Informe de Laboratorio Proyecto Final

Tema: Proyecto Final

Nota

Estudiante	Escuela	Asignatura
Christian Mestas Zegarra cmestasz@unsa.edu.pe	Escuela Profesional de Ingeniería de Sistemas	Fundamentos de la Programación 2 Semestre: II Código: 1701213

Laboratorio	Tema	Duración
Proyecto Final	Proyecto Final	04 horas

Semestre académico	Fecha de inicio	Fecha de entrega
2023 - B	Del 22 Enero 2024	Al 29 Enero 2024

1. Tarea

- **Enunciado:**
  • Cree una versión del videojuego de estrategia usando componentes de GUI, bases de datos y archivos.

2. Equipos, materiales y temas utilizados

- Sistema Operativo Microsoft Windows 10 Pro 64 bits
- Visual Studio Code 1.82.2
- Java Development Kit 17.0.1
- JavaFX sdk 21.0.1
- Git 2.41.0.windows.1
- Windows PowerShell 5.1.19041.3031
- Cuenta en GitHub con el correo institucional.
- Programación Orientada a Objetos
- HashMap de Objetos
- Agregación y composición
- Herencia y polimorfismo
- Interfaces
- Miembros de clase e instancia
- Interfaz gráfica de usuario
- Bases de datos
- Archivos

3. URL de Repositorio Github

- URL del Repositorio GitHub para clonar o recuperar.
- `https://github.com/cmestasz/fp2-23b.git`
- URL del proyecto final en el Repositorio GitHub.
- `https://github.com/cmestasz/fp2-23b/tree/main/fase03/proyecto_final`

4. Actividades con el repositorio GitHub

commits.bash

```bash
$ git add .gitignore

$ git commit -m ".gitignore actualizado para solo publicar la carpeta src"
[main e0cc937] .gitignore actualizado para solo publicar la carpeta src
 1 file changed, 3 insertions(+)

$ git add .

$ git commit -m "Ejemplo dado en la documentacion de JavaFX"
[main 118a77f] Ejemplo dado en la documentacion de JavaFX
 9 files changed, 155 insertions(+)
 create mode 100644 .vscode/settings.json
 create mode 100644 fase03/proyecto_final/VIDEOGAME/.vscode/launch.json
 create mode 100644 fase03/proyecto_final/VIDEOGAME/.vscode/settings.json
 create mode 100644 fase03/proyecto_final/VIDEOGAME/README.md
 create mode 100644 fase03/proyecto_final/VIDEOGAME/src/FX/Controller.java
 create mode 100644 fase03/proyecto_final/VIDEOGAME/src/FX/Main.fxml
 create mode 100644 fase03/proyecto_final/VIDEOGAME/src/FX/Videogame.java
 create mode 100644 fase03/proyecto_final/VIDEOGAME/src/FX/style.css
 create mode 100644 fase03/proyecto_final/VIDEOGAME/src/FX/test.fxml
 create mode 100644 fase03/proyecto_final/VIDEOGAME/src/img/4.png

$ git push -f
Enumerating objects: 24, done.
Counting objects: 100% (24/24), done.
Delta compression using up to 4 threads
Compressing objects: 100% (19/19), done.
Writing objects: 100% (21/21), 3.69 MiB | 1.04 MiB/s, done.
Total 21 (delta 1), reused 0 (delta 0), pack-reused 0
remote: Resolving deltas: 100% (1/1), done.
To https://github.com/cmestasz/fp2-23b.git
 + af4068d...118a77f main -> main

$ git add .

$ git commit -m "Menú principal"
[main e9a3e21] Menú principal
 9 files changed, 92 insertions(+), 34 deletions(-)
 create mode 100644 fase03/proyecto_final/VIDEOGAME/src/FX/Main Menu.fxml
 create mode 100644 fase03/proyecto_final/VIDEOGAME/src/img/barrack.png
 create mode 100644 fase03/proyecto_final/VIDEOGAME/src/img/barrack.psd
 create mode 100644 fase03/proyecto_final/VIDEOGAME/src/img/waiting.png
 create mode 100644 fase03/proyecto_final/VIDEOGAME/src/img/waiting.psd

$ git add .

$ git commit -m "Controlador del menú principal"
[main 84ae869] Controlador del menú principal
 5 files changed, 131 insertions(+), 14 deletions(-)
 create mode 100644 fase03/proyecto_final/VIDEOGAME/src/FX/MainMenu/Main Menu.fxml
 create mode 100644 fase03/proyecto_final/VIDEOGAME/src/FX/MainMenu/MainMenuController.java
 create mode 100644 fase03/proyecto_final/VIDEOGAME/src/Utils/Resolution.java
```

