Universidad de Granada

Facultad de Ingeniería Informática

Practica 3: Pruebas software

UNIVERSIDAD DE GRANADA

DS: Grupo 1.7

Emanuel Giraldo Herrera
Thomas Lang
Timur Sorokin
Alejando Iborra Morán

(2023-2024)

Contents

1 Planificación

1.1 "Creador de personajes"

- **Test 1**: CFG de ClaseBuilder
  Propósito: Verificar que la función loadCFG() de ClaseBuilder funciona correctamente.
- **Test 2**: CFG de PersonajeBuilder
  Propósito: Confirmar que la función loadCFG() de PersonajeBuilder funciona correctamente.
- **Test 3**: Creación correcta del personaje
  Propósito: Asegurar que la creación de un personaje utilizando PersonajeBuilder se realiza correctamente.
- **Test 4**: Exportación Correcta del personaje
  Propósito: Verificar que la exportación del personaje se realiza correctamente.
- **Test 5**: Funcionamiento del director
  Propósito: Confirmar que el director puede crear un personaje correctamente.
- **Test 6**: Funcionamiento de la fachada
  Propósito: Verificar que la fachada funciona como se espera.

1.2 "Gestor de personajes"

- **Test 1**: Añadir Personaje
  Propósito: Confirmar que se pueden agregar personajes a la lista del gestor.
- **Test 2**: Eliminar Personaje
  Propósito: Verificar que se pueden eliminar personajes de la lista del gestor.
- **Test 3**: Operaciones sobre la lista
  Propósito: Confirmar que las operaciones sobre la lista de personajes funcionan correctamente.

2 Análisis

2.1 CFG de ClaseBuilder

Elemento a Probar:

La función loadCFG() de la clase ClaseBuilder.

Condiciones para la Prueba:

Se requiere que exista un archivo de configuración válido para la clase en cuestión. La función loadCFG() debe cargar correctamente los valores del archivo de configuración.

Datos Necesarios para la Prueba: Un archivo de configuración válido que contenga los atributos de la clase con sus respectivos valores. Valores esperados para cada atributo de la clase según el archivo de configuración.

2.2 CFG de PersonajeBuilder

Elemento a Probar:

La función loadCFG() de la clase PersonajeBuilder.

Condiciones para la Prueba:

Se requiere que exista un archivo de configuración válido para el personaje en cuestión. La función loadCFG() debe cargar correctamente los valores del archivo de configuración.

Datos Necesarios para la Prueba:

Un archivo de configuración válido que contenga los atributos del personaje con sus respectivos valores. Valores esperados para cada atributo del personaje según el archivo de configuración.

2.3 Creación correcta del personaje

Elemento a Probar:

El proceso de creación de un personaje utilizando PersonajeBuilder.

Condiciones para la Prueba: Se deben cargar correctamente los valores de los archivos de configuración de la clase y el personaje. Los atributos del personaje creado deben coincidir con los valores esperados según las configuraciones de clase y personaje.

Datos Necesarios para la Prueba: Archivos de configuración válidos para la clase y el personaje. Valores esperados para los atributos del personaje basados en las configuraciones de clase y personaje.

2.4 Exportación Correcta del personaje

Elemento a Probar:

El proceso de exportación de un personaje.

Condiciones para la Prueba:

El personaje debe haber sido creado correctamente. La función de exportación debe generar un archivo con la estructura correcta y los datos del personaje.

Datos Necesarios para la Prueba:

Un personaje creado y válido. Valores esperados para los atributos del personaje en el archivo de exportación.

2.5 Funcionamiento del director

Elemento a Probar:

El proceso de creación de un personaje a través del director.

Condiciones para la Prueba: El director debe ser capaz de crear un personaje utilizando un constructor específico.

Datos Necesarios para la Prueba:

Un constructor válido para el tipo de personaje deseado. Valores esperados para los atributos del personaje creado.

2.6 Funcionamiento de la fachada

Elemento a Probar: El funcionamiento de la fachada en una operación específica.

Condiciones para la Prueba:

La fachada debe ser capaz de realizar la operación deseada correctamente.

Datos Necesarios para la Prueba:

Datos de entrada necesarios para la operación. Valores esperados para el resultado de la operación.

3 Diseño

3.1 Creador de personajes

3.1.1 Grupo: Archivos de configuración

TLDR: Estos casos de prueba aseguran que los archivos de configuración necesarios para construir personajes estén presentes en el sistema y puedan ser accedidos correctamente por el programa.

- **Casos de prueba**
  - Configuración de clase: verificar que los archivos de configuración de clase existen en la ruta especificada
  - Configuración de raza: verificar que los archivos de configuración de raza existen en la ruta especificada
- **Entorno de prueba**
  No se requieren herramientas ni infraestructuras adicionales.
- **Datos de prueba**
  - Se requiere ClaseBuilder debidamente inicializado.
  - Se requiere PersonajeBuidler debidamente inicializado.
  - Se requieren rutas donde se ubican los archivos de configuración.
- **Condición base**
  Los archivos lib/cfg/* existen y son accesibles.

