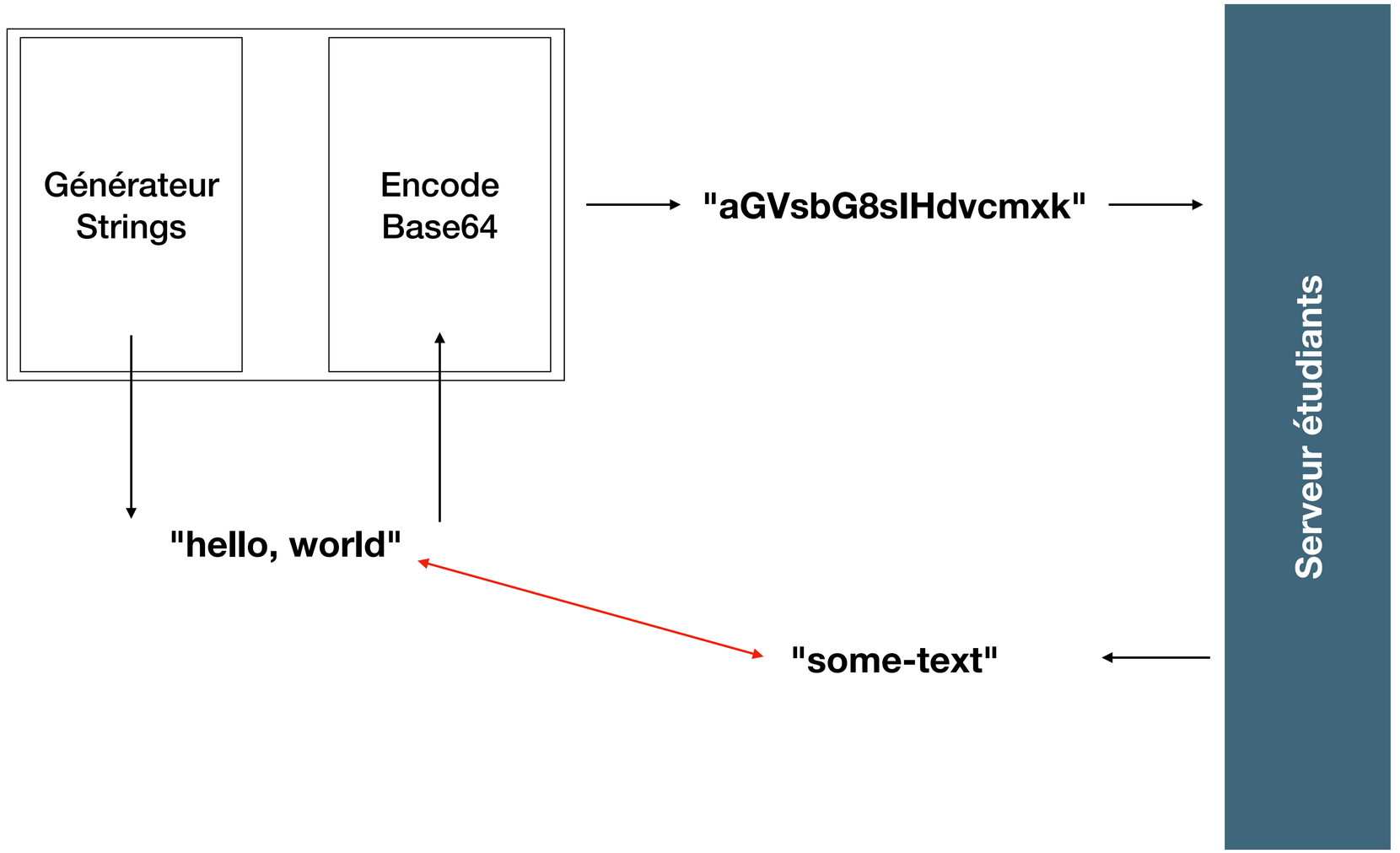
```
POST /api/secret-base64
Content-Type: application/json

{"to-translate":"aGVsbG8sIHdvcmxk"}
```

```
Status : 200
{"text":"hello, world"}
```