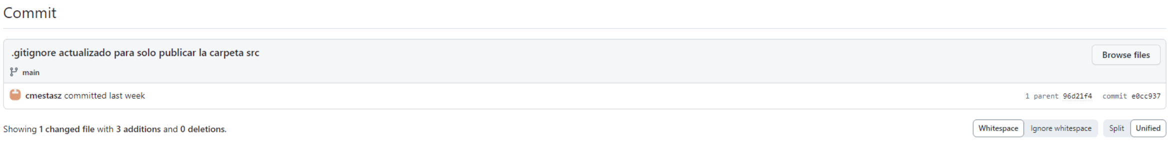
```

$ git add .

$ git commit -m "Modelo de servidor para manejar varias instancias locales del videojuego"
[main 5e5d1a0] Modelo de servidor para manejar varias instancias locales del videojuego
 4 files changed, 133 insertions(+), 56 deletions(-)
 delete mode 100644 fase03/proyecto_final/VIDEOGAME/src/FX/Main Menu.fxml
 create mode 100644 fase03/proyecto_final/VIDEOGAME/src/FX/MainMenu/MainMenuServer.java

$ git add .

$ git commit -m "Servidor completo"
[main 1d626a9] Servidor completo
 1 file changed, 33 insertions(+), 11 deletions(-)

$ git add .

$ git commit -m "Funcionalidad de servidor para el menú principal completa"
[main abc9d8d] Funcionalidad de servidor para el menú principal completa
 5 files changed, 198 insertions(+), 58 deletions(-)
 create mode 100644 fase03/proyecto_final/VIDEOGAME/src/FX/MainMenu/Connection.java
 create mode 100644 fase03/proyecto_final/VIDEOGAME/src/FX/MainMenu/Operation.java
 create mode 100644 fase03/proyecto_final/VIDEOGAME/src/Utils/Utils.java

$ git push
Enumerating objects: 76, done.
Counting objects: 100% (76/76), done.
Delta compression using up to 4 threads
Compressing objects: 100% (59/59), done.
Writing objects: 100% (65/65), 30.02 KiB | 2.73 MiB/s, done.
Total 65 (delta 32), reused 0 (delta 0), pack-reused 0
remote: Resolving deltas: 100% (32/32), completed with 7 local objects.
To https://github.com/cmestasz/fp2-23b.git
   118a77f..abc9d8d  main -> main

$ git add .

$ git commit -m "Inicio del juego vinculado entre ambas instancias"
[main a9a8d44] Inicio del juego vinculado entre ambas instancias
 9 files changed, 132 insertions(+), 64 deletions(-)
 rename fase03/proyecto_final/VIDEOGAME/src/FX/{ => MainGame}/Controller.java (94%)
 rename fase03/proyecto_final/VIDEOGAME/src/FX/{ => MainGame}/Main.fxml (98%)
 delete mode 100644 fase03/proyecto_final/VIDEOGAME/src/FX/test.fxml

$ git add .

$ git commit -m "Cambio de estructura y vinculo entre ambos controladores"
[main 0be0029] Cambio de estructura y vinculo entre ambos controladores
 8 files changed, 77 insertions(+), 35 deletions(-)
 delete mode 100644 fase03/proyecto_final/VIDEOGAME/src/FX/MainGame/Controller.java
 rename fase03/proyecto_final/VIDEOGAME/src/FX/MainGame/{Main.fxml => Main Game.fxml} (98%)
 create mode 100644 fase03/proyecto_final/VIDEOGAME/src/FX/MainGame/MainGameController.java
 rename fase03/proyecto_final/VIDEOGAME/src/FX/{MainMenu/MainMenuServer.java =>
    MainServer.java} (96%)
 rename fase03/proyecto_final/VIDEOGAME/src/{FX/MainMenu => Utils}/Connection.java (96%)
 rename fase03/proyecto_final/VIDEOGAME/src/{FX/MainMenu => Utils}/Operation.java (90%)
```

```

$ git add .

$ git commit -m "Comentarios descriptivos en las partes confusas"
[main 38e2aaf] Comentarios descriptivos en las partes confusas
 3 files changed, 21 insertions(+), 12 deletions(-)

$ git add .

$ git commit -m "Estructura del controlador del juego principal"
[main 8c4d1bd] Estructura del controlador del juego principal
 11 files changed, 86 insertions(+), 82 deletions(-)
 delete mode 100644 fase03/proyecto_final/VIDEOGAME/src/FX/MainGame/Main Game.fxml
 create mode 100644 fase03/proyecto_final/VIDEOGAME/src/FX/MainGame/MainGame.fxml
 rename fase03/proyecto_final/VIDEOGAME/src/FX/MainMenu/{Main Menu.fxml => MainMenu.fxml}
    (100%)
 delete mode 100644 fase03/proyecto_final/VIDEOGAME/src/img/4.png
 delete mode 100644 fase03/proyecto_final/VIDEOGAME/src/img/barrack.png
 delete mode 100644 fase03/proyecto_final/VIDEOGAME/src/img/barrack.psd
 create mode 100644 fase03/proyecto_final/VIDEOGAME/src/img/tile.png
 create mode 100644 fase03/proyecto_final/VIDEOGAME/src/img/tile.psd

$ git push
Enumerating objects: 72, done.
Counting objects: 100% (72/72), done.
Delta compression using up to 4 threads
Compressing objects: 100% (57/57), done.
Writing objects: 100% (61/61), 14.12 KiB | 2.02 MiB/s, done.
Total 61 (delta 29), reused 0 (delta 0), pack-reused 0
remote: Resolving deltas: 100% (29/29), completed with 5 local objects.
To https://github.com/cmestasz/fp2-23b.git
   abc9d8d..8c4d1bd  main -> main

$ git add .

$ git commit -m "Estructura visual del juego principal"
[main 49914f7] Estructura visual del juego principal
 6 files changed, 124 insertions(+), 35 deletions(-)
 create mode 100644 fase03/proyecto_final/VIDEOGAME/src/FX/MainGame/Board.java

$ git add .

$ git commit -m "Modelo de base de datos y cambios en el menu principal"
[main 5b878e5] Modelo de base de datos y cambios en el menu principal
 6 files changed, 135 insertions(+), 37 deletions(-)
 create mode 100644 fase03/proyecto_final/VIDEOGAME/src/Utils/DBConnector.java
 rename fase03/proyecto_final/VIDEOGAME/src/Utils/{Connection.java => ServerConnection.java}
    (91%)

$ git push
Enumerating objects: 48, done.
Counting objects: 100% (48/48), done.
Delta compression using up to 4 threads
Compressing objects: 100% (29/29), done.
Writing objects: 100% (31/31), 21.49 KiB | 4.30 MiB/s, done.
Total 31 (delta 13), reused 0 (delta 0), pack-reused 0
```

```
remote: Resolving deltas: 100% (13/13), completed with 9 local objects.
To https://github.com/cmestasz/fp2-23b.git
   8c4d1bd..5b878e5  main -> main

$ git add .

$ git commit -m "Clase que permite enviar y solicitar datos a la base de datos"
[main e1375c5] Clase que permite enviar y solicitar datos a la base de datos
 3 files changed, 74 insertions(+), 14 deletions(-)

$ git add .

$ git commit -m "Implementacion de la clase DBConnector"
[main acca9f6] Implementacion de la clase DBConnector
 5 files changed, 101 insertions(+), 70 deletions(-)

$ git add .

$ git commit -m "Clases para el juego principal"
[main dfbc1be] Clases para el juego principal
 17 files changed, 186 insertions(+), 23 deletions(-)
 create mode 100644 fase03/proyecto_final/VIDEOGAME/src/FX/MainGame/BoardGUI.java
 create mode 100644 fase03/proyecto_final/VIDEOGAME/src/FX/MainGame/Classes/Archer.java
 create mode 100644 fase03/proyecto_final/VIDEOGAME/src/FX/MainGame/Classes/Knight.java
 create mode 100644 fase03/proyecto_final/VIDEOGAME/src/FX/MainGame/Classes/Soldier.java
 create mode 100644 fase03/proyecto_final/VIDEOGAME/src/FX/MainGame/Classes/Spearman.java
 create mode 100644 fase03/proyecto_final/VIDEOGAME/src/FX/MainGame/Classes/Swordsman.java
 create mode 100644 fase03/proyecto_final/VIDEOGAME/src/img/tile_archer.png
 create mode 100644 fase03/proyecto_final/VIDEOGAME/src/img/tile_archer.psd
 create mode 100644 fase03/proyecto_final/VIDEOGAME/src/img/tile_knight.png
 create mode 100644 fase03/proyecto_final/VIDEOGAME/src/img/tile_knight.psd
 create mode 100644 fase03/proyecto_final/VIDEOGAME/src/img/tile_spearman.png
 create mode 100644 fase03/proyecto_final/VIDEOGAME/src/img/tile_spearman.psd
 create mode 100644 fase03/proyecto_final/VIDEOGAME/src/img/tile_swordsman.png
 create mode 100644 fase03/proyecto_final/VIDEOGAME/src/img/tile_swordsman.psd

$ git add .

$ git commit -m "Eleccion de reino y cambios en el menu principal"
[main 790dc08] Eleccion de reino y cambios en el menu principal
 5 files changed, 119 insertions(+), 32 deletions(-)
 create mode 100644 fase03/proyecto_final/VIDEOGAME/src/img/settings.png

$ git add .

$ git commit -m "El tablero ahora se conecta entre ambos jugadores"
[main 5f9c49f] El tablero ahora se conecta entre ambos jugadores
 13 files changed, 227 insertions(+), 95 deletions(-)
 delete mode 100644 fase03/proyecto_final/VIDEOGAME/src/FX/MainGame/BoardGUI.java
 create mode 100644 fase03/proyecto_final/VIDEOGAME/src/Utils/SerializableColor.java

$ git add .

 $ git commit -m "Envio de mensajes y conexion en el juego principal"
[main 45de600] Envio de mensajes y conexion en el juego principal
 10 files changed, 270 insertions(+), 75 deletions(-)
 create mode 100644 fase03/proyecto_final/VIDEOGAME/src/Utils/MainGameOperation.java
 create mode 100644 fase03/proyecto_final/VIDEOGAME/src/Utils/MainMenuOperation.java
 delete mode 100644 fase03/proyecto_final/VIDEOGAME/src/Utils/Operation.java

$ git add .

$ git commit -m "Nuevo sistema de mensajes"
[main 5207444] Nuevo sistema de mensajes
 3 files changed, 35 insertions(+), 16 deletions(-)

$ git add .

$ git commit -m "Chat de colores y mejor tratado"
[main 1afcb77] Chat de colores y mejor tratado
 3 files changed, 67 insertions(+), 26 deletions(-)

 $ git add .

$ git commit -m "Representacion visual de los soldados en ambos tableros"
[main cc5ebe5] Representacion visual de los soldados en ambos tableros
 13 files changed, 185 insertions(+), 78 deletions(-)
 create mode 100644 fase03/proyecto_final/VIDEOGAME/src/Utils/BetterColor.java
 delete mode 100644 fase03/proyecto_final/VIDEOGAME/src/Utils/SerializableColor.java
 create mode 100644 fase03/proyecto_final/VIDEOGAME/src/Utils/Tile.java
 create mode 100644 fase03/proyecto_final/VIDEOGAME/src/Utils/VideogameConstants.java
 rename fase03/proyecto_final/VIDEOGAME/src/img/{tile.png => tile_tile.png} (100%)
 rename fase03/proyecto_final/VIDEOGAME/src/img/{tile.psd => tile_tile.psd} (100%)

$ git push
Enumerating objects: 173, done.
Counting objects: 100% (173/173), done.
Delta compression using up to 4 threads
Compressing objects: 100% (145/145), done.
Writing objects: 100% (154/154), 247.29 KiB | 7.98 MiB/s, done.
Total 154 (delta 87), reused 0 (delta 0), pack-reused 0
remote: Resolving deltas: 100% (87/87), completed with 10 local objects.
To https://github.com/cmestasz/fp2-23b.git
   5b878e5..cc5ebe5  main -> main

$ git add .

$ git commit -m "Implementacion de los tipos de soldado y sus posibles acciones"
[main 70d26d2] Implementacion de los tipos de soldado y sus posibles acciones
 23 files changed, 52 insertions(+), 43 deletions(-)
 rename fase03/proyecto_final/VIDEOGAME/src/{FX => }/MainServer.java (99%)
 rename fase03/proyecto_final/VIDEOGAME/src/{FX => }/Videogame.java (98%)
 create mode 100644 fase03/proyecto_final/VIDEOGAME/src/img/action_charge.png
 create mode 100644 fase03/proyecto_final/VIDEOGAME/src/img/action_charge.psd
 create mode 100644 fase03/proyecto_final/VIDEOGAME/src/img/action_dismount.png
 create mode 100644 fase03/proyecto_final/VIDEOGAME/src/img/action_dismount.psd
 create mode 100644 fase03/proyecto_final/VIDEOGAME/src/img/action_mount.png
 create mode 100644 fase03/proyecto_final/VIDEOGAME/src/img/action_mount.psd
 create mode 100644 fase03/proyecto_final/VIDEOGAME/src/img/action_move.png
 create mode 100644 fase03/proyecto_final/VIDEOGAME/src/img/action_move.psd
 create mode 100644 fase03/proyecto_final/VIDEOGAME/src/img/action_schiltrom.png
 create mode 100644 fase03/proyecto_final/VIDEOGAME/src/img/action_schiltrom.psd
 create mode 100644 fase03/proyecto_final/VIDEOGAME/src/img/action_shoot.png
 create mode 100644 fase03/proyecto_final/VIDEOGAME/src/img/action_shoot.psd
 create mode 100644 fase03/proyecto_final/VIDEOGAME/src/img/action_sworddance.png
 create mode 100644 fase03/proyecto_final/VIDEOGAME/src/img/action_sworddance.psd

$ git add .

$ git commit -m "Implementacion de movimientos y ataques, se cancelaron bastantes planes que eran demasiado ambiciosos"
[main 4fc2a3f] Implementacion de movimientos y ataques, se cancelaron bastantes planes que eran demasiado ambiciosos
 26 files changed, 320 insertions(+), 107 deletions(-)
 create mode 100644 fase03/proyecto_final/VIDEOGAME/src/img/action_attack.png
 rename fase03/proyecto_final/VIDEOGAME/src/img/{action_shoot.psd => action_attack.psd} (61%)
 delete mode 100644 fase03/proyecto_final/VIDEOGAME/src/img/action_charge.png
 delete mode 100644 fase03/proyecto_final/VIDEOGAME/src/img/action_charge.psd
 delete mode 100644 fase03/proyecto_final/VIDEOGAME/src/img/action_dismount.png
 delete mode 100644 fase03/proyecto_final/VIDEOGAME/src/img/action_dismount.psd
 delete mode 100644 fase03/proyecto_final/VIDEOGAME/src/img/action_mount.png
 delete mode 100644 fase03/proyecto_final/VIDEOGAME/src/img/action_mount.psd
 delete mode 100644 fase03/proyecto_final/VIDEOGAME/src/img/action_schiltrom.png
 delete mode 100644 fase03/proyecto_final/VIDEOGAME/src/img/action_schiltrom.psd
 delete mode 100644 fase03/proyecto_final/VIDEOGAME/src/img/action_shoot.png
 delete mode 100644 fase03/proyecto_final/VIDEOGAME/src/img/action_sworddance.png
 delete mode 100644 fase03/proyecto_final/VIDEOGAME/src/img/action_sworddance.psd

$ git add .

$ git commit -m "Primera prueba de toda la funcionalidad completa"
[main 6520ee5] Primera prueba de toda la funcionalidad completa
 6 files changed, 85 insertions(+), 62 deletions(-)

$ git push
Enumerating objects: 98, done.
Counting objects: 100% (98/98), done.
Delta compression using up to 4 threads
Compressing objects: 100% (73/73), done.
Writing objects: 100% (76/76), 71.05 KiB | 5.46 MiB/s, done.
Total 76 (delta 46), reused 0 (delta 0), pack-reused 0
remote: Resolving deltas: 100% (46/46), completed with 13 local objects.
To https://github.com/cmestasz/fp2-23b.git
   cc5ebe5..6520ee5  main -> main

$ git add .

$ git commit -m "Correccion de los ultimos errores"
[main 924da03] Correccion de los ultimos errores
 12 files changed, 134 insertions(+), 72 deletions(-)

$ git push
Enumerating objects: 46, done.
Counting objects: 100% (46/46), done.
Delta compression using up to 4 threads
Compressing objects: 100% (23/23), done.
Writing objects: 100% (24/24), 25.04 KiB | 6.26 MiB/s, done.
Total 24 (delta 16), reused 0 (delta 0), pack-reused 0
```

```
remote: Resolving deltas: 100% (16/16), completed with 16 local objects.
To https://github.com/cmestasz/fp2-23b.git
   6520ee5..924da03  main -> main

$ git add .

$ git commit -m "Implementacion de más utilidades"
[main ea1dca9] Implementacion de más utilidades
 5 files changed, 27 insertions(+), 13 deletions(-)

$ git push
Enumerating objects: 28, done.
Counting objects: 100% (28/28), done.
Delta compression using up to 4 threads
Compressing objects: 100% (14/14), done.
Writing objects: 100% (15/15), 4.20 KiB | 1.40 MiB/s, done.
Total 15 (delta 11), reused 0 (delta 0), pack-reused 0
remote: Resolving deltas: 100% (11/11), completed with 10 local objects.
To https://github.com/cmestasz/fp

$ git add .

$ git commit -m "Ultimas correcciones de la logica y funciones"
[main f0cf82b] Ultimas correcciones de la logica y funciones
 2 files changed, 4 insertions(+), 1 deletion(-)

$ git push
Enumerating objects: 20, done.
Counting objects: 100% (20/20), done.
Delta compression using up to 4 threads
Compressing objects: 100% (10/10), done.
Writing objects: 100% (11/11), 3.40 KiB | 1.70 MiB/s, done.
Total 11 (delta 8), reused 0 (delta 0), pack-reused 0
remote: Resolving deltas: 100% (8/8), completed with 7 local objects.
To https://github.com/cmestasz/fp2-23b.git
   ea1dca9..f0cf82b  main -> main

$ git add .

$ git commit -m "Correcciones de la base de datos"
[main 12becf8] Correcciones de la base de datos
 3 files changed, 15 insertions(+), 7 deletions(-)

$ git push
Enumerating objects: 23, done.
Counting objects: 100% (23/23), done.
Delta compression using up to 4 threads
Compressing objects: 100% (11/11), done.
Writing objects: 100% (12/12), 1.12 KiB | 1.12 MiB/s, done.
Total 12 (delta 8), reused 0 (delta 0), pack-reused 0
remote: Resolving deltas: 100% (8/8), completed with 8 local objects.
To https://github.com/cmestasz/fp2-23b.git
   f0cf82b..12becf8  main -> main

$ git add .

$ git commit -m "Implementacion de mejoras visuales y forma final"
[main fd709bc] Implementacion de mejoras visuales y forma final
 47 files changed, 155 insertions(+), 99 deletions(-)
 create mode 100644 fase03/proyecto_final/VIDEOGAME/src/img/White_Hammer.png
 create mode 100644 fase03/proyecto_final/VIDEOGAME/src/img/White_Moving.png
 create mode 100644 fase03/proyecto_final/VIDEOGAME/src/img/background_beach.png
 create mode 100644 fase03/proyecto_final/VIDEOGAME/src/img/background_beach.psd
 create mode 100644 fase03/proyecto_final/VIDEOGAME/src/img/background_data.png
 create mode 100644 fase03/proyecto_final/VIDEOGAME/src/img/background_data.psd
 create mode 100644 fase03/proyecto_final/VIDEOGAME/src/img/background_desert.png
 create mode 100644 fase03/proyecto_final/VIDEOGAME/src/img/background_desert.psd
 create mode 100644 fase03/proyecto_final/VIDEOGAME/src/img/background_forest.png
 create mode 100644 fase03/proyecto_final/VIDEOGAME/src/img/background_forest.psd
 create mode 100644 fase03/proyecto_final/VIDEOGAME/src/img/background_meadow.png
 create mode 100644 fase03/proyecto_final/VIDEOGAME/src/img/background_meadow.psd
 create mode 100644 fase03/proyecto_final/VIDEOGAME/src/img/background_mountain.png
 create mode 100644 fase03/proyecto_final/VIDEOGAME/src/img/background_mountain.psd
 create mode 100644 fase03/proyecto_final/VIDEOGAME/src/img/beach.jpg
 create mode 100644 fase03/proyecto_final/VIDEOGAME/src/img/desert.png
 create mode 100644 fase03/proyecto_final/VIDEOGAME/src/img/icon_big_attack.png
 create mode 100644 fase03/proyecto_final/VIDEOGAME/src/img/icon_big_defence.png
 create mode 100644 fase03/proyecto_final/VIDEOGAME/src/img/icon_big_helmet.png
 create mode 100644 fase03/proyecto_final/VIDEOGAME/src/img/icon_big_target.png
 create mode 100644 fase03/proyecto_final/VIDEOGAME/src/img/meadow.jpg
 create mode 100644 fase03/proyecto_final/VIDEOGAME/src/img/mountain.jpg

$ git push
Enumerating objects: 94, done.
Counting objects: 100% (94/94), done.
Delta compression using up to 4 threads
Compressing objects: 100% (58/58), done.
Writing objects: 100% (59/59), 43.14 MiB | 2.47 MiB/s, done.
Total 59 (delta 16), reused 0 (delta 0), pack-reused 0
remote: Resolving deltas: 100% (16/16), completed with 13 local objects.
To https://github.com/cmestasz/fp2-23b.git
   12becf8..fd709bc  main -> main

$ git add .

$ git commit -m "Correcciones para el ejecutable y proyecto exportado"
[main 83baf9f] Correcciones para el ejecutable y proyecto exportado
 9 files changed, 67 insertions(+), 25 deletions(-)
 create mode 100644 fase03/proyecto_final/VIDEOGAME/SERVER.jar
 create mode 100644 fase03/proyecto_final/VIDEOGAME/VIDEOGAME.jar
 create mode 100644 fase03/proyecto_final/VIDEOGAME/data/dblogin.dat
 create mode 100644 fase03/proyecto_final/VIDEOGAME/src/Main.java

$ git add .

$ git commit -m "Ejecutables del videojuego en una carpeta separada"
[main 68308c2] Ejecutables del videojuego en una carpeta separada
 46 files changed, 569 insertions(+), 1 deletion(-)
 create mode 100644 fase03/proyecto_final/EJECUTABLES/SERVER.jar
 rename fase03/proyecto_final/{VIDEOGAME/SERVER.jar => EJECUTABLES/VIDEOGAME.jar} (77%)
 create mode 100644 fase03/proyecto_final/EJECUTABLES/data/dblogin.dat
 create mode 100644 fase03/proyecto_final/SERVER/.vscode/settings.json
 create mode 100644 fase03/proyecto_final/SERVER/README.md
 create mode 100644 fase03/proyecto_final/SERVER/SERVER.jar
 create mode 100644 fase03/proyecto_final/SERVER/src/FX/MainGame/Board.java
 create mode 100644 fase03/proyecto_final/SERVER/src/FX/MainGame/Classes/Archer.java
 create mode 100644 fase03/proyecto_final/SERVER/src/FX/MainGame/Classes/Knight.java
 create mode 100644 fase03/proyecto_final/SERVER/src/FX/MainGame/Classes/Soldier.java
 create mode 100644 fase03/proyecto_final/SERVER/src/FX/MainGame/Classes/Spearman.java
 create mode 100644 fase03/proyecto_final/SERVER/src/FX/MainGame/Classes/Swordsman.java
 rename fase03/proyecto_final/{VIDEOGAME => SERVER}/src/MainServer.java (99%)
 create mode 100644 fase03/proyecto_final/SERVER/src/Utils/BetterColor.java
 create mode 100644 fase03/proyecto_final/SERVER/src/Utils/DBConnector.java
 create mode 100644 fase03/proyecto_final/SERVER/src/Utils/MainGameOperation.java
 create mode 100644 fase03/proyecto_final/SERVER/src/Utils/MainMenuOperation.java
 rename fase03/proyecto_final/{VIDEOGAME => SERVER}/src/Utils/ServerConnection.java (100%)
 create mode 100644 fase03/proyecto_final/SERVER/src/Utils/Utils.java
 create mode 100644 fase03/proyecto_final/SERVER/src/Utils/VideogameConstants.java
 delete mode 100644 fase03/proyecto_final/VIDEOGAME/src/img/White_Hammer.png
 delete mode 100644 fase03/proyecto_final/VIDEOGAME/src/img/White_Moving.png
 delete mode 100644 fase03/proyecto_final/VIDEOGAME/src/img/action_attack.psd
 delete mode 100644 fase03/proyecto_final/VIDEOGAME/src/img/action_move.psd
 delete mode 100644 fase03/proyecto_final/VIDEOGAME/src/img/background_beach.psd
 delete mode 100644 fase03/proyecto_final/VIDEOGAME/src/img/background_data.psd
 delete mode 100644 fase03/proyecto_final/VIDEOGAME/src/img/background_desert.psd
 delete mode 100644 fase03/proyecto_final/VIDEOGAME/src/img/background_forest.psd
 delete mode 100644 fase03/proyecto_final/VIDEOGAME/src/img/background_meadow.psd
 delete mode 100644 fase03/proyecto_final/VIDEOGAME/src/img/background_mountain.psd
 delete mode 100644 fase03/proyecto_final/VIDEOGAME/src/img/beach.jpg
 delete mode 100644 fase03/proyecto_final/VIDEOGAME/src/img/desert.png
 delete mode 100644 fase03/proyecto_final/VIDEOGAME/src/img/icon_big_attack.png
 delete mode 100644 fase03/proyecto_final/VIDEOGAME/src/img/icon_big_defence.png
 delete mode 100644 fase03/proyecto_final/VIDEOGAME/src/img/icon_big_helmet.png
 delete mode 100644 fase03/proyecto_final/VIDEOGAME/src/img/icon_big_target.png
 delete mode 100644 fase03/proyecto_final/VIDEOGAME/src/img/meadow.jpg
 delete mode 100644 fase03/proyecto_final/VIDEOGAME/src/img/mountain.jpg
 delete mode 100644 fase03/proyecto_final/VIDEOGAME/src/img/tile_archer.psd
 delete mode 100644 fase03/proyecto_final/VIDEOGAME/src/img/tile_knight.psd
 delete mode 100644 fase03/proyecto_final/VIDEOGAME/src/img/tile_spearman.psd
 delete mode 100644 fase03/proyecto_final/VIDEOGAME/src/img/tile_swordsman.psd
 delete mode 100644 fase03/proyecto_final/VIDEOGAME/src/img/tile_tile.psd
 delete mode 100644 fase03/proyecto_final/VIDEOGAME/src/img/waiting.psd

$ git push
Enumerating objects: 62, done.
Counting objects: 100% (62/62), done.
Delta compression using up to 4 threads
Compressing objects: 100% (41/41), done.
Writing objects: 100% (48/48), 137.06 MiB | 1.74 MiB/s, done.
Total 48 (delta 15), reused 0 (delta 0), pack-reused 0
remote: Resolving deltas: 100% (15/15), completed with 7 local objects.
remote: warning: See https://gh.io/lfs for more information.
remote: warning: File fase03/proyecto_final/EJECUTABLES/VIDEOGAME.jar is 50.32 MB this is larger than GitHub's recommended maximum file size of 50.00 MB
remote: warning: File fase03/proyecto_final/VIDEOGAME/VIDEOGAME.jar is 86.40 MB this is larger than GitHub's recommended maximum file size of 50.00 MB
remote: warning: GH001: Large files detected. You may want to try Git Large File Storage - https://git-lfs.github.com.
```

```
To https://github.com/cmestasz/fp2-23b.git
   fd709bc..68308c2  main -> main
```



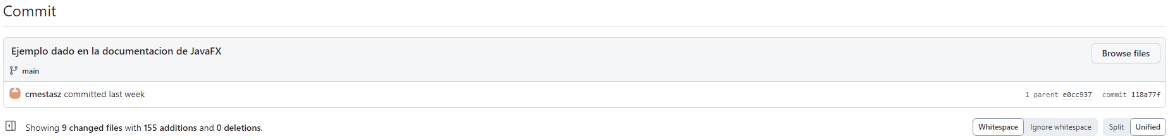
Commit

.gitignore actualizado para solo publicar la carpeta src

main

cmestasz committed last week 1 parent 96d21f4 commit e0cc937

Showing 1 changed file with 3 additions and 0 deletions.



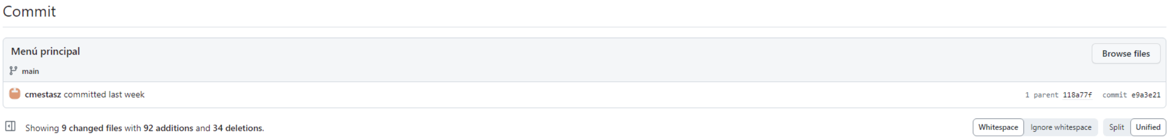
Commit

Ejemplo dado en la documentacion de JavaFX

main

cmestasz committed last week 1 parent e0cc937 commit 118a77f

Showing 9 changed files with 155 additions and 0 deletions.



