

Informe de Testing Y Pruebas de Código

Introducción

En el desarrollo de software, el testing y las pruebas de código son una parte muy importante para garantizar la calidad de las aplicaciones.

Conceptos Básicos Definición y Diferencias entre Testing y Pruebas de código

- **Testing:** Es un proceso en el cual se evalúa el código para detectar posibles errores.
- **Pruebas de Código:** Se centra en evaluar la calidad y corrección del código.

La finalidad de las pruebas es que valide que todo está correcto, que identifique errores y que el rendimiento sea óptimo.

Tipos de pruebas

→ Pruebas unitarias

Son de bajo nivel y evalúan métodos y funciones individuales de las clases, las pruebas unitarias son bastante baratas de automatizar y se pueden ejecutar rápidamente mediante un servidor de integración continua.

→ Pruebas de integración

Las pruebas de integración verifican que los distintos módulos o servicios utilizados por tu aplicación funcionan bien en conjunto. Estos tipos de pruebas son más costosos de ejecutar, ya que requieren que varias partes de la aplicación estén en marcha.

→ Pruebas funcionales

Se centran en los requisitos empresariales de una aplicación. Solo verifican el resultado de una acción y no comprueban los estados intermedios del sistema al realizar dicha acción.

→ Pruebas de extremo a extremo

Replican el comportamiento de un usuario con el software en un entorno de aplicación completo. verifican que diversos flujos de usuario funcionen según lo previsto.

→ Pruebas de aceptación

Son pruebas formales que verifican si un sistema satisface los requisitos empresariales Requieren que se esté ejecutando toda la aplicación durante las pruebas y se centran en replicar las conductas de los usuarios.

→ Pruebas de rendimiento

Evalúan el rendimiento de un sistema con una carga de trabajo determinada. Ayudan a medir la fiabilidad, la velocidad, la escalabilidad y la capacidad de respuesta de una aplicación.

→ Pruebas de humo

Son pruebas básicas que sirven para comprobar el funcionamiento básico de la aplicación. pueden resultar útiles justo después de realizar una compilación nueva para decidir si se pueden ejecutar o no pruebas más caras.

Técnicas de Testing

TDD (Test Driven Development): TDD es una técnica de desarrollo de software que practica la escritura de pruebas que provocan «fallos» para luego refactorizarlo.

BDD(Behaviour Driven Development): Se considera una extensión de Test Driven Development. El Desarrollo Dirigido por el Comportamiento es una técnica que practica la creación de escenarios simples sobre cómo debe comportarse una aplicación desde la perspectiva del usuario final.

ATDD (Acceptance Test Driven Development): Es una metodología de desarrollo basada en la comunicación entre el cliente empresarial, el desarrollador y el tester.

