Ejercicio Nivel 12
CupiBlog

Descripción global

El equipo Cupi2 desea construir una aplicación que permita manejar un sistema de Blogs llamado CupiBlog, un espacio que les permite a los usuarios crear artículos con contenidos de su interés y calificar artículos de otros usuarios

Objetivos

El objetivo de este ejercicio es que el estudiante comprenda y adquiera práctica en:

- El desarrollo de aplicaciones siguiendo un proceso incremental.
- El manejo de la persistencia de información en un manejador de base de datos.
- La comunicación entre programas a través de sockets.
- El desarrollo de interfaces gráficas interactivas.
- El desarrollo de pruebas unitarias en jUnit para las clases del ejercicio.

Los siguientes pasos conforman el plan sugerido para desarrollar el ejercicio. La idea es ir desarrollando y probando incrementalmente los métodos de las clases.

Preparación

Esta sección presenta una lista de chequeo de todas las tareas necesarias para la preparación del ejercicio. Por favor, revise que cada tarea haya sido completada **antes** de pasar a la siguiente sección de esta guía de trabajo.

1. Para conocer el funcionamiento esperado de la aplicación, descargue y/o ejecute el archivo demo del ejercicio que se encuentra en el siguiente enlace: https://cupi2.virtual.uniandes.edu.co/ejercicios-del-semestre-apo2/ejercicio-n12.

2. Descomprima este archivo e importe el proyecto llamado **n12_cupiBlog** en Eclipse. No olvide borrar el archivo comprimido para evitar problemas.

3. Lea el enunciado del problema disponible en: **n12_cupiBlog/docs/specs/Descripcion.pdf**.

4. Estudie el documento de requerimientos funcionales disponible en: **n12_cupiBlog/docs/specs/RequerimientosFuncionales.pdf**.

5. Estudie el documento de requerimientos no funcionales disponible en: **n12_cupiBlog/docs/specs/RequerimientosNoFuncionales.pdf**.

6. Estudie el modelo del mundo del cliente diseñado para este ejercicio. Este modelo se encuentra en: **n12_cupiBlog/docs/specs/ModeloConceptualCliente.jpg**. Identifique las clases, relaciones entre clases, constantes, atributos y métodos.

7. Estudie el modelo de la interfaz del cliente diseñado para este ejercicio. Este modelo se encuentra en: **n12_cupiBlog/docs/specs/ModeloInterfazCliente.jpg**.

8. Estudie los archivos de apoyo en: **n12_cupiBlog/data/**

    - **servidor.properties** (properties para la conexión con la base de datos).
    - **imagenes/** (carpeta que contiene las imágenes requeridas por la aplicación)

Parte1: Diseño del servidor

En este ejercicio usted deberá implementar completamente el servidor para un sistema de blog distribuido. Dado que hay un protocolo de comunicación definido, cualquier servidor que siga este mismo protocolo podría ser usado para que los clientes que están dados en el esqueleto se comuniquen entre ellos a través del servidor.

Tenga en cuenta los siguientes aspectos para la implementación del servidor:

- El servidor debe estar siempre esperando nuevas conexiones por parte de los clientes.
- El servidor tiene que efectuar varias tareas simultáneamente: lo recomendable es que usted tenga un hilo en el cual se reciben las conexiones de los clientes (hilo principal) y un hilo más por cada cliente conectado. Este último hilo se encarga de recibir los mensajes enviados por el cliente.
- El servidor debe almacenar en una base de datos la información de los usuarios. Es necesario tener una tabla con los datos de los usuarios.
- El servidor no debe hacer manejo de errores (recuperación en caso de error). Los posibles errores que se pueden presentar, son debidos a problemas de comunicación. Por ejemplo, si un cliente interrumpe su ejecución de manera inesperada o si la conexión falla. En este caso, el servidor debe cerrar la sesión del usuario y continuar disponible para los otros jugadores
- En la página de cupi2 se encuentra el ejemplo de la Batalla Naval, que podrá servirle de guía para el uso de sockets y de la base de datos.

Parte2: Diseño de pruebas

Diseñe las pruebas que va a construir para verificar la implementación del servidor. No se espera (para la evaluación del ejercicio) que construya pruebas automáticas para todos los aspectos del

servidor, pero usted debe implementar las que le sean de mayor utilidad para apoyar la implementación de su proyecto. Revise las pruebas del cliente.

Parte3: Creación de esqueletos del servidor

Cree los esqueletos de las clases del servidor. Esto es, creación de las clases con declaración de constantes, atributos y signaturas de los métodos. Recuerde documentar a medida que incluye nuevos elementos en los esqueletos.

Parte4: Implementación de pruebas de servidor

Cree las clases de pruebas del servidor que definió en la parte 2 de este enunciado. Estas pruebas aún no pueden ser usadas porque el servidor solo contiene los esqueletos de las clases (no los métodos implementados).

Parte5: Implementación del servidor

Complete los esqueletos de las clases del servidor definidas en la parte 3 de este enunciado.

Validación

1. Ejecute las pruebas que diseñó para el servidor.
2. Interactúe con la aplicación y verifique su correcto funcionamiento.

Entrega

1. Indente el código fuente de todas las clases del mundo. En el siguiente enlace https://youtu.be/cwQ9QiauaSc encuentra un video que explica cómo indentar el código fuente de su ejercicio.

2. Limpie el proyecto para que la entrega no contenga archivos ejecutables ni temporales (https://youtu.be/mbcpY46wXS0).

3. Construya el archivo entregable con su ejercicio desarrollado y validado completamente. En el siguiente video https://youtu.be/xuSDFfEZW78 se explica detalladamente el proceso para producir el comprimido del ejercicio y enviarlo a SicuaPlus. Renombre el archivo a entregar con su login de la siguiente forma:

n<*nivel del ejercicio*>_<*login estudiante*>**.zip**
Por ejemplo: **n12_tsuarez.zip**

La no indentación del código fuente o el nombramiento incorrecto del ejercicio en su entrega es una acción penalizada en la plantilla de calificación del mismo.

4. Entregue el archivo del ejercicio vía SicuaPlus, de acuerdo con las normas, fecha y hora de entrega.