**PROGRAMACIÓN ORIENTADA A OBJETOS**

**INTERFAZ**

**2023-2**

**Laboratorio 5/6**

# **OBJETIVOS**

1. Desarrollar una mini aplicación gráfica considerando el patrón MVC.
2. Implementar el esquema de manejo de eventos con clases anónimas
3. Experimentar el comportamiento de las ventanas JFrame, JDialog y JOptionPane
4. Seleccionar los lienzos más apropiados para un diseño: JPanel, JScrollPane, JTabbedPane
5. Revisar las posibilidades de los estilos:FlowLayout, BorderLayout y GridLayout
6. Apropiar algunos componentes básicos: JLabel, JTextField, JButton, JMenuBar,
7. Apropiar algunos componentes especiales: JFileChooser y JColorChooser
8. Vivenciar las prácticas XP :[Acceptance tests](http://www.extremeprogramming.org/rules/functionaltests.html) *are run often and the score is published*

*When* [a bug is found](http://www.extremeprogramming.org/rules/bugs.html) *tests are create*

**ENTREGA**

1. Incluyan en un archivo .zip los archivos correspondientes al laboratorio. El nombre debe ser los dos apellidos de los miembros del equipo ordenados alfabéticamente.
2. En el espacio de entrega deben indicar el estado de avance de su laboratorio y los problemas pendientes por resolver.
3. Deben publicar el avance al final de la sesión y la versión definitiva en la fecha indicada, en los espacios preparados para tal fin.

**CONTEXTO**

El objetivo es implementar el juego  **Vintage** en versión usuario-computador.

# El trabajo se debe hacer desde **CONSOLA**

|  |  |
| --- | --- |
| El propuesto por ustedes  VintageGUI | El acordado en laboratorio Vintage |
| **Vista - Controlador** | **Modelo** |

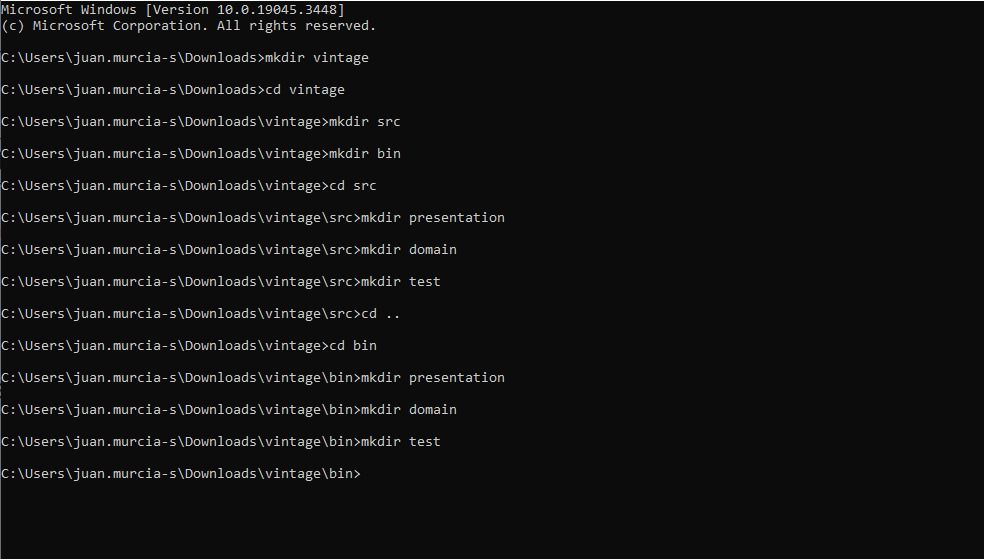
**Para la capa de presentación NO deben hacer pruebas de unidad ni diagramas de secuencia**

## DESARROLLO

### Directorios

El objetivo de este punto es construir un primer esquema para el juego **Vintage**.

1. Preparen un directorio llamado **Vintage**  con los directorios src y bin y los subdirectorios para presentación, dominio y pruebas de unidad.



