PONTIFICIA UNIVERSIDAD CATÓLICA DE CHILE
ESCUELA DE INGENIERÍA
DEPARTAMENTO DE CIENCIA DE LA COMPUTACIÓN
IIC2233 - PROGRAMACIÓN AVANZADA

Actividad 13

1^o semestre 2018
14 de junio

Networking

Introducción

Todas las tardes después de recolectar actividades, la reina Barrios llega a su castillo a jugar *PrograHotel*. El malvado Devil Kawas, al enterarse de esto, compró todas las acciones de este masivo juego e implementó un sistema de suscripción mensual para poder jugar. Enfurecida por esta tarifa, Florencia se puso un nuevo objetivo de vida: crear el nuevo *PrograHotel* gratuito.

Instrucciones

Deberán desarrollar una arquitectura cliente-servidor que haga funcionar el hotel.

Funcionamiento de *PrograHotel*

El hotel consiste en una sola habitación representada por la interfaz que viene incluida con este enunciado (ver figura 1). Esta trae los métodos suficientes para mostrar y mover un personaje en 4 direcciones, presionando las flechas del teclado. Una descripción más técnica de cada método puede ser encontrada en el *docstring* de cada función. Para garantizar que todos los usuarios puedan disfrutar equitativamente de *PrograHotel*, todos deben ver lo mismo en sus pantallas y por lo tanto no se permite modificar nada de la interfaz gráfica entregada.

Figura 1: Interfaz de *PrograHotel* con un personaje.

Tu trabajo será conectar esta interfaz a la estructura cliente-servidor que vas a crear.

Implementación

Para la estructura cliente-servidor, se debe seguir el protocolo especificado en este enunciado.

Cada vez que el **cliente** desea hacer algo, manda un diccionario serializado con la librería `json`, y no hace nada hasta que el servidor le responda. El diccionario está estructurado de la siguiente forma:

- Cuando el cliente se conecta al servidor, debe mandar un mensaje con la estructura {`'orden':'crear'`}. Luego tiene que esperar la respuesta del servidor para saber en qué posición crear al personaje.
- Al presionar una tecla de flecha para hacer un movimiento, el cliente envía al servidor un mensaje con la siguiente estructura: {`'orden': 'mover', 'posicion':(x, y)`} donde `x` e `y` indican los puntos de la nueva posición del personaje. En el archivo `Client.py` el método `pedir_mover` es llamado cuando se presiona una tecla de flecha, el cual recibe la nueva posición `x`, `y`. El cliente queda esperando la respuesta del servidor para desplazar al personaje.
- Finalmente, cuando el cliente desea desconectarse, debe enviar el mensaje {`'orden':'desconectar'`}. En el archivo `Client.py` el método `desconectar` es llamado cuando se cierra la ventana del juego. En este método deben enviar el mensaje al servidor indicando que se van a desconectar.

El **servidor**, al recibir estas órdenes, debe manejarlas y responder con la siguiente estructura:

- Al recibir la orden `crear`, debe responder con {`'orden': 'crear', 'posicion':(x, y)`}, donde las coordenadas `x` e `y` de la posición deben escogerse al azar entre 0 y 500 para cada dimensión . Al recibir la respuesta, el cliente deberá crear a su entidad en esa misma posición de su ventana.
- Al recibir la orden `mover`, debe responder con {`'orden': 'mover', 'posicion':(x, y)`}, donde `x` e `y` son las coordenadas respectivas de la nueva posición. El cliente no se mueve hasta que recibe la respuesta del servidor.

El servidor sólo debe permitir a un cliente conectado a la vez. Además, debes considerar lo siguiente:

- La comunicación debe realizarse sobre TCP.
- Los diccionarios deben ser enviados serializados con `json`.

Métodos de la interfaz

En el archivo `ventana_principal.py` se encuentra la clase `MiVentana`, que deberás usar en el cliente para desplegar la interfaz gráfica. Esta clase recibe dos funciones: la primera recibe dos parámetros y será llamada cada vez que su personaje se mueva, la segunda no recibe parámetros y será llamada cuando se cierre la ventana. El archivo `Client.py` instancia `MiVentana` y le entrega las funciones `pedir_mover`, que recibe la posición nueva `(x, y)`, y `desconectar` que no recibe parámetros. Estas funciones deben ser desarrolladas por ti para que el cliente envíe la información necesaria al servidor.

Para interactuar con la ventana puedes usar los siguientes métodos:

- `agregar_mi_personaje(self, x, y)`: Agrega a la ventana el personaje. Recibe la posición inicial `(x, y)` en la que comienza el personaje.
- `actualizar_posicion_personaje(self, x, y)`: Actualiza la posición en la interfaz del personaje del cliente en la posición `(x, y)`.

Bonus

Como la reina Barrios quiere que *PrograHotel* sea rápido compró un computador último modelo para correrlo, pero es demasiado grande como para moverlo de un lugar a otro. Por esta razón, se te pide como requisito adicional poder conectarse a este desde otro equipo a través de una red local. Para ello la pareja debe mostrar 2 computadores distintos, uno con el servidor y otro con el cliente operando al mismo tiempo, es decir, ambos equipos deben comunicarse entre sí.

Notas

- Se espera que se implemente un servidor y un cliente (ambos deben poder funcionar independientemente y sin importarse entre sí).
- Modificar la interfaz entregada significará un descuento en la nota final.
- Se debe utilizar la librería `json` para el envío y recepción de la información.
- Podrán optar al bonus solo aquellos que tengan una nota superior o igual a 6,0.
- Para cobrar el bonus deben mostrar que funcione correctamente la conexión al cuerpo docente presente en la sala.

Requerimientos

- (3.00 pts) Servidor
  - (1.00 pt) El servidor existe de manera independiente y es capaz de conectarse de manera correcta con el cliente.
  - (0.50 pt) El servidor recibe y envía los mensajes con el cliente.
  - (0.50 pt) Se utiliza el protocolo especificado al enviar y recibir mensajes.
  - (1.00 pt) El servidor maneja la desconexión del cliente.
- (3.00 pts) Cliente
  - (1.00 pt) El cliente existe de manera independiente pudiendo recibir órdenes del servidor.
  - (1.00 pt) Se implementan de manera correcta las funciones de la interfaz.
  - (0.50 pt) El cliente recibe y envía los mensajes al servidor.
  - (0.50 pt) Se utiliza el protocolo especificado para el envío y recepción de información.
- (0.50 pts) Bonus
  - Conectar el cliente y el servidor en computadores distintos a través de internet.

Entrega

- **Lugar:** En su repositorio de GitHub en la **carpeta** `Actividades/AC13/`
- **Hora:** 16:40
- Si está trabajando en pareja, basta con que un miembro suba la actividad. Si se suben actividades distintas, se corregirá una de las dos al azar.