PONTIFICIA UNIVERSIDAD CATÓLICA DE CHILE
ESCUELA DE INGENIERÍA
DEPARTAMENTO DE CIENCIA DE LA COMPUTACIÓN

IIC2233 - Programación Avanzada
1º semestre 2015

Tarea 6

iChat DCC

1. Objetivos

- Aplicar conceptos de Networking para sincronización de programas en red.
- Interfaz gráfica para la implementación de un sistema de mensajería.
- Aplicar conocimiento de Threads en un entorno de mensajería.
- Envío de archivos a través de sockets.
- Implementación de una capa de seguridad para la protección de los usuarios del programa.
- Uso regex para la validacion y busqueda de patrones en strings.
- Serialización para guardar estados del programa.
- Interacción con un API conocido para realizar algunos servicios web.

2. Problema

En esta tarea deberá implementar un sistema de mensajería llamado iChat DCC que quizás le sea un poco difícil de concebir, ya que no existe ninguno igual tal que permita la comunicación personal y grupal. Además este sistema debe admitir el registro de nuevos usuarios, la validación y certificación de éstos al momento del login.

3. Características del chat

3.1. Usuario

Cada usuario debe contar con un nombre y un color[1] específico de texto, que lo identifique del resto. Por lo que no pueden haber dos usuarios con el mismo color o nombre. Cada vez que un usuario se conecte o desconecte, el programa deberá de tener la capacidad de seguir funcionando, ergo su programa deberá tener un buen manejo en las conexiones.

[1] no necesariamente a su elección

3.2. Lobby

El lobby es el menú principal que verá el usuario al conectarse a su programa. En éste podrá ver a todas las personas conectas al servidor, además de visualizar las conversaciones activas en las que participa. Cada vez que el usuario reciba algún mensaje se deberá indicar en el lobby que tiene mensajes sin leer, y el número de éstos (queda a su criterio si muestra el último mensaje recibido). El usuario deberá poder contar con la posibilidad de crear dos tipos de chat con el resto de los usuarios, además de poder navegar desde el lobby a algún chat, y viceversa, sin perder la conexión del chat. Los tipos chat se detallan a continuación:

- **Chat Grupales**: los chats grupales deberán tener un nombre designado por el usuario que lo creó, además solo podrán ingresar usuarios que han sido invitados por el creador del chat. Cada usuario, a excepción del creador, tendrá la opción de abandonar el chat (al hacerlo notificará su salida a los demás participantes).
- **Chat Individual**: el chat individual generará una conexión entre dos usuarios, los cuales pueden abandonar el chat en cualquier momento. El usuario que ha sido abandonado podrá continuar visualizando el chat, sin embargo no podrá comunicarse con el mismo usuario a menos de que cree una nueva conexión.

4. Servidor

4.1. Sign up & Login

Para poder usar el servicio, cada cliente deberá conectarse a su usuario. Para esto deberán poder crear una cuenta con un mail de la PUC. Esto implica que el mail utilizado debe ser de la forma *usuario@uc.cl* o bien *usuario@puc.cl*. El cliente deberá ser capaz de crear una cuenta que posea un nombre de usuario a su elección (que aún no haya sido utilizado) y una clave de seguridad. Es de suma importancia que usted **jamás** guarade la clave que el usuario decida usar, es por esto que usted deberá utilizar el protocolo HASH + SALT para guardar la información de los usuarios en algún archivo.

En el caso de que el cliente ya tenga una cuenta, este podrá acceder al servicio ingresando su usuario, o su mail, y la clave asociada a la cuenta. Al hacer un login exitoso el usuario entrará al lobby, en donde podrá visualizar todos los **Chat Grupales** que quedaron activos de su última sesión.

4.2. Sign Out

Al momento de generar un sign out del sistema de mensajería usted deberá guardar los **Chat Grupales** en que el usuario es partícipe para su próximo login, sin embargo no es necesario que guarde las conversación de estos.

5. Funcionalidades del Chat

5.1. Archivos

En cada uno de los chat descritos anteriormente se debe contar con la posibilidad de enviar imágenes y archivos de texto. Al enviar imágenes, se deberá ver una preview de la imagen, la cual debe estar integrada al resto de los mensajes del chat correspondiente. [2]

[2]Ayuda: Para poder generar las funcionalidades pedidas usted deberá generar un sistema de codificación para que sean interpretados por el servidor.

5.2. Imágenes de Google

Dentro de los chats, si un usuario decide enviar un mensaje que posee un substring envuelto por `$$`, los usuarios NO verán ese substring sino que un resultado a la busqueda de imágenes de Google con ese substring.
Por ejemplo, si un usuario decide enviar::

```
Me encanta $$python language logo$$, es el mejor lenguaje!
```

en el chat se deberá ver algo como

Me encanta 

, es el mejor lenguaje!

6. Regex

Algunos de los requerimientos de esta tarea involucran el chequeo de strings. Es **obligatorio** que todos estos chequeos se realicen mediante el uso expresiones regulares y no mediante el simple uso de los métodos de la clase `str`. Estos chequeos involucran los requerimientos nombrados en las secciones `4.1` y `5.2`.

7. Bonus

Tome en consideración que puede optar hasta un máximo de 15 % de bonus.

- **5 % (NUEVO)** No utilizar la librería `select`. Este bonus se aplica sólo si se obtiene una nota igual o superior a 4.
- **5 %** Avatar por usuario, elegible por ellos, teniendo uno por defecto. Este avatar se verá en el chat cada vez que el usuario envíe un mensaje (tipo facebook chat).
- **10 %** El administrador del **Chat Grupal** tendrá la capacidad de eliminar miembros del grupo y hacer administrador a otros usuarios. Los administradores no pueden eliminarse entre ellos. En caso de salirse el administrador y de no existir otro en el grupo, la administración pasara al miembro con mayor antigüedad.
- **5 %** Perfil de usuarios: el cliente puede ser capaz de clickear sobre los usuarios en el lobby y ver su perfil: nombre de usuario, foto elegida y grupos que comparten.

8. Restricciones y alcances

- Tu programa debe ser desarrollado en Python 3.4
- Su código debe seguir la guía de estilos `PEP8`
- Si no se encuentra especificado en el enunciado, asuma que el uso de cualquier librería Python está prohibido. Pregunte por foro si se pueden usar librerías específicas.
- Todo error debe ser controlado, para que el programa no se caiga.
- Debe usar la librería `PyQt4` para realizar la tarea.
- El ayudante puede castigar el puntaje[3] de tu tarea, si le parece adecuado. Se recomienda ordenar el código y ser lo más claro y eficiente posible en la creación algoritmos.

[3] Hasta −5 décimas.

- Debe adjuntar un archivo `README.md` donde comente sus alcances y el funcionamiento de su sistema (*i.e.* manual de usuario) de forma *concisa* y *clara*.
- Cree un módulo para cada conjunto de clases. Divídalas por las relaciones y los tipos que poseen en común.
- Cualquier aspecto no especificado queda a su criterio, siempre que no pase por encima de ningún otro.

9. Entrega

- **Fecha/hora:** Viernes 26 de Junio de 2015 / 23:59.
- **Lugar:** GIT - Carpeta: Tareas/Tarea_06

Tareas que no cumplan con las restricciones señaladas en este enunciado tendrán la calificación mínima (1.0).