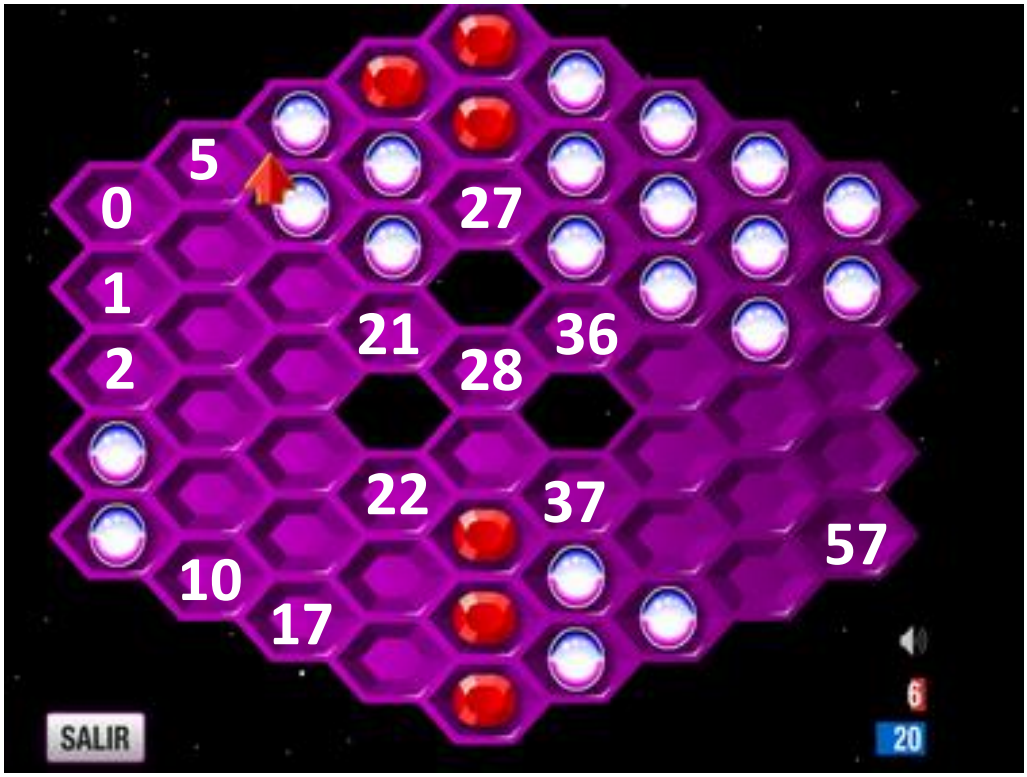
PROTOCOLO DE COMUNICACIÓN

Representación lógica del tablero

Las casillas en el tablero de juego estarán numeradas de forma ascendente a partir de la posición 0 en la esquina superior izquierda del tablero hasta la posición en la esquina inferior derecha. (Ver imagen).



NOTA: Esta es simplemente la numeración del tablero para estandarizar el protocolo, luego de su traducción en cada programa Ud. puede interpretar cada casilla como mejor le convenga.

Mensajes mediante TCP

- Puerto: El puerto a utilizar será el 5000.
- Formato: El formato de transmisión que tendrá la información será JSON (por ser un estándar de comunicación reconocido y por su simplicidad). Por favor revisar los siguientes enlaces que contienen información acerca del formato de serialización JSON, su implementación y otras cosas (por favor prestar atención a los tipos de datos y otras generalidades).
    1. http://www.json.org/json-es.html
    2. http://www.w3schools.com/js/js_json_intro.asp
    3. https://msdn.microsoft.com/es-es/library/bb412179(v=vs.110).aspx

Jesús A. Reyes V.
Preparador de Programación II, Semestre 2016-1
jesus.reyes@unet.edu.ve
@jesusareyesv en Telegram

4. http://stackoverflow.com/questions/3142495/deserialize-json-into-c-sharp-dynamic-object
5. http://stackoverflow.com/questions/6620165/how-can-i-parse-json-with-c
6. http://www.oracle.com/technetwork/articles/java/json-1973242.html

NOTA: La forma de implementar internamente los JSON es propia e independiente en cada lenguaje de programación. Para efectos de éste protocolo sólo nos interesa la forma serializada en bytes (o string) del objeto JSON. Para saber cómo implementa JSON su lenguaje de programación utilice el primer link de la lista de arriba, al final aparecen algunas APIs.