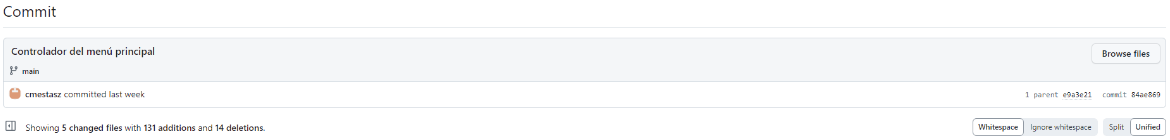
Commit

Menú principal

main

cmestasz committed last week 1 parent 118a77f commit e9a3e21

Showing 9 changed files with 92 additions and 34 deletions.



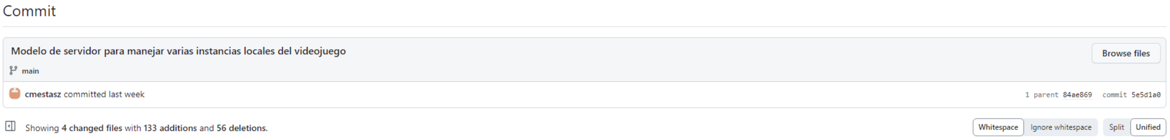
Commit

Controlador del menú principal

main

cmestasz committed last week 1 parent e9a3e21 commit 84ae869

Showing 5 changed files with 131 additions and 14 deletions.



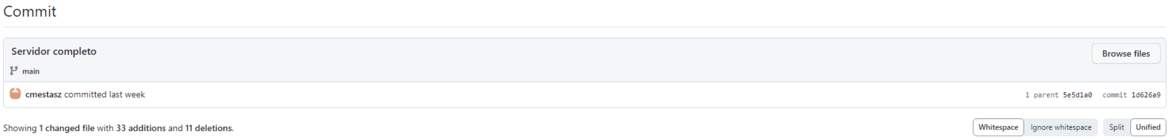
Commit

Modelo de servidor para manejar varias instancias locales del videojuego

main

cmestasz committed last week 1 parent 84ae869 commit 5e5d1a0

Showing 4 changed files with 133 additions and 56 deletions.



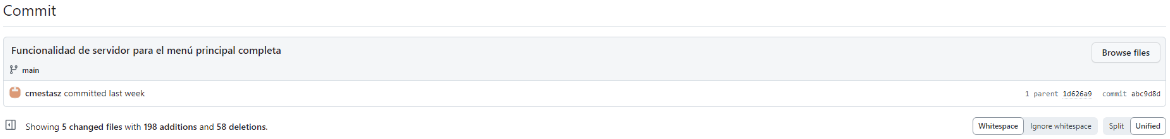
Commit

Servidor completo

main

cmestasz committed last week 1 parent 5e5d1a0 commit 1d626a9

Showing 1 changed file with 33 additions and 11 deletions.



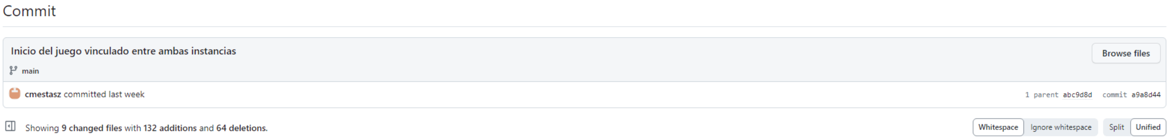
Commit

Funcionalidad de servidor para el menú principal completa

main

cmestasz committed last week 1 parent 1d626a9 commit abc9d8d

Showing 5 changed files with 198 additions and 58 deletions.



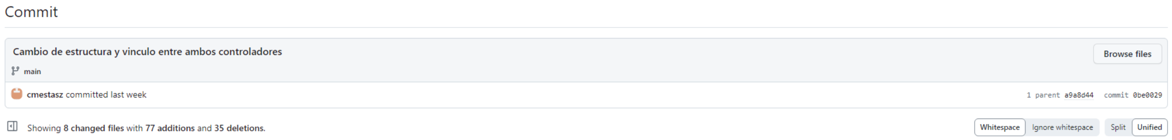
Commit

Inicio del juego vinculado entre ambas instancias

main

cmestasz committed last week 1 parent abc9d8d commit a9a8d44

Showing 9 changed files with 132 additions and 64 deletions.



Commit

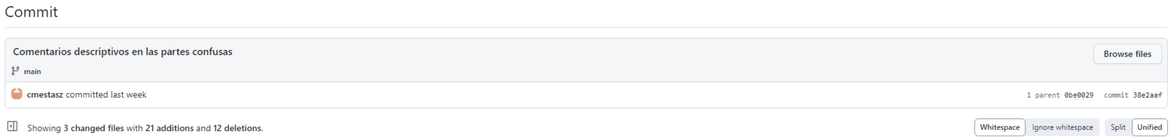
Cambio de estructura y vinculo entre ambos controladores

main

cmestasz committed last week

1 parent a9a8d44 commit 0be0029

Showing 8 changed files with 77 additions and 35 deletions.



Commit

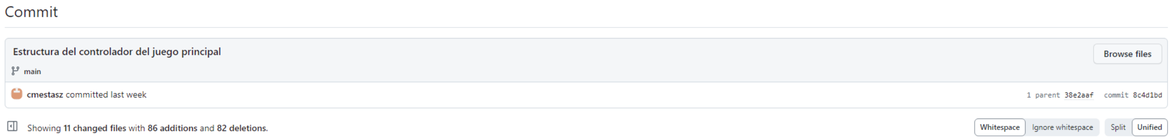
Comentarios descriptivos en las partes confusas

main

cmestasz committed last week

1 parent 0be0029 commit 38e2aaf

Showing 3 changed files with 21 additions and 12 deletions.



Commit

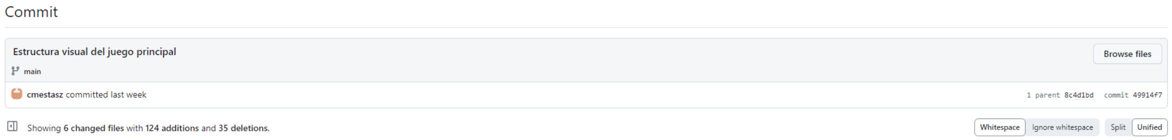
Estructura del controlador del juego principal

main

cmestasz committed last week

1 parent 38e2aaf commit 8c4d1bd

Showing 11 changed files with 86 additions and 82 deletions.



Commit

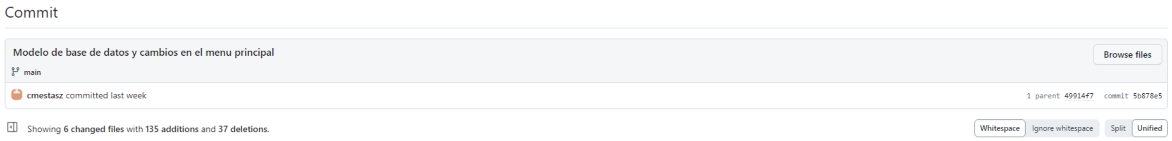
Estructura visual del juego principal

main

cmestasz committed last week

1 parent 8c4d1bd commit 49914f7

Showing 6 changed files with 124 additions and 35 deletions.



Commit

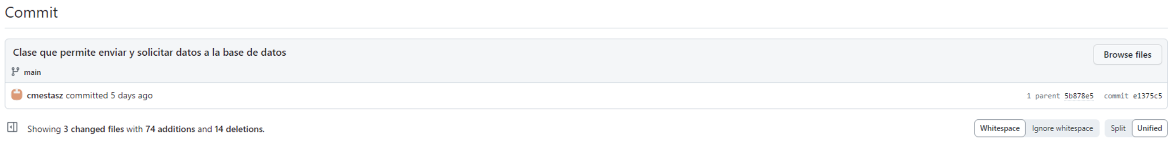
Modelo de base de datos y cambios en el menu principal

main

cmestasz committed last week

1 parent 49914f7 commit 5b878e5

Showing 6 changed files with 135 additions and 37 deletions.



Commit

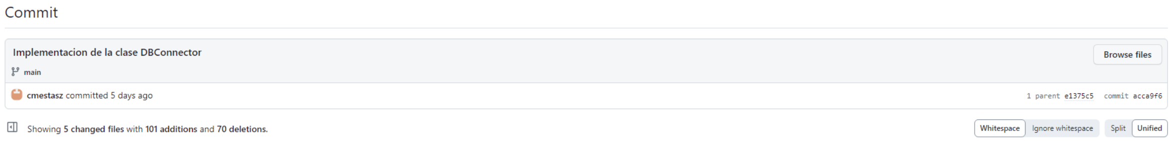
Clase que permite enviar y solicitar datos a la base de datos

main

cmestasz committed 5 days ago

1 parent 5b878e5 commit e1375c5

Showing 3 changed files with 74 additions and 14 deletions.



Commit

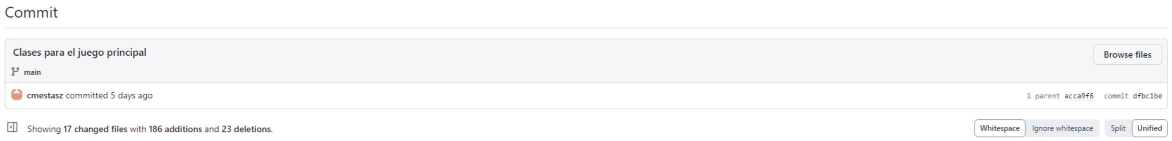
Implementacion de la clase DBConnector

main

cmestasz committed 5 days ago

1 parent e1375c5 commit acca9f6

Showing 5 changed files with 101 additions and 70 deletions.



Commit

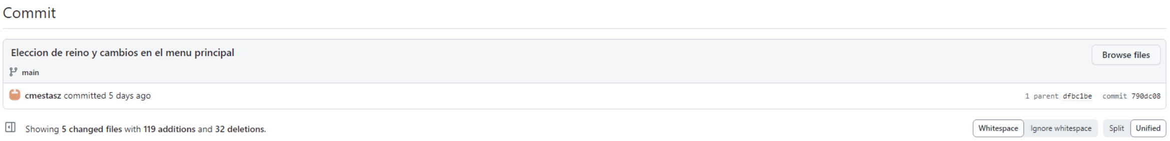
Clases para el juego principal

main

cmestasz committed 5 days ago

1 parent acca9f6 commit dfbc1be

Showing 17 changed files with 186 additions and 23 deletions.



Commit

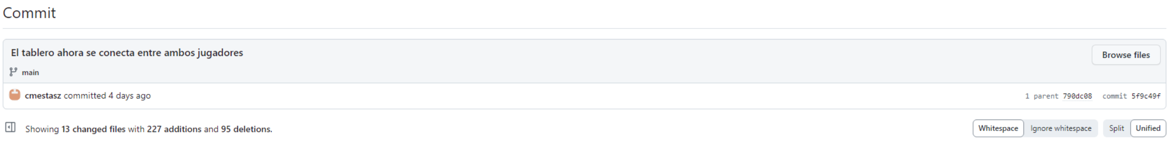
Elección de reino y cambios en el menu principal

main

cmestasz committed 5 days ago

1 parent dfbc1be commit 790dc08

Showing 5 changed files with 119 additions and 32 deletions.



Commit

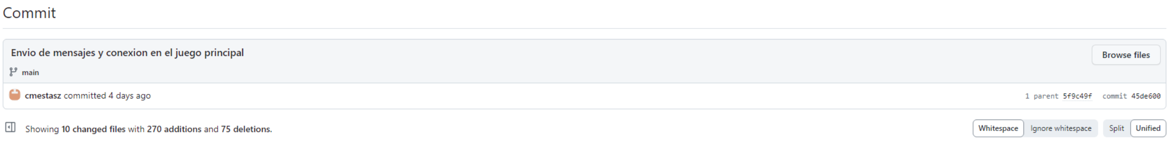
El tablero ahora se conecta entre ambos jugadores

main

cmestasz committed 4 days ago

1 parent 790dc08 commit 5f9c49f

Showing 13 changed files with 227 additions and 95 deletions.



Commit

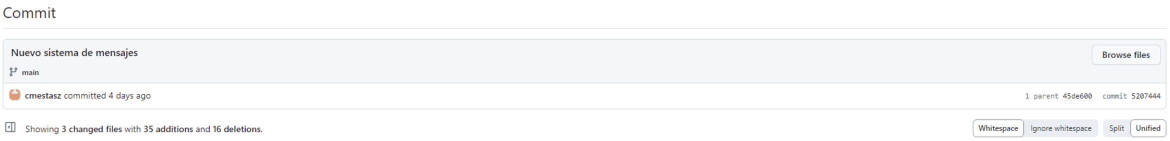
Envio de mensajes y conexion en el juego principal

main

cmestasz committed 4 days ago

1 parent 5f9c49f commit 45de600

Showing 10 changed files with 270 additions and 75 deletions.



Commit

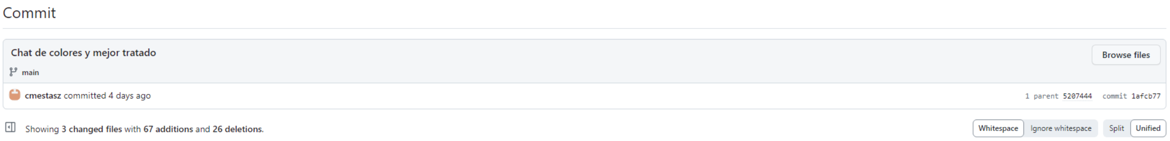
Nuevo sistema de mensajes

main

cmestasz committed 4 days ago

1 parent 45de600 commit 5207444

Showing 3 changed files with 35 additions and 16 deletions.



Commit

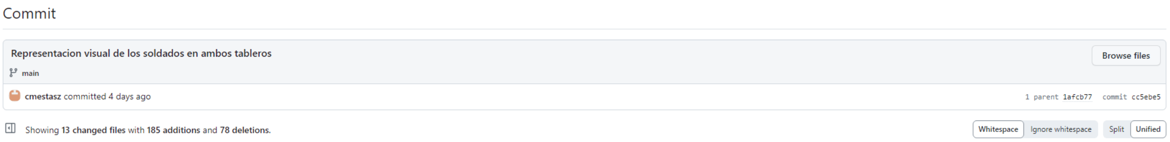
Chat de colores y mejor tratado

main

cmestasz committed 4 days ago

1 parent 5207444 commit 1afcb77

Showing 3 changed files with 67 additions and 26 deletions.



Commit

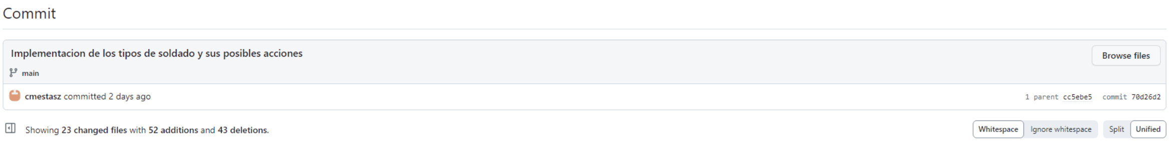
Representacion visual de los soldados en ambos tableros

main

cmestasz committed 4 days ago

1 parent 1afcb77 commit cc5ebe5

Showing 13 changed files with 185 additions and 78 deletions.



Commit

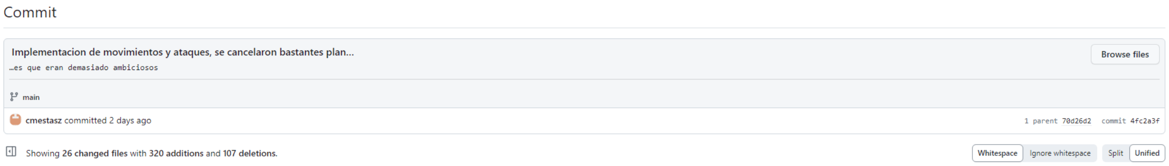
Implementacion de los tipos de soldado y sus posibles acciones

main

cmestasz committed 2 days ago

1 parent cc5ebe5 commit 70d26d2

Showing 23 changed files with 52 additions and 43 deletions.



Commit

Implementacion de movimientos y ataques, se cancelaron bastantes plan…

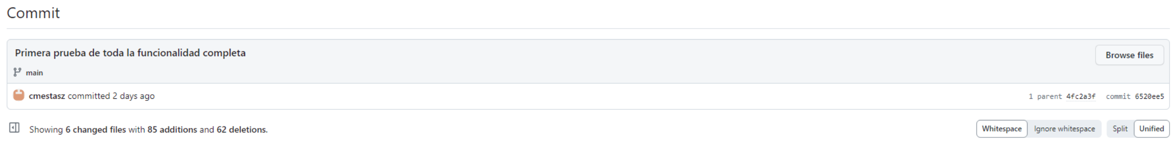
…es que eran demasiado ambiciosos

main

cmestasz committed 2 days ago

1 parent 70d26d2 commit 4fc2a3f

Showing 26 changed files with 320 additions and 107 deletions.