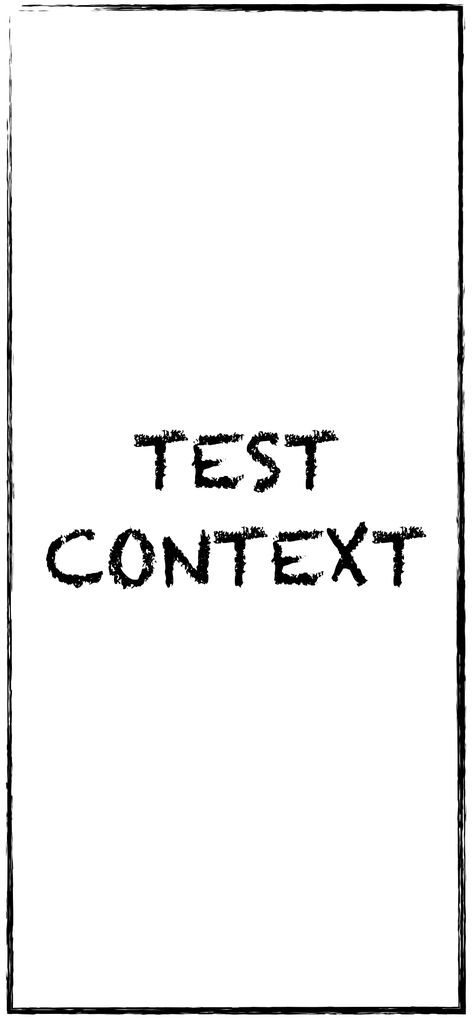
/api/secret-base64



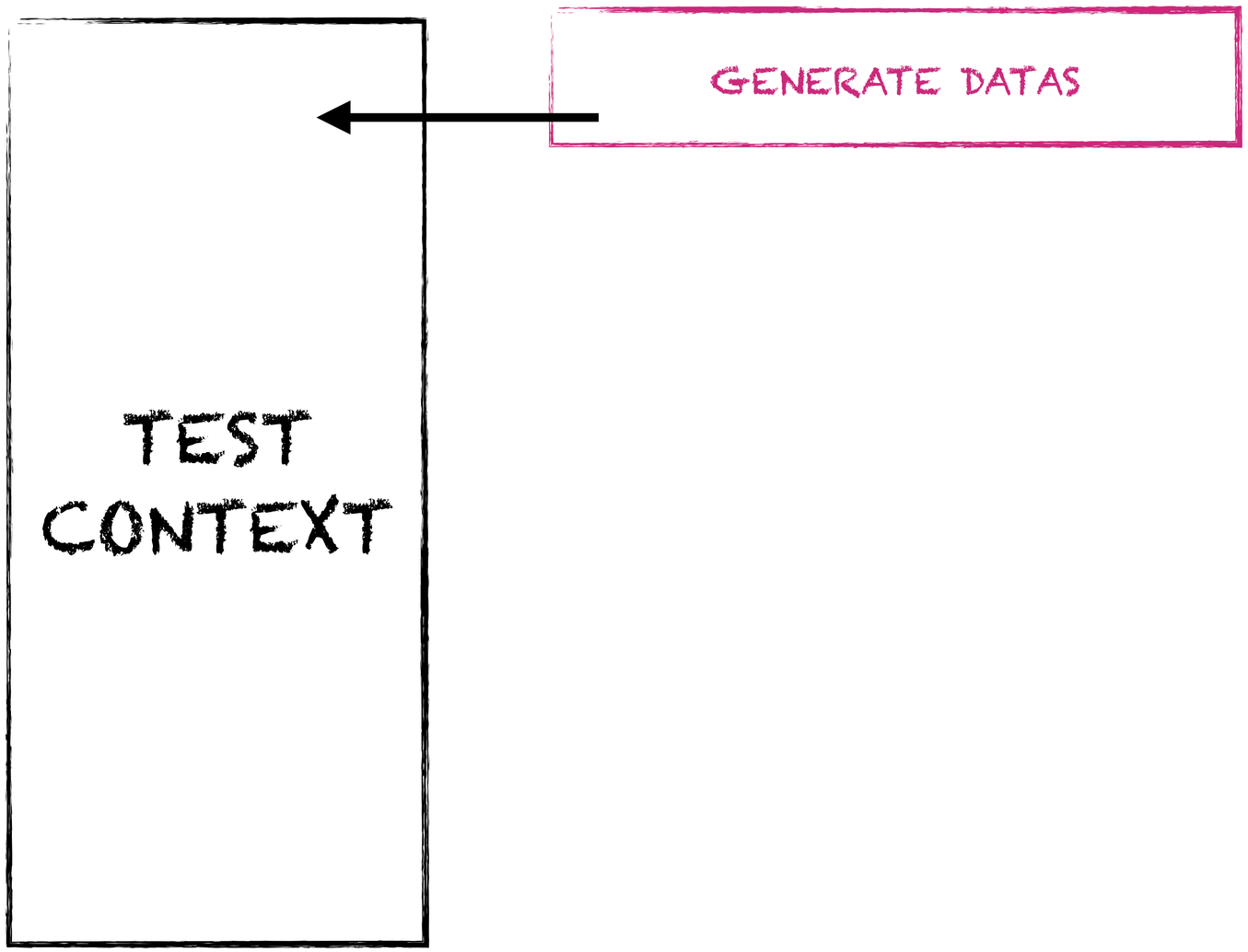
/api/secret-base64



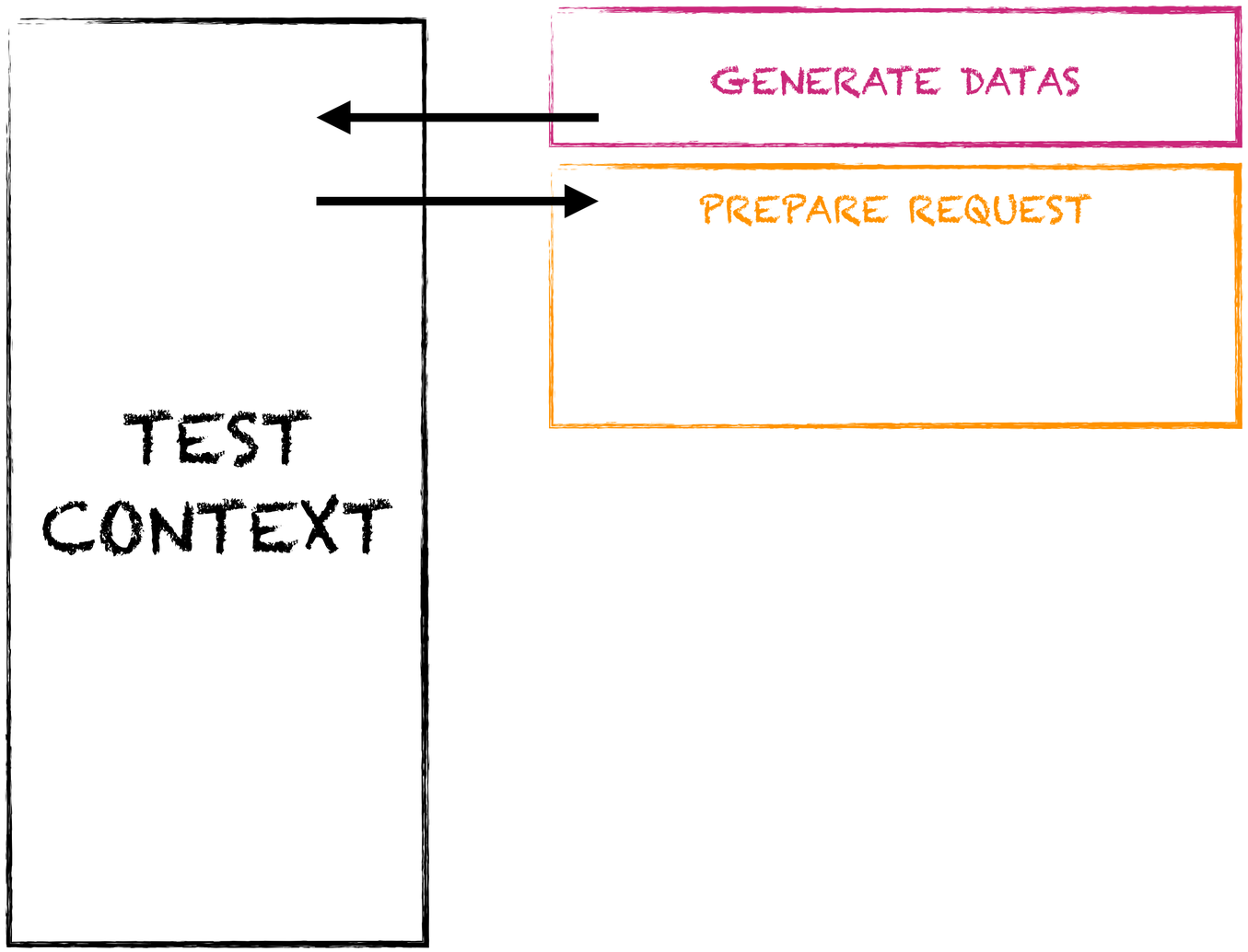
DÉROULEMENT D'UN TEST



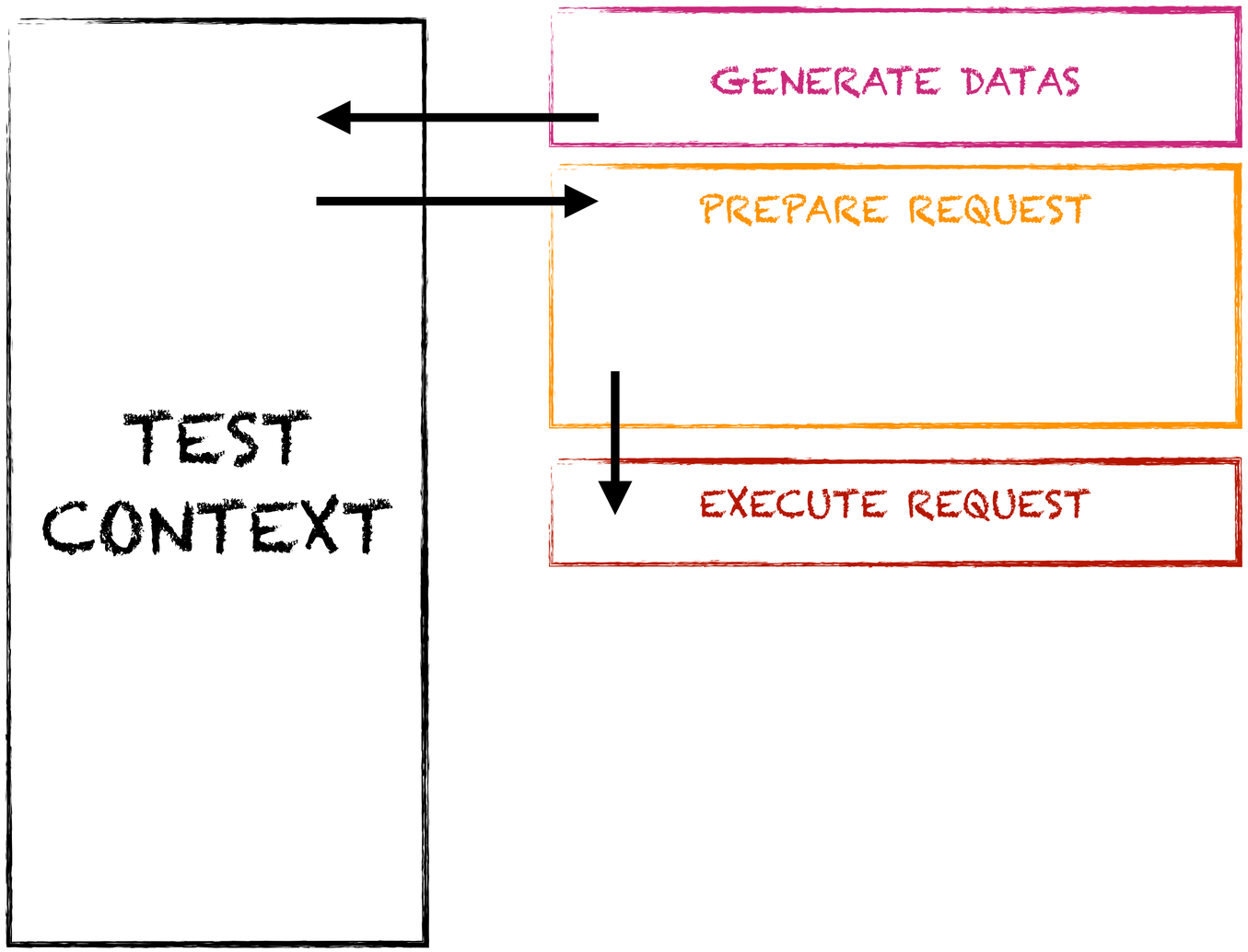
DÉROULEMENT D'UN TEST



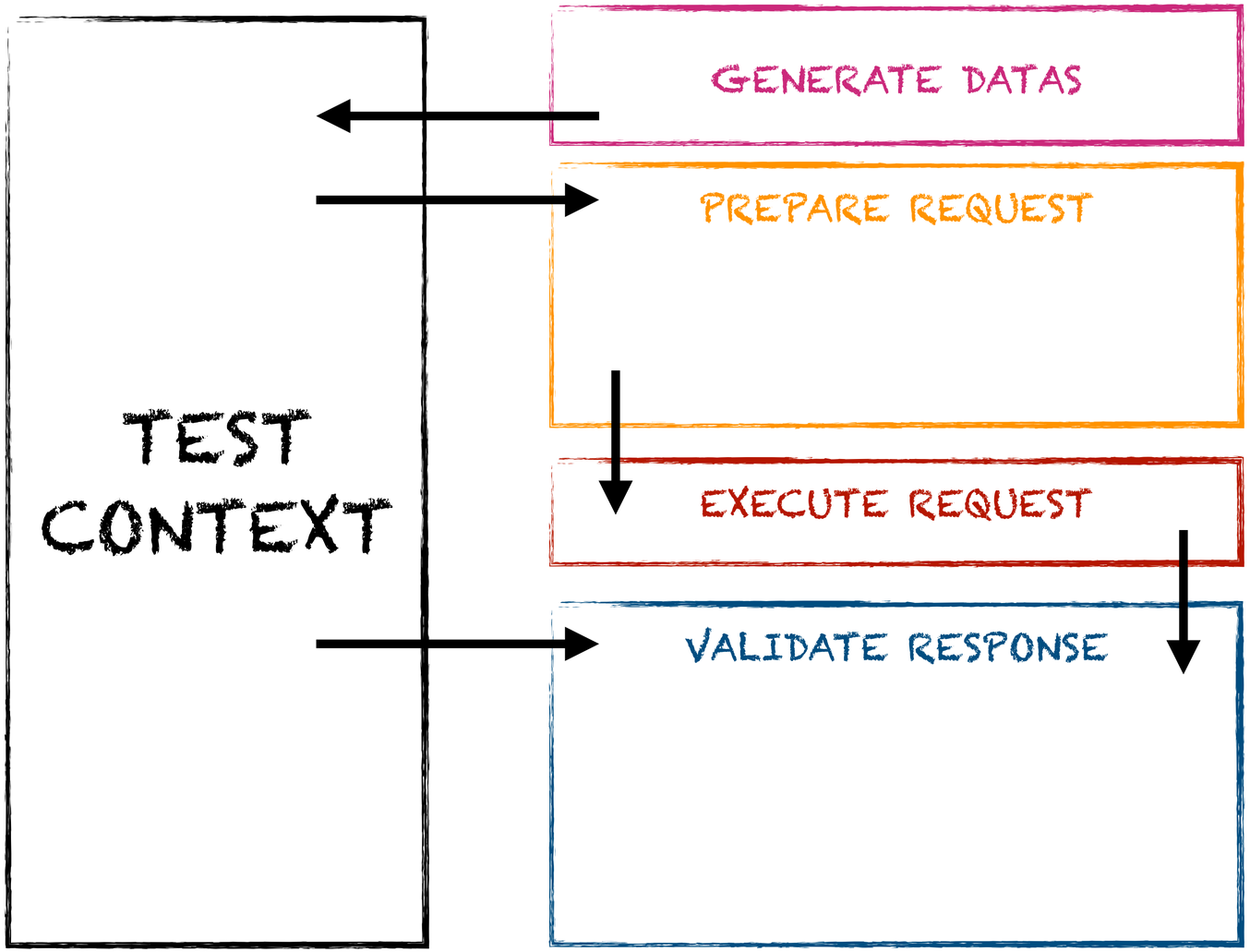
DÉROULEMENT D'UN TEST



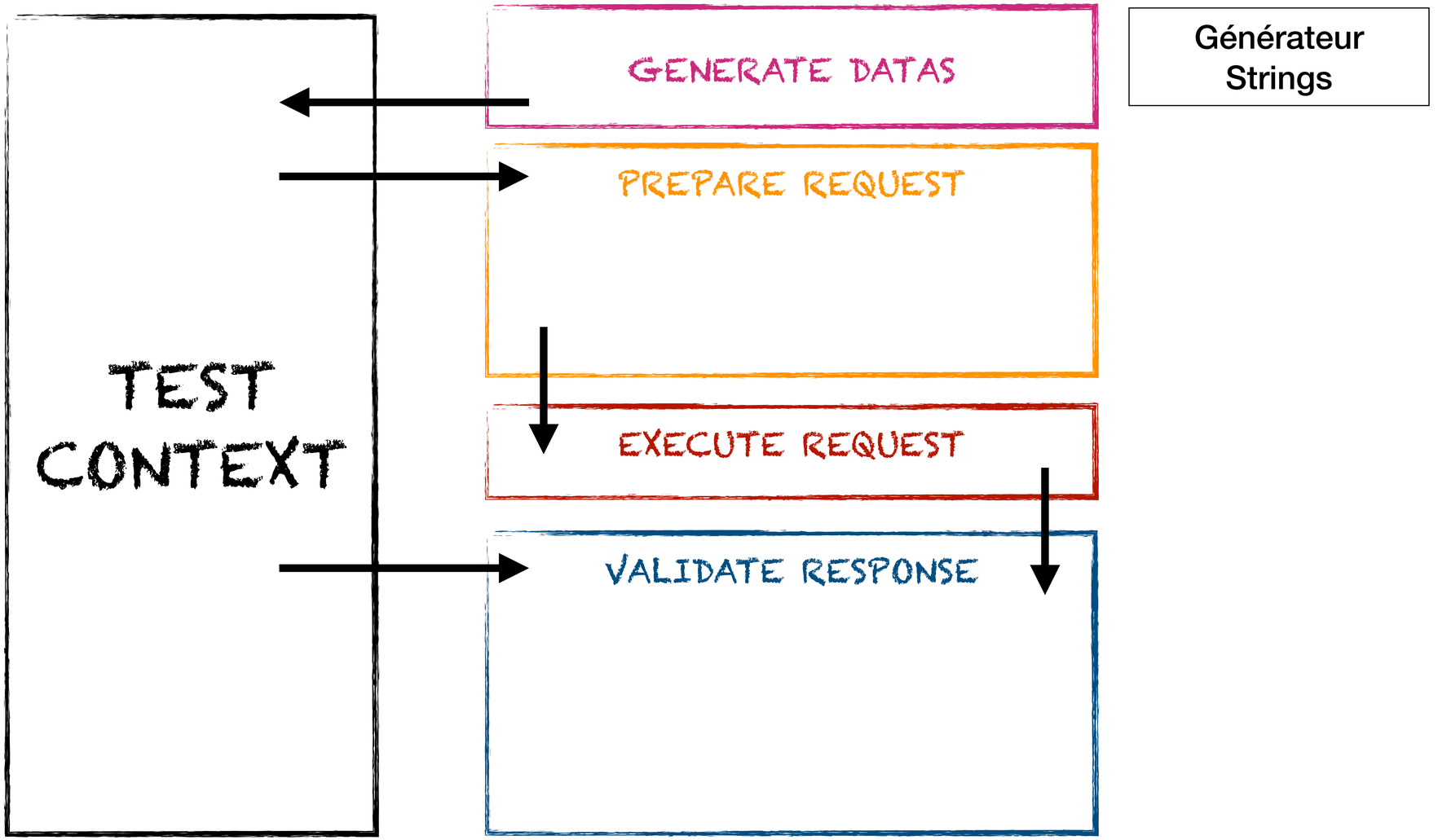
DÉROULEMENT D'UN TEST



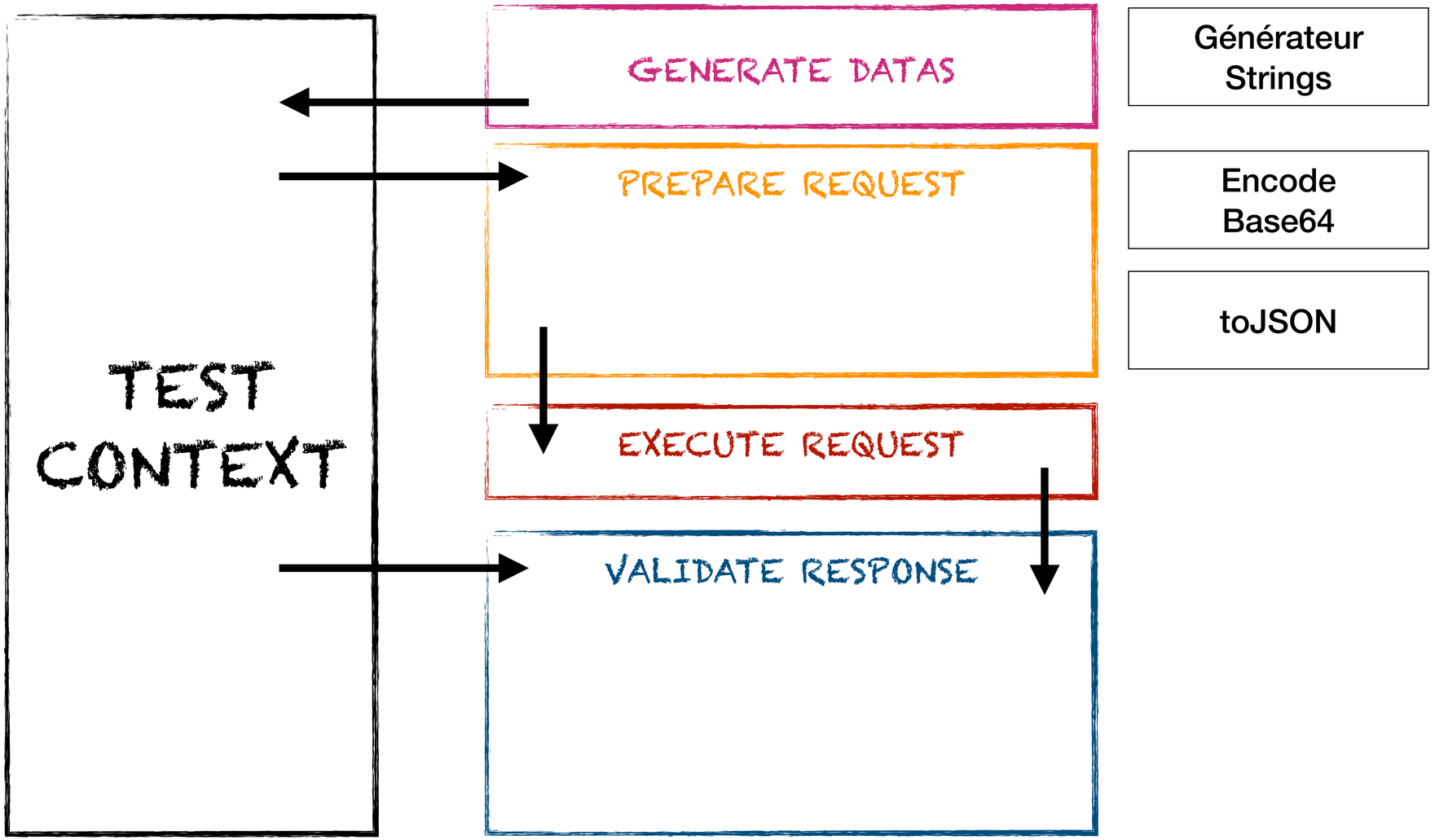
DÉROULEMENT D'UN TEST



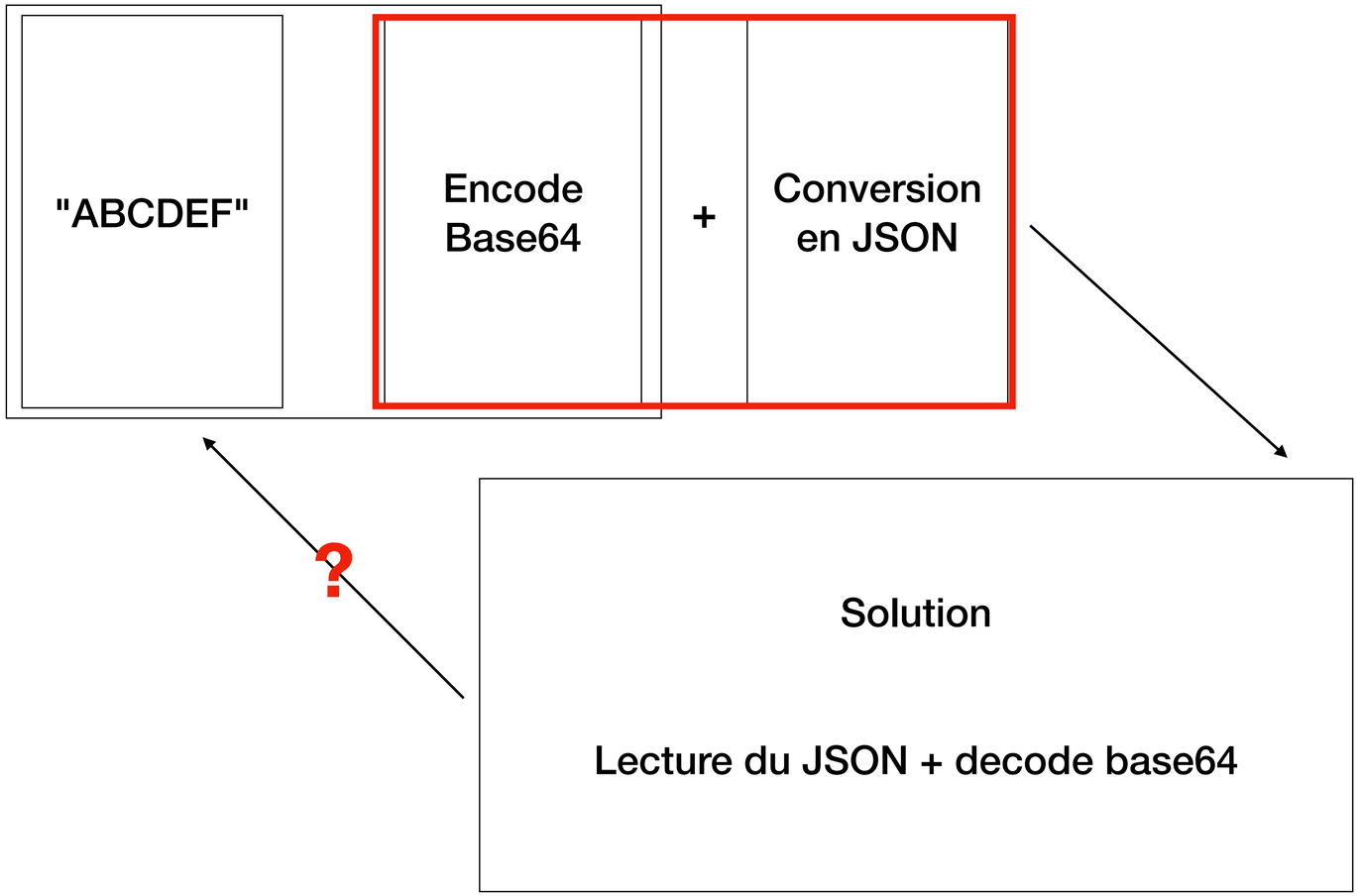
DÉROULEMENT D'UN TEST



