



UNIVERSIDAD NACIONAL DE LA MATANZA

Departamento de Ingeniería e Investigaciones Tecnológicas

Programación Avanzada **Trabajo Final de Laboratorio**

ENTREGA 21/11/2015

GRUPO Nº 8

INTEGRANTES

- Di Giacomo, Gastón
- Blanco, Juan Manuel
- Martin, Gonzalo Javier
- Miranda, Cristian Nahuel

CURSO

Martes, Viernes y Sábado - Turno noche

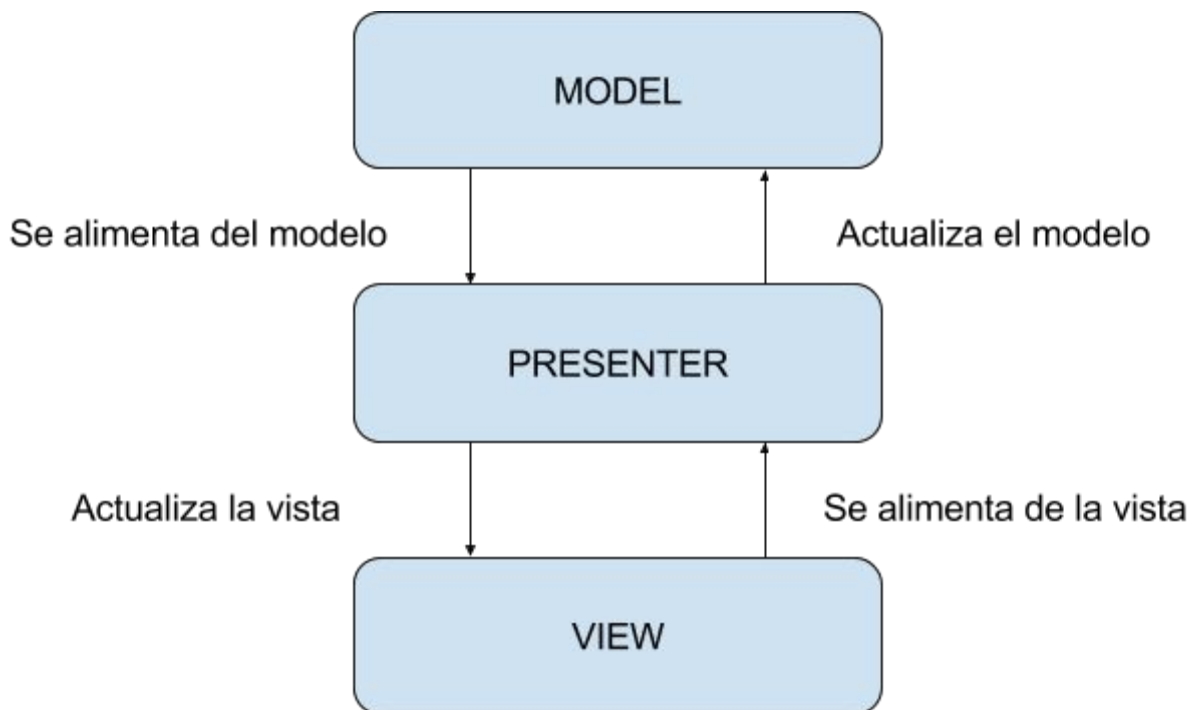
Análisis y Diseño

A continuación se describe la arquitectura elegida, componentes principales y mecanismos de comunicación.