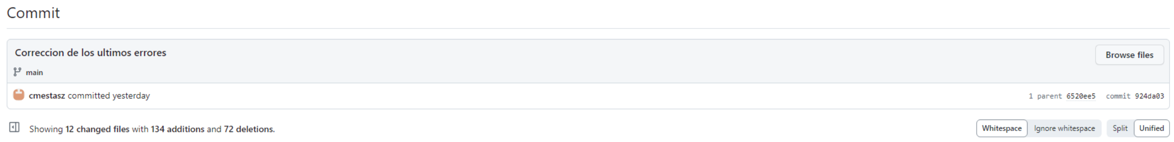

Commit

Primera prueba de toda la funcionalidad completa

main

cmestasz committed 2 days ago — 1 parent 4fc2a3f commit 6520ee5

Showing 6 changed files with 85 additions and 62 deletions.

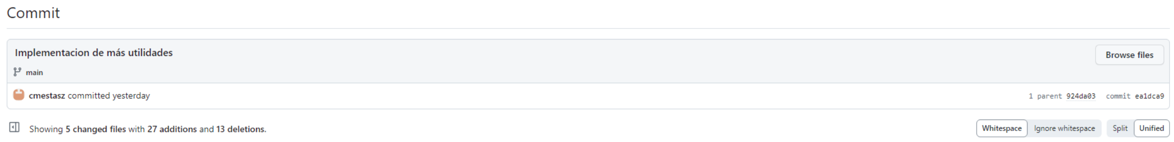

Commit

Correccion de los ultimos errores

main

cmestasz committed yesterday — 1 parent 6520ee5 commit 924da03

Showing 12 changed files with 134 additions and 72 deletions.

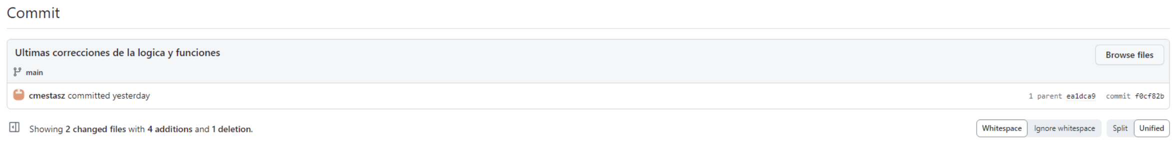

Commit

Implementacion de más utilidades

main

cmestasz committed yesterday — 1 parent 924da03 commit ea1dca9

Showing 5 changed files with 27 additions and 13 deletions.

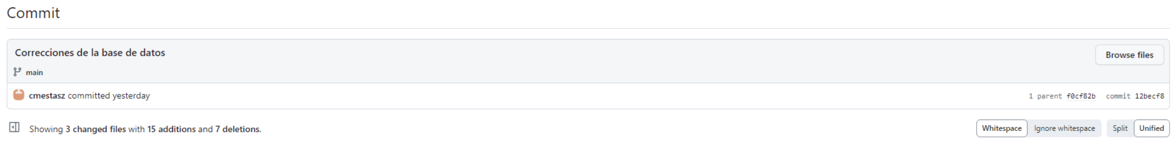

Commit

Ultimas correcciones de la logica y funciones

main

cmestasz committed yesterday — 1 parent ea1dca9 commit f0cf82b

Showing 2 changed files with 4 additions and 1 deletion.

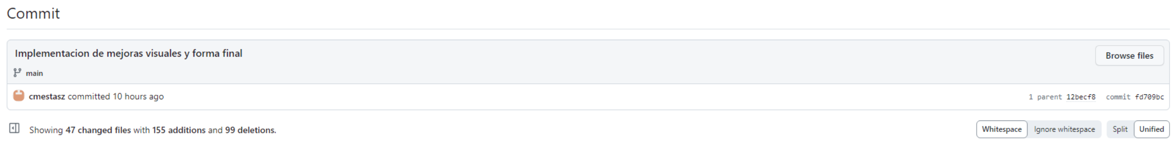

Commit

Correcciones de la base de datos

main

cmestasz committed yesterday — 1 parent f0cf82b commit 12becf8

Showing 3 changed files with 15 additions and 7 deletions.

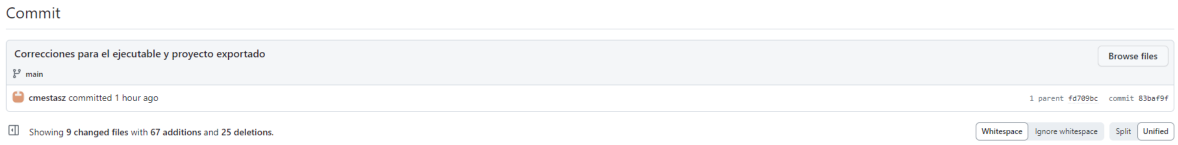

Commit

Implementacion de mejoras visuales y forma final

main

cmestasz committed 10 hours ago — 1 parent 12becf8 commit fd709bc

Showing 47 changed files with 155 additions and 99 deletions.

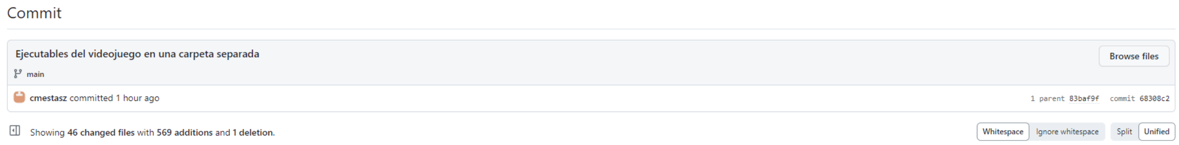

Commit

Correcciones para el ejecutable y proyecto exportado

main

cmestasz committed 1 hour ago — 1 parent fd709bc commit 83baf9f

Showing 9 changed files with 67 additions and 25 deletions.


Commit

Ejecutables del videojuego en una carpeta separada

main

cmestasz committed 1 hour ago — 1 parent 83baf9f commit 68308c2

Showing 46 changed files with 569 additions and 1 deletion.

5. Código desarrollado

5.1. Servidor

MainServer.java

```java

import Utils.*;
import java.io.*;
import java.util.*;
import javax.swing.JOptionPane;
import FX.MainGame.Board;

public class MainServer extends Thread implements MainMenuOperation, MainGameOperation {
    private ArrayList<ServerConnection> connectionsList = new ArrayList<ServerConnection>();
    private HashMap<Integer, Long> lastModifiedMap = new HashMap<Integer, Long>(); // Guarda CUALQUIER archivo que es recibido o cambiado
    private int totalConnections;
    private boolean active = true;
    private int tickRate;

    private HashMap<String, int[]> matches = new HashMap<String, int[]>();

    public MainServer(int tickRate) {
        this.tickRate = tickRate;
    }

    public void run() {
        File directory = new File("connections");
        if (!directory.exists()) {
            directory.mkdirs();
        }
        try {
            while (active) {
                int newTotalConnections = 0;
                for (File file : directory.listFiles())
                    if (file.getName().endsWith(".dat"))
                        newTotalConnections++;
                System.out.println(newTotalConnections);

                if (totalConnections != newTotalConnections) {
                    for (int id = totalConnections; id < newTotalConnections; id++) {
                        // Se crea una nueva conexión y se agrega a la lista.
                        ServerConnection connection = new ServerConnection(totalConnections);
                        System.out.println("connecting: " + connection);
                        connectionsList.add(connection);
                        lastModifiedMap.put(totalConnections, connection.getLastModified());
                    }
                    totalConnections = newTotalConnections;

                }

                for (int id = 0; id < totalConnections; id++)
                    respond(id);

                sleep(tickRate);
            }

            for (ServerConnection connection : connectionsList) {
                connection.deleteDataConnection();
                connection = null;
            }

        } catch (Exception e) {
            e.printStackTrace();
        }
    }

    public void end() {
        active = false;
    }

    private void respond(int id) {
        ServerConnection connection = connectionsList.get(id);
        System.out.println("responding: " + connection);
        long lastModified = connection.getLastModified();

        // Se verifica si la conexión ha sido modificada desde la última respuesta.
        if (lastModifiedMap.get(id) != lastModified) {
            try {
                if (!connection.isInitialized())
                    connection.initialize();

                DataInputStream in = connection.getDataInputStream();
                int operation = in.readInt();
                String code = Utils.readString(in);

                int[] ids;
                int idOther;
                ServerConnection other;
                DataOutputStream toHost;
                DataOutputStream toGuest;
                DataOutputStream toOther;
                switch (operation) {
                    case OPERATION_CREATE:
                        // Se almacena la información de la conexión que ha creado un nuevo código.
                        matches.put(code, new int[] { connection.getId(), -1 });
                        lastModifiedMap.put(id, lastModified);
                        break;

                    case OPERATION_JOIN:
                        // Se intenta unir dos conexiones con el código.
                        Utils.readString(in);
                        Utils.readString(in);
                        int otherId = in.readInt();
                        ids = matches.get(code);
                        toGuest = connection.getDataOutputStream();
                        toGuest.writeInt(RESPONSE_GUEST);
                        if (ids != null && ids[1] == -1) {
                            ServerConnection host = connectionsList.get(ids[0]);
                            ids[1] = connection.getId();
                            Utils.writeString(toGuest, host.getName());
                            Utils.writeString(toGuest, host.getKingdom());

                            toHost = host.getDataOutputStream();
                            toHost.writeInt(RESPONSE_HOST);
                            Utils.writeString(toHost, connection.getName());
                            Utils.writeString(toHost, connection.getKingdom());
                            toHost.writeInt(otherId);

                            toHost.close();
                            lastModifiedMap.put(host.getId(), host.getLastModified());
                        }
                        toGuest.writeChar(0);
                        toGuest.close();
                        lastModifiedMap.put(id, connection.getLastModified());
                        break;

                    case OPERATION_START:
                        // Se inicia la conexión del invitado con el código.
                        ObjectInputStream inObj = connection.getObjectInputStream();
                        Board board = (Board) inObj.readObject();
                        board.invertBoard();
                        inObj.close();
                        connection.deleteObjConnection();

                        int idGuest = matches.get(code)[1];
                        ServerConnection guest = connectionsList.get(idGuest);

                        toGuest = guest.getDataOutputStream();
                        ObjectOutputStream toGuestObj = guest.getObjectOutputStream();
                        toGuest.writeInt(RESPONSE_START);
                        toGuestObj.writeObject(board);
                        toGuest.close();
                        toGuestObj.close();

                        lastModifiedMap.put(idGuest, guest.getLastModified());
                        lastModifiedMap.put(id, connection.getLastModified());
                        break;

                    case OPERATION_CHAT:
                        String message = Utils.readString(in);
                        message.replaceAll("\n", "");
                        ids = matches.get(code);
                        idOther = id == ids[0] ? ids[1] : ids[0];
                        other = connectionsList.get(idOther);

                        toOther = other.getDataOutputStream();
                        toOther.writeInt(RESPONSE_CHAT);
                        Utils.writeString(toOther, message);
                        toOther.close();

                        lastModifiedMap.put(id, connection.getLastModified());
                        lastModifiedMap.put(idOther, other.getLastModified());
                        break;

                    case OPERATION_MOVE:
                    case OPERATION_ATTACK:
                        int sI = in.readInt();
                        int sJ = in.readInt();
                        int oI = in.readInt();
                        int oJ = in.readInt();

                        ids = matches.get(code);
                        idOther = id == ids[0] ? ids[1] : ids[0];
                        other = connectionsList.get(idOther);

                        toOther = other.getDataOutputStream();
                        toOther.writeInt(operation == OPERATION_MOVE ? RESPONSE_MOVE :
                            RESPONSE_ATTACK);
                        Utils.writeIdxs(toOther, sI, sJ, oI, oJ);
                        toOther.close();

                        lastModifiedMap.put(id, connection.getLastModified());
                        lastModifiedMap.put(idOther, other.getLastModified());
                        break;

                }
                in.close();
            } catch (Exception e) {
                e.printStackTrace();
            }
        }
    }

    public static void main(String[] args) {
        new DBConnector();
        int tickRate = Integer.parseInt(JOptionPane.showInputDialog("Ingrese el tiempo entre
            ticks (en milisegundos):"));
        MainServer server = new MainServer(tickRate);
        server.start();
        JOptionPane.showMessageDialog(null, "El servidor esta ejecutandose
            correctamente\nPresione ok para detenerlo");
        server.end();
    }
}
```

- Clase que se encarga de recibir las peticiones de los clientes y responderlas.
- Al momento de abrir una instancia del videojuego, se crea una conección que se conecta con el servidor.
- Mediante el uso de hilos, se responde todas las peticiones de los clientes.

DBConnector.java

```
package Utils;

import java.io.*;
import java.sql.*;

import javax.swing.JOptionPane;

```

```java
public class DBConnector {
    private String user;
    private String password;
    private Connection connection;

    public static void main(String[] args) {
        new DBConnector();
    }

    public DBConnector() {
        try {
            Class.forName("com.mysql.cj.jdbc.Driver");

            Connection connection = readLogin();
            if (connection == null) {
                while (connection == null) {
                    user = JOptionPane.showInputDialog(null, "Ingrese usuario", "Primera
                        conexión",
                            JOptionPane.QUESTION_MESSAGE);
                    password = JOptionPane.showInputDialog(null, "Ingrese clave", "Primera
                        conexión",
                            JOptionPane.QUESTION_MESSAGE);
                    // usuario = "query";
                    // clave = "123456789";
                    connection = login("");
                }
                createDatabase();
                writeLogin();
            }

            // connection.prepareStatement("CREATE DATABASE test").execute();
            // ResultSet resultados = conexion.prepareStatement("SELECT * FROM alumnos WHERE
            // nombre=’Juan’")
            // .executeQuery();

            System.out.println("Conexion exitosa");
        } catch (Exception e) {
            e.printStackTrace();
        }
    }

    private Connection readLogin() throws IOException {
        File dblogin = new File("data/dblogin.dat");
        if (dblogin.exists()) {
            DataInputStream in = new DataInputStream(new FileInputStream(dblogin));
            user = Utils.readString(in);
            password = Utils.readString(in);
            return login("videogame");
        }
        return null;
    }

    private Connection login(String database) {
        String url = "jdbc:mysql://localhost:3306/" + database;
        try {
            connection = DriverManager.getConnection(url, user, password);
            return connection;
        } catch (Exception e) {
            JOptionPane.showMessageDialog(null, "Usuario o clave incorrectos", "Error",
                JOptionPane.ERROR_MESSAGE);
            return null;
        }
    }

    private void writeLogin() throws IOException {
        File directory = new File("data/");
        if (!directory.exists())
            directory.mkdirs();
        DataOutputStream out = new DataOutputStream(new FileOutputStream("data/dblogin.dat"));
        out.writeChars(user);
        out.writeChar(0);
        out.writeChars(password);
        out.writeChar(0);
        out.close();
    }

    private void createDatabase() throws SQLException {
        connection.prepareStatement("CREATE DATABASE videogame;\n").execute();
        connection = login("videogame");
        connection.prepareStatement(
                "CREATE TABLE players (id int NOT NULL AUTO_INCREMENT, name varchar(30) NOT
                    NULL, password varchar(30) NOT NULL, PRIMARY KEY (id))")
                .execute();
        connection.prepareStatement(
                "CREATE TABLE matches (id int NOT NULL AUTO_INCREMENT, winner_id int NOT NULL,
                    loser_id int NOT NULL, PRIMARY KEY (id), INDEX winner_id (winner_id),
                    INDEX loser_id (loser_id))")
                .execute();
        connection.prepareStatement(
                "ALTER TABLE matches ADD CONSTRAINT winner_id FOREIGN KEY (winner_id)
                    REFERENCES players(id) ON DELETE RESTRICT ON UPDATE RESTRICT")
                .execute();
        connection.prepareStatement(
                "ALTER TABLE matches ADD CONSTRAINT loser_id FOREIGN KEY (loser_id) REFERENCES
                    players(id) ON DELETE RESTRICT ON UPDATE RESTRICT")
                .execute();
    }

    public int loginPlayer(String name, String password) {
        try {
            String query = String.format("SELECT id FROM players WHERE name = '%s' AND
                password = '%s'", name,
                   password);
            ResultSet results = connection.prepareStatement(query).executeQuery();
            if (results.next())
                return results.getInt(1);
        } catch (Exception e) {
            e.printStackTrace();
        }
        return -1;
    }