DÉROULEMENT D'UN TEST



/api/secret-base64

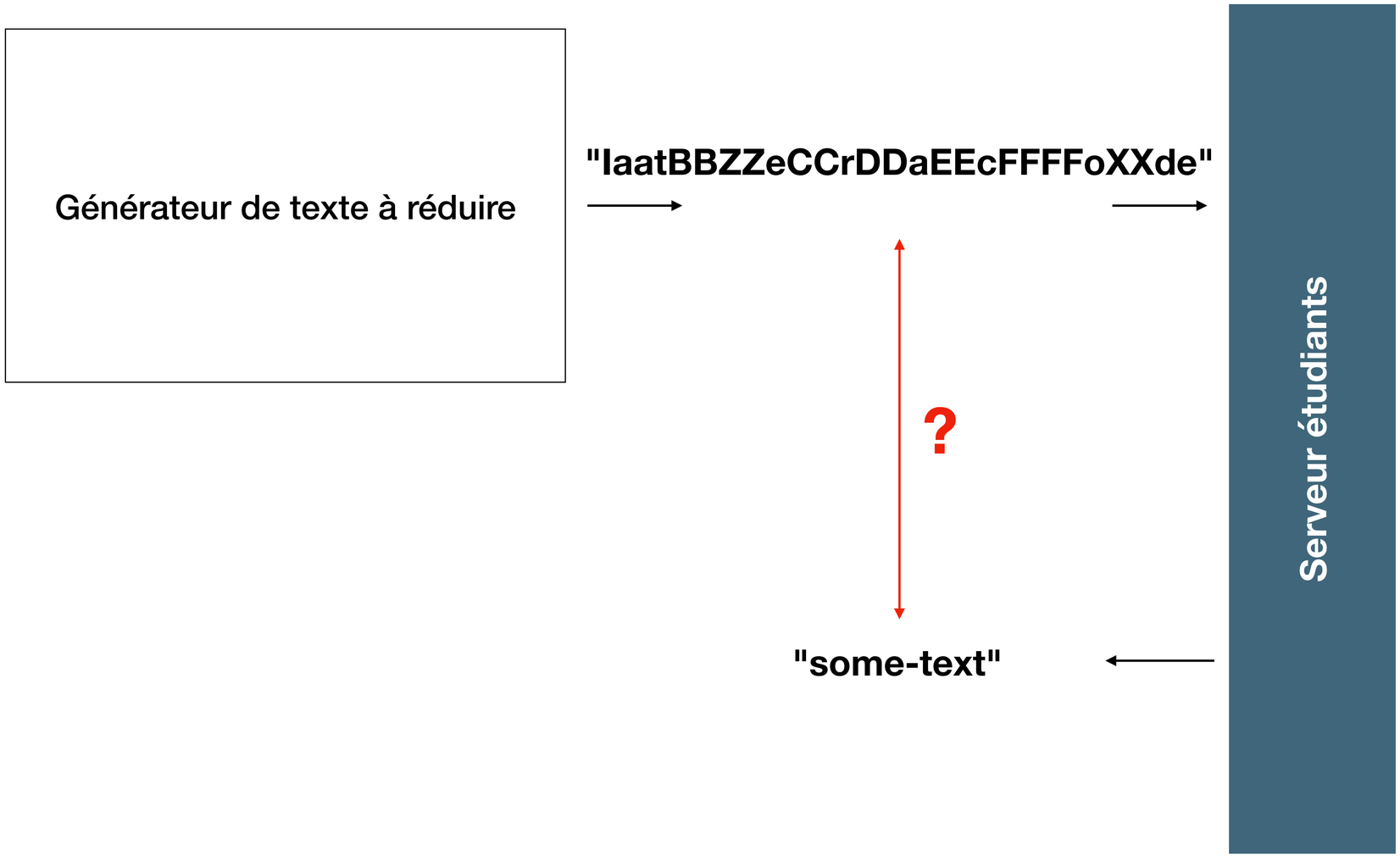


/api/secret-reduce »

```
POST /api/secret-reduced
Content-Type: application/json

{"to-translate"
:"IaatBBZZeCCrDDaEEcFFFFoXXde"}
```

```
Status : 200
{"text":"Iteracode"}
```

/api/secret-reduce »



/api/secret-reduce »

Générateur de texte à réduire

/api/secret-reduce »

Générateur de texte à réduire

Liste de caractères distincts

↓

Conversion en chaîne

↓

Entre chaque caractère, ajoute 0 à 4 PAIRES d'un même caractère

/api/secret-reduce »