Arquitectura

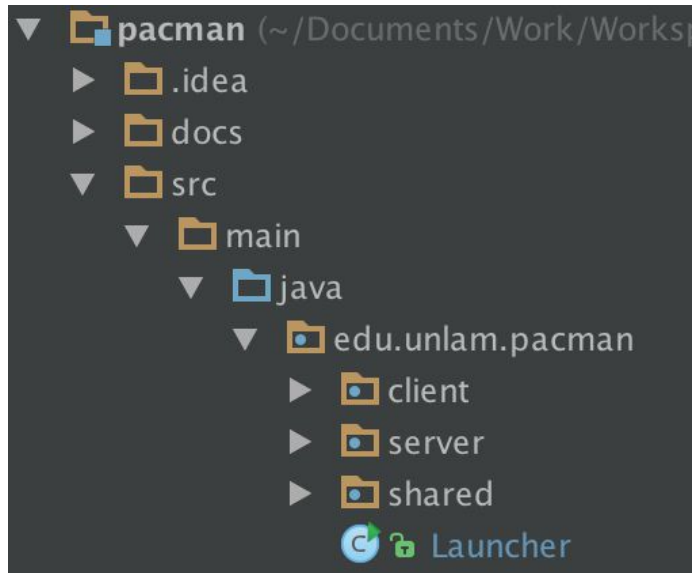
Se utiliza una variante al patrón de arquitectura MVC conocido como MVP (Model - View - Presenter). Toda la lógica se encuentra en los presenters, los cuales saben cómo actualizar la vista conociendo al modelo.

Este patrón facilita el unit testing de los presenters ya que estos últimos no contienen nada relacionado a la interfaz de usuario lo cuál facilita su abstracción.



Módulos

El proyecto se encuentra dividido en tres partes:



Un paquete **"client"** donde se encuentran todas las clases correspondientes al front-end de la aplicación. Esto incluye los módulos MVP.

Un paquete **"server"** que contiene los componentes del back-end de la aplicación (lógica de negocio y acceso a base de datos).

Un paquete **"shared"** donde se encuentran las clases compartidas entre client y server.

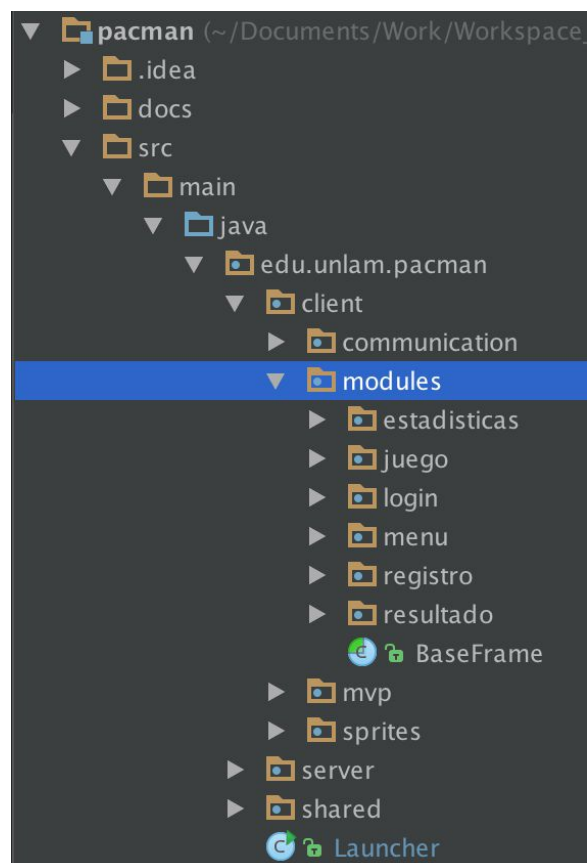
Client

En el paquete **"client"** existe un módulo para cada pantalla, el cuál está compuesto de una clase modelo, una vista y su correspondiente presenter.

Aquí también se encuentran las clases base del patrón MVP.

Server

En cuanto a servicios, el paquete **"server"** contiene una clase por cada uno de ellos las cuales encapsulan la lógica de negocio abstrayéndose de acceso a datos. Este último es efectuado por los DAO (Data Access Object) quienes son consumidos por los servicios y son los encargados de realizar las consultas a la base de datos.



Eventos

Para comunicar a los presenters entre sí utilizamos una biblioteca llamada [guava](#) que nos permite enviar mensajes entre distintos puntos conectados a través de un "event bus". Los presenters se encuentran registrados a este canal para escuchar los eventos que viajan a través de él.

