

Programación III

Laboratorio III

Trabajo práctico

Prof. Mariano Kaimakamian Carrau

Año: 2021



Índice

Tabla de revisión	2
Objetivo	2
Fechas de entrega y contenido	2
Requerimiento: funcionalidad y consideraciones	3
Kokumo no monogatari - reglas del juego	3
Consideraciones	4
Sugerencia	5
Links de interés	5

Tabla de revisión

Fecha	Agregado
03/05/2021	Creación del documento.
05/05/2021	Se clarifica el punto 1 del requerimiento y se incorpora el punto 4. También se revisan las reglas del juego para que sea más fácil de entender
08/05/2021	Se agrega un nuevo punto en el apartado de requerimientos.
16/05/2021	Se aclara que HttpRequest aparece en Java 11 y que HttpURLConnection está disponible para Java 8.

Objetivo

Aplicar los conocimientos teóricos adquiridos durante la cursada, en el desarrollo un video juego para dos personas, basado en turnos, implementado sobre una arquitectura del tipo cliente-servidor, empleando protocolo HTTP.

Será fundamental llevar adelante prácticas de investigación que permitan expandir el alcance resolutivo, actividad que resulta fundamental dentro de un contexto laboral.

Fechas de entrega y contenido


- **1era revisión:** 17/05/2021
 - Al menos debe entregarse el diagrama de clases.
- **2da revisión:** 31/05/2021
 - Diagrama de clases corregido y código acorde (repositorio git).
- **Final:** 14/06/2021
 - La entrega final, sobre la que se evaluará el trabajo, deberá contener.
 1. Enunciado
 2. Diagramas de clases
 3. Código fuente (repositorio git)
 4. Archivo JAR (el juego debe funcionar y ceñirse a las especificaciones)

Requerimiento: funcionalidad y consideraciones

1. El juego debe poder ser ejecutado en modo servidor o cliente, lo que implica que:
 - a. **Modo servidor:** el jugador creará una partida y deberá tener la posibilidad de invitar a un tercero o aguardar que éste envíe la solicitud de unión.
 - b. **Modo cliente:** deberá contar con la posibilidad de enviar una petición para unirse a una partida creada o de aguardar a ser invitado.
2. En cualquiera de los dos casos, es necesario conocer la ip y el puerto en el que sendos programas estarán escuchando; si bien el protocolo a utilizar es el http, se recomienda usar un puerto distinto al tradicional 80 u 8080.
3. La lógica de negocio deberá ejecutarse en el programa que oficie como servidor.
4. Tener presente que algunos procesos requerirán que el usuario aguarde unos instantes, motivo por el cual deberá ser informado oportunamente para que sepa cual es el estado del aplicativo.
5. El juego debe ser lo suficientemente robusto como para no cerrarse inesperadamente ante excepciones, de modo tal que si - por ejemplo - alguno de los dos extremos pierde la comunicación, el aplicativo pueda informarlo y dar por cancelada la partida.
6. Se recomienda el uso de la clase `HttpServer` (Java 8 & 11) y `HttpRequest` (Java 11) o `URLConnection` (Java 8) para realizar la comunicación entre aplicativos, aunque queda a criterio del desarrollador el uso de bibliotecas / librerías.
7. El formato de intercambio entre el cliente y el servidor debe ser Json.
8. Se deberá utilizar la terminal como interfaz de comunicación entre el usuario y el software, aunque queda a criterio del desarrollador incorporar una interfaz gráfica adicional.
9. El código debe estar disponible en algún repositorio git.
10. El código debe estar escrito en inglés; incluir comentarios cuando lo crean oportuno, como también incluir javadocs cuando sea posible (sobre todo para las funciones cuyos nombres no sean muy claros o no se pueda determinar qué realiza).

Kokumo no monogatari - reglas del juego

1. Al inicio de la partida, cada jugador deberá posicionar sus tres ninjas dentro de una cuadrícula de 5 x 5 casilleros.
2. En ningún momento el oponente deberá conocer las ubicaciones de las tropas enemigas; cada jugador verá dos cuadrículas:
 - a. La cuadrícula principal, en la que se hallan desplegados sus ninjas.