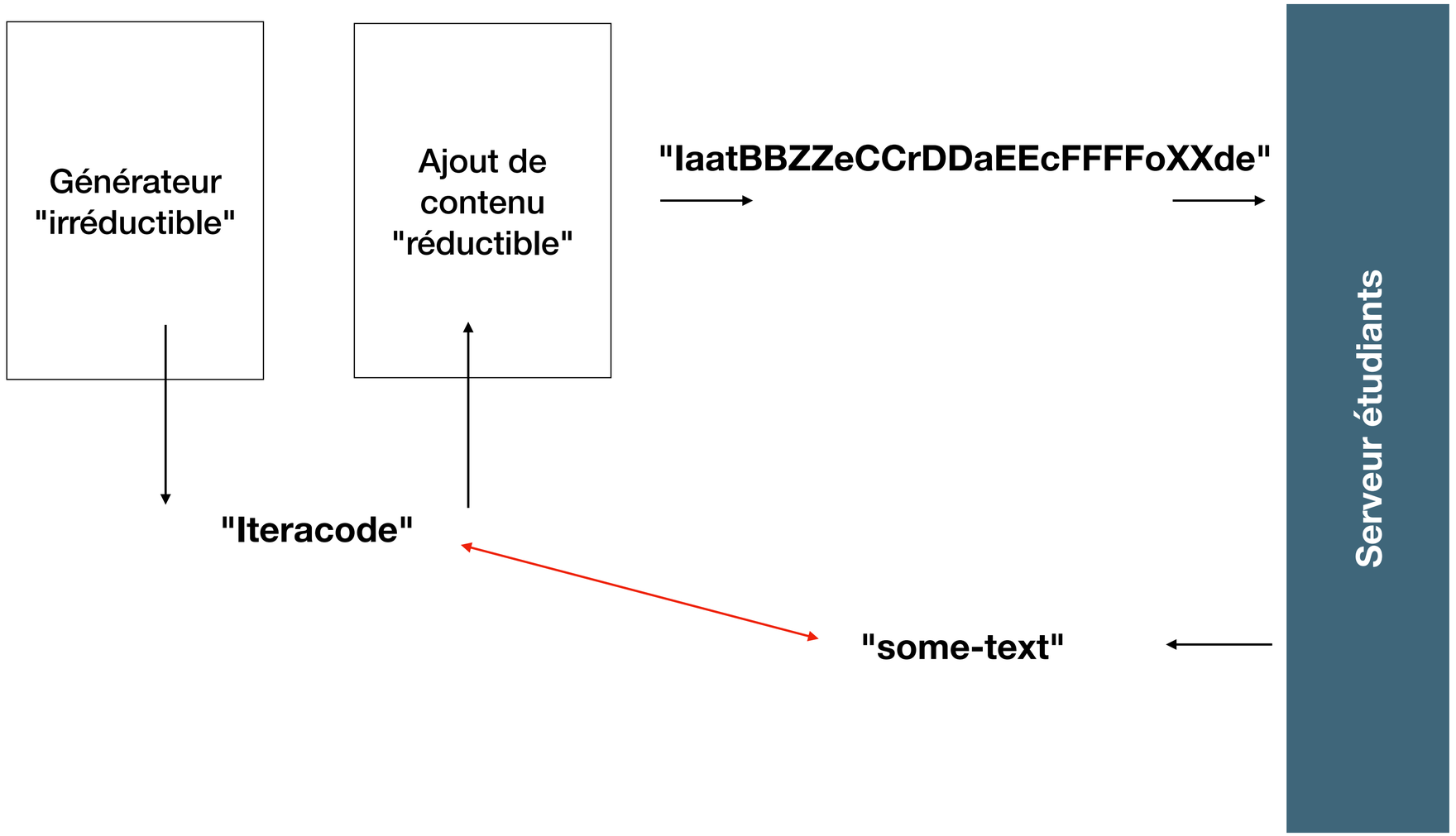
Générateur de texte à réduire

Liste de caractères distincts

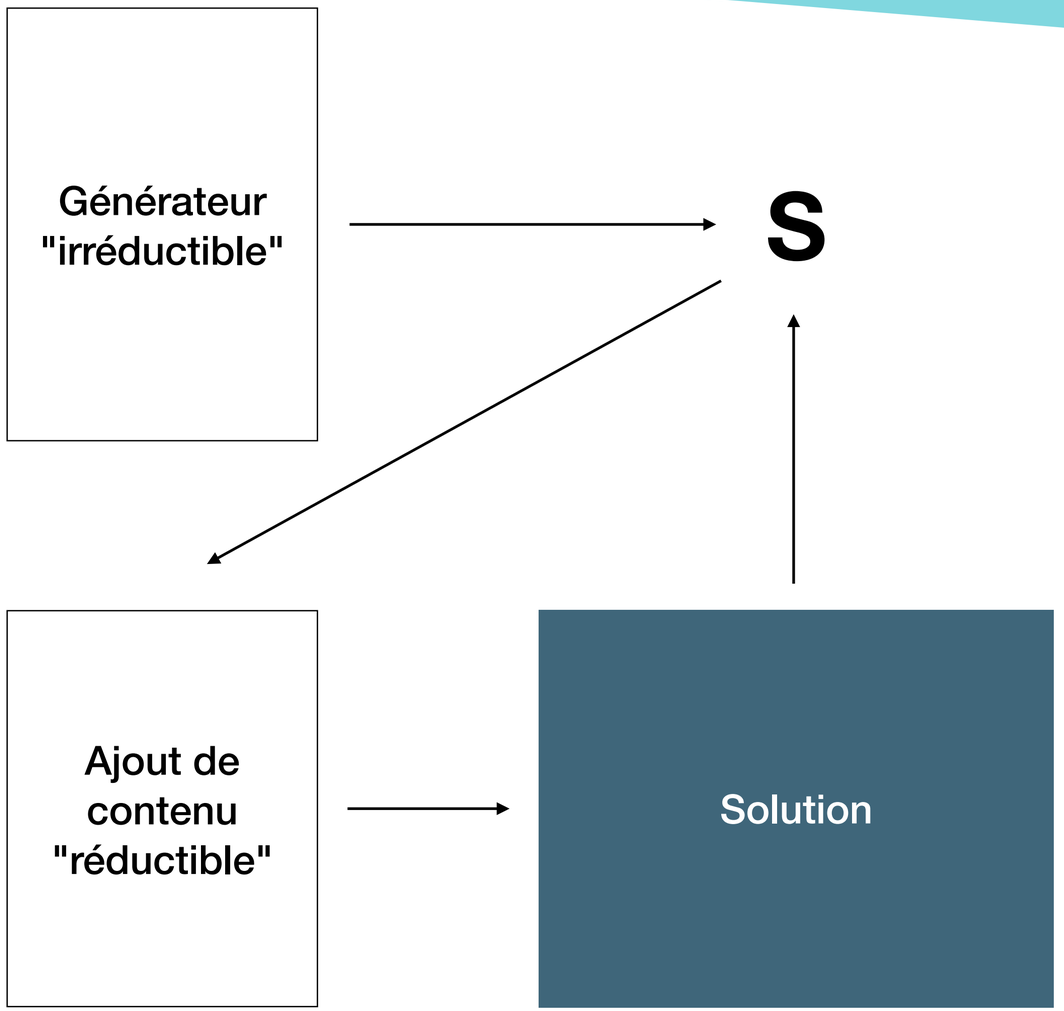
Conversion en chaîne

Entre chaque caractère, ajoute 0 à 4 PAIRES d'un même caractère

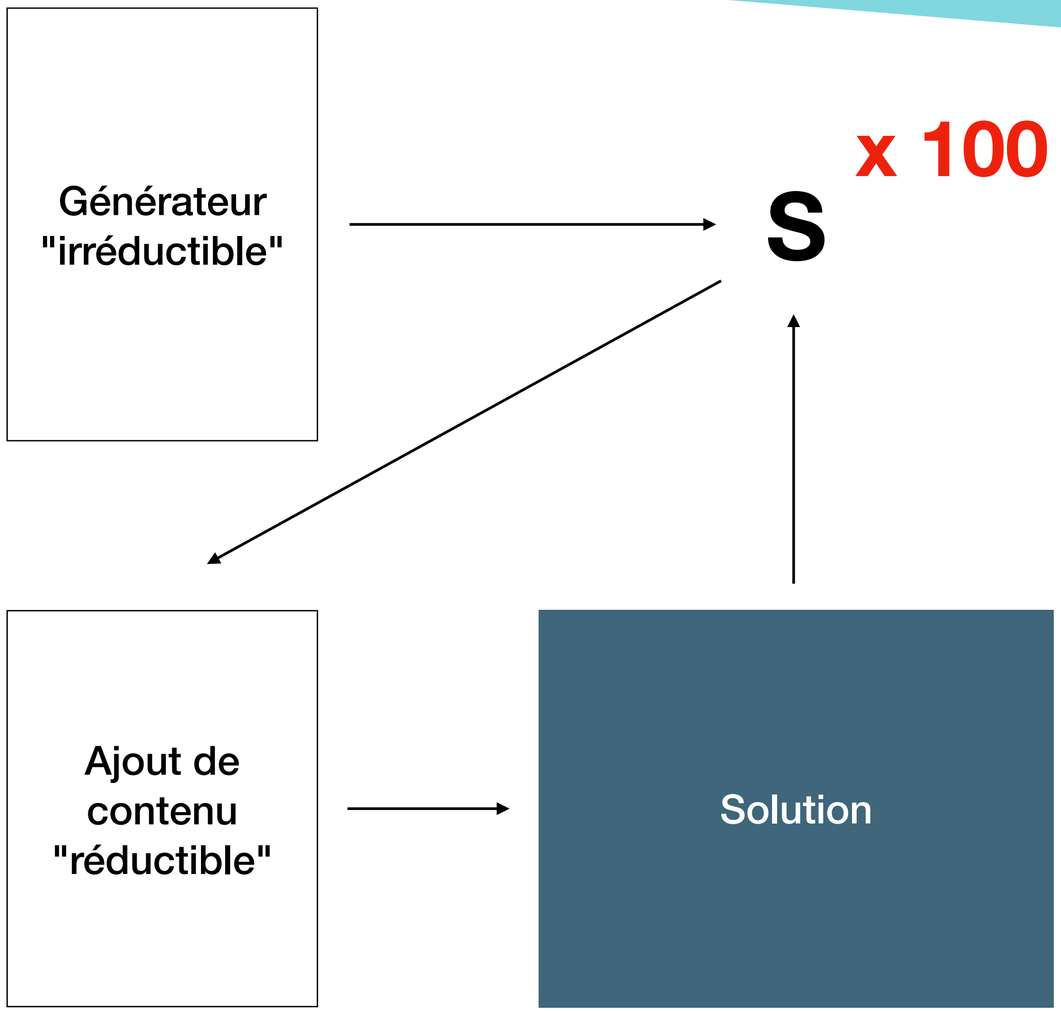
/api/secret-reduce »



/api/secret-base64



/api/secret-base64



/api/exemple-report »

```
POST /api/expense-report
Content-Type: text/plain

1 + 2 + 3 - 5
```

```
Status : 200
{"value":1}
```

/api/exemple-report »

```
POST /api/expense-report?type=postfix
Content-Type: text/plain

1 2 3 + + 5 -
```

```
Status : 200
{"value":1}
```

/api/exemple-report »

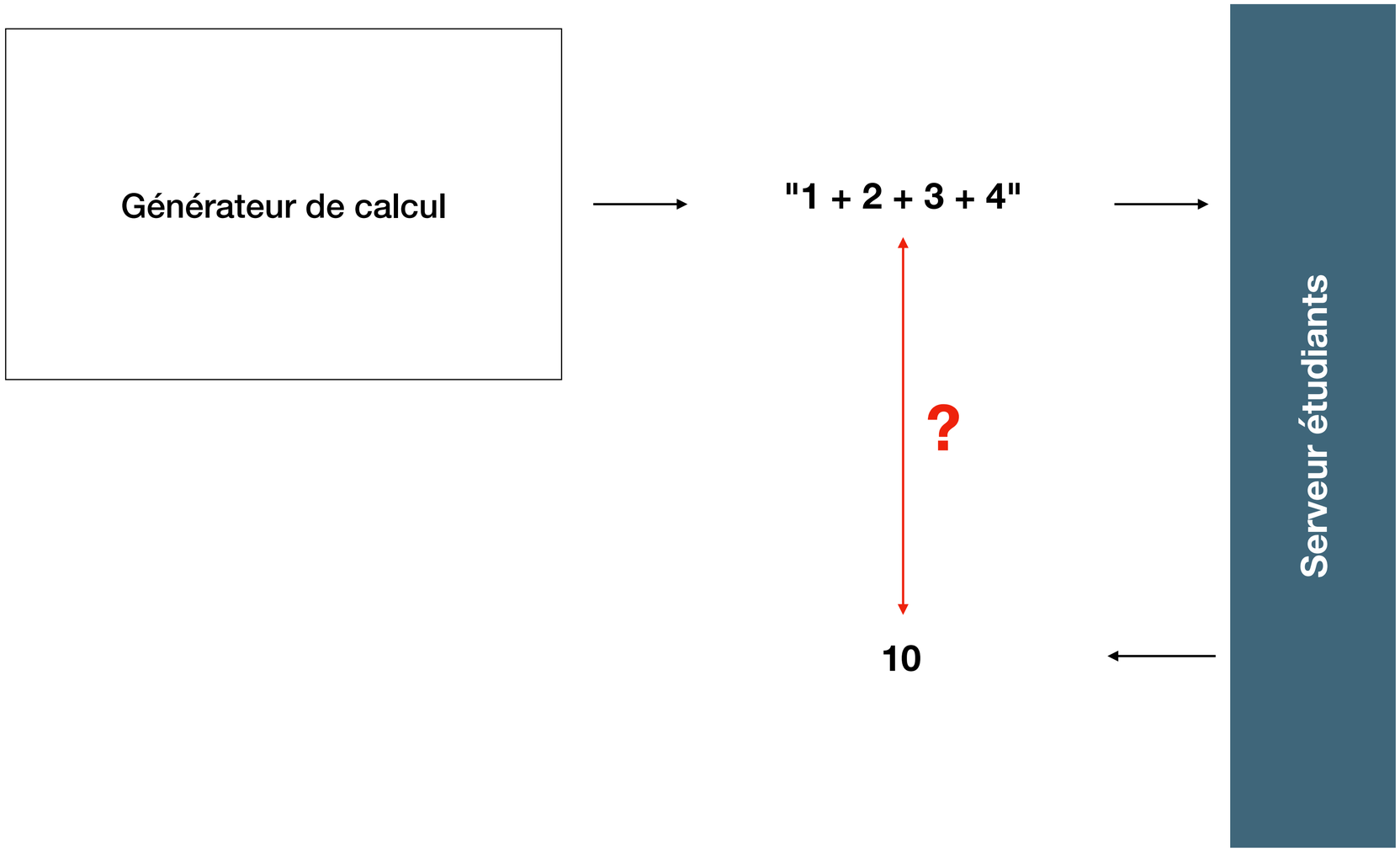
Générateurs de listes

```
OP = '+' ou '-'

INFIX = [ENTIER [OP ENTIER]+]

OPERANDE = ENTIER ou POSTFIX
POSTFIX = (OPERANDE OPERANDE OP)
```

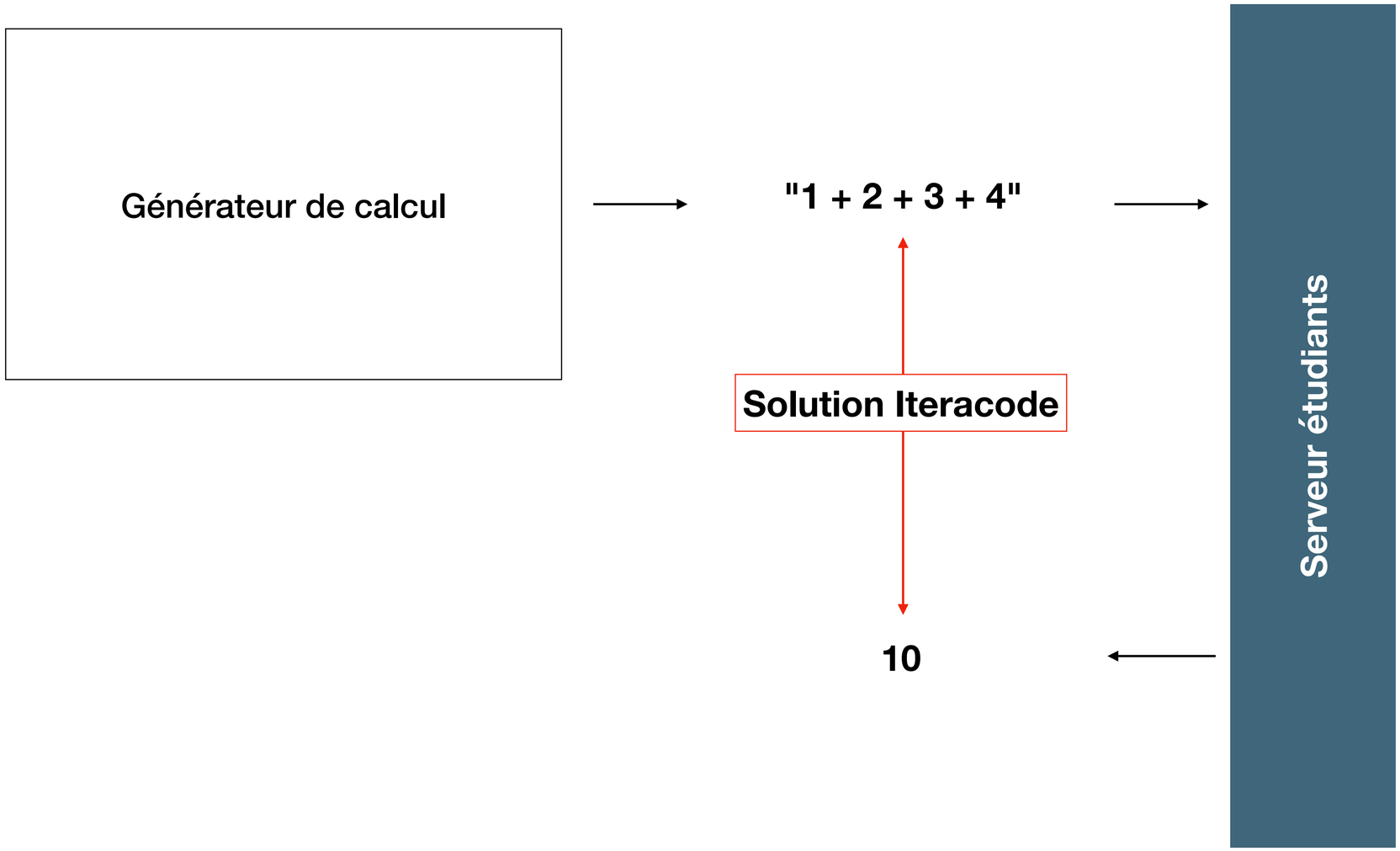
Liste -> Texte

```
(1 '+ 2 '+ 3 '- 4)    --->  "1 + 2 + 3 - 4"

((1 2 +) (3 4 +) -)   --->  "1 2 + 3 4 + -"
```

