

Cypress와 쿼리

실무에 바로 적용하는 프론트엔드 테스트


사용 브랜치



shopping-mall-e2e-test-example-with-answer

```
$ git checkout shopping-mall-e2e-test-example-with-answer
```

subject



```
cy.findAllByRole('table')  
  .eq(1)  
  .findAllByRole('row')  
  .as('itemList');
```

Cypress에서는 커맨드를 Promise 체이닝을 통해 관리하며,
각 커맨드는 체인이 종료되거나 오류가 발생할 때 까지 멈춤

→ 이런 동작을 제어하기 위해 내부적으로 다음 커맨드의 대상을 **subject**로 관리

subject

```
cy.findAllByRole('table')  
  .eq(1)  
  .findAllByRole('row')  
  .as('itemList');
```

Cypress에서는 커맨드를 Promise 체이닝을 통해 관리하며,
각 커맨드는 체인이 종료되거나 오류가 발생할 때 까지 멈춤

→ 이런 동작을 제어하기 위해 내부적으로 다음 커맨드의 대상을 **subject**로 관리

테스트의 시작 시점 또는 대상이 되는 요소

Retry-ability

- 🤔 모두 다른 타이밍에 DOM이 렌더링 된다면?
- 🤔 API 응답을 기다리는 중이라면?
- 🤔 계산이 오래 걸리는 로직이 존재한다면?

언제 테스트를 시작해야 할까?
특정 요소가 렌더링 되었는지 매번 체크해야 할까?

Retry-ability

- 🤔 모두 다른 타이밍에 DOM이 렌더링 된다면?
- 🤔 API 응답을 기다리는 중이라면?
- 🤔 계산이 오래 걸리는 로직이 존재한다면?

Retry-ability 재시도로 타이밍 제어!

잠재적으로 업데이트 가능성이 있다고 판단해
원하는 상황이 발생할 때까지 특정 시간동안 재시도

Retry-ability 시간 변경하기



```
// 인위적으로 10초 동안 대기  
cy.findByLabelText('이메일', { timeout: 10000 })
```

기본 대기시간은 4초

defaultCommandTimeout로 전역 재시도 시간을 변경할 수도 있음

Retry-ability

커맨드에서 내부적으로 몇 번을 재시도하던
subject가 yield한 결과를 체이닝 형태로 받기 때문에
일관된 환경에서 테스트가 가능해짐

→ 선행 동작이 실패하면 이후의 동작은 실행되지 않음

Retry-ability 여부에 따른 API 구분

- Query: 전체 체이닝 로직을 재시도하며 실행
 - ex) get, cypress-testing-library, find 등
- Assertion: 단언을 수행하는 특별한 쿼리의 유형
 - ex) should 등
- Non-Query: 재시도를 하지 않으며, 단 한 번만 실행되는 커맨드
 - ex) visit, click, then 등

정리 (1/2)

cy.get()

- jQuery와 유사한 CSS Selector를 사용해 DOM에 접근할 수 있다.
- cy.as()로 선언한 별칭(alias)과 함께 사용하면 테스트 코드를 간결하게 작성할 수 있다.

체이닝과 subject

- subject 객체는 테스트의 시작 지점 또는 대상이 되는 요소를 의미
- 이를 통해 체이닝 명령 수행이나 종료 또는 오류를 제어
- Cypress의 커맨드 실행이 완료되는 타이밍을 맞추기 위해서는 subject 체이닝 형태로 연속해서 커맨드를 실행하거나 then() API를 사용. 이러한 실행 결과 전달을 yield라고 한다.

정리 (2/2)

Retry-ability

- `cy.get()`, `cy.should()` 등 잠재적으로 업데이트 가능성이 있다고 판단해 재시도를 실행하는 API
- `subject` 덕분에 타이밍 문제 없이 재시도하여 얻은 결과를 순차적으로 사용할 수 있다.
- 테스트 타이밍에 대한 다양한 고민 요소를 해결해주는 핵심 기능

Retry-ability 기능 여부에 따른 API 구분

- Query: 전체 체이닝 로직을 재시도하며 수행
- Assertion: 단언을 수행하는 특별한 쿼리의 유형
- Non-Query: 재시도를 하지 않으며, 단 한 번만 실행되는 커맨드