```

```java
    public int[] getWinsLoses(int id) {
        try {
            int[] totals = new int[2];
            String query = String.format("SELECT COUNT(*) from matches WHERE winner_id =
                ’%d’", id);
            ResultSet results = connection.prepareStatement(query).executeQuery();
            if (results.next())
                totals[0] = results.getInt(1);

            query = String.format("SELECT COUNT(*) from matches WHERE loser_id = ’%d’", id);
            results = connection.prepareStatement(query).executeQuery();
            if (results.next())
                totals[1] = results.getInt(1);

            return totals;
        } catch (Exception e) {
            e.printStackTrace();
            return null;
        }
    }

    public void registerPlayer(String name, String password) {
        try {
            String query = String.format("INSERT INTO players (name, password) VALUES (’%s’,
                ’%s’)", name, password);
            connection.prepareStatement(query).execute();
        } catch (Exception e) {
            e.printStackTrace();
        }
    }

    public void createMatch(int winner_id, int loser_id) {
        try {
            String query = String.format("INSERT INTO matches (winner_id, loser_id) VALUES
                (’%d’, ’%d’)", winner_id,
                    loser_id);
            System.out.println(query);
            connection.prepareStatement(query).execute();
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

- Clase que se encarga de conectarse a la base de datos y realizar las operaciones.
- La primera vez que se inicia, crea la base de datos y guarda el usuario que puede acceder en un archivo.
- Posee todos los métodos que permiten interactuar con la base de datos a lo largo de todo el juego.

ServerConnection.java

```java
package Utils;

```

```java
import java.io.*;

public class ServerConnection {
    private int id;
    private String name;
    private String kingdom;
    private File connectionDataFile;
    private File connectionObjFile;
    private boolean initialized;

    public ServerConnection(int id) {
        this.id = id;
        this.connectionDataFile = new File("connections/" + id + ".dat");
        this.connectionObjFile = new File("connections/" + id + ".obj");
    }

    public void initialize() throws IOException {
        DataInputStream in = getDataInputStream();
        in.readInt();
        Utils.readString(in);
        this.name = Utils.readString(in);
        this.kingdom = Utils.readString(in);
        in.close();
        initialized = true;
    }

    public int getId() {
        return id;
    }

    public String getName() {
        return name;
    }

    public String getKingdom() {
        return kingdom;
    }

    public boolean isInitialized() {
        return initialized;
    }

    public File getConnectionDataFile() {
        return connectionDataFile;
    }

    public File getConnectionObjFile() {
        return connectionObjFile;
    }

    public long getLastModified() {
        return connectionDataFile.lastModified();
    }

    public DataInputStream getDataInputStream() throws IOException {
        return new DataInputStream(new FileInputStream(connectionDataFile));
    }

    public ObjectInputStream getObjectInputStream() throws IOException {
        return new ObjectInputStream(new FileInputStream(connectionObjFile));
    }

    public DataOutputStream getDataOutputStream() throws IOException {
        return new DataOutputStream(new FileOutputStream(connectionDataFile));
    }

    public ObjectOutputStream getObjectOutputStream() throws IOException {
        return new ObjectOutputStream(new FileOutputStream(connectionObjFile));
    }

    public void deleteDataConnection() {
        connectionDataFile.delete();
    }

    public void deleteObjConnection() {
        connectionObjFile.delete();
    }

    public String toString() {
        return id + ": " + name;
    }

}
```

- Clase de utilidad para mantener una conección con el servidor.

- Mantiene la id de la conección, el nombre y el reino del jugador.

- Permite generar lectores y escritores de archivos para realizar el envio de datos.

5.2. Videojuego

Archer.java

```java
package FX.MainGame.Classes;

public class Archer extends Soldier {
    private static final int HEALTH = 5;
    private static final int ATTACK = 7;
    private static final int DEFENSE = 3;
    private int arrows = 10;

    public Archer(String name, int team, String type, String typeFile) {
        super(name, team, HEALTH, ATTACK, DEFENSE, type, typeFile);
    }

    // Lamentablemente no llegare a implementar esto
    public void shoot(Soldier other) {
        attack(other);
        arrows--;
    }

    public int getArrows() {
        return arrows;
    }
}
```

- Clase que almacena un arquero, y sus estadísticas.
- Los arqueros tiene un mayor rango de ataque.

Knight.java

```java
package FX.MainGame.Classes;

public class Knight extends Soldier {
    private static final int HEALTH = 3;
    private static final int ATTACK = 10;
    private static final int DEFENSE = 7;
    private boolean mounted = false;

    public Knight(String name, int team, String type, String typeFile) {
        super(name, team, HEALTH, ATTACK, DEFENSE, type, typeFile);
    }

    // Lamentablemente no llegare a implementar esto
    public void mount() {
        mounted = true;
        modifyAttack(1);
        modifyDefense(-1);
    }

    // Lamentablemente no llegare a implementar esto
    public void dismount() {
        mounted = false;
        modifyAttack(-1);
        modifyDefense(1);
    }

    public boolean isMounted() {
        return mounted;
    }
}
```

- Clase que almacena un caballero, y sus estadísticas.
- Los arqueros tiene un mayor rango de movimiento.

Spearman.java

```java
package FX.MainGame.Classes;

public class Spearman extends Soldier {
    private static final int HEALTH = 8;
    private static final int ATTACK = 5;
    private static final int DEFENSE = 10;

    public Spearman(String name, int team, String type, String typeFile) {
        super(name, team, HEALTH, ATTACK, DEFENSE, type, typeFile);
    }

    // Lamentablemente no llegare a implementar esto
    public void schiltrom() {
        modifyDefense(1);
    }
}
```

- Clase que almacena un lancero, y sus estadísticas.

Swordsman.java

```java
package FX.MainGame.Classes;

public class Swordsman extends Soldier {
    private static final int HEALTH = 10;
    private static final int ATTACK = 10;
    private static final int DEFENSE = 8;

    public Swordsman(String name, int team, String type, String typeFile) {
        super(name, team, HEALTH, ATTACK, DEFENSE, type, typeFile);
    }

    // Lamentablemente no llegare a implementar esto
    public void swordDance() {
        modifyAttack(1);
    }
}
```

- Clase que almacena un espadachín, y sus estadísticas.

Soldier.java

```java
package FX.MainGame.Classes;

import java.io.Serializable;

public abstract class Soldier implements Serializable {
    private String name;
    private int team;
    private int initialHealth;
    private int currentHealth;
    private int attack;
    private int defense;
    private String type;
    private String typeFile;

    public Soldier(String name, int team, int initialHealth, int attack, int defense, String
         type, String typeFile) {
        this.name = name;
        this.team = team;
        this.initialHealth = initialHealth;
        this.currentHealth = initialHealth;
        this.attack = attack;
        this.defense = defense;
        this.type = type;
        this.typeFile = typeFile;
    }

    public String getName() {
        return name;
    }

    public int getTeam() {
        return team;
    }

    public int getInitialHealth() {
        return initialHealth;
    }

    public int getCurrentHealth() {
        return currentHealth;
    }

    public int getAttack() {
        return attack;
    }

    public int getDefense() {
        return defense;
    }

    public String getType() {
        return type;
    }

    public String getTypeFile() {
        return typeFile;
    }

    public void heal() {
        currentHealth++;
    }

    public int attack(Soldier other) {
        int damage = Math.max(1, attack - other.getDefense() / 2);
        other.hurt(damage);
        return damage;
    }

    public void hurt(int damage) {
        currentHealth -= damage;
    }

    public void modifyAttack(int change) {
```

```java
        attack += change;
    }

    public void modifyDefense(int change) {
        defense += change;
    }

    public String toString() {
        return name;
    }
}
```

- Clase que almacena un soldado.
- Superclase de todas las clases de soldados.
- Almacena nombre, equipo, vida, ataque, defensa, y tipo.
- Posee métodos para simular el comportamiento de un soldado.

Board.java

```java
package FX.MainGame;

import FX.MainGame.Classes.*;
import Utils.*;
import java.io.Serializable;
import java.util.*;

public class Board implements Serializable, VideogameConstants {
    private final Random RANDOM = new Random();

    private String terrain;
    private String terrainFile;
    private BetterColor background;
    private HashMap<String, Soldier> army1 = new HashMap<String, Soldier>();
    private HashMap<String, Soldier> army2 = new HashMap<String, Soldier>();
    private String kingdomPlayer;
    private String kingdomEnemy;

    public Board(String kingdom1, String kingdom2) {
        int idxTerrain = RANDOM.nextInt(TERRAINS.length);
        terrain = TERRAINS[idxTerrain];
        terrainFile = TERRAIN_FILES[idxTerrain];

        this.kingdomPlayer = kingdom1;
        this.kingdomEnemy = kingdom2;

        initSoldiers(army1, 1);
        initSoldiers(army2, 2);
    }

    public void invertBoard() {
        HashMap<String, Soldier> armyt = army1;
        army1 = army2;
        army2 = armyt;

        String kingdomt = kingdomPlayer;
        kingdomPlayer = kingdomEnemy;
        kingdomEnemy = kingdomt;
    }

    public HashMap<String, Soldier> getArmy1() {
        return army1;
    }

    public HashMap<String, Soldier> getArmy2() {
        return army2;
    }

    public String getTerrain() {
        return terrain;
    }

    public String getTerrainFile() {
        return terrainFile;
    }

    public String getKingdomPlayer() {
        return kingdomPlayer;
    }

    public String getKingdomEnemy() {
        return kingdomEnemy;
    }

    public BetterColor getBackground() {
        return background;
    }

    private void initSoldiers(HashMap<String, Soldier> map, int team) {
        for (int i = 0; i < TOTAL_SOLDIERS; i++) {
            int idx = RANDOM.nextInt(TYPES.length);
            String type = TYPES[idx];
            String fileType = TYPE_FILES[idx];
            int row, col;
            String key;
            do {
                row = RANDOM.nextInt(SIZE);
                col = RANDOM.nextInt(SIZE);
                key = generateKey(row, col);
            } while (army1.containsKey(key) || army2.containsKey(key));
            String name = type + i + "X" + team;
            Soldier soldier = null;
            switch (type) {
                case "CABALLERO":
                    soldier = new Knight(name, team, type, fileType);
                    break;
                case "ARQUERO":
                    soldier = new Archer(name, team, type, fileType);
                    break;
                case "ESPADACHIN":
```

```java
                    soldier = new Swordsman(name, team, type, fileType);
                    break;
                case "LANCERO":
                    soldier = new Spearman(name, team, type, fileType);
                    break;
            }
            map.put(key, soldier);
        }
    }

    private String generateKey(int i, int j) {
        return i + "," + j;
    }

    public String toString() {
        return "a board!";
    }
}
```

- Clase que almacena un tablero.
- Almacena el terreno, los ejércitos y los reinos.
- Es serializable para permitir ser enviado entre ambos jugadores al momento de iniciar el juego.

MainGameController.java

```java
package FX.MainGame;

import java.io.*;
import java.util.HashMap;
import FX.MainGame.Classes.Soldier;
import FX.MainMenu.MainMenuController;
import Utils.*;
import javafx.application.Platform;
import javafx.collections.ObservableList;
import javafx.fxml.FXML;
import javafx.scene.Node;
import javafx.scene.control.*;
import javafx.scene.image.*;
import javafx.scene.input.MouseEvent;
import javafx.scene.layout.*;
import javafx.scene.text.*;
import javafx.stage.Stage;

public class MainGameController implements MainGameOperation, VideogameConstants {
    private Stage gameStage;
    private Stage menuStage;
    private Resolution resolution;
    private int width;
    private int height;
    private MainMenuController menuController;
    private Board board;
    private String kingdomPlayer;
    private String kingdomEnemy;
    private File connectionFile;
    private String path;
    private int idConnection;
    private int idPlayer;
    private int idEnemy;
    private DataReceiver dataReceiver;
    private String matchCode;
    private String pName;
    private String eName;
    private Tile[][] tiles = new Tile[SIZE][SIZE];
    private String selectedAction = "MOVER";
    private Tile selectedTile;
    private HashMap<String, Soldier> army1;
    private HashMap<String, Soldier> army2;
    private DBConnector dbConnector;
    private boolean gameEnded;
    private boolean playerTurn = true;

    @FXML
    private GridPane uiBoard;
    @FXML
    private ImageView boardBackground;
    @FXML
    private ImageView dataBackground;
    @FXML
    private TextArea playerData;
    @FXML
    private TextArea enemyData;
    @FXML
    private ScrollPane chatOutputPane;
    @FXML
    private VBox chatOutput;
    @FXML
    private TextField chatInput;
    @FXML
    private TilePane actionsPane;
    @FXML
    private Pane messagePane;
    @FXML
    private TextArea messageOutput;
    @FXML
    private VBox moveActionPane;
    @FXML
    private VBox attackActionPane;

    public void init(MainMenuController menuController, Resolution resolution, Stage menuStage, Stage gameStage,
            Board board,
            int idConnection, String matchCode, String pName, String eName, int idPlayer, int idEnemy) {
        this.menuController = menuController;
        this.resolution = resolution;
        this.width = resolution.getWidth();
        this.height = resolution.getHeight();
        this.menuStage = menuStage;
        this.gameStage = gameStage;
        this.board = board;
        army1 = board.getArmy1();
        army2 = board.getArmy2();
        this.idConnection = idConnection;
        this.kingdomPlayer = board.getKingdomPlayer();
        this.kingdomEnemy = board.getKingdomEnemy();
        this.matchCode = matchCode;
        this.pName = pName;
        this.eName = eName;
        this.idPlayer = idPlayer;
        this.idEnemy = idEnemy;

        initButtons();
        initBackground();
        initDataFields();
        initChat();

        actionsPane.setPrefWidth(width * 0.15);
        actionsPane.setPrefHeight(width * 0.05);
        setStyleColor(moveActionPane, SELECTED_COLOR);

        dbConnector = new DBConnector();
        setConnection();
    }

    public void initialize() {

    }

    public void sendMessage() {
        String message = String.format("%s: %s%n", pName, chatInput.getText());
        printMessage(message, PLAYER_COLOR);
        try {
            DataOutputStream out = new DataOutputStream(new FileOutputStream(connectionFile));
            out.writeInt(OPERATION_CHAT);
            Utils.writeStrings(out, new String[] { matchCode, message });
            out.close();
        } catch (Exception e) {
            e.printStackTrace();
        }
        chatInput.setText("");
    }

    public void setActionMove() {
        setStyleColor(moveActionPane, SELECTED_COLOR);
        setStyleColor(attackActionPane, null);
        selectedAction = "MOVER";
    }

    public void setActionAttack() {
        setStyleColor(attackActionPane, SELECTED_COLOR);
        setStyleColor(moveActionPane, null);
        selectedAction = "ATACAR";
    }

    public void closeMessage() {
        messagePane.setVisible(false);
```

```java
        if (gameEnded) {
            dataReceiver.endGame();
            menuStage.show();
            menuController.restartMenu();
            gameStage.close();
        }
    }

    private void initButtons() {
        for (int i = 0; i < SIZE; i++) {
            for (int j = 0; j < SIZE; j++) {
                String key = generateKey(i, j);
                double size = 1.0 * resolution.getHeight() / SIZE;
                HashMap<String, Soldier> army1 = board.getArmy1();
                HashMap<String, Soldier> army2 = board.getArmy2();
                Tile tile;
                Soldier soldier;

                if (army1.containsKey(key)) {
                    soldier = army1.get(key);
                    tile = new Tile(soldier.getCurrentHealth(), soldier.getTypeFile(), size,
                        i, j);
                    setStyleColor(tile, PLAYER_COLOR_TRANS);
                } else if (army2.containsKey(key)) {
                    soldier = army2.get(key);
                    tile = new Tile(soldier.getCurrentHealth(), soldier.getTypeFile(), size,
                        i, j);
                    setStyleColor(tile, ENEMY_COLOR_TRANS);
                } else {
                    tile = new Tile(0, "tile", size, i, j);
                }
                tiles[i][j] = tile;

                tile.setOnMouseClicked(this::handleClick);
                uiBoard.add(tile, i, j);
            }
        }
    }

    private void setStyleColor(Region pane, BetterColor color) {
        if (color == null) {
            pane.setStyle("-fx-background-color: none;");
        } else {
            pane.setStyle(String.format("-fx-background-color: %s;", color.getRGBA()));
        }
    }

    private void initBackground() {
        boardBackground.setFitWidth(width);
        boardBackground.setFitHeight(height);
        boardBackground.setImage(new Image(String.format("img/background_%s.png",
            board.getTerrainFile())));

        dataBackground.setFitWidth(width - height);
        dataBackground.setFitHeight(height);
        dataBackground.setImage(new Image("img/background_data.png"));
    }

    private void initDataFields() {
        playerData.setText(String.format("%s: %s%n", pName, kingdomPlayer));
        enemyData.setText(String.format("%s: %s%n", eName, kingdomEnemy));
    }

    private void initChat() {
        chatOutput.setPrefHeight(height * 0.4);
    }

    private void handleClick(MouseEvent event) {
        Tile tile = (Tile) event.getSource();

        if (!tryDoAction(tile)) {
            if (board.getArmy1().containsKey(tile.getKey())) {
                selectedTile = tile;
                showActionsMenu();
            } else {
                selectedTile = null;
                removeActionsMenu();
            }
        }
    }

    private void showActionsMenu() {
        actionsPane.setVisible(true);
    }

    private void removeActionsMenu() {
        actionsPane.setVisible(false);
    }

    private boolean tryDoAction(Tile otherTile) {
        if (selectedTile != null && playerTurn) {
            String otherKey = otherTile.getKey();
            try {
                DataOutputStream out;
                int sI = selectedTile.getI();
                int sJ = selectedTile.getJ();
                int oI = otherTile.getI();
                int oJ = otherTile.getJ();
                Soldier selectedSoldier = army1.get(generateKey(sI, sJ));
                int distance;
                switch (selectedAction) {
                    case "MOVER":
                        distance = selectedSoldier.getTypeFile().equals("knight") ? 2 : 1;
                        if (selectedTile.isConnected(otherTile, distance) &&
                             !army1.containsKey(otherKey)
                                && !army2.containsKey(otherKey)) {
                            moveSoldier(true, sI, sJ, oI, oJ);
                            removeActionsMenu();