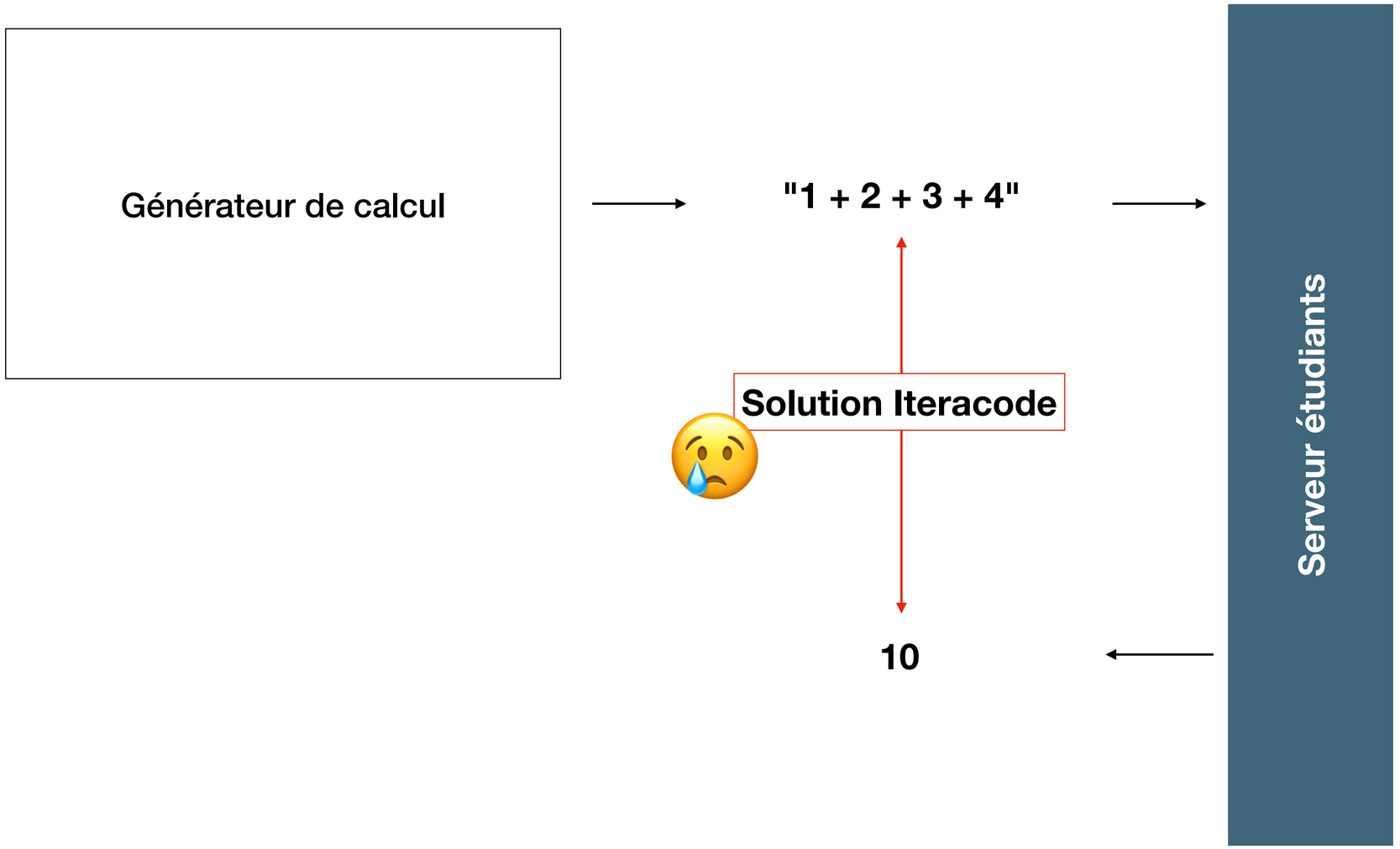
/api/exemple-report »



/api/exemple-report »



/api/exemple-report »



/api/exemple-report »

- Example Based Testing sur notre solution

```
infix-solver "1 + 2 + 3 + 4 + 5" = 15
...
postfix-solver "7 8 + 3 2 + +" = 20
...
```

/api/exemple-report »

- Example Based Testing sur notre solution

```
infix-solver "1 + 2 + 3 + 4 + 5" = 15
...
postfix-solver "7 8 + 3 2 + +" = 20
...
```

- Property Based Testing

```
infix-generator + infix-solver = entier

postfix-generator + postfix-solver = entier
```

/api/exemple-report »

- Example Based Testing sur notre solution

```
infix-solver "1 + 2 + 3 + 4 + 5" = 15
...
postfix-solver "7 8 + 3 2 + +" = 20
...
```

- Property Based Testing

```
infix-generator + infix-solver = entier
```

x 200

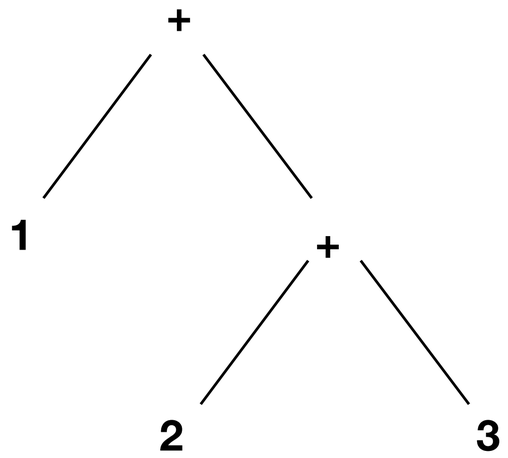
```
postfix-generator + postfix-solver = entier
```

x 200

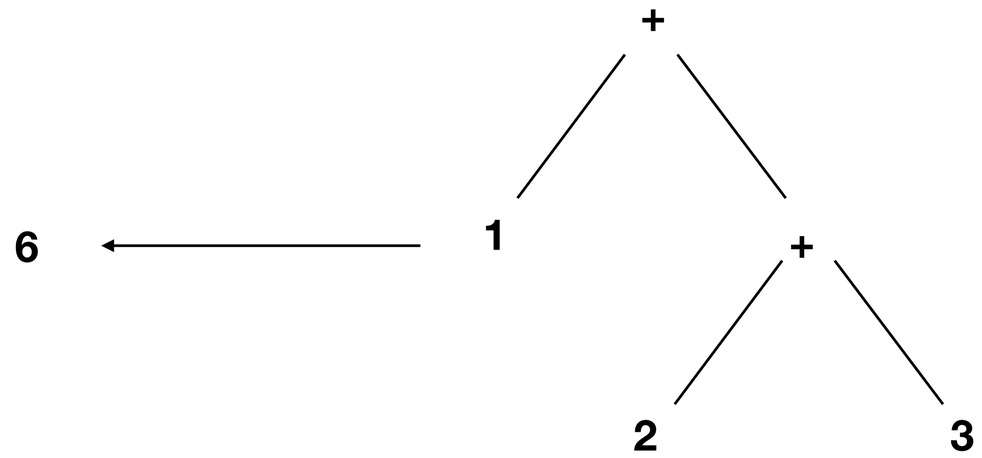
/api/exemple-report »

Peut-on faire un meilleur générateur ?

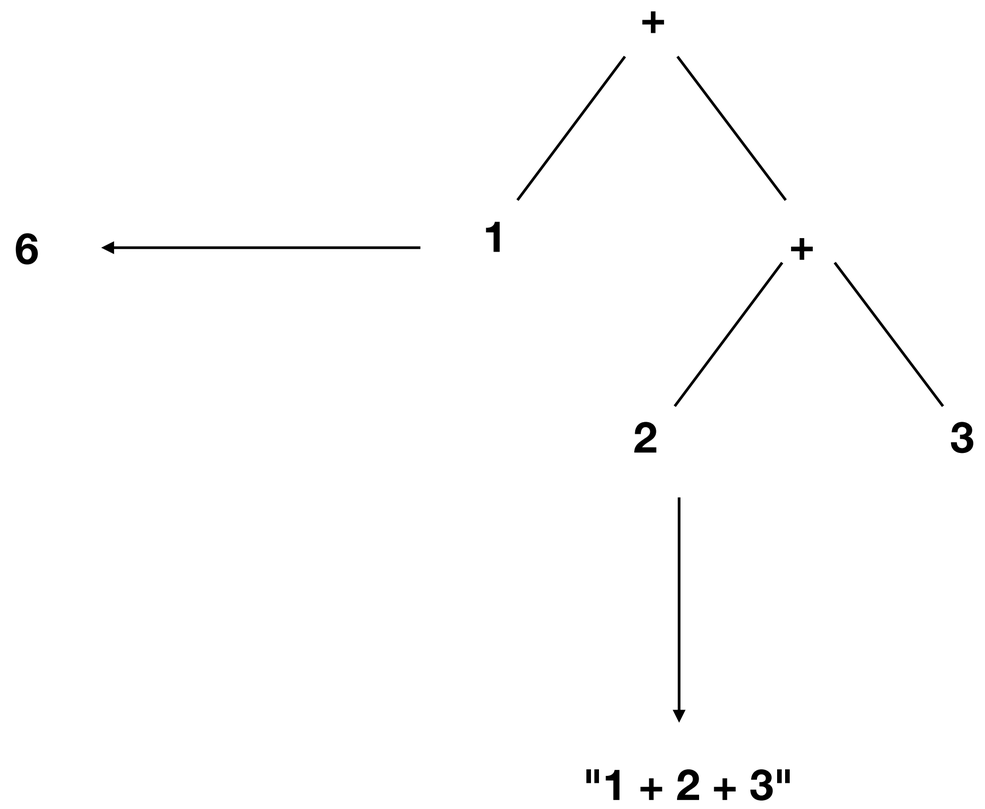
/api/exemple-report »



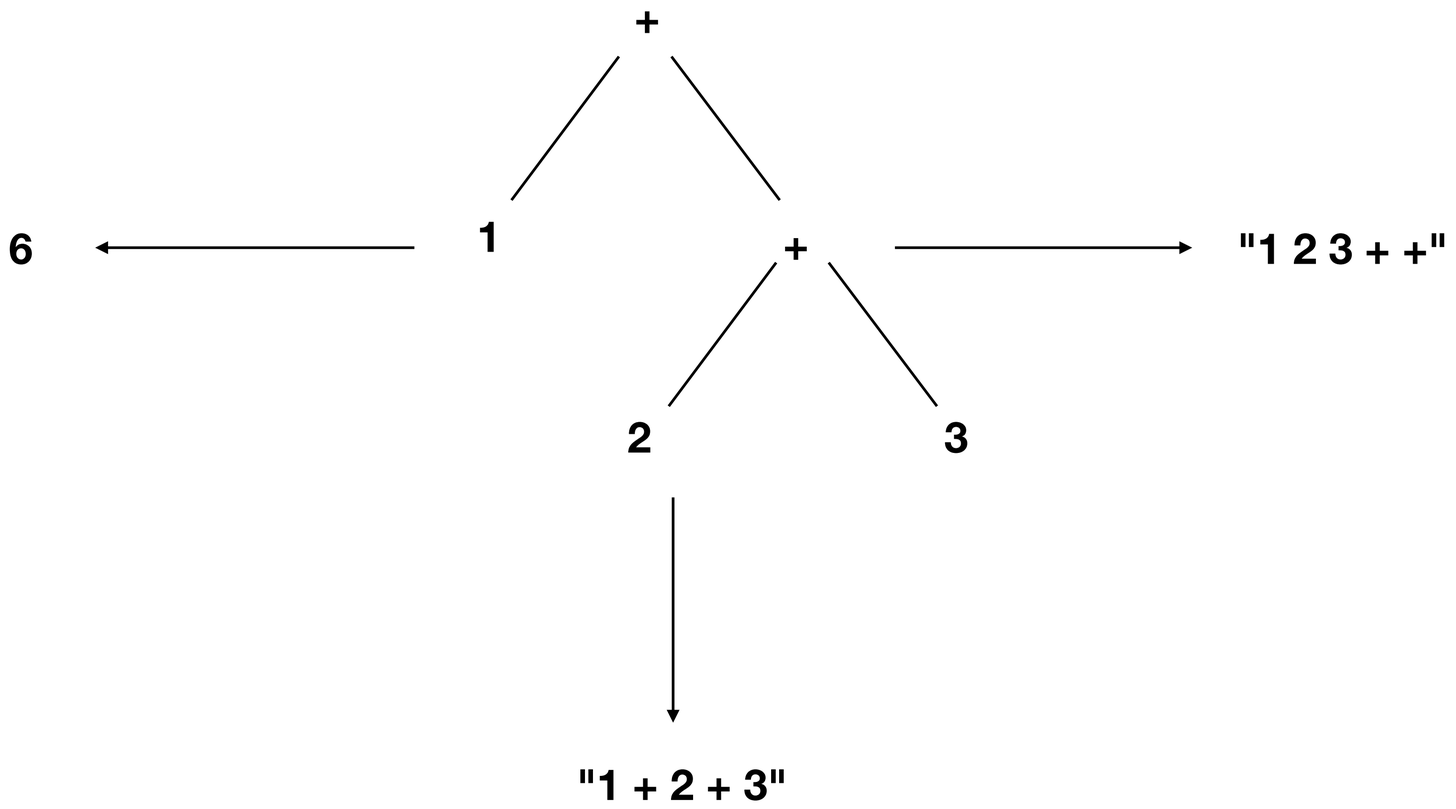
/api/exemple-report »