**Ciclo 0: Ventana vacía – Salir**

### [En \*.java y lab05.doc]

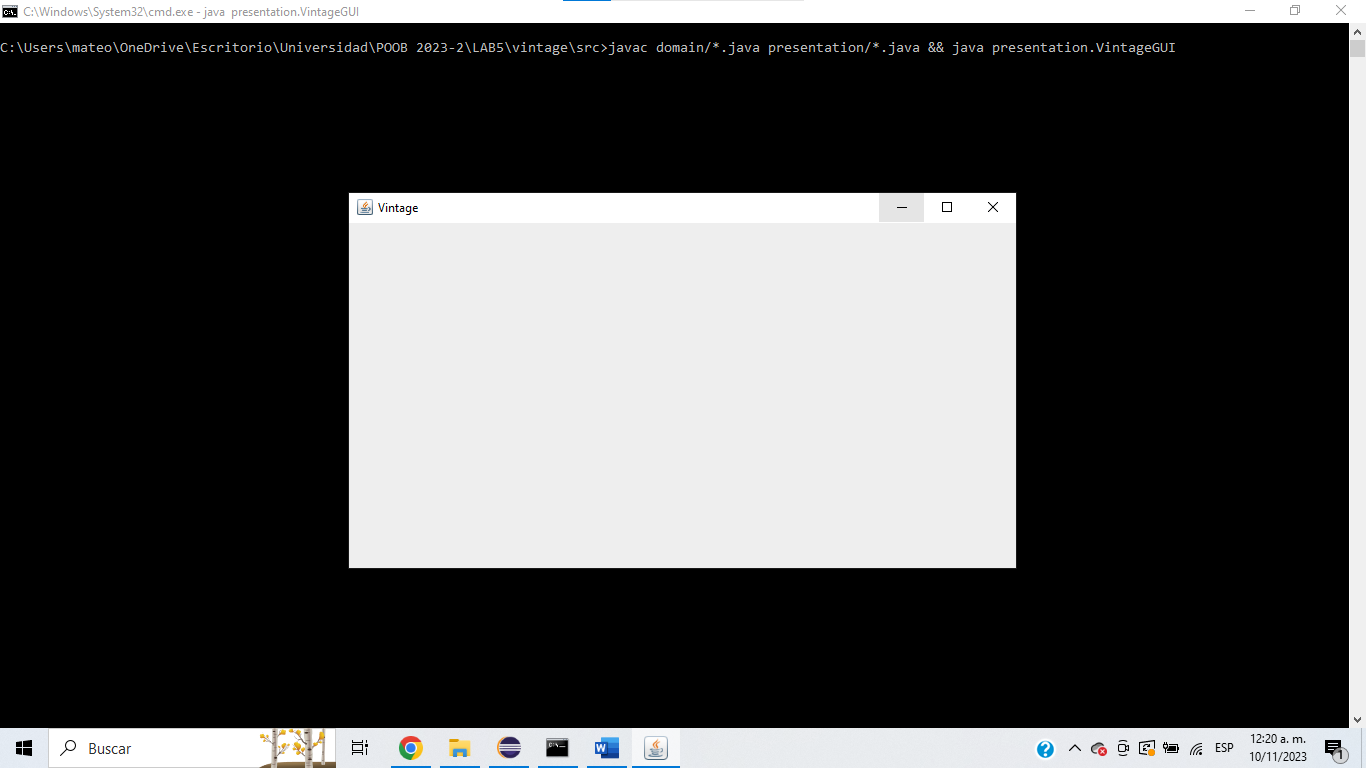
El objetivo es implementar la ventana principal de **Vintage** con un final adecuado desde el icono de cerrar. Utilizar el esquema de prepareElements-prepareActions.

1. Construyan el primer esquema de la ventana de **Vintage** únicamente con el título “**Vintage**”. Para esto cree la clase  **Vintage** como un JFrame con su creador (que sólo coloca el título) y el método main que crea un objeto **Vintage** y lo hace visible. Ejecútenlo. Capturen la pantalla.

(Si la ventana principal no es la inicial en su diseño, después deberán mover el main al componente visual correspondiente)



1. Modifiquen el tamaño de la ventana para que ocupe un cuarto de la pantalla y ubíquenla en el centro. Para eso inicien la codificación del método prepareElements. Capturen esa pantalla.



1. Traten de cerrar la ventana. ¿Termina la ejecución? ¿Qué deben hacer para terminar la ejecución?

Actualmente cuando intentamos cerrar la ventana generada por VintageGUI nos damos cuenta que se cierra la ventana de JFrame pero en la consola de windows vemos que el programa sigue en ejecuccion aunque el JFrame se cierre.

1. Estudien en JFrame el método setDefaultCloseOperation. ¿Para qué sirve? ¿Cómo lo usarían si queremos confirmar el cierre de la aplicación? ¿Cómo lo usarían si queremos simplemente cerrar la aplicación?

setDefaultCloseOperation es un metodo de JFrame que nos ayuda a configurar la operacion que va a suceder cuando se intente cerrar el programa. Para confirmer el cierre de la aplicacion se deberia usar la operacion de doNothing para configurar el cerrado de manera automatica mediante un oyente que nos ayude a manejar el cerrado de la aplicacion. Para simplemente cerrar la aplicacion lo que se deberia hacer es usar la operacion EXIT\_ON\_CLOSE cuya opcion esta definida para que el cerrado se haga automaticamente aplicando un System.exti para que el cerrado termine la ejecucion del programa.