                            out = new DataOutputStream(new FileOutputStream(connectionFile));
                            out.writeInt(OPERATION_MOVE);
                            Utils.writeString(out, matchCode);
                            Utils.writeIdxs(out, sI, sJ, oI, oJ);
                            out.close();

                            playerTurn = false;
                            selectedTile = null;
                            out.close();
                            return true;
                        }
                        showMessage("Movimiento no valido.");
                        break;
                    case "ATACAR":
                        distance = selectedSoldier.getTypeFile().equals("archer") ? 2 : 1;
                        if (selectedTile.isConnected(otherTile, distance) && army2.containsKey(otherKey)) {
                            attackSoldier(true, sI, sJ, oI, oJ);
                            removeActionsMenu();

                            out = new DataOutputStream(new FileOutputStream(connectionFile));
                            out.writeInt(OPERATION_ATTACK);
                            Utils.writeString(out, matchCode);
                            Utils.writeIdxs(out, sI, sJ, oI, oJ);
                            out.close();

                            playerTurn = false;
                            selectedTile = null;
                            out.close();
                            return true;
                        }
                        showMessage("Ataque no valido.");
                        break;
                }
            } catch (Exception e) {
                e.printStackTrace();
            }
        }
        return false;
    }

    // Métodos que funcionan en ambos sentidos, host -> guest o guest -> host
    private void moveSoldier(boolean isPlayer, int iSelected, int jSelected, int iOther, int jOther) {
        Tile selectedTile = tiles[iSelected][jSelected];
        Tile otherTile = tiles[iOther][jOther];
        String selectedKey = selectedTile.getKey();
        String otherKey = otherTile.getKey();

        HashMap<String, Soldier> army = null;
        BetterColor color = null;
        if (isPlayer) {
            army = army1;
            color = PLAYER_COLOR_TRANS;
        } else {
            army = army2;
            color = ENEMY_COLOR_TRANS;
        }

```

```java
        Soldier soldier = army.remove(selectedKey);
        army.put(otherKey, soldier);
        selectedTile.setImageAndhealth("tile", 0);
        setStyleColor(selectedTile, null);
        otherTile.setImageAndhealth(soldier.getTypeFile(), soldier.getCurrentHealth());
        setStyleColor(otherTile, color);

        String message = soldier + " se mueve." + "\n";
        if (isPlayer)
            playerData.appendText(message);
        else
            enemyData.appendText(message);
    }

    private void attackSoldier(boolean isPlayer, int iSelected, int jSelected, int iOther, int jOther) {
        Tile selectedTile = tiles[iSelected][jSelected];
        Tile otherTile = tiles[iOther][jOther];
        String selectedKey = selectedTile.getKey();
        String otherKey = otherTile.getKey();

        Soldier soldierAttacks = null;
        Soldier soldierReceives = null;
        if (isPlayer) {
            soldierAttacks = army1.get(selectedKey);
            soldierReceives = army2.get(otherKey);
        } else {
            soldierAttacks = army2.get(selectedKey);
            soldierReceives = army1.get(otherKey);
        }

        int damage = soldierAttacks.attack(soldierReceives);
        otherTile.setImageAndhealth(soldierReceives.getTypeFile(), soldierReceives.getCurrentHealth());
        String message = String.format("%s ataca a %s con %d de daño%n", soldierAttacks, soldierReceives, damage);
        if (isPlayer)
            playerData.appendText(message);
        else
            enemyData.appendText(message);

        if (soldierReceives.getCurrentHealth() <= 0) {
            soldierAttacks.heal();
            selectedTile.setImageAndhealth(soldierAttacks.getTypeFile(), soldierAttacks.getCurrentHealth());
            otherTile.setImageAndhealth("tile", 0);
            setStyleColor(otherTile, null);

            message = soldierReceives + " ha muerto!\n";
            if (isPlayer) {
                playerData.appendText(message);
                army2.remove(otherKey);
                if (army2.size() == 0) {
                    if (idEnemy != 0)
                        dbConnector.createMatch(idPlayer, idEnemy);
                    endGame(pName, kingdomPlayer);
                }
            } else {
                enemyData.appendText(message);
                army1.remove(otherKey);
                if (army1.size() == 0) {
                    if (idEnemy != 0)
                        dbConnector.createMatch(idEnemy, idPlayer);
                    endGame(eName, kingdomEnemy);
                }
            }
        }

    }

    private void endGame(String name, String kingdom) {
        showMessage(String.format("%s ha ganado con el reino %s!", name, kingdom));
        gameEnded = true;
    }

    private void printMessage(String message, BetterColor color) {
        Text messageText = new Text(message);
        messageText.setFont(Font.font("Book Antiqua"));
        messageText.setFill(color.getColor());
        messageText.setWrappingWidth(width - height);

        ObservableList<Node> children = chatOutput.getChildren();
        children.add(children.size() - 1, messageText);
        chatOutputPane.setVvalue(1);
    }

    private void setConnection() {
        path = "connections/" + idConnection + ".dat";
        connectionFile = new File(path);
        try {
            connectionFile.createNewFile();
            dataReceiver = new DataReceiver();
            dataReceiver.start();
        } catch (Exception e) {
            e.printStackTrace();
        }
    }

    private String generateKey(int i, int j) {
        return i + "," + j;
    }

    private void showMessage(String message) {
        messagePane.setVisible(true);
        messageOutput.setText(message);
    }

    private class DataReceiver extends Thread {
        private File matchFile = new File(path);
        private long lastModified = matchFile.lastModified();
        private boolean gameEnded;

        public void run() {
            try {
                while (!gameEnded) {
                    // Comprueba si el archivo de la partida ha sido modificado
                    if (matchFile.lastModified() != lastModified) {
                        DataInputStream in = new DataInputStream(new
                             FileInputStream(matchFile));
                        int response = in.readInt();
                        switch (response) {
                            // Mensaje de chat
                            case RESPONSE_CHAT:
                                String message = Utils.readString(in);
                                Platform.runLater(() -> {
                                    printMessage(message, ENEMY_COLOR);
                                });
                                break;

                            // Movimientos y ataques
                            case RESPONSE_MOVE:
                            case RESPONSE_ATTACK:
                                int sI = in.readInt();
                                int sJ = in.readInt();
                                int oI = in.readInt();
                                int oJ = in.readInt();

                                Platform.runLater(() -> {
                                    showActionsMenu();
                                    playerTurn = true;
                                    if (response == RESPONSE_MOVE)
                                        moveSoldier(false, sI, sJ, oI, oJ);
                                    else
                                        attackSoldier(false, sI, sJ, oI, oJ);
                                });
                                break;
                        }
                        lastModified = matchFile.lastModified();
                        in.close();
                    }

                    sleep(500);
                }
            } catch (Exception e) {
                e.printStackTrace();
            }
        }

        public void endGame() {
            gameEnded = true;
        }
    }
}
```

- Clase que controla el videojuego principal, se encarga de la manipulación de todos los elementos FX.
- Realiza la inicialización de todos los elementos del escenario.

- Posee métodos que permiten enviar al servidor los datos tanto de chats como de movimientos.
- Posee la clase interna DataReceiver que permite recibir las respuestas del servidor y ejecutarlas.

MainGame.fxml

```xml
<?xml version="1.0" encoding="UTF-8"?>

<?import javafx.geometry.Insets?>
<?import javafx.scene.control.Button?>
<?import javafx.scene.control.Label?>
<?import javafx.scene.control.ScrollPane?>
<?import javafx.scene.control.TextArea?>
<?import javafx.scene.control.TextField?>
<?import javafx.scene.control.TitledPane?>
<?import javafx.scene.image.Image?>
<?import javafx.scene.image.ImageView?>
<?import javafx.scene.layout.AnchorPane?>
<?import javafx.scene.layout.ColumnConstraints?>
<?import javafx.scene.layout.GridPane?>
<?import javafx.scene.layout.HBox?>
<?import javafx.scene.layout.Pane?>
<?import javafx.scene.layout.RowConstraints?>
<?import javafx.scene.layout.StackPane?>
<?import javafx.scene.layout.TilePane?>
<?import javafx.scene.layout.VBox?>
<?import javafx.scene.text.Font?>
<?import javafx.scene.text.Text?>

<StackPane xmlns="http://javafx.com/javafx/21" xmlns:fx="http://javafx.com/fxml/1" fx:controller="FX.MainGame.MainGameController">
   <children>
      <ImageView fx:id="boardBackground" fitHeight="150.0" fitWidth="200.0" pickOnBounds="true" preserveRatio="true" />
      <HBox>
         <children>
            <GridPane fx:id="uiBoard" maxHeight="-Infinity" maxWidth="-Infinity" minHeight="-Infinity" minWidth="-Infinity">
              <columnConstraints>
                  <ColumnConstraints hgrow="SOMETIMES" />
                  <ColumnConstraints hgrow="SOMETIMES" />
                  <ColumnConstraints hgrow="SOMETIMES" />
                  <ColumnConstraints hgrow="SOMETIMES" />
                  <ColumnConstraints hgrow="SOMETIMES" />
                  <ColumnConstraints hgrow="SOMETIMES" />
                  <ColumnConstraints hgrow="SOMETIMES" />
                  <ColumnConstraints hgrow="SOMETIMES" />
                <ColumnConstraints hgrow="SOMETIMES" />
                <ColumnConstraints hgrow="SOMETIMES" />
              </columnConstraints>
              <rowConstraints>
                  <RowConstraints vgrow="SOMETIMES" />
                  <RowConstraints vgrow="SOMETIMES" />
                  <RowConstraints vgrow="SOMETIMES" />
                  <RowConstraints vgrow="SOMETIMES" />
                  <RowConstraints vgrow="SOMETIMES" />
                  <RowConstraints vgrow="SOMETIMES" />
                  <RowConstraints vgrow="SOMETIMES" />
                <RowConstraints vgrow="SOMETIMES" />
                <RowConstraints vgrow="SOMETIMES" />
                <RowConstraints vgrow="SOMETIMES" />
              </rowConstraints>
            </GridPane>
            <StackPane>
               <children>
                  <ImageView fx:id="dataBackground" fitHeight="150.0" fitWidth="200.0"
                      pickOnBounds="true" preserveRatio="true" />
                  <VBox>
                     <children>
                        <HBox>
                          <children>
                             <TextArea fx:id="playerData" blendMode="MULTIPLY">
                                <font>
                                  <Font name="Book Antiqua" size="10.0" />
                                </font></TextArea>
                             <TextArea fx:id="enemyData" blendMode="MULTIPLY">
                                <font>
                                  <Font name="Book Antiqua" size="10.0" />
                                </font></TextArea>
                          </children>
                        </HBox>
                        <VBox alignment="CENTER">
                          <children>
                             <ScrollPane fx:id="chatOutputPane" blendMode="MULTIPLY"
                                 hbarPolicy="NEVER" vbarPolicy="ALWAYS">
                                <content>
                                  <VBox fx:id="chatOutput" style="-fx-background-color:
                                       white;">
                                     <children>
                                        <Text strokeType="OUTSIDE" strokeWidth="0.0">
                                           <font>
                                             <Font size="18.0" />
                                           </font>
                                        </Text>
                                     </children>
                                  </VBox>
                                </content>
                             </ScrollPane>
                             <TextField fx:id="chatInput" blendMode="MULTIPLY"
                                 onAction="#sendMessage" promptText="Envia un mensaje!">
                                <VBox.margin>
                                  <Insets bottom="15.0" />
                                </VBox.margin>
                                <font>
                                  <Font name="Book Antiqua" size="12.0" />
                                </font></TextField>
                             <TilePane fx:id="actionsPane" alignment="CENTER" hgap="30.0">
                                <children>
                                  <VBox fx:id="moveActionPane" onMouseClicked="#setActionMove"
                                       TilePane.alignment="CENTER">
                                     <children>
                                        <Label alignment="CENTER" prefWidth="50.0" text="Mover">
                                           <font>
```

```
                                               <Font name="Book Antiqua" size="12.0" />
                                            </font></Label>
                                         <ImageView fitHeight="50.0" fitWidth="50.0"
                                             pickOnBounds="true" preserveRatio="true">
                                            <image>
                                               <Image url="@../../img/action_move.png" />
                                            </image>
                                         </ImageView>
                                      </children>
                                   </VBox>
                                   <VBox fx:id="attackActionPane"
                                       onMouseClicked="#setActionAttack"
                                       TilePane.alignment="CENTER">
                                      <children>
                                         <Label alignment="CENTER"
                                             maxWidth="1.7976931348623157E308" prefWidth="50.0"
                                             text="Atacar">
                                            <font>
                                               <Font name="Book Antiqua" size="12.0" />
                                            </font></Label>
                                         <ImageView fitHeight="50.0" fitWidth="50.0"
                                             pickOnBounds="true" preserveRatio="true">
                                            <image>
                                               <Image url="@../../img/action_attack.png" />
                                            </image>
                                         </ImageView>
                                      </children>
                                   </VBox>
                                </children>
                             </TilePane>
                          </children>
                       </VBox>
                    </children>
                 </VBox>
              </children>
           </StackPane>
        </children>
     </HBox>
     <Pane fx:id="messagePane" visible="false">
        <children>
           <TitledPane animated="false" collapsible="false" layoutX="274.0" layoutY="74.0"
               prefHeight="139.0" prefWidth="279.0" text="Mensaje">
              <content>
                 <AnchorPane minHeight="0.0" minWidth="0.0" prefHeight="180.0"
                     prefWidth="200.0">
                    <children>
                       <TextArea fx:id="messageOutput" layoutX="-1.0" prefHeight="81.0"
                           prefWidth="279.0" wrapText="true">
                          <font>
                             <Font name="Book Antiqua" size="14.0" />
                          </font>
                       </TextArea>
                       <Button layoutX="122.0" layoutY="84.0" mnemonicParsing="false"
                           onAction="#closeMessage" text="OK">
                          <font>
                             <Font name="Book Antiqua" size="12.0" />
                           </font>
                        </Button>
                     </children>
                  </AnchorPane>
               </content>
               <font>
                  <Font name="Book Antiqua" size="12.0" />
               </font>
            </TitledPane>
         </children>
      </Pane>
   </children>
</StackPane>
```

- Clase FXML de JavaFX que posee el juego principal.
- Posee las maquetas que luego son rellenadas con el controlador para el juego principal.

MainMenuController.java

```java
package FX.MainMenu;

import Utils.*;
import java.io.*;
import FX.MainGame.Board;
import FX.MainGame.MainGameController;
import javafx.application.*;
import javafx.collections.*;
import javafx.fxml.*;
import javafx.scene.*;
import javafx.scene.control.*;
import javafx.scene.layout.Pane;
import javafx.stage.Stage;

public class MainMenuController implements MainMenuOperation {
    private final ObservableList<Resolution> RESOLUTIONS =
        FXCollections.observableArrayList();
    private final ObservableList<String> KINGDOMS = FXCollections.observableArrayList();
    private final int CODE_LENGTH = 6;

    private String pName;
    private String eName;
    private String pKingdom;
    private String eKingdom;
    private Resolution resolution;
    private int idConnection;
    private int idPlayer;
    private int idEnemy;
    private String path;
    private File connectionFile;
    private DataReceiver dataReceiver;
    private Stage stage;
    private DBConnector dbConnector;
    private Board board;
    private String matchCode;

    @FXML
    private TextField nameInput;
    @FXML
    private TextField passwordInput;
    @FXML
    private Pane settingsPane;
    @FXML
    private ComboBox<Resolution> resolutionInput;
    @FXML
    private ComboBox<String> kingdomInput;
    @FXML
    private TextField createMatchCode;
    @FXML
    private TextField joinMatchCode;
    @FXML
    private Label playerName;
    @FXML
    private Label enemyName;
    @FXML
    private Label playerKingdom;
    @FXML
    private Label enemyKingdom;
    @FXML
    private Button startButton;
    @FXML
    private TitledPane messagePane;
    @FXML
    private TextArea messageOutput;

    public void setStage(Stage stage) {
        this.stage = stage;
    }

    public void initialize() throws IOException {
        try {
            RESOLUTIONS.addAll(new Resolution(850, 480), new Resolution(1280, 720), new
                Resolution(1366, 768),
                   new Resolution(1920, 1080));
            resolutionInput.setItems(RESOLUTIONS);
            resolutionInput.setValue(RESOLUTIONS.get(0));
            resolution = resolutionInput.getValue();