/api/exemple-report »



/api/exemple-report »



/api/exemple-report »

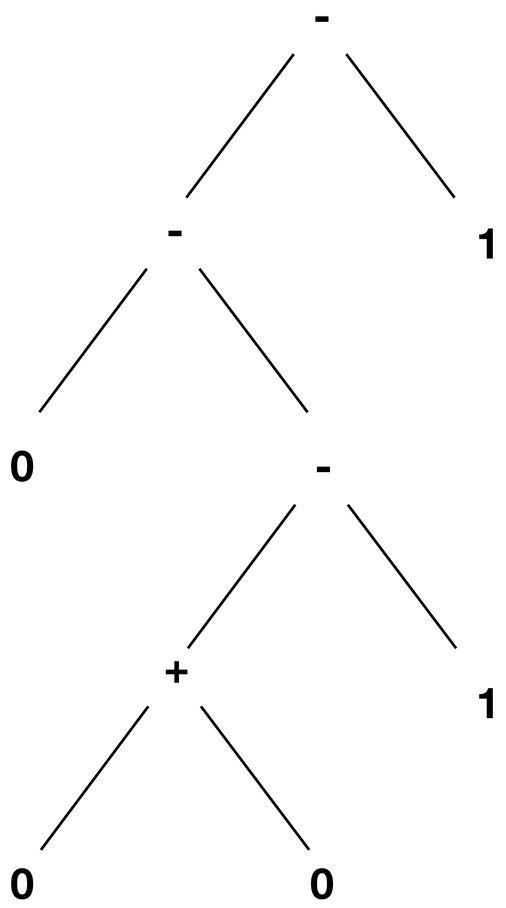
Property Based Testing

```
arbre + evaluate
=
arbre + to-infix + infix-solver
=
arbre + to-postfix + postfix-solver
```

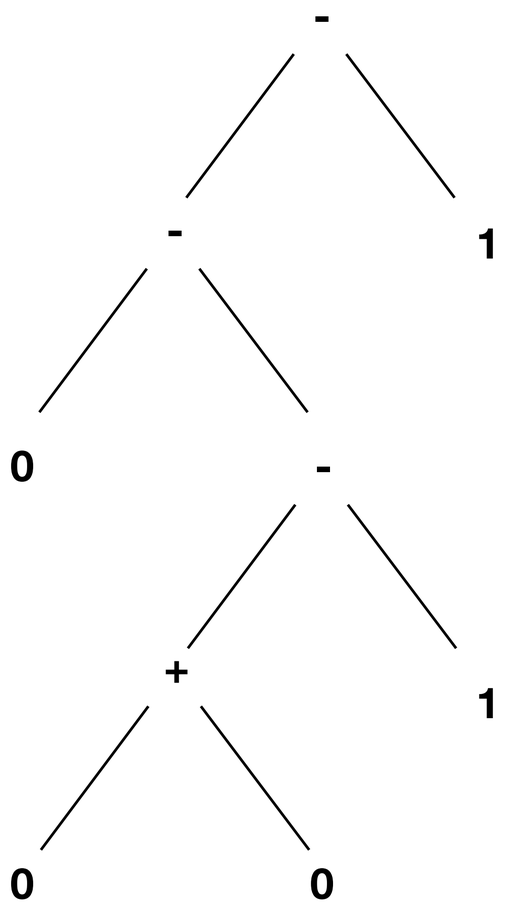
/api/exemple-report »

Property Based Testing

/api/exemple-report »



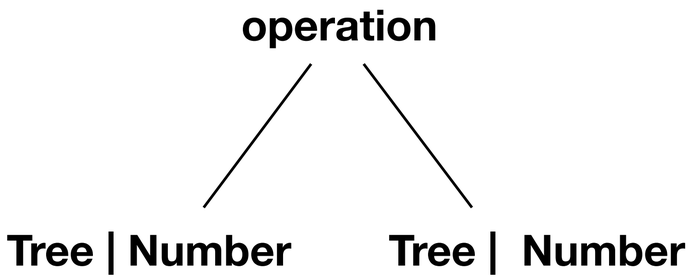
/api/exemple-report »



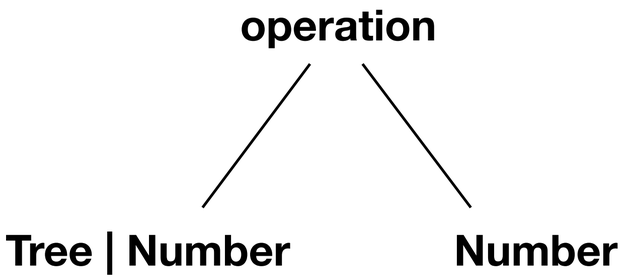
((0 - ((0 + 0) - 1)) - 1) != 0 - 0 + 0 - 1 - 1

/api/exemple-report »

Calculation Tree



« Infix » Tree

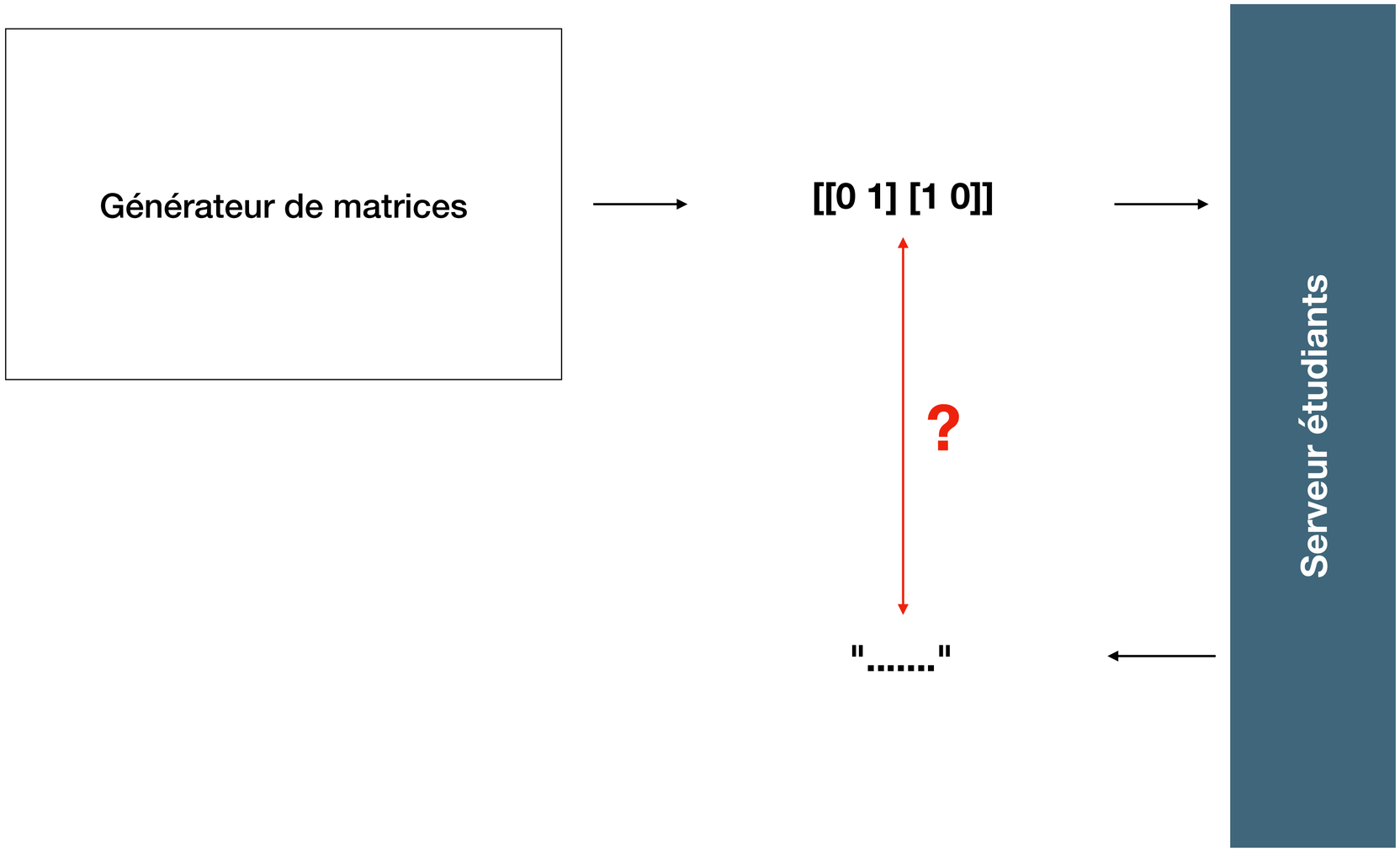


/api/drawing »

```
POST /api/drawing
Content-Type: application/json

{"payload": [[0 1] [1 0]]}
```

```
Status : 200
{"image":"image en base 64"}
```

/api/drawing »



/api/drawing »

Matrice

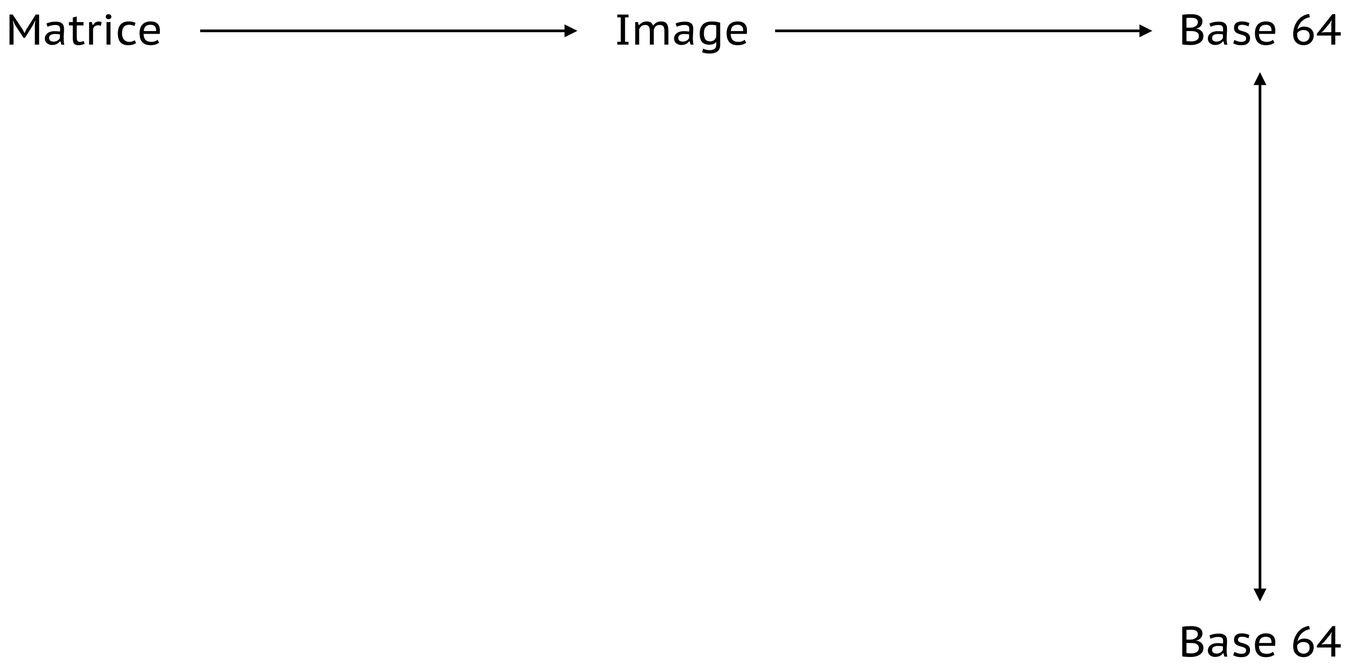
Base 64

/api/drawing »