1. Preparen el “oyente” correspondiente al icono cerrar que le pida al usuario que confirme su selección. Para eso inicien la codificación del método prepareActions y el método asociado a la acción (exit). Ejecuten el programa y cierren el programa. Capturen las pantallas.

Interfaz de usuario gráfica

Descripción generada automáticamente

**Ciclo 1: Ventana con menú – Salir**

### [En \*.java y lab05.doc]

El objetivo es implementar un menú clásico para la aplicación con un final adecuado desde la opción del menú para salir. El menú debe ofrecer mínimo las siguientes opciones :Nuevo, Abrir – Salvar y Salir . Incluyan los separadores de opciones.

1. Expliquen los componentes visuales necesarios para este menú. ¿Cuáles serían atributos y cuáles podrían ser varibles del método prepareElements? Justifique.

En este caso los atributos que vamos a usar son los correspondientes a los JMenuItem a los cuales les debemos implementar ciertos actionListener por ende los debemos usar como atributos de clase, las variables que se usaran de manera unica para el metodo prepareElements son el JMenuBar y el JMenu debido a que estos elementos ya estan preprogramados por ende no necesitamos guardarlos debido a que nosotros no haremos uso de estos.

1. Construya la forma del menú propuesto (prepareElements - prepareElementsMenu) . Ejecuten. Capturen la pantalla.

Interfaz de usuario gráfica, Texto, Aplicación

Descripción generada automáticamente

1. Preparen el “oyente” correspondiente al icono cerrar con confirmación (prepareActions - prepareActionsMenu). Ejecuten el programa y salgan del programa. Capturen las pantallas.

Imagen que contiene Rectángulo

Descripción generada automáticamente

Interfaz de usuario gráfica, Aplicación

Descripción generada automáticamente

**Ciclo 2: Salvar y abrir**

### [En \*.java y lab05.doc]

El objetivo es preparar la interfaz para las funciones de persistencia

1. Detalle el componente JFileChooser especialmente los métodos : JFileChooser, showOpenDialog, showSaveDialog, getSelectedFile.

JFileChooser: Este constructor crea un objeto JFileChooser que apunta al directorio predeterminado del usuario. El directorio predeterminado varía según el sistema operativo, siendo típicamente la carpeta "Mis documentos" en Windows y el directorio de inicio del usuario en Unix.

showOpenDialog: Este método muestra un cuadro de diálogo para seleccionar un archivo con la etiqueta "Abrir archivo". Requiere un componente padre y devuelve el estado de la selección del archivo.

showSaveDialog: Este método muestra un cuadro de diálogo para guardar un archivo. Requiere un componente padre y devuelve el estado de la operación de guardar.

getSelectedFile: Este método devuelve el archivo seleccionado, el cual puede ser establecido por el programador mediante setSelectedFile o por una acción del usuario, como escribir el nombre del archivo en la interfaz de usuario o seleccionar el archivo de una lista en la interfaz de usuario.

1. Implementen parcialmente los elementos necesarios para salvar y abrir. Al seleccionar los archivos indique que las funcionalidades están en construcción detallando la acción y el nombre del archivo seleccionado.
2. Ejecuten las dos opciones y capturen las pantallas más significativas.

Interfaz de usuario gráfica, Aplicación, Word

Descripción generada automáticamente Interfaz de usuario gráfica, Texto, Aplicación, Correo electrónico

Descripción generada automáticamente

Interfaz de usuario gráfica, Aplicación, Word

Descripción generada automáticamente con confianza media Interfaz de usuario gráfica, Texto, Aplicación, Correo electrónico

Descripción generada automáticamente

**Ciclo 3: Forma de la ventana principal**

### [En \*.java y lab05.doc]

El objetivo es codificar el diseño de la ventana principal (todos los elementos de primer nivel)

1. Presenten el bosquejo del diseño de interfaz con todos los componentes necesarios.
2. Continúe con la implementación definiendo los atributos necesarios y extendiendo el método prepareElements().

Para la zona del tablero defina un método prepareElementsBoard() y un método refresh() que actualiza la vista del tablero considerando, por ahora, el tablero inicial por omisión. Este método lo vamos a implementar realmente en otros ciclos.