            KINGDOMS.addAll("INGLATERRA", "FRANCIA", "CASTILLA-ARAGÓN", "MOROS", "SACRO
                IMPERIO");
            kingdomInput.setItems(KINGDOMS);

            dbConnector = new DBConnector();
            setConnection();
        } catch (Exception e) {
            FileWriter writer = new FileWriter("error.log");
            writer.write(e.getMessage());
            writer.close();
        }
    }

    public void setKingdom() {
        pKingdom = kingdomInput.getValue();
        playerKingdom.setText(pKingdom);
    }

    public void toggleSettings() {
        settingsPane.setVisible(!settingsPane.isVisible());
    }

    public void setResolution() {
        resolution = resolutionInput.getValue();
    }

    public void createMatch() {
        if (checkName() && checkKingdom()) {
            matchCode = "";
            for (int i = 0; i < CODE_LENGTH; i++)
                matchCode += (char) ('A' + (int) (Math.random() * 26));

            createMatchCode.setText(matchCode);
            try {
                DataOutputStream out = new DataOutputStream(new
                    FileOutputStream(connectionFile));
                out.writeInt(OPERATION_CREATE);
                Utils.writeStrings(out, new String[] { matchCode, pName, pKingdom });
                out.close();
            } catch (Exception e) {
                e.printStackTrace();
            }
        }
    }

    public void joinMatch() {
        matchCode = joinMatchCode.getText();
        if (checkName() && checkKingdom() && matchCode.length() == CODE_LENGTH) {
            try {
                DataOutputStream out = new DataOutputStream(new
                    FileOutputStream(connectionFile));
                out.writeInt(OPERATION_JOIN);
                Utils.writeStrings(out, new String[] { matchCode, pName, pKingdom });
                out.writeInt(idPlayer);
            } catch (Exception e) {
                e.printStackTrace();
            }
        }
    }

    public void startMatch() {
        if (checkName() && checkEnemy() && checkKingdom()) {
            try {
                DataOutputStream out = new DataOutputStream(new
                    FileOutputStream(connectionFile));
                out.writeInt(OPERATION_START);
                Utils.writeString(out, matchCode);
                out.close();

                board = new Board(pKingdom, eKingdom);
```

```java
                ObjectOutputStream outObj = new ObjectOutputStream(
                        new FileOutputStream("connections/" + idConnection + ".obj"));
                outObj.writeObject(board);
                outObj.close();
            } catch (Exception e) {
                e.printStackTrace();
            }

            createGameStage();
        }
    }

    public void login() {
        String name = nameInput.getText();
        idPlayer = dbConnector.loginPlayer(name, passwordInput.getText());
        if (idPlayer == -1) {
            showMessage("Usuario no encontrado.");
        } else {
            showMessage("Acceso correcto.");
            pName = name;
            playerName.setText(pName);
            nameInput.setText("");
            passwordInput.setText("");
        }
    }

    public void register() {
        pName = nameInput.getText();
        dbConnector.registerPlayer(pName, passwordInput.getText());
        showMessage("Usuario creado correctamente.");
        login();
    }

    public void getStatistics() {
        if (checkName()) {
            int[] status = dbConnector.getWinsLoses(idPlayer);
            showMessage(String.format("W: %d | L: %d", status[0], status[1]));
        }
    }

    public void closeMessage() {
        messagePane.setVisible(false);
    }

    public void restartMenu() {
        createMatchCode.setText("");
        joinMatchCode.setText("");
        enemyName.setText("");
        enemyKingdom.setText("");
        startButton.setDisable(false);
        dataReceiver = new DataReceiver();
        dataReceiver.start();
    }

    private void setConnection() {
        if (connectionFile == null) {
            path = "connections/" + idConnection + ".dat";
            connectionFile = new File(path);
            while (connectionFile.exists()) {
                idConnection++;
                path = "connections/" + idConnection + ".dat";
                connectionFile = new File(path);
            }
        }
        try {
            connectionFile.createNewFile();
            dataReceiver = new DataReceiver();
            dataReceiver.start();
        } catch (Exception e) {
            e.printStackTrace();
        }
    }

    private boolean checkName() {
        boolean nameSet = pName != null;
        if (!nameSet)
            showMessage("Crea o accede a tu cuenta!");
        return nameSet;
    }

    private boolean checkEnemy() {
        boolean enemySet = eName != null;
        if (!enemySet)
            showMessage("Crea o únete a una partida!");
        return enemySet;
    }

    private boolean checkKingdom() {
        boolean kingdomSet = pKingdom != null;
        if (!kingdomSet)
            showMessage("Escoge un reino!");
        return kingdomSet;
    }

    private void createGameStage() {
        dataReceiver.startGame();
        stage.hide();
        new MainGame(this);
    }

    private void showMessage(String message) {
        messagePane.setVisible(true);
        messageOutput.setText(message);
    }

    // Clase interna para el receptor de datos en un hilo separado
    private class DataReceiver extends Thread {
        private File matchFile = new File(path);
        private long lastModified = matchFile.lastModified();
        private boolean gameStarted;

        public void run() {
            try {
                while (!gameStarted) {
                    // Comprueba si el archivo de la partida ha sido modificado
                    if (matchFile.lastModified() != lastModified) {
                        DataInputStream in = new DataInputStream(new
                            FileInputStream(matchFile));
                        int response = in.readInt();
                        String name, kingdom;
                        switch (response) {
                            // Respuesta del anfitrión
                            case RESPONSE_HOST:
                                name = Utils.readString(in);
                                kingdom = Utils.readString(in);
                                int idOther = in.readInt();
                                // Actualiza el nombre del oponente en la interfaz de usuario
                                Platform.runLater(() -> {
                                    setEnemy(name, kingdom);
                                    idEnemy = idOther;
                                });
                                break;
                            // Respuesta del invitado
                            case RESPONSE_GUEST:
                                name = Utils.readString(in);
                                kingdom = Utils.readString(in);
                                if (name.equals("")) {
                                    showMessage("La partida no existe.");
                                } else {
                                    // Actualiza el nombre del oponente en la interfaz de
                                        usuario y desactiva el
                                    // botón de inicio
                                    Platform.runLater(() -> {
                                        setEnemy(name, kingdom);
                                        startButton.setDisable(true);
                                    });
                                }
                                break;
                            // Respuesta de inicio de la partida
                            case RESPONSE_START:
                                File objFile = new File("connections/" + idConnection + ".obj");
                                ObjectInputStream inObj = new ObjectInputStream(new
                                    FileInputStream(objFile));
                                board = (Board) inObj.readObject();
                                inObj.close();
                                objFile.delete();
                                // Inicia el juego principal
                                Platform.runLater(() -> {
                                    createGameStage();
                                });
                        }
                        lastModified = matchFile.lastModified();
                        in.close();
                    }

                    sleep(1000);
                }
            } catch (Exception e) {
```

```java
                e.printStackTrace();
            }
        }

        public void startGame() {
            gameStarted = true;
        }

        private void setEnemy(String name, String kingdom) {
            eName = name;
            eKingdom = kingdom;
            enemyName.setText(eName);
            enemyKingdom.setText(eKingdom);
        }
    }

    private class MainGame {
        public MainGame(MainMenuController mainMenuController) {
            try {
                // Carga el archivo FXML del juego principal y configura la escena
                FXMLLoader loader = new
                    FXMLLoader(getClass().getResource("/FX/MainGame/MainGame.fxml"));
                Parent root = loader.load();

                Stage mainGame = new Stage();
                mainGame.setTitle("Main Game");
                mainGame.setScene(new Scene(root, resolution.getWidth(),
                    resolution.getHeight()));
                mainGame.setResizable(false);
                mainGame.show();

                MainGameController controller = loader.getController();
                controller.init(mainMenuController, resolution, stage, mainGame, board,
                    idConnection, matchCode, pName,
                        eName,
                        idPlayer, idEnemy);
            } catch (Exception e) {
                e.printStackTrace();
            }
        }
    }
}
```

- Clase que controla el menú principal, se encarga de la manipulación de todos los elementos FX.
- Posee métodos que permiten enviar al servidor los datos tanto de chats como de movimientos.
- Es el encargado de realizar la conección con la base de datos usando la clase DBConnector y con el servidor usando un archivo.
- Posee la clase interna DataReceiver que permite recibir las respuestas del servidor y ejecutarlas.

MainMenu.fxml

```xml
<?xml version="1.0" encoding="UTF-8"?>

```

```
<?import javafx.scene.control.Button?>
<?import javafx.scene.control.ComboBox?>
<?import javafx.scene.control.Label?>
<?import javafx.scene.control.PasswordField?>
<?import javafx.scene.control.TextArea?>
<?import javafx.scene.control.TextField?>
<?import javafx.scene.control.TitledPane?>
<?import javafx.scene.image.Image?>
<?import javafx.scene.image.ImageView?>
<?import javafx.scene.layout.AnchorPane?>
<?import javafx.scene.layout.Pane?>
<?import javafx.scene.text.Font?>

<AnchorPane maxHeight="-Infinity" maxWidth="-Infinity" minHeight="-Infinity"
    minWidth="-Infinity" prefHeight="600.0" prefWidth="325.0"
    xmlns="http://javafx.com/javafx/21" xmlns:fx="http://javafx.com/fxml/1"
    fx:controller="FX.MainMenu.MainMenuController">
  <children>
    <Label layoutX="99.0" layoutY="9.0" text="VIDEOJUEGO">
      <font>
        <Font name="Book Antiqua" size="20.0" />
      </font>
    </Label>
    <Button layoutX="27.0" layoutY="248.0" mnemonicParsing="false" onAction="#createMatch"
        text="Crear partida">
      <font>
        <Font name="Book Antiqua" size="12.0" />
      </font></Button>
    <Button layoutX="27.0" layoutY="286.0" mnemonicParsing="false" onAction="#joinMatch"
        text="Unirse a partida">
      <font>
        <Font name="Book Antiqua" size="12.0" />
      </font></Button>
    <TextField fx:id="createMatchCode" editable="false" layoutX="150.0" layoutY="248.0"
        prefHeight="25.0" prefWidth="150.0" promptText="Código">
      <font>
        <Font name="Book Antiqua" size="12.0" />
      </font></TextField>
    <TextField fx:id="joinMatchCode" layoutX="150.0" layoutY="286.0" prefHeight="25.0"
        prefWidth="150.0" promptText="Código">
      <font>
        <Font name="Book Antiqua" size="12.0" />
      </font></TextField>
    <Label layoutX="113.0" layoutY="223.0" text="SALA DE ESPERA">
      <font>
        <Font name="Book Antiqua" size="14.0" />
      </font></Label>
    <ImageView fitHeight="225.0" fitWidth="225.0" layoutX="52.0" layoutY="326.0"
        pickOnBounds="true" preserveRatio="true">
      <image>
        <Image url="@../../img/waiting.png" />
      </image>
    </ImageView>
    <Button fx:id="startButton" layoutX="141.0" layoutY="564.0" mnemonicParsing="false"
        onAction="#startMatch" text="Iniciar">
      <font>
```

```
            <Font name="Book Antiqua" size="12.0" />
         </font></Button>
      <TextField fx:id="nameInput" layoutX="60.0" layoutY="66.0" prefHeight="25.0" prefWidth="212.0" promptText="Nombre">
         <font>
            <Font name="Book Antiqua" size="12.0" />
         </font></TextField>
      <Label fx:id="playerName" layoutX="71.0" layoutY="343.0" prefHeight="17.0" prefWidth="150.0">
         <font>
            <Font name="Book Antiqua" size="16.0" />
         </font></Label>
      <Label layoutX="141.0" layoutY="416.0" text="VS">
         <font>
            <Font name="Gill Sans MT" size="40.0" />
         </font>
      </Label>
      <Label fx:id="enemyName" alignment="CENTER_RIGHT" layoutX="110.0" layoutY="512.0" prefHeight="17.0" prefWidth="150.0">
         <font>
            <Font name="Book Antiqua" size="16.0" />
         </font></Label>
      <Label layoutX="131.0" layoutY="39.0" text="USUARIO">
         <font>
            <Font name="Book Antiqua" size="14.0" />
         </font>
      </Label>
      <PasswordField fx:id="passwordInput" layoutX="61.0" layoutY="100.0" prefHeight="25.0" prefWidth="212.0" promptText="Contraseña">
         <font>
            <Font name="Book Antiqua" size="12.0" />
         </font></PasswordField>
      <Button layoutX="10.0" layoutY="135.0" mnemonicParsing="false" onAction="#register" text="Crear cuenta">
         <font>
            <Font name="Book Antiqua" size="12.0" />
         </font></Button>
      <Button layoutX="102.0" layoutY="135.0" mnemonicParsing="false" onAction="#login" text="Ingresar a cuenta">
         <font>
            <Font name="Book Antiqua" size="12.0" />
         </font></Button>
      <Button layoutX="218.0" layoutY="135.0" mnemonicParsing="false" onAction="#getStatistics" text="Ver estadísticas">
         <font>
            <Font name="Book Antiqua" size="12.0" />
         </font></Button>
      <Label fx:id="playerKingdom" layoutX="71.0" layoutY="363.0" prefHeight="17.0" prefWidth="150.0">
         <font>
            <Font name="Book Antiqua" size="12.0" />
         </font></Label>
      <Label fx:id="enemyKingdom" alignment="CENTER_RIGHT" layoutX="110.0" layoutY="492.0" prefHeight="17.0" prefWidth="150.0">
         <font>
            <Font name="Book Antiqua" size="12.0" />
```

```
       </font></Label>
    <ComboBox fx:id="kingdomInput" layoutX="90.0" layoutY="189.0" onAction="#setKingdom"
        prefWidth="150.0" promptText="Reino" />
    <Label layoutX="143.0" layoutY="165.0" text="REINO">
       <font>
          <Font name="Book Antiqua" size="14.0" />
       </font>
    </Label>
    <Pane fx:id="settingsPane" prefHeight="116.0" prefWidth="325.0"
        style="-fx-background-color: white;" visible="false">
       <children>
          <Label layoutX="93.0" layoutY="14.0" text="CONFIGURACIONES">
             <font>
                <Font name="Book Antiqua" size="14.0" />
             </font>
          </Label>
          <Label layoutX="133.0" layoutY="41.0" text="Resolución">
             <font>
                <Font name="Book Antiqua" size="12.0" />
             </font></Label>
          <ComboBox fx:id="resolutionInput" layoutX="88.0" layoutY="63.0"
              onAction="#setResolution" prefWidth="150.0" promptText="Resolución" />
       </children>
    </Pane>
    <Pane layoutX="295.0" layoutY="7.0" onMouseClicked="#toggleSettings" prefHeight="25.0"
        prefWidth="25.0">
       <children>
          <ImageView fitHeight="25.0" fitWidth="25.0" layoutX="-1.0" pickOnBounds="true"
              preserveRatio="true">
             <image>
                <Image url="@../../img/settings.png" />
             </image>
          </ImageView>
       </children>
    </Pane>
    <TitledPane fx:id="messagePane" animated="false" collapsible="false" layoutX="25.0"
        layoutY="231.0" prefHeight="139.0" prefWidth="279.0" text="Mensaje" visible="false">
      <content>
        <AnchorPane minHeight="0.0" minWidth="0.0" prefHeight="180.0" prefWidth="200.0">
             <children>
                <TextArea fx:id="messageOutput" layoutX="-1.0" prefHeight="81.0"
                    prefWidth="279.0" wrapText="true">
                   <font>
                      <Font name="Book Antiqua" size="14.0" />
                   </font></TextArea>
                <Button layoutX="122.0" layoutY="84.0" mnemonicParsing="false"
                    onAction="#closeMessage" text="OK">
                   <font>
                      <Font name="Book Antiqua" size="12.0" />
                   </font></Button>
             </children>
          </AnchorPane>
      </content>
       <font>
          <Font name="Book Antiqua" size="12.0" />
       </font>
```

```
        </TitledPane>
    </children>
</AnchorPane>
```

- Clase FXML de JavaFX que posee el juego principal.
- Posee todo el menú principal ya ordenado, puesto que el menú principal no es reescalable.

BetterColor.java

```
package Utils;

import java.io.Serializable;
import javafx.scene.paint.Color;

public class BetterColor implements Serializable {
    private double redF;
    private double greenF;
    private double blueF;
    private double alphaF;
    private int redD;
    private int greenD;
    private int blueD;
    private int alphaD;

    public BetterColor(double red, double green, double blue, double alpha) {
        this.redF = red;
        this.greenF = green;
        this.blueF = blue;
        this.alphaF = alpha;

        this.redD = (int) (redF * 255);
        this.greenD = (int) (greenF * 255);
        this.blueD = (int) (blueF * 255);
        this.alphaD = (int) (alphaF * 255);
    }

    public Color getColor() {
        return new Color(redF, greenF, blueF, alphaF);
    }

    public String getRGBA() {
        return String.format("rgba(%d, %d, %d, %d)", redD, blueD, greenD, alphaD);
    }
}
```

- Clase de apoyo que contiene un color serializable.
- Permite generar su representación en color FX y como rgba para los estilos.

MainGameOperation.java