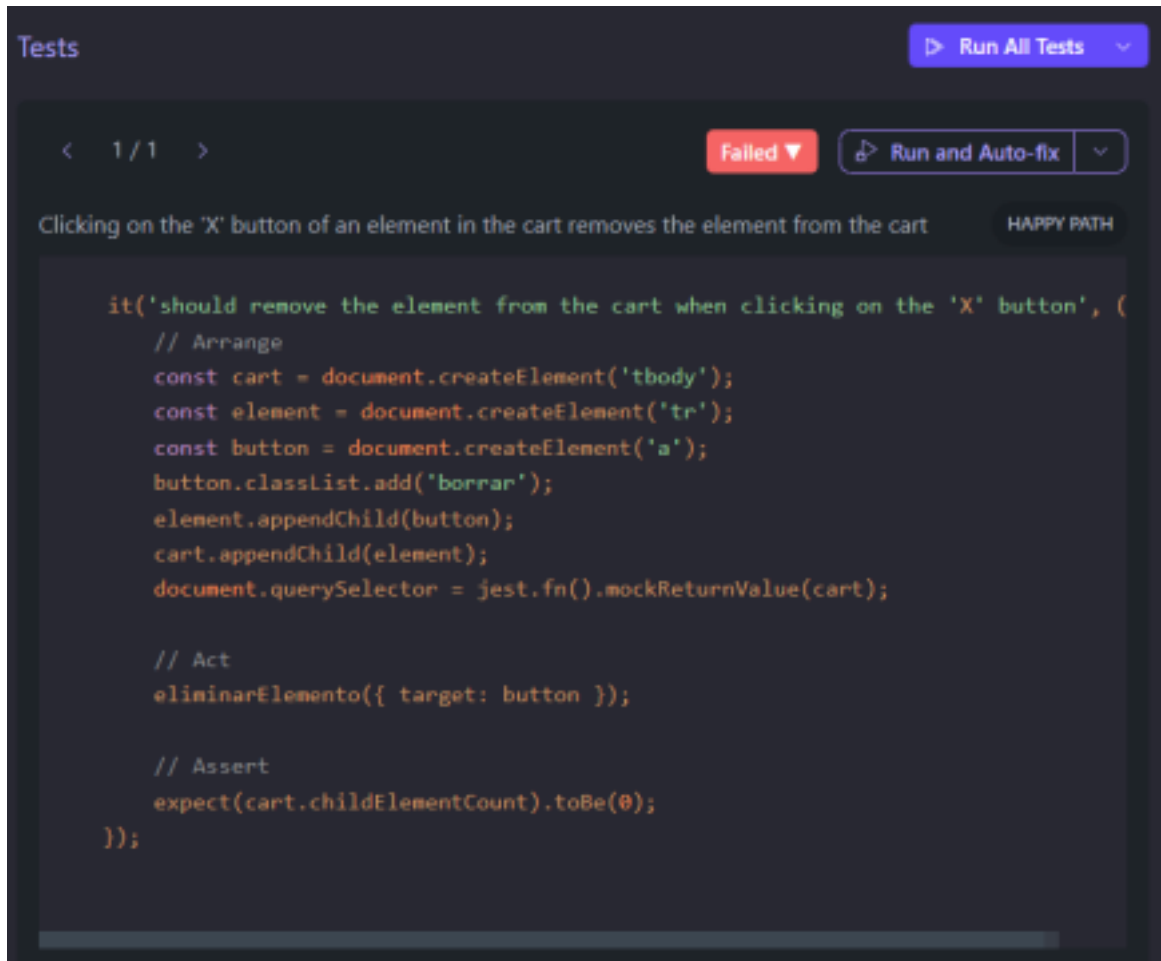
Automatización de Pruebas

La automatización de pruebas acelera el proceso de testing y facilita la detección de errores

Herramientas y Frameworks

- **JUnit:** Herramienta de testing de unidad para Java.
- **Selenium:** Herramienta de testing de integración para aplicaciones web. ●
- Cucumber:** Herramienta de testing de aceptación para aplicaciones web.

Casos de Uso y Ejemplos



The screenshot shows a Jest test runner interface. At the top, there's a 'Tests' tab and a 'Run All Tests' button. Below that, a navigation bar shows '< 1 / 1 >' and a 'Failed' status with a dropdown arrow. To the right of the status is a 'Run and Auto-fix' button. The main area displays the test description: 'Clicking on the 'X' button of an element in the cart removes the element from the cart' with a 'HAPPY PATH' label. The test code is shown in a dark-themed editor with syntax highlighting. The code is as follows:

```
it('should remove the element from the cart when clicking on the 'X' button', () {
  // Arrange
  const cart = document.createElement('tbody');
  const element = document.createElement('tr');
  const button = document.createElement('a');
  button.classList.add('borrar');
  element.appendChild(button);
  cart.appendChild(element);
  document.querySelector = jest.fn().mockReturnValue(cart);

  // Act
  eliminarElemento({ target: button });

  // Assert
  expect(cart.childElementCount).toBe(0);
});
```

Conclusión

Creo que el testing y las pruebas son una parte muy importante de la programación, ya que garantizan la calidad del software, creo que esto es una parte fundamental de cuando desarrollamos una aplicación. Esto nos sirve para detectar errores y poder solucionarlos fácilmente, también para validar que lo que hemos hecho esté correcto y así poder mejorar el rendimiento del software.

Referencias

Hiberus. (2023, 28 noviembre). *Las mejores herramientas para cada tipo de testing*. Blog de hiberus.

<https://www.hiberus.com/crecemos-contigo/las-mejores-herramientas-para-cada-tipo-de-testing/>

De Informática Profesional SA, S. (2023, 17 febrero). *TDD vs BDD vs ATDD*.

<https://www.linkedin.com/pulse/tdd-vs-bdd-atdd-servicios-de-informatica-profesion/?originalSubdomain=es>

Atlassian. (s. f.). *Los distintos tipos de pruebas en software* | Atlassian.

<https://www.atlassian.com/es/continuous-delivery/software-testing/types-of-software-testing>