El sistema que provee la biblioteca genera un "pool" de N hilos en estado inactivo que se van utilizando a medida que los presenters procesan los eventos recibidos.

Sockets

Se crea un socket por cada aplicación (dos para el cliente que actúa de servidor).

Se serializan los objetos utilizando [Gson](#) para generar una cadena de tipo JSON que pueda viajar a través de la red.

Base de Datos

Utilizamos MySQL como sistema de base de datos de tipo relacional.

La base de datos es identificada como **PACMAN**, la cual contiene una tabla llamada **JUGADORES** donde se almacenan los jugadores registrados.

Compilación

Utilizamos [Apache Maven](#) para compilar y empaquetar la aplicación. Se utiliza un archivo "pom.xml" donde se especifica información fundamental en la construcción del ejecutable tal como nombre, versión y recursos a incluir dentro del paquete.

Para compilar la aplicación se debe tener instalado Maven y ejecutar el siguiente comando: *mvn clean package*

Ejecución

El juego se encuentra empaquetado en un archivo de tipo "jar" el cual puede ser ejecutado en consola. No se requiere ninguna dependencia extra. Solamente se debe tener instalada la última versión de la JRE (versión 8 o posterior) y ejecutar el siguiente comando: *java -jar pacman.jar [server|client]*

Pruebas realizadas

Login

1. Login con usuario vacío.
*dado que el usuario se encuentra en la pantalla de Login
y el campo "usuario" se encuentra vacío
cuando hago click en el botón "Login"
entonces un mensaje indica "El nombre de usuario no puede ser vacío"*
2. Login con password vacía.
*dado que el usuario se encuentra en la pantalla de Login
y el campo "password" se encuentra vacío
cuando hago click en el botón "Login"
entonces un mensaje indica "La contraseña no puede ser vacía"*
3. Login con usuario y password inválidos.
*dado que el usuario se encuentra en la pantalla de Login
y el campo "usuario" contiene un username inexistente
y el campo "password" contiene una password inexistente
cuando hago click en el botón "Login"
entonces un mensaje indica "Jugador no existente"*
4. Login con usuario y password válidos.
*dado que el usuario se encuentra en la pantalla de Login
y el campo "usuario" contiene un username válido
y el campo "password" contiene una password válido
cuando hago click en el botón "Login"
entonces el usuario debe ser llevado al Menú principal*

Registro

1. Registro con usuario vacío.
*dado que el usuario se encuentra en la pantalla de Registro
y el campo "usuario" se encuentra vacío
cuando hago click en el botón "Registrar"
entonces un mensaje indica "El nombre de usuario no puede ser vacío"*
2. Registro con password vacía.
*dado que el usuario se encuentra en la pantalla de Registro
y el campo "password" se encuentra vacío
cuando hago click en el botón "Registrar"
entonces un mensaje indica "La contraseña no debe ser vacía"*

3. Registro con confirmación de password vacía.
*dado que el usuario se encuentra en la pantalla de Registro
y el campo "confirmar password" se encuentra vacío
cuando hago click en el botón "Registrar"
entonces un mensaje indica "Las contraseñas deben ser idénticas"*
4. Registro con password y confirmación de password no idénticas.
*dado que el usuario se encuentra en la pantalla de Registro
y el campo "password" y "confirmar password" no son iguales
cuando hago click en el botón "Registrar"
entonces un mensaje indica "Las contraseñas deben ser idénticas"*
5. Registro con datos válidos.
*dado que el usuario se encuentra en la pantalla de Registro
y todos los campos son válidos
cuando hago click en el botón "Registrar"
entonces un mensaje indica "Jugador registrado con éxito"
y el usuario vuelve a la pantalla de Login*