Los JSON contendrán 2 campos cuyas claves serán “tipo” y “nodos”.

- **“tipo”** será de tipo string con 2 valores posibles:
  - **“saltar”** -> que le indicará al programa que recibe que la gema a mover realizará un salto (revisar las reglas en el documento del proyecto).
  - **“clonar”** -> que indicará que la ficha indicada debe ser clonada.
- **“nodos”** será del tipo array con 2 valores internos **[Ni, Nf]**
  - **Ni** indicará el nodo en el que se encuentra la gema que se pretende manipular.
  - **Nf** indicará el nodo hacia el cuál se desplazará (o clonará) la gema posicionada en **Ni**.
  - **El orden de los nodos es importante, siempre será primero el inicial y luego el final.**

El formato estaría estructurado de la siguiente forma:

{

“tipo” : valor ,

“nodos” : [Ni, Nf]

}

Un ejemplo de mensaje para saltar de una posición a otra sería:

{

“tipo” : “saltar” ,

“nodos” : [3, 8]

}

Para clonar simplemente sería:

{

“tipo” : “clonar” ,

“nodos” : [12, 16]

}

NOTA: El protocolo fue diseñado para que fuera simple y cómodo. Las validaciones de movimientos y demás elementos concernientes a la lógica del programa (puntaje, fin de juego, etc.) deben ser realizadas por el programador.

Mensajes mediante UDP

Para cumplir con el requisito de que cada programa cliente podrá visualizar una lista de los servidores de juego disponibles se utilizará la comunicación un socket UDP con casting (específicamente multicast). Multicast es una forma de difundir información desde un solo socket UDP a muchos receptores, así que es propicio para esta tarea.

NOTA DE CONDOLENCIA: Siento no haber discutido este punto el viernes en la reunión, pero todavía no tenía claro si se implementaría o no.

Multicast funciona (de forma somera) asignando a los datagramas una dirección denominada dirección multicast, que define el grupo de computadores entre los cuales se intercambian los datagramas (grupo multicast).

- **Puerto:** para el socket UDP se usará 5001.
- **Formato de comunicación:** se usará JSON de nuevo pero con otros campos.

Los nuevos campos del JSON serán:

- **“hostname”** será de tipo string y contendrá el nombre del servidor. Ejemplo**: “TerminatorIA”.**
- **“ip”** será de tipo string y guardará la dirección IP de la PC que hospeda al servidor TCP de la partida.
- **“puerto”** será de tipo number y guardará el número de puerto en el que se encuentra escuchando el servidor TCP (por si cambia el bloqueo de puertos del laboratorio Valerio Wong durante la entrega).

El formato de los JSON sería:

{

“hostname” : “foo”,

“ip” : “bar”,

“puerto” : puerto

}

Un ejemplo sería:

{

“hostname” : “TerminatorIA”,

“ip” : “10.0.6.4”,

“puerto” : 5000

}

- **Dirección multicast:** para efectos del proyecto se usarán 2 direcciones de grupo: 230.0.0.0 o 230.0.4.20, si no encuentran hosts activos en una dentro de cierto periodo de tiempo (10 segundos) procederán a buscar en la siguiente y así sucesivamente.

NOTA: El viernes 28/10/16 habrá una preparaduría sobre UDP y subnetting. Por favor rellene el formulario para seleccionar el mejor horario https://goo.gl/forms/PbkDnHh0DrYUjgBp1

Consideracion final

Por favor respete los protocolos establecidos si desea que su programa pueda comunicarse con los de cualquier otra persona de ambas secciones (de lo contrario su código sufrirá de desorden del espectro autista).

Jesús A. Reyes V.
Preparador de Programación II, Semestre 2016-1
jesus.reyes@unet.edu.ve
@jesusareyesv en Telegram