- 
- b. La cuadrícula secundaria, en la que se marcarán los ataques realizados contra su oponente.
 3. Una vez dispuestos los guerreros en el campo de batalla, se debe decidir qué jugador iniciará la partida para, luego, cederle el turno al contrincante (y así sucesivamente).
 4. En su turno, cada jugador podrá realizar una de las siguientes acciones por cada ninja vivo:
 - a. **Moverlo 1 casillero:** en cualquier dirección y con la limitante de no poder optar por esta maniobra en turnos consecutivos.
 - b. **Realizar un ataque:** indicar la coordenada del terreno enemigo, al cual dirigirá el ataque.
 5. Si el ataque del oponente es dirigido hacia una casilla ocupada por un ninja, éste pierde la vida; en cambio, si está despoblada, el terreno queda intransitable.
 6. La partida termina cuando uno de los dos jugadores elimina a la facción enemiga.

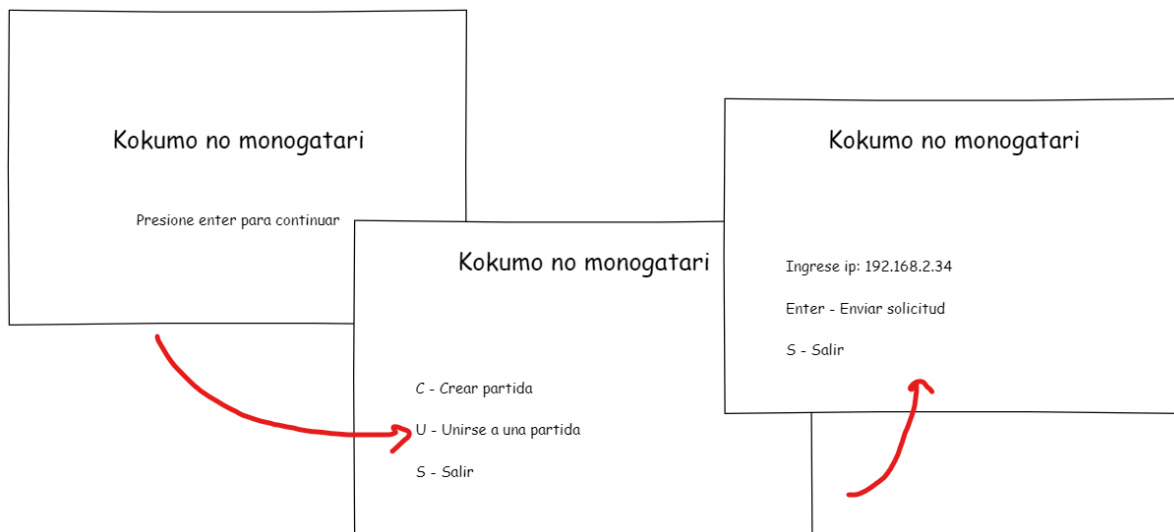
Consideraciones

- Uno de los tres ninjas será el comandante del escuadrón y poseerá dos vidas.
- Todo ninja puede moverse siempre y cuando el comandante siga con vida.
- Pueden crearse ambientes de combate con mayor complejidad; no necesariamente tienen que ser terrenos cuadrados y pueden incorporarse otro tipo de obstáculos como ríos, piedras y árboles, con distintos efectos defensivos u ofensivos (pero esto es opcional y se sugiere explorar esta alternativa cuando el juego base esté creado).
- El oponente nunca debe conocer la posición de los ninjas.

Sugerencia

Iniciar por el flujo de las ventanas; esto favorecerá conocer qué datos se necesitan, en qué momento, y cómo diseñar el comportamiento de navegación.

Por ejemplo...



Links de interés

<https://docs.oracle.com/javase/8/docs/jre/api/net/httpserver/spec/com/sun/net/httpserver/HttpServer.html>

<https://docs.oracle.com/javame/8.0/api/httpclient/api/com/oracle/httpclient/HttpRequest.html>

<https://es.wikipedia.org/wiki/JSON>