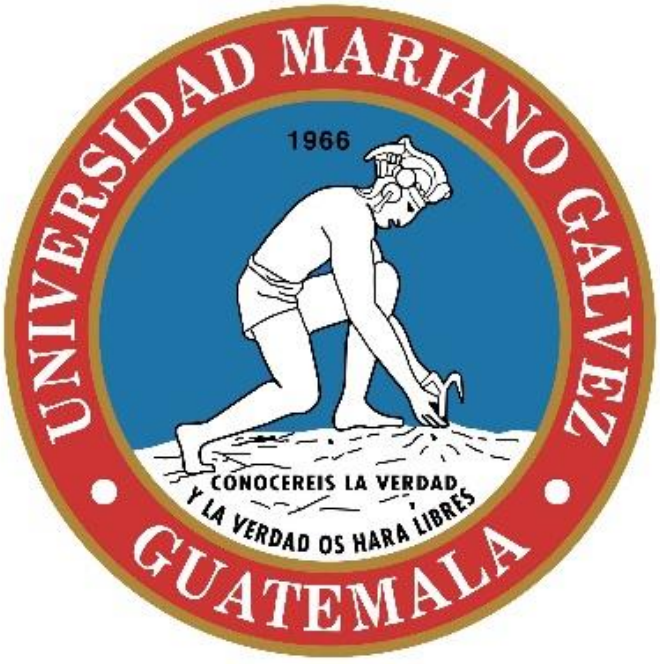
JEST



Marvin Orellana 1890-17-2063 Julio Rodas 1890-17-23483 Darling Catalán 1890-17-19455 Jasson Herrera 1890-17-2264 Samuel Ortiz 1890-17-20405

UNIVERSIDAD MARIANO GALVEZ DE GUATEMALA 28-08-2021

Que es Jest

Jest es un framework creado por Facebook y mantenido por la comunidad con apoyo de Facebook. Es fácil de instalar y no requiere de una configuración muy compleja para poder añadirlo en nuestros proyectos. Jest es una librería que nos permite escribir y ejecutar tests, es desarrollada por Facebook y usada por plataformas como airbnb, twitter, spotify, resuelve, etc.

Lo mejor de Jest es que funciona de inmediato. Simplemente instálelo a través de npm o yarn, escriba su prueba y ejecute jest. Es así de simple. Si desea cambiar las configuraciones, Jest le permite hacerlo en el package.json con muchas opciones de configuración.

Jest ha ganado mucha popularidad en los últimos años para las pruebas tanto de front-end como de back-end. Muchas grandes empresas, incluidas Twitter, Instagram, Pinterest y Airbnb, utilizan Jest para las pruebas de React.

¿Qué son las pruebas Unitarias?

La prueba unitaria es una prueba de software en la que se prueban unidades individuales (componentes) de un software. El propósito de las pruebas unitarias es validar que cada unidad del software funcione según lo diseñado. Una unidad es la parte comprobable más pequeña de cualquier software.

Las pruebas unitarias comprueban casos estándares (suposición explícita) es decir, no son perfectas.

Las características de las pruebas unitarias son:

- **Automatizable:** Deben funcionar sin ningún proceso manual.
- **Total, Cobertura:** Debemos de pasar por cada bloque escrito.
- **Reutilizables:** Podemos usarlas para probar otros bloques.
- **Independientes:** No pueden depender de otra prueba para funcionar.

- **Rápidas de crear:** Deben de ser algo conciso y que prueben algo muy particular.

Debemos tener en cuenta que realizando el testing lo que estamos asegurando es que nuestro código funcione correctamente, no que la lógica de negocio o el procedimiento sea correcto.

Ejecución de Pruebas

Jest ejecuta sus pruebas simultáneamente en paralelo, lo que proporciona una ejecución de prueba más suave y rápida. también proporciona pruebas de instantáneas. La prueba de instantáneas es una excelente manera de asegurarse de que su IU no cambie inesperadamente. Las instantáneas representan un componente de la interfaz de usuario, toman una captura de pantalla y comparan el componente con una imagen de referencia almacenada con la prueba. Las instantáneas hacen que probar los componentes básicos de la interfaz de usuario sea extremadamente simple con una línea de código: esperar (componente). toMatchSnapshot (); Hecho.