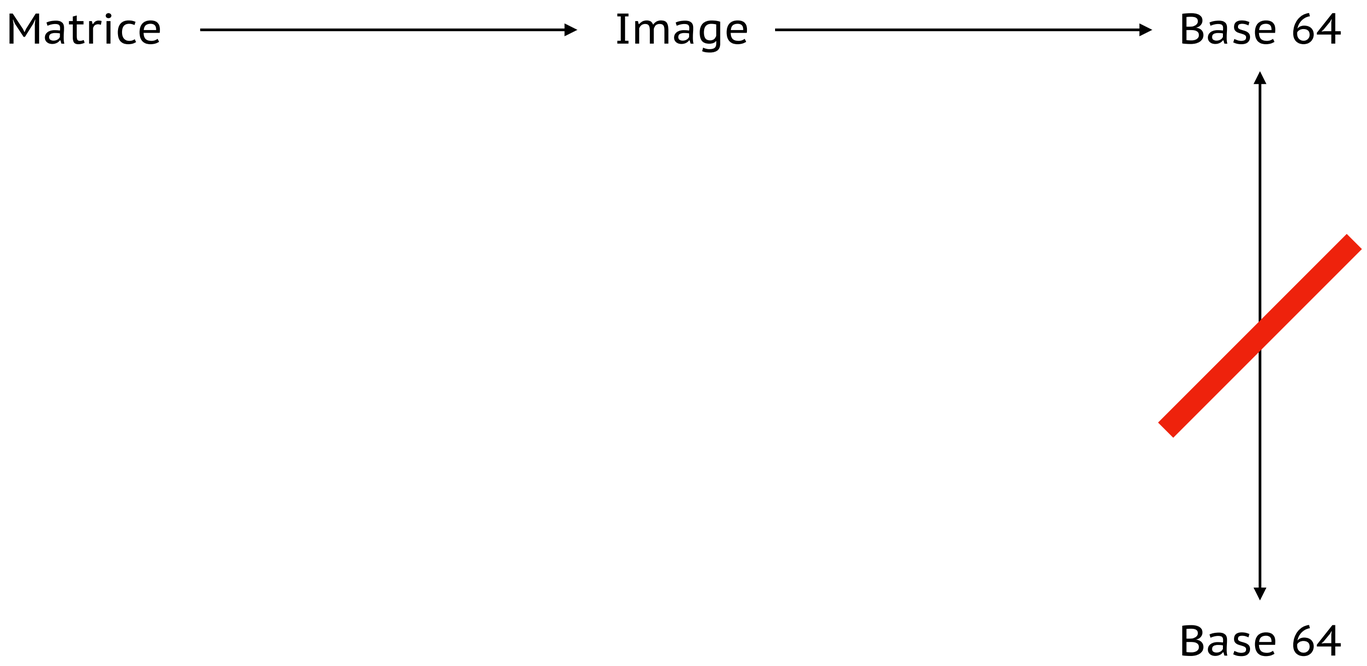
Matrice → Image → Base 64

Base 64

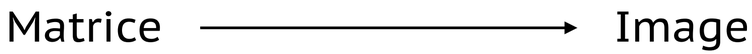
/api/drawing »



/api/drawing »

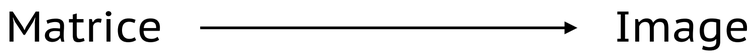


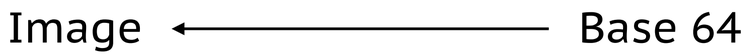
/api/drawing »



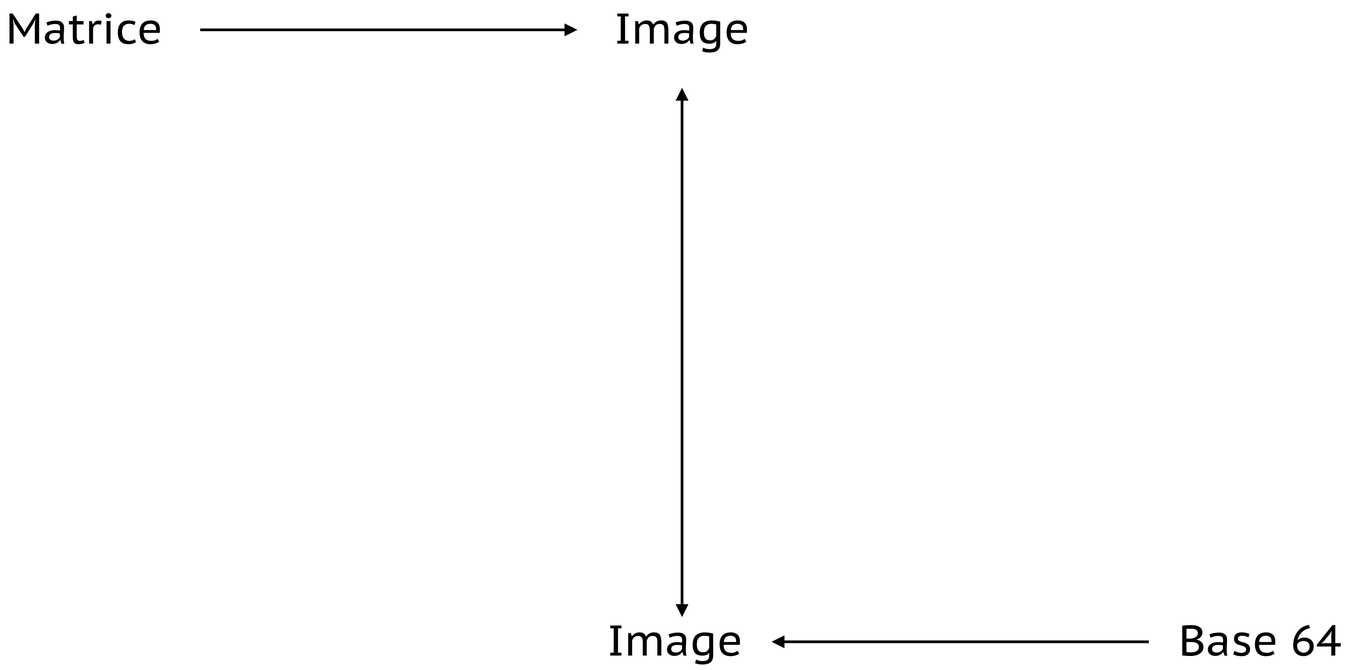
Base 64

/api/drawing »

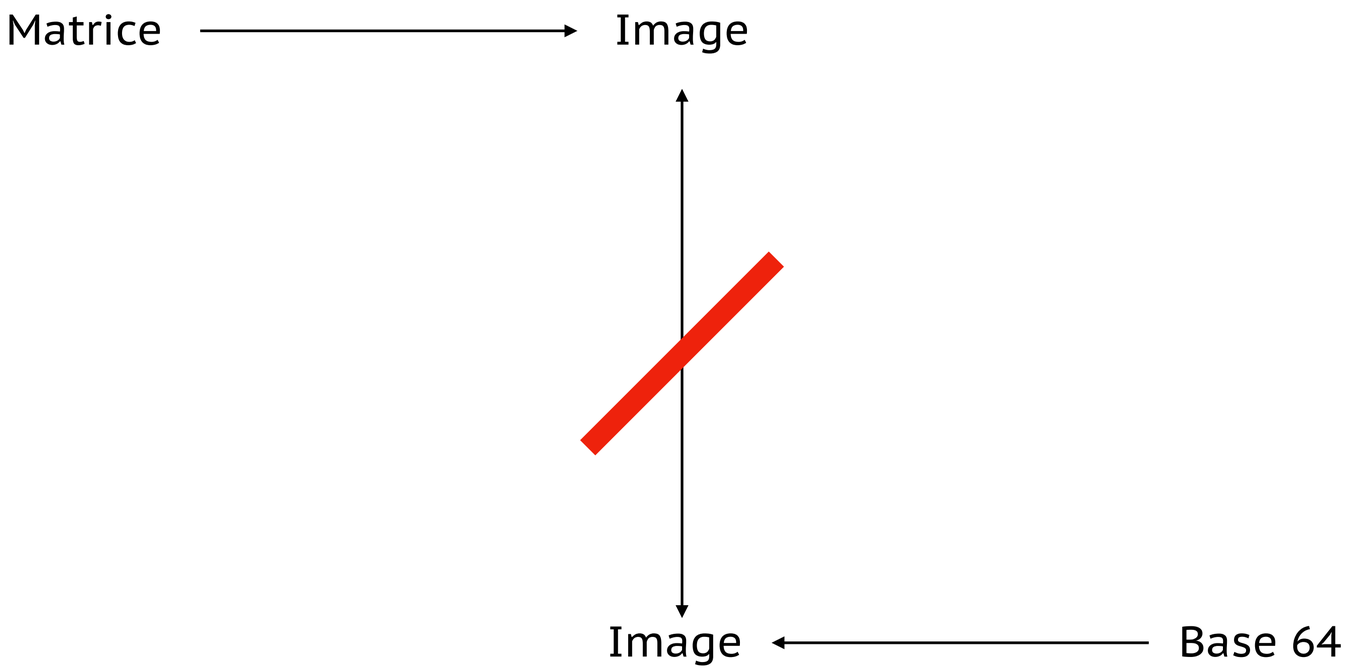




/api/drawing »



/api/drawing »



/api/drawing »

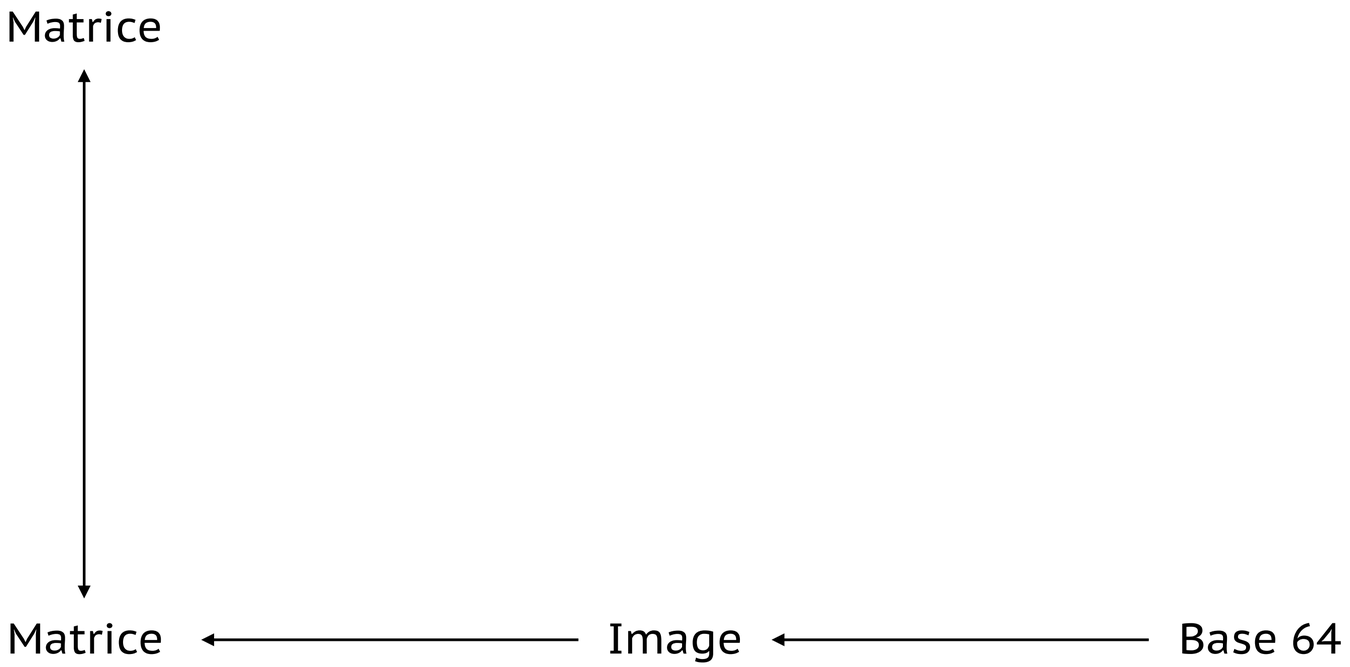
Matrice

Image ⟵ Base 64

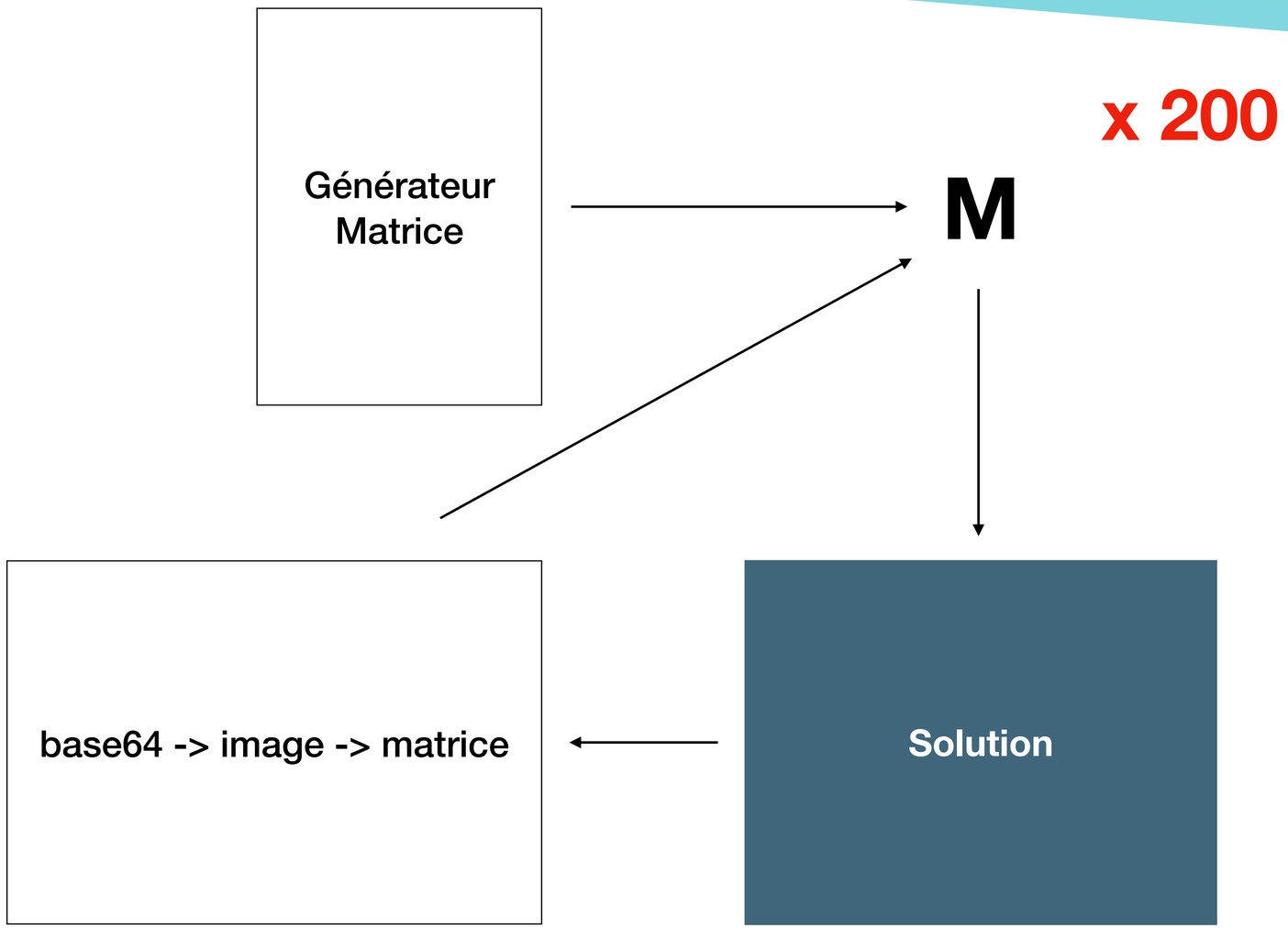
/api/drawing »

Matrice

/api/drawing »



/api/drawing »



ENCORE DES TESTS

Test d’intégration

DEFI SUM - CAS OK
/api/sum + 200?

