

03-Aspect asynchrone et jeux de données de test



Philosophie du 'retry'

1

Un timeout global, configurable et par commande

2

Un 'retry' pour les commandes et les assertions

2

Pas pour toutes les commandes

invoke()

invoke.js

```
describe('Formstest suite',()=>{  
  
  it('phone numbershouldinclude06 ', () => {  
  
    cy.get('#phone-number')  
      .invoke('text')  
      .then(phoneNumber=> {  
        expect(phoneNumber).to.include('06');  
      });  
  })  
})
```

wrap()

wrap.is

```
describe('Nomdelasuitede test',()=>{  
  
  it('Scénarionominal ', () => {  
  
    cy.get('#firstName').wrap() type('mohammed.ali@wildcodeschool.com')  
    cy.get('#login-email').wrap () type('hahahatuveuxmonmotdepasse?')  
      })    cy.get('#login-password').  
  })  
})
```



Alias

Donner un alias aux
utilisateurs stockés
dans ce fichier :
usersGroup

Utiliser le contenu du
fichier dans les tests avec
son aléa

```
1 describe('Control All', () => {  
2   before(() => {  
3     cy.fixture('users.json')  
4     .as('usersGroup')  
5   });  
6  
7   it('Load users', () => {  
8     cy.get('@usersGroup').then(users => {  
9       users.forEach(user => {  
10        cy.log(`User email : ${user.email}`)  
11      });  
12    })  
13  });  
14 })
```

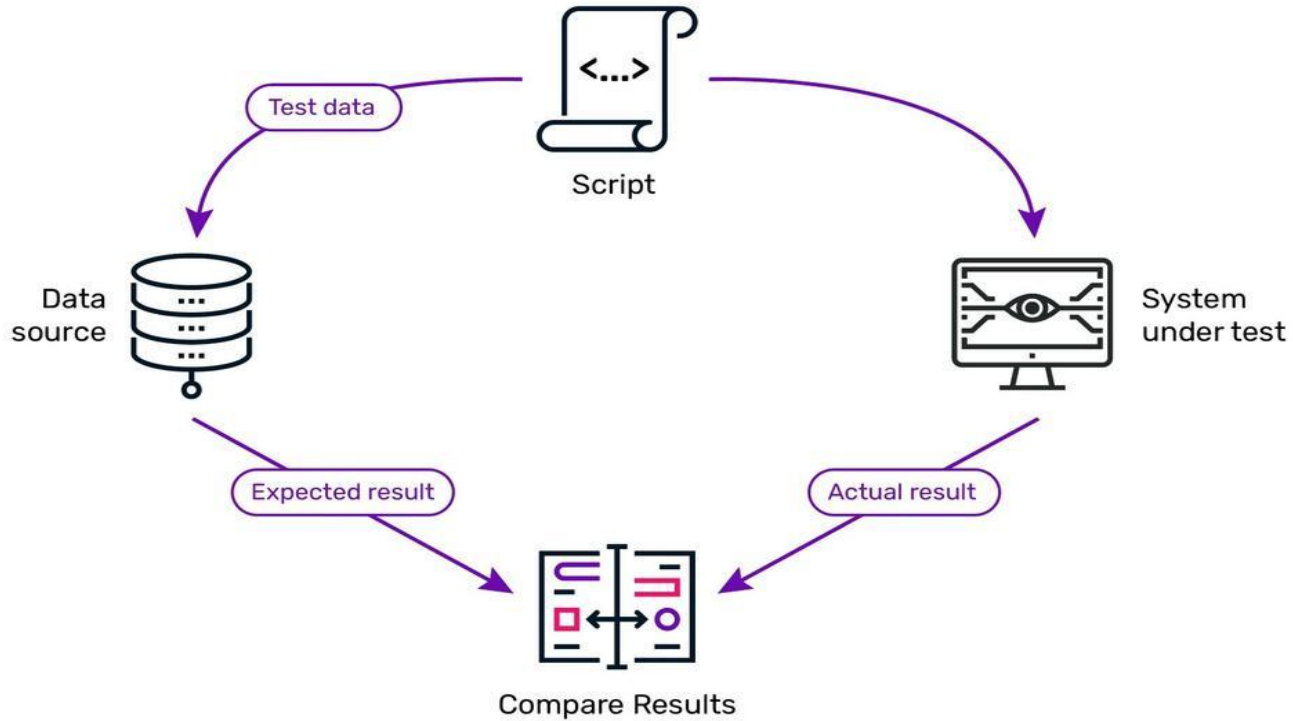
intercept()

nominal.js

```
describe('Test d'interception d'une requête', () => {  
  it('Scénario nominal', () => {  
    cy.intercept('/api/getUser').as('getUser')  
  
    cy.get('#getUserButton').click()  
    cy.wait('@getUser').its('response.body')  
      .should('include', { name: 'John Doe' })  
  })  
})
```



Data Driven Testing-concept





Data Driven Testing–Cypress

Charger les utilisateurs d'un fichier
JSON

Créer un test pour chaque
utilisateur

```
Exercices > cypress > integration > presentation > usersFixture.js > ...
1  describe('Control All', () => {
2
3      const users = require('../../fixtures/users')
4
5      users.forEach( user => {
6          it('Should contain valid data', () => {
7              cy.wrap(user.lastName).should('be.a', 'string')
8              cy.log(`Name : ${user.lastName} ${user.firstName}`)
9              /*
10               * Do some work here to retrieve validity state of user
11               */
12              expect(true).to.deep.equal(user.isValid)
13          });
14      });
15 }
```





Fixture

Charger le fichier users.json
localisé dans le dossier
cypr ess/ f i x t ur es

Donner un aléa aux
utilisateurs stockés
dans ce fichier :
usersGroup

Utiliser le contenu du
fichier dans les tests avec
son aléa

```
1 describe('Control All', () => {  
2   before(() => {  
3     cy.fixture('users.json')  
4     .as('usersGroup')  
5   });  
6  
7   it('Load users', () => {  
8     cy.get('@usersGroup').then(users => {  
9       users.forEach(user => {  
10        cy.log(`User email : ${user.email}`)  
11      });  
12    })  
13  });  
14 })
```



DEMO

Data DrivenTesting



Lève la main ! 🖐️

As-tu compris ?



I completely understand
(can teach it).



I mostly understand
(can show it).



I understand pretty well.



I need more practice
and examples.



I need help.



I don't understand at all.

Assertions: Expect

AssertionsChai

```
expect(name).to.not.equal('Jane')
```

```
expect(obj).to.deep.equal({name:'Jane'})
```

```
expect(10).to.be.greaterThan(5)
```

```
expect(1).to.satisfy((num)=>{return num>0})
```

```
expect(myVar).to.exist
```

```
expect(null).to.be.null
```

assertions.js



APP ACTIONS



Cypresscommands

Emplacement du
script

commands.js

Nom de la commande
personnalisée

Paramètres

Instructions

```
14  
15 Cypress.Commands.add("generateForm", () => {  
16   cy.get('#generate-btn').click()  
17 })  
18  
19 Cypress.Commands.add("completeDestinationData", (city, department) => {  
20   cy.get('#field-destinationtown').type(city)  
21   cy.get('#field-destinationcounty').type(department)  
22 })  
23
```