Incluso hay una herramienta CLI que puede utilizar desde la línea de comandos. Para dar un ejemplo, la herramienta CLI le permite ejecutar solo pruebas específicas que coincidan con un patrón. Además de eso, alberga mucha más funcionalidad, que puede encontrar en la documentación de CLI. Ejecución

En resumen, esto significa que Jest ofrece un ejecutor de pruebas, una biblioteca de afirmaciones, una herramienta CLI y un gran soporte para diferentes técnicas de burla. Todo esto lo convierte en un marco y no solo en una biblioteca.

Características

Configuración cero: "Jest tiene como objetivo funcionar desde el primer momento, sin configuración, en la mayoría de los proyectos de JavaScript". Esto significa que simplemente puede instalar Jest como una dependencia para su proyecto, y con ajustes mínimos o nulos, puede comenzar a escribir su primera prueba.

Aislado: el aislamiento es una propiedad muy importante al ejecutar pruebas. Garantiza que las diferentes pruebas no influyan en los resultados de las demás. Para Jest, las pruebas se ejecutan en

paralelo, cada una en su propio proceso. Esto significa que no pueden interferir con otras pruebas y Jest actúa como el orquestador que recopila los resultados de todos los procesos de prueba.

Instantáneas: las instantáneas son una característica clave para las pruebas de front-end porque le permiten verificar la integridad de los objetos grandes. Esto significa que no tiene que escribir pruebas grandes llenas de afirmaciones para verificar si todas las propiedades están presentes en un objeto y tienen el tipo correcto. Simplemente puede crear una instantánea y Jest hará la magia. Más adelante.

API enriquecido: Jest es conocido por tener una API enriquecida que ofrece muchos tipos de afirmaciones específicas para necesidades muy específicas. Además de eso, su excelente documentación debería ayudarlo a comenzar rápidamente.

Algunas de las Ventajas que creemos hacen a jest la mejor opción hoy, son:

- Casi cero configuraciones para comenzar a usarlo, depende de las cosas que uses en el proyecto
- Es extremadamente rápido, usar workers para paralelizar su ejecución
- Permite usar snapshops testing
- Los mensajes de error/feedback son muy claros
- Detecta y utiliza la configuración de babel
- Puedes tener el reporte del coverage sin necesidad de instalar algo más
- Es extendible, puedes crear tus matchers personalizados o incluso correr pruebas de otros lenguajes.
- Se pueden hacer tests de frontend y backend, corriendo en paralelo si tiene un monorepo
- Ofrece una herramienta CLI para controlar sus pruebas fácilmente
- Viene con un modo interactivo que ejecuta automáticamente todas las pruebas afectadas para los cambios de código que ha realizado en su última confirmación.
- Proporciona sintaxis para probar una sola prueba u omitir pruebas con. only y. skip. Esta función es útil al depurar pruebas individuales
- Proporciona una excelente documentación con muchos ejemplos y una comunidad de apoyo. Puede unirse a la comunidad de Jest a través de Discord o hacer preguntas en Stack Overflow.

- Facilita la Mocking a los desarrolladores, ya que es una de las cosas más dolorosas para los ingenieros de pruebas. Explicamos con más detalle en este post cómo funciona la burla de broma.

Términos de Jest

Existen dos de los términos Jest más utilizados que también se utilizan en otras herramientas de prueba: simulacro y espía.

Simulacro

De la documentación de Jest, podemos encontrar la siguiente descripción para un simulacro de Jest: “Las funciones simuladas facilitan la prueba de los enlaces entre el código borrando la implementación real de una función, capturando llamadas a la función (y los parámetros pasados en esas llamadas). "

Además, podemos usar un simulacro para devolver lo que queramos que devuelva. Esto es muy útil para probar todas las rutas en nuestra lógica porque podemos controlar si una función devuelve un valor correcto, un valor incorrecto o incluso arroja un error.

Espía

Funciones Mockeadas también son conocidos como "espías", porque permiten espiar el comportamiento de una función que se llama indirectamente por algunos otros códigos, en lugar de sólo la salida de la prueba. Puedes crear una falsa función con jest.fn(). Si no se da la implementación, la función mockeda devolverá undefined cuando se invoque.

Conclusiones

- Jest es una herramienta muy útil en el mejoramiento de la calidad de software, gracias a que integra muchas herramientas en una sola.

- Utilizando en muchas empresas mundialmente para la ejecución de sus pruebas.

- Fomenta el uso de buenas prácticas de programación y mejora el rendimiento, gracias a que utiliza en sus bases de datos sugerencias y reglas de buena programación.

Egrafia

https://www.testim.io/blog/jest-testing-a-helpful-introductory-tutorial/

https://www.valentinog.com/blog/jest/

https://yeisondaza.com/configurar-jest