**Pacotes importantes - Cypress**

**Faker js**

**O que é?**

Transforma dados estáticos para dados dinâmicos randômicos.

* **Endereços** - Gere endereços válidos, códigos postais, nomes de ruas, estados e países;
* **Dados baseados no tempo** - passado, presente, futuro, recente, em breve... quando quiser;
* **Localização** - Defina um local para gerar nomes, endereços e números de telefone realistas;
* **Finanças** - Crie detalhes da conta, transações e endereços criptográficos;
* **Produtos** - Gere preços, nomes de produtos, adjetivos e descrições;
* **Nomes** - Gere humanos virtuais com uma identidade online e offline completa;
* **Números** - Claro, também podemos gerar números aleatórios e strings.

**Instalação:**

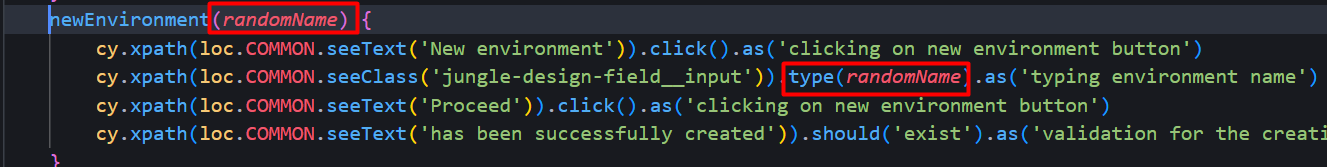
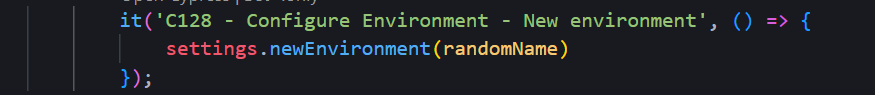
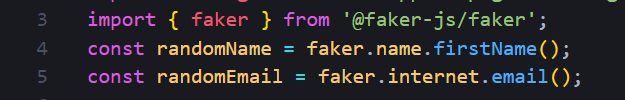
*npm install --save-dev @faker-js/faker*

**Uso:**

Importar o pacote no início da spec ou do arquivo desejado.

Colocar as variáveis com o tipo de dados desejados, você pode checar elas nesta documentação: <https://fakerjs.dev/api/>

Chame a variável no teste escolhido.



**Links:**

<https://www.npmjs.com/package/@faker-js/faker>

<https://fakerjs.dev/>

**Faker br**

**O que muda do Faker js?**

Os dados gerados são em implantados no Brasil em língua portuguesa, alguns exemplos de diferenciais:

* Gerador de CPF;
* Gerador de CNPJ;
* Gerador de cartão de crédito válido.

**Instalação:**

*npm install faker-br*

**Uso:**

**—>** Importar o pacote no início da spec ou do arquivo desejado.

var faker = require('faker-br');

**—>** Colocar as variáveis com o tipo de dados desejados, você pode checar elas nesta documentação:<https://github.com/tamnil/Faker-br#api-methods>

var randomcpf = faker.br.cpf();

var randomCnpj = faker.br.cnpj();

**—>** Chame a variável no teste escolhido.

console.log(faker.br.cnpj());

**Links**:

<https://www.npmjs.com/package/faker-br>

<https://github.com/tamnil/Faker-br>

**Mochawesome Reporter**

**O que é?**

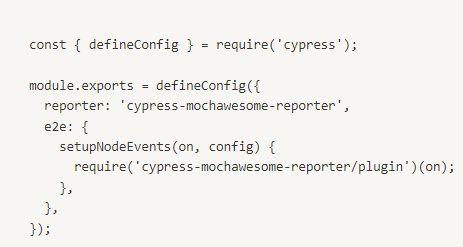
O pacote "cypress-mochawesome-reporter" é um plugin para o framework de testes Cypress que gera relatórios de testes no formato "mochawesome". Esse formato permite que os resultados dos testes sejam apresentados de forma clara e organizada, com tabelas que facilitam a compreensão dos resultados.

**Instalação:**

* No terminal, executar:

npm i --save-dev cypress-mochawesome-reporter

* Editar no arquivo cypress.config.js:



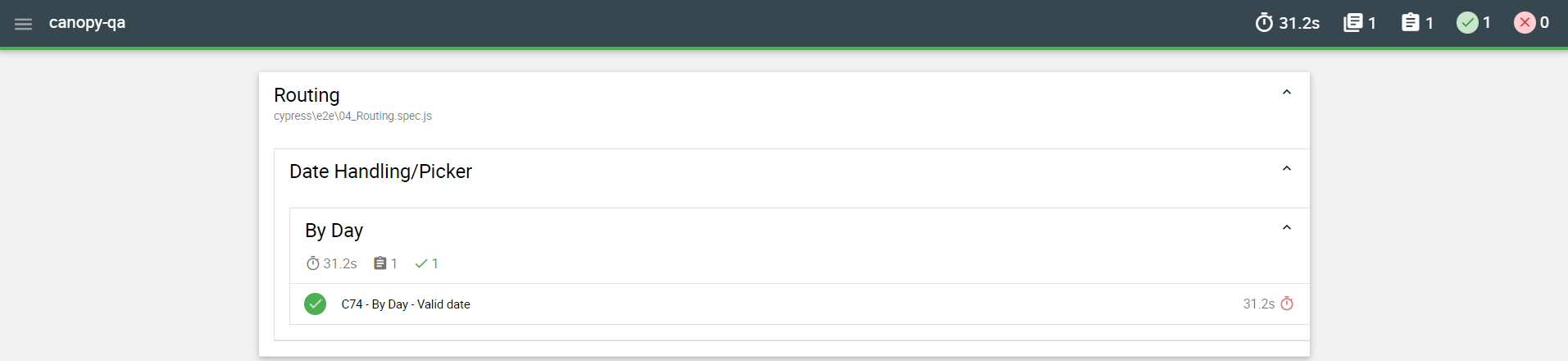
* Adicionar no arquivo cypress/support/e2e.js:

import 'cypress-mochawesome-reporter/register';

**Uso:**

Run Cypress —> gerará um arquivo .html toda vez que os testes forem executados em modo headless na pasta cypress/reports/html.

**Exemplo:**

****

**Links:**

<https://www.npmjs.com/package/cypress-mochawesome-reporter>

**Cypress real events**

**O que é?**

No Cypress, os eventos padrão são simulados por meio de JavaScript, o que faz com que esses eventos sejam considerados não confiáveis (o valor de event.isTrusted é falso) e possam apresentar comportamentos distintos dos eventos nativos. Contudo, em algumas situações, como no preenchimento de um alerta nativo ou na cópia para a área de transferência, o uso de eventos simulados não é possível. Para solucionar esse problema, um plugin específico foi desenvolvido.

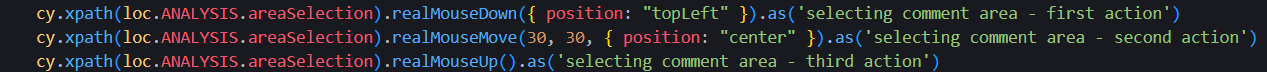
**Instalação:**

*npm install cypress-real-events*

**Usos:**

* cy.realClick
* cy.realHover
* cy.realPress
* cy.realTouch
* cy.realType
* cy.realSwipe
* cy.realMouseDown
* cy.realMouseUp
* cy.realMouseMove
* cy.realMouseWheel

**Exemplos:**



**Links:**

<https://github.com/dmtrKovalenko/cypress-real-events>