Test d’intégration

TEST FAIL ?

DEFI SUM - CAS OK
/api/sum + 200?

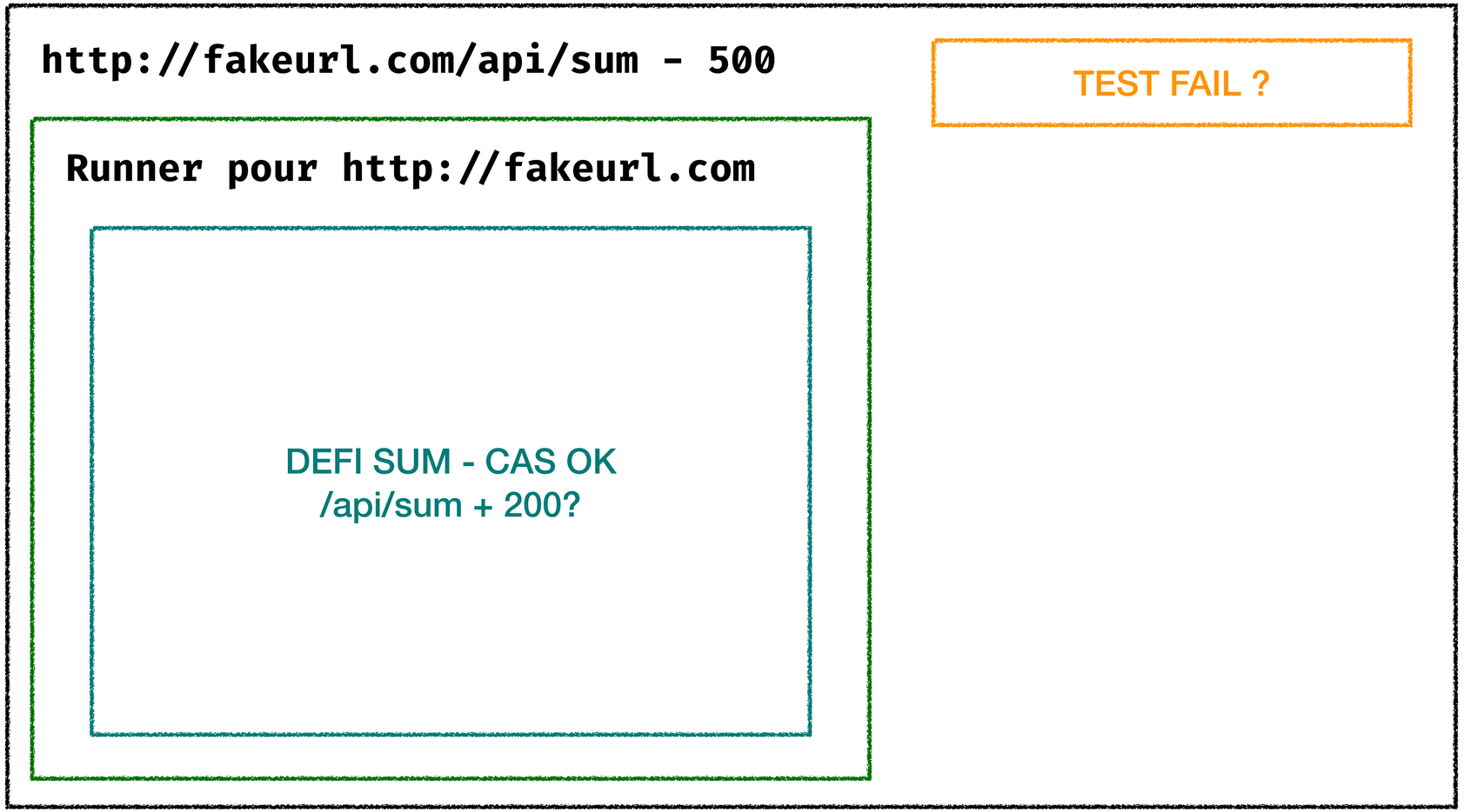
Test d’intégration

TEST FAIL ?

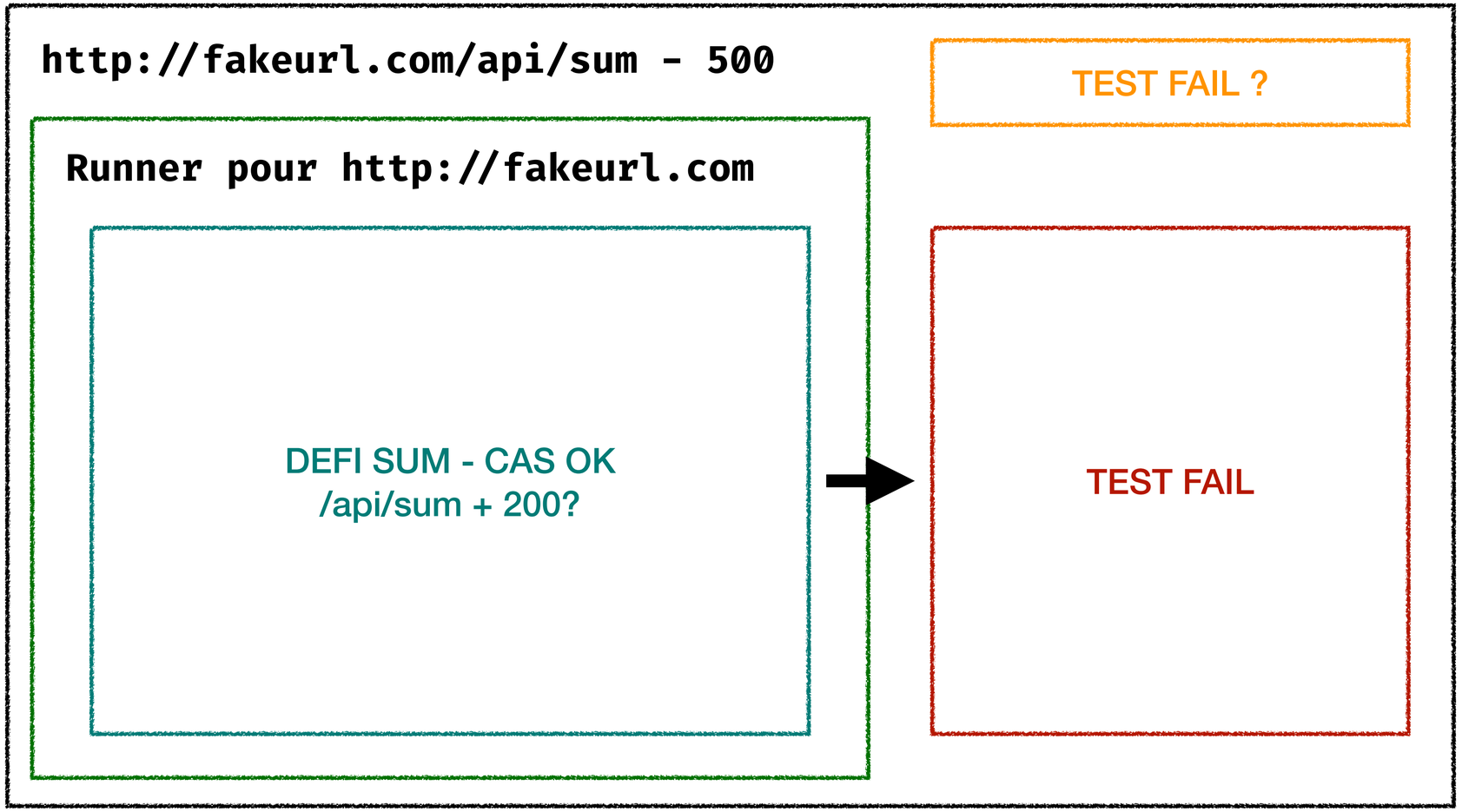
Runner pour http://fakeurl.com

DEFI SUM - CAS OK
/api/sum + 200?

Test d’intégration



Test d’intégration



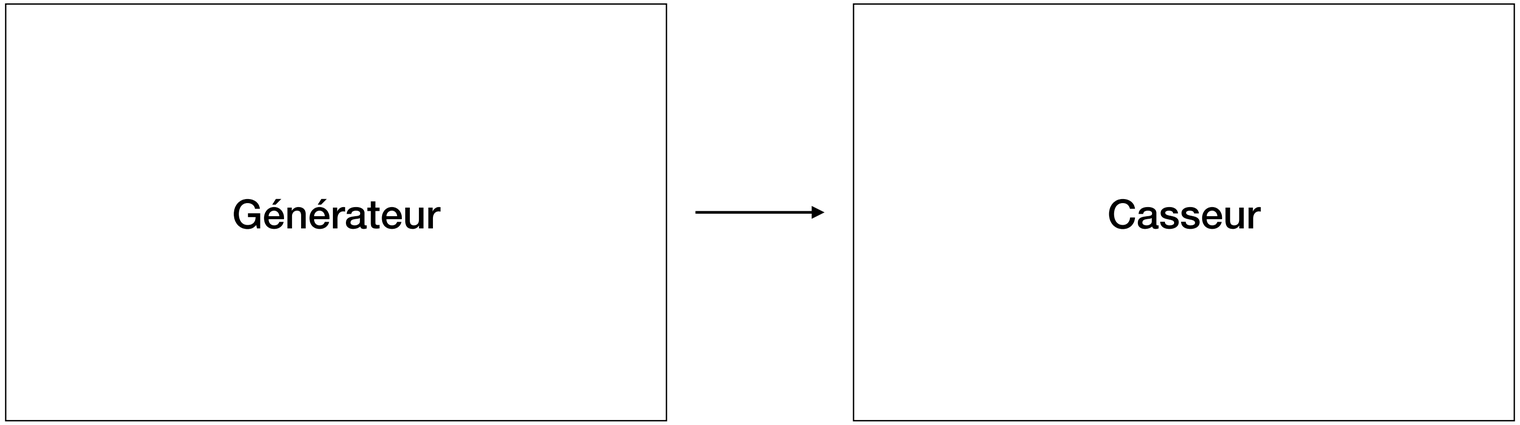
400

Chaîne vide

Matrice incomplète

Expression pas mathématique

400



Property Based Testing

x 200

```
matrix-generator + matrix-breaker
      « n'a pas une forme de matrice »
```

x 200

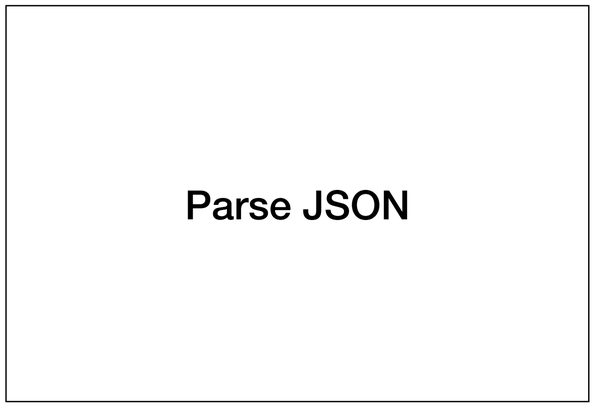
```
postfix-generator + math-breaker
      « ne peut pas être parsé »
```

x 200

```
infix-generator + math-breaker
      « ne peut pas être parsé »
```

Les étudiants contre-attaquent

```
200
Content-Type: application/json

<html>...</html>
```

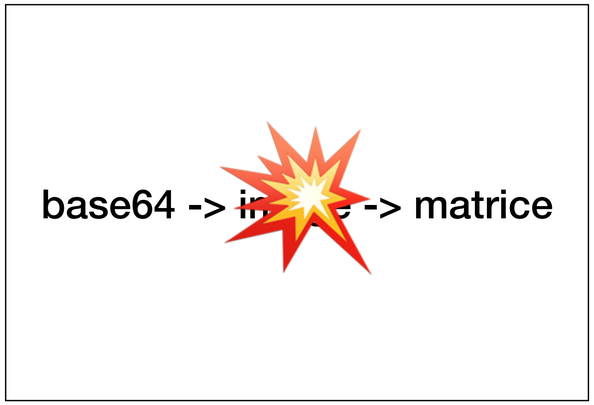



Les étudiants contre-attaquent

```
200
Content-Type: application/json

<html>...</html>
```

Les étudiants contre-attaquent

Les étudiants contre-attaquent

"PAS-DU-TOUT-BASE-64" ⟶ 

QUELQUES LIENS

Exemples d'API + Démo des premières routes

https://github.com/lteracode/nightcode-2020-api-examples

Résultats de l'année 2019

https://github.com/lteracode/nightcode-2019-results

MERCI DE VOTRE ATTENTION DES QUESTIONS ?

Studio d'innovation digitale, en mode Agile.

Charles Fourdrignier - Lead Dev

charles@iteracode.fr

iteracode.fr - 09 81 36 43 87

6 Rue des Hautes Cornes - 80000 Amiens

CLOJURE