3.1.2 Grupo: ClaseBuilder CFG

TLDR: Verifica que la función `loadCFG()` de ClaseBuilder carga correctamente los atributos de clase.

- **Casos de prueba:**
  - Lectura de atributos de clase: verificar que la función `loadCFG()` de ClaseBuilder funciona correctamente y carga los atributos de clase esperados.
- **Entorno de prueba:**
  - No se requieren herramientas ni infraestructuras adicionales.

- **Datos de prueba:**
  - Se requiere ClaseBuilder debidamente inicializado.
- **Condición base:**
  - ClaseBuilder está inicializado correctamente y ha cargado los valores desde el archivo de configuración correctamente.

3.1.3 Grupo: PersonajeBuilder CFG

TLDR: Verifica que la función `loadCFG()` de PersonajeBuilder carga correctamente los atributos de personaje.

- **Casos de prueba:**
  - Lectura de atributos de personaje: verificar que la función `loadCFG()` de PersonajeBuilder funciona correctamente y carga los atributos de personaje esperados.
- **Entorno de prueba:**
  - No se requieren herramientas ni infraestructuras adicionales.
- **Datos de prueba:**
  - Se requiere PersonajeBuilder debidamente inicializado.
- **Condición base:**
  - PersonajeBuilder está inicializado correctamente y cha cargado los valores desde el archivo de configuración correctamente.

3.1.4 Grupo: Creación de Personaje

TLDR: Verifica que la creación de un personaje utilizando los constructores de ClaseBuilder y PersonajeBuilder produce un personaje con los atributos esperados.

- **Casos de prueba:**
  - Verificar que los atributos del personaje son los esperados después de su creación.
- **Entorno de prueba:**
  - No se requieren herramientas ni infraestructuras adicionales.
- **Datos de prueba:**
  - Se requieren ClaseBuilder y PersonajeBuilder debidamente inicializados.
- **Condición base:**
  - ClaseBuilder y PersonajeBuilder están inicializados correctamente y la construcción de los atributos en el builder da el mismo resultado que `loadCFG()`.

3.1.5 Grupo: Exportación del estado

TLDR: Verifica que la exportación del estado de un personaje se realiza correctamente y que se genera el archivo esperado.

- **Casos de prueba:**
  - Verificar que el archivo de exportación del estado del personaje se genera correctamente en la ubicación especificada.
- **Entorno de prueba:**

- No se requieren herramientas ni infraestructuras adicionales.

- **Datos de prueba:**
  - Se requiere un personaje correctamente construido y una ubicación de archivo de exportación válida.

- **Condición base:**
  - El personaje está correctamente construido y la ubicación de archivo de exportación es válida. Finalmente que el archivo efectivamente se ha creado.

3.1.6 Grupo: Funcionamiento Director

TLDR: Verifica que el director pueda crear un personaje correctamente y que el personaje creado tenga el nombre esperado.

- **Casos de prueba:**
  - Verificar que el director pueda crear un personaje con el nombre especificado.
  - Verificar que el nombre del personaje creado coincida con el nombre especificado.

- **Entorno de prueba:**
  - No se requieren herramientas ni infraestructuras adicionales.

- **Datos de prueba:**
  - Se requiere un director correctamente inicializado y un nombre de personaje válido.

- **Condición base:**
  - El director está correctamente inicializado y puede crear un personaje con el nombre especificado.

3.1.7 Grupo: Funcionamiento de la Fachada

TLDR: Verifica que la fachada pueda crear un personaje correctamente con los constructores proporcionados y que el personaje creado tenga el nombre esperado.

- **Casos de prueba:**
  - Verificar que la fachada pueda crear un personaje utilizando los constructores de ClaseBuilder y PersonajeBuilder.
  - Verificar que el nombre del personaje creado coincida con el nombre especificado.

- **Entorno de prueba:**
  - No se requieren herramientas ni infraestructuras adicionales.

- **Datos de prueba:**
  - Se requiere una fachada correctamente inicializada, un constructor de ClaseBuilder y PersonajeBuilder válidos, y un nombre de personaje válido.

- **Condición base:**
  - La fachada está correctamente inicializada y puede crear un personaje utilizando los constructores proporcionados.

3.2 Gestor personajes

Se encarga de verificar el funcionamiento adecuado del gestor de personajes en el sistema. Este gestor gestiona una la lista de personajes, permitiendo añadir, eliminar y realizar operaciones como ordenar y filtrar por diferentes criterios, como nombre, clase, raza o atributo.

3.2.1 Añadir Personaje

TLDR: Este caso de prueba verifica que el gestor de personajes pueda añadir correctamente un personaje a la lista de personajes.

- **Casos de prueba:**
  - Se añade un personaje a la lista del gestor de personajes.

  arduino
- **Entorno de prueba:**
  - Se necesita un gestor de personajes inicializado y un personaje válido.
- **Datos de prueba:**
  - Gestor de personajes correctamente inicializado.
  - Personaje válido.
- **Condición base:**
  - El gestor de personajes está correctamente inicializado y la lista de personajes está vacía.