Menú Principal

1. Unirse a partida.
*dado que el usuario se encuentra en la pantalla de Menú Partido
cuando hago click en el botón "Unirse a Partida"
entonces el jugador se encuentra registrado para jugar
y el botón se vuelve inactivo*
2. Empezar partida con un número insuficiente de jugadores unidos.
*dado que el usuario ADMIN se encuentra en la pantalla de Menú Partido
y hay dos jugadores unidos a la partida
cuando hago click en el botón "Empezar partida"
entonces un mensaje indica "La partida solo puede comenzar cuando hay un mínimo
de 3 y un máximo de 5 jugadores en ella."*
3. Empezar partida con un número válido de jugadores unidos.
*dado que el usuario ADMIN se encuentra en la pantalla de Menú Partido
cuando hago click en el botón "Empezar partida"
entonces se inicia el juego
y el usuario es redirigido a la pantalla de Juego*

Construcción de componentes en el juego

1. Comprobar aparición de personajes en el tablero.
*dado que el usuario se encuentra jugando
cuando la pantalla de juego aparece
entonces los personajes deben aparecer en lugares separados sobre el tablero*

2. Comprobar construcción del tablero.
*dado que el usuario se encuentra jugando
cuando la pantalla de juego aparece
entonces el tablero debe ser visible*
3. Comprobar aparición de frutas y frutas especiales.
*dado que el usuario se encuentra jugando
cuando la pantalla de juego aparece
entonces las frutas comunes deben rellenar todo el espacio "vacío" del tablero
y una o más frutas especiales deben ser posicionadas en lugares aleatorios*
4. Comprobar aparición de puntaje debajo del tablero.
*dado que el usuario se encuentra jugando
cuando la pantalla de juego aparece
entonces el puntaje de cada jugador debe ser visible en la parte inferior del tablero*
5. Inicio de cronómetro.
*dado que el usuario se encuentra jugando
cuando la pantalla de juego aparece
entonces el cronómetro debe iniciar a descontar segundos*

Dinámica de juego

1. Comprobar lectura de teclas.
*dado que el usuario se encuentra jugando
cuando el usuario presiona las flechas del teclado
entonces el personaje debe responder con su movimiento correspondiente*
2. Movimiento de personaje frente a paredes
*dado que el usuario se encuentra jugando
cuando su personaje se encuentra frente a una pared
entonces este no debe avanzar*
3. Pacman comiendo frutas comunes.
*dado que el usuario se encuentra jugando
cuando pacman se mueve sobre frutas comunes
entonces estas deben desaparecer del tablero ya que pacman las comió*
4. Personajes comiendo frutas especiales.
*dado que el usuario se encuentra jugando
cuando su personaje se mueve sobre frutas especiales
entonces estas deben desaparecer del tablero ya que el personaje las comió*

5. Colisión entre Pacman y Fantasma (ambos en modo normal)
*dado que el usuario se encuentra jugando
cuando su personaje (pacman) colisiona con un fantasma
y ambos se encuentran en modo normal
entonces el fantasma debe comer al pacman
y sumar un punto en el puntaje correspondiente
y el pacman debe revivir en el lugar más alejado posible de los demás fantasmas*
6. Colisión entre Pacman (modo cazador) y Fantasma (modo normal)
*dado que el usuario se encuentra jugando
cuando su personaje (pacman) colisiona con un fantasma
y el pacman se encuentra en modo cazador
y el fantasma se encuentra en modo normal
entonces el pacman debe comer al fantasma
y sumar un punto en el puntaje correspondiente
y el fantasma debe revivir en el lugar más alejado posible de los demás personajes*
7. Colisión entre Pacman (modo normal) y Fantasma (modo cazador)
*dado que el usuario se encuentra jugando
cuando su personaje (pacman) colisiona con un fantasma
y el pacman se encuentra en modo normal
y el fantasma se encuentra en modo cazador
entonces el fantasma debe comer al pacman
y sumar un punto en el puntaje correspondiente
y el pacman debe revivir en el lugar más alejado posible de los demás fantasmas*
8. Colisión entre dos Fantasmas (ambos en modo normal)
*dado que el usuario se encuentra jugando
cuando su fantasma colisiona con otro fantasma
y ambos se encuentran en modo normal
entonces ambos fantasmas deben bloquearse un segundo sin poder moverse
y luego seguir la dirección que llevaban previamente*
9. Colisión entre Fantasma (modo cazador) y Fantasma (modo normal)
*dado que el usuario se encuentra jugando
cuando su fantasma colisiona con otro fantasma
y su personaje se encuentra en modo cazador
y el otro fantasma en modo normal
entonces su fantasma debe comer al otro fantasma
y sumar un punto en el puntaje correspondiente
y el fantasma debe revivir en el lugar más alejado posible de los demás personajes*