1. Ejecuten y capturen esta pantalla.

Interfaz de usuario gráfica, Aplicación

Descripción generada automáticamente

**Ciclo 4: Cambiar color**

### [En \*.java y lab05.doc]

El objetivo es implementar este caso de uso.

1. Expliquen los elementos (vista – controlador) necesarios para implementar este caso de uso.
2. Detalle el comportamiento de JColorChooser especialmente el método estático showDialog
3. Implementen los componentes necesarios para cambiar el color del tablero y de las fichas.
4. Ejecuten el caso de uso y capture las pantallas más significativas.

Interfaz de usuario gráfica

Descripción generada automáticamente

Interfaz de usuario gráfica

Descripción generada automáticamente

**Interfaz de usuario gráfica, Aplicación

Descripción generada automáticamente**

**Ciclo 5: Modelo Vintage**

### [En \*.java y lab05.doc]

El objetivo es implementar la capa de dominio para **Vintage**.

1. Construya los métodos básicos del juego (**No olvide MDD y TDD)**
2. Ejecuten las pruebas y capturen el resultado.

**Ciclo 6: Jugar**

### [En \*.java y lab05.doc]

El objetivo es implementar el caso de uso jugar.

1. Adicione a la capa de presentación el atributo correspondiente al modelo.
2. Perfeccionen el método refresh() considerando la información del modelo de dominio.
3. Expliquen los elementos necesarios para implementar este caso de uso.
4. Para implementar est caso de uso debiamos tener metodos que actualizaran los paneles de jugador board y tablero, los metodos que realizaban esta acciones los definimos como prepareElementsBoardGameGamer1();

prepareElementsBoardGameGamer2(); prepareElementsBoardGame();

1. Implementen los componentes necesarios para jugar .¿Cuántos oyentes necesitan?

¿Por qué?

Para jugar necesitabamos n\*n oyentes los cuales correspondian a cada uno de los botones que podia dar click el usuario, estos los asignamos con un bucle a nuestra matriz de JButton que tenian los botones.

1. Ejecuten el caso de uso y capture las pantallas más significativas.

Interfaz de usuario gráfica, Aplicación

Descripción generada automáticamente

Interfaz de usuario gráfica, Aplicación

Descripción generada automáticamente

Interfaz de usuario gráfica, Aplicación

Descripción generada automáticamente

**Ciclo 7: Reiniciar**

### [En \*.java y lab05.doc]

El objetivo es implementar este caso de uso.

1. Expliquen los elementos a usar para implementar este caso de uso.

Los elementos para este caso son un prepare elements que nos permita inicializar de cero los elementos en el tablero, estso nos permitira mostrar una instancia nueva cada vez que sea necesario, este metodo los llamamos prepareElementsBoardInGame();

1. Implementen los elementos necesarios para reiniciar
2. Ejecuten el caso de uso y capture las pantallas más significativas.

Interfaz de usuario gráfica, Aplicación

Descripción generada automáticamente

Interfaz de usuario gráfica

Descripción generada automáticamente

**Ciclo 8: Cambiar el tamaño**

### [En \*.java y lab05.doc]

El objetivo es implementar este caso de uso.

1. Expliquen los elementos a usar para implementar este caso de uso
2. Implementen los elementos necesarios para cambiar el tamaño del juego
3. Ejecuten el caso de uso y capture las pantallas más significativas.

Interfaz de usuario gráfica, Aplicación

Descripción generada automáticamente

Interfaz de usuario gráfica, Aplicación

Descripción generada automáticamente

**RETROSPECTIVA**

1. ¿Cuál fue el tiempo total invertido en el laboratorio por cada uno de ustedes? (Horas/Hombre)

Le invertimos mas de 30 horas a este laboratoria de manera conjunta para que quedara un Proyecto de gran calidad y cumpliera los requesitos funcionales

1. ¿Cuál es el estado actual del laboratorio? ¿Por qué?

Terminado

1. Considerando la práctica XP del laboratorio ¿por qué consideran que es importante?

Utilizamos la practica de los bugs deben ser encontrados primero debido a que hicimos todo lo possible env er los casos de errores que implcaban fallo a nuestro codigo permitiendonos hacer un laboratorio que no tuviera bugs.

1. ¿Cuál consideran fue su mayor logro? ¿Por qué? ¿Cuál consideran que fue su mayor problema? ¿Qué hicieron para resolverlo?

El mayor logro fue lograr un laboratorio que esta completamente functional en el cual puedes jugar de manera concreta y es un juego con buena estetica y logica estable.

1. ¿Qué hicieron bien como equipo? ¿Qué se comprometen a hacer para mejorar los resultados?

El trabajo a par donde pudimos completer la funcionalidad del juego que era requerida.