10.Actualización de puntajes.

*dado que el usuario se encuentra jugando
cuando su personaje como a otro
entonces el puntaje correspondiente a ese personaje debe aumentarse en uno*

11.Fin de partida por tiempo.

*dado que el usuario se encuentra jugando
cuando se acaba el tiempo del cronómetro
entonces el mensaje deberá ver una pantalla con el mensaje "FIN DE PARTIDA"*

12.Fin de partida porque no hay más frutas.

*dado que el usuario se encuentra jugando
cuando se acaban las frutas del tablero
entonces el mensaje deberá ver una pantalla con el mensaje "FIN DE PARTIDA"*

Pantallas



A screenshot of a 'Login' window. It features a title bar with three colored buttons (red, yellow, green) and the title 'Login'. The main area contains two input fields: 'Usuario:' and 'Contraseña:'. Below these fields are two buttons: 'Login' and 'Registrarse'.

Usuario:

Contraseña:

Login

Registrarse



A screenshot of a 'Login' window, similar to the first one but with an additional field. It has a title bar with three colored buttons and the title 'Login'. The main area contains three input fields: 'Usuario:', 'Contraseña:', and 'IP Servidor:'. The 'IP Servidor' field contains the text '127.0.0.1'. Below these fields are two buttons: 'Login' and 'Registrarse'.

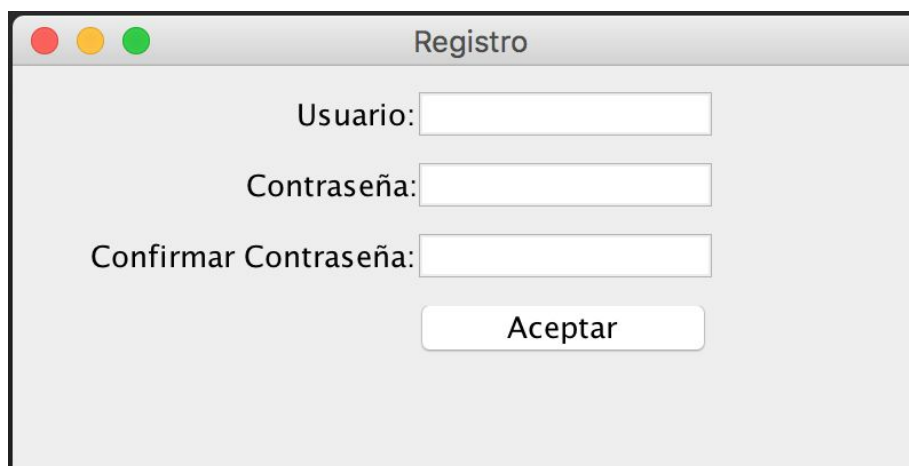
Usuario:

Contraseña:

IP Servidor:

Login

Registrarse



A screenshot of a 'Registro' (Registration) window. It features a title bar with three colored buttons and the title 'Registro'. The main area contains three input fields: 'Usuario:', 'Contraseña:', and 'Confirmar Contraseña:'. Below these fields is a single button: 'Aceptar'.

Usuario:

Contraseña:

Confirmar Contraseña:

Aceptar