```
package Utils;

```

```java
public interface MainGameOperation {
    int OPERATION_CHAT = 300;
    int OPERATION_MOVE = 301;
    int OPERATION_ATTACK = 302;
    int RESPONSE_CHAT = 400;
    int RESPONSE_MOVE = 401;
    int RESPONSE_ATTACK = 402;
}
```

- Interfaz que mantiene los códigos de operación y respuesta para el servidor (Del menú principal).

MainMenuOperation.java

```java
package Utils;

public interface MainMenuOperation {
    int OPERATION_CREATE = 100;
    int OPERATION_JOIN = 101;
    int OPERATION_START = 102;
    int RESPONSE_HOST = 200;
    int RESPONSE_GUEST = 201;
    int RESPONSE_START = 202;
}
```

- Interfaz que mantiene los códigos de operación y respuesta para el servidor (Del juego principal).

Resolution.java

```java
package Utils;

public class Resolution {
    private int width;
    private int height;

    public Resolution(int width, int height) {
        this.width = width;
        this.height = height;
    }

    public int getWidth() {
        return width;
    }

    public int getHeight() {
        return height;
    }

    public String toString() {
        return width + " x " + height;
    }
}
```

- Clase de apoyo que contiene una resolución (ancho x alto).

Tile.java

```java
package Utils;

import javafx.scene.control.Label;
import javafx.scene.image.*;
import javafx.scene.layout.Pane;

import java.util.HashMap;

public class Tile extends Pane implements VideogameConstants {
    private HashMap<String, ImageView> images = new HashMap<String, ImageView>();
    private Label health;
    private ImageView image;
    private String type;
    private int i;
    private int j;

    public Tile(int health, String type, double size, int i, int j) {
        this.i = i;
        this.j = j;
        this.type = type;
        for (int n = 0; n < TYPE_FILES.length; n++)
            images.put(TYPE_FILES[n], generateImageView(size, TYPE_FILES[n]));
        this.health = generateHealthLabel(size);
        setImageAndhealth(type, health);
    }

    public void setImageAndhealth(String type, int hp) {
        while (getChildren().size() > 0)
            getChildren().remove(0);

        image = images.get(type);
        getChildren().add(image);

        if (type.equals("tile"))
            health.setText("");
        else
            health.setText(hp + "");
        getChildren().add(health);

    }

    public int getI() {
        return i;
    }

    public int getJ() {
        return j;
    }

    public String getType() {
        return type;
    }

    public boolean isConnected(Tile other, int distance) {
        return Math.abs(other.getI() - i) <= distance && Math.abs(other.getJ() - j) <=
           distance;
    }

    private ImageView generateImageView(double size, String type) {
        Image image = new Image(String.format("img/tile_%s.png", type));
        ImageView imageView = new ImageView(image);
        imageView.setFitWidth(size);
        imageView.setFitHeight(size);
        images.put(type, imageView);
        return imageView;
    }

    private Label generateHealthLabel(double size) {
        Label label = new Label();
        label.setLayoutX(size / 18);
        label.setLayoutY(size / 18);
        label.setTextFill(BACKGROUND_COLOR.getColor());
        return label;
    }

    public String getKey() {
        return i + "," + j;
    }

    public String toString() {
        return "a " + type + "!: " + i + ", " + j;
    }
}
```

- Clase que mantiene una celda del tablero.
- Permite cambiar la imagen y la vida de cada celda.
- Posee otros métodos de utilidad como retornar posicion en el tablero y hallar distancia entre casillas.

Utils.java

```java
package Utils;

import java.io.*;

public class Utils {
    public static String readString(DataInputStream in) throws IOException {
        char c;
        String str = "";
        while ((c = in.readChar()) != 0)
            str += c;
        return str;
    }

    public static void writeStrings(DataOutputStream out, String[] strings) throws
        IOException {
        for (String str : strings) {
            out.writeChars(str);
            out.writeChar(0);
        }
    }

    public static void writeString(DataOutputStream out, String str) throws IOException {
        out.writeChars(str);
        out.writeChar(0);
    }

    public static void writeIdxs(DataOutputStream out, int sI, int sJ, int oI, int oJ) throws
         IOException {
        out.writeInt(sI);
        out.writeInt(sJ);
        out.writeInt(oI);
        out.writeInt(oJ);
    }
}
```

- Clase de utilidad que posee diferentes atajos de lectura y escritura de archivos.

VideogameConstants.java

```java
package Utils;

public interface VideogameConstants {
    String[] TERRAINS = { "BOSQUE", "CAMPO ABIERTO", "MONTAA", "DESIERTO", "PLAYA" };
    String[] TERRAIN_FILES = { "forest", "meadow", "mountain", "desert", "beach" };
    String[] TYPES = { "CABALLERO", "ARQUERO", "ESPADACHIN", "LANCERO" };
    String[] TYPE_FILES = { "knight", "archer", "swordsman", "spearman", "tile" };

    BetterColor PLAYER_COLOR = new BetterColor(0.27, 0.51, 1, 1); // #4580ff
    BetterColor ENEMY_COLOR = new BetterColor(1, 0.27, 0.27, 1); // #ff4545
    BetterColor PLAYER_COLOR_TRANS = new BetterColor(0.27, 0.51, 1, 0.1); // #4580ff, op 10%
    BetterColor ENEMY_COLOR_TRANS = new BetterColor(1, 0.27, 0.27, 0.1); // #ff4545, op 10%
    BetterColor BACKGROUND_COLOR = new BetterColor(0.1, 0.1, 0.1, 1); // #1a1a1a
    BetterColor SELECTED_COLOR = new BetterColor(0.8, 0.8, 0.8, 0.05); // #cccccc, op 5%
    int TOTAL_SOLDIERS = 5;
    int SIZE = 10;
}
```

- Interfaz que mantiene valores predeterminados por el juego.

Videogame.java

```java

import java.io.*;
import FX.MainMenu.MainMenuController;
import javafx.application.Application;
import javafx.fxml.FXMLLoader;
import javafx.scene.*;
import javafx.stage.*;

public class Videogame extends Application {

    @Override
   public void start(Stage primaryStage) throws IOException {
       try {
           // Carga el archivo FXML del menú principal y configura la escena
           FXMLLoader loader = new
               FXMLLoader(getClass().getResource("FX/MainMenu/MainMenu.fxml"));
           Parent root = loader.load();

           MainMenuController controller = loader.getController();
           controller.setStage(primaryStage);

           primaryStage.setTitle("Main Menu");
           primaryStage.setScene(new Scene(root, 325, 600));
           primaryStage.setResizable(false);
           primaryStage.show();
       } catch (Exception e) {
           FileWriter writer = new FileWriter("error.log");
           writer.write(e.getMessage());
           writer.close();
       }
   }

   public static void main(String[] args) {
       launch(args);
   }
}
```

- Clase principal, que instancia la ventana e inicia el juego.

Main.java

```
public class Main {
    public static void main(String[] args) {
        Videogame.main(args);
    }
}
```

- Clase que llama al main de la clase principal (necesario para la exportación a jar).

6. Ejecución del código

6.1. Video de ejecución

`https://drive.google.com/file/d/1ZT1aK9tdZdqz731l37N5dQ56h89uFjGI/view?usp=sharing`

7. Diagrama UML

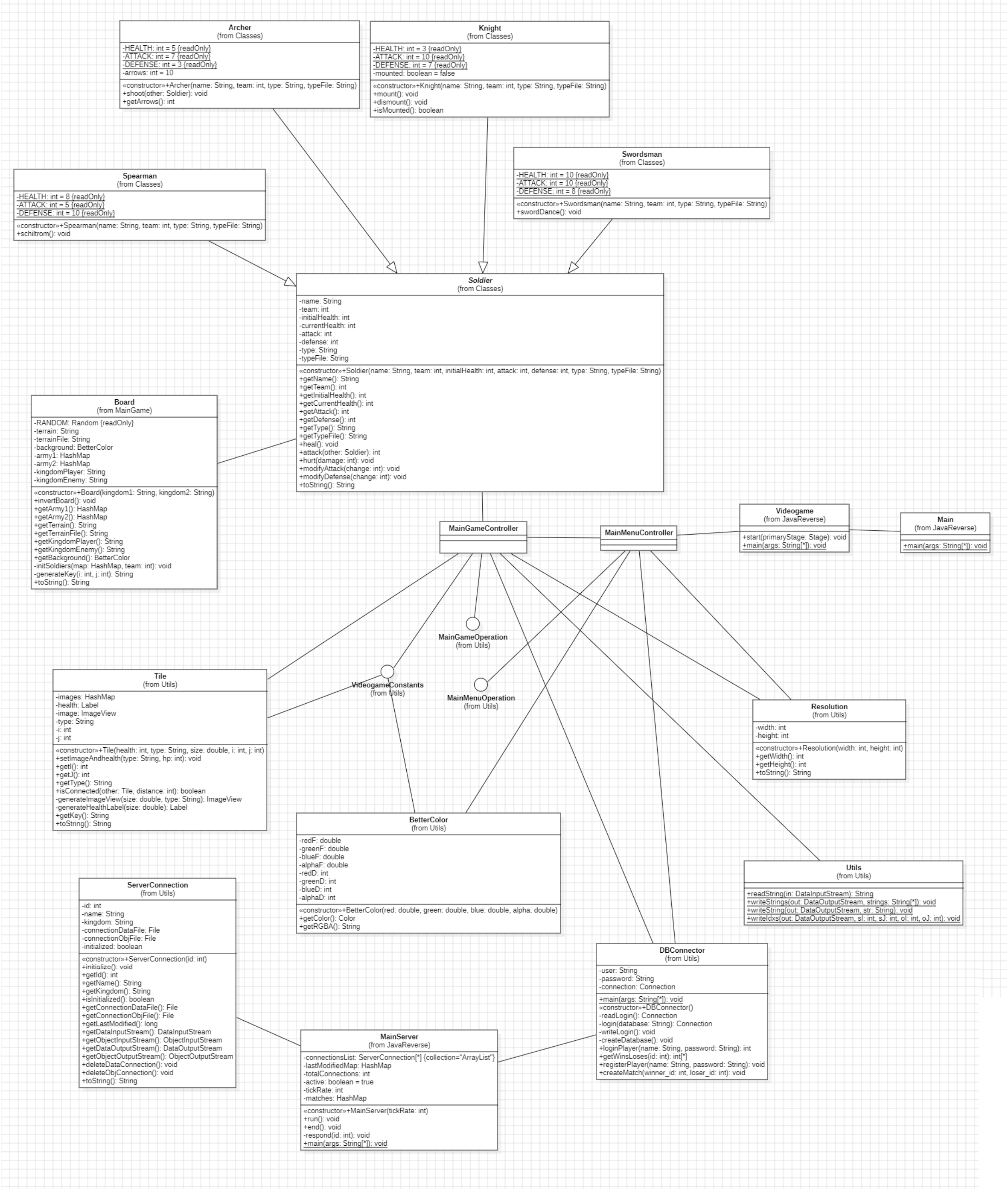

Archer
(from Classes)
-HEALTH: int = 5 {readOnly}
-ATTACK: int = 7 {readOnly}
-DEFENSE: int = 3 {readOnly}
-arrows: int = 10
«constructor»+Archer(name: String, team: int, type: String, typeFile: String)
+shoot(other: Soldier): void
+getArrows(): int
Knight
(from Classes)
-HEALTH: int = 3 {readOnly}
-ATTACK: int = 10 {readOnly}
-DEFENSE: int = 7 {readOnly}
-mounted: boolean = false
«constructor»+Knight(name: String, team: int, type: String, typeFile: String)
+mount(): void
+dismount(): void
+isMounted(): boolean
Swordsman
(from Classes)
-HEALTH: int = 10 {readOnly}
-ATTACK: int = 10 {readOnly}
-DEFENSE: int = 8 {readOnly}
«constructor»+Swordsman(name: String, team: int, type: String, typeFile: String)
+swordDance(): void
Spearman
(from Classes)
-HEALTH: int = 8 {readOnly}
-ATTACK: int = 5 {readOnly}
-DEFENSE: int = 10 {readOnly}
«constructor»+Spearman(name: String, team: int, type: String, typeFile: String)
+schiltrom(): void
Soldier
(from Classes)
-name: String
-team: int
-initialHealth: int
-currentHealth: int
-attack: int
-defense: int
-type: String
-typeFile: String
«constructor»+Soldier(name: String, team: int, initialHealth: int, attack: int, defense: int, type: String, typeFile: String)
+getName(): String
+getTeam(): int
+getInitialHealth(): int
+getCurrentHealth(): int
+getAttack(): int
+getDefense(): int
+getType(): String
+getTypeFile(): String
+heal(): void
+attack(other: Soldier): int
+hurt(damage: int): void
+modifyAttack(change: int): void
+modifyDefense(change: int): void
+toString(): String
Board
(from MainGame)
-RANDOM: Random {readOnly}
-terrain: String
-terrainFile: String
-background: BetterColor
-army1: HashMap
-army2: HashMap
-kingdomPlayer: String
-kingdomEnemy: String
«constructor»+Board(kingdom1: String, kingdom2: String)
+invertBoard(): void
+getArmy1(): HashMap
+getArmy2(): HashMap
+getTerrain(): String
+getTerrainFile(): String
+getKingdomPlayer(): String
+getKingdomEnemy(): String
+getBackground(): BetterColor
-initSoldiers(map: HashMap, team: int): void
-generateKey(i: int, j: int): String
+toString(): String
MainGameController
MainMenuController
Videogame
(from JavaReverse)
+start(primaryStage: Stage): void
+main(args: String[*]): void
Main
(from JavaReverse)
+main(args: String[*]): void
MainGameOperation
(from Utils)
VideogameConstants
(from Utils)
MainMenuOperation
(from Utils)
Tile
(from Utils)
-images: HashMap
-health: Label
-image: ImageView
-type: String
-i: int
-j: int
«constructor»+Tile(health: int, type: String, size: double, i: int, j: int)
+setImageAndhealth(type: String, hp: int): void
+getI(): int
+getJ(): int
+getType(): String
+isConnected(other: Tile, distance: int): boolean
-generateImageView(size: double, type: String): ImageView
-generateHealthLabel(size: double): Label
+getKey(): String
+toString(): String
Resolution
(from Utils)
-width: int
-height: int
«constructor»+Resolution(width: int, height: int)
+getWidth(): int
+getHeight(): int
+toString(): String
BetterColor
(from Utils)
-redF: double
-greenF: double
-blueF: double
-alphaF: double
-redD: int
-greenD: int
-blueD: int
-alphaD: int
«constructor»+BetterColor(red: double, green: double, blue: double, alpha: double)
+getColor(): Color
+getRGBA(): String
Utils
(from Utils)
+readString(in: DataInputStream): String
+writeStrings(out: DataOutputStream, strings: String[*]): void
+writeString(out: DataOutputStream, str: String): void
+writeIdxs(out: DataOutputStream, sI: int, sJ: int, oI: int, oJ: int): void
ServerConnection
(from Utils)
-id: int
-name: String
-kingdom: String
-connectionDataFile: File
-connectionObjFile: File
-initialized: boolean
«constructor»+ServerConnection(id: int)
+initialize(): void
+getId(): int
+getName(): String
+getKingdom(): String
+isInitialized(): boolean
+getConnectionDataFile(): File
+getConnectionObjFile(): File
+getLastModified(): long
+getDataInputStream(): DataInputStream
+getObjectInputStream(): ObjectInputStream
+getDataOutputStream(): DataOutputStream
+getObjectOutputStream(): ObjectOutputStream
+deleteDataConnection(): void
+deleteObjConnection(): void
+toString(): String
DBConnector
(from Utils)
-user: String
-password: String
-connection: Connection
+main(args: String[*]): void
«constructor»+DBConnector()
-readLogin(): Connection
-login(database: String): Connection
-writeLogin(): void
-createDatabase(): void
+loginPlayer(name: String, password: String): int
+getWinsLoses(id: int): int[*]
+registerPlayer(name: String, password: String): void
+createMatch(winner_id: int, loser_id: int): void
MainServer
(from JavaReverse)
-connectionsList: ServerConnection[*] {collection="ArrayList"}
-lastModifiedMap: HashMap
-totalConnections: int
-active: boolean = true
-tickRate: int
-matches: HashMap
«constructor»+MainServer(tickRate: int)
+run(): void
+end(): void
-respond(id: int): void
+main(args: String[*]): void

Diagrama UML.

8. Estructura de laboratorio Proyecto Final

- **El contenido que se entrega en este laboratorio es el siguiente:**

```
proyecto_final/
|--- EJECUTABLES
   |--- SERVER.jar
   |--- VIDEOGAME.jar
|--- INFORME
   |--- img
      |--- commit_01.png
      |--- commit_02.png
      |--- commit_03.png
      |--- commit_04.png
      |--- commit_05.png
      |--- commit_06.png
      |--- commit_07.png
      |--- commit_08.png
      |--- commit_09.png
      |--- commit_10.png
      |--- commit_11.png
      |--- commit_12.png
      |--- commit_13.png
      |--- commit_14.png
      |--- commit_15.png
      |--- commit_16.png
      |--- commit_17.png
      |--- commit_18.png
      |--- commit_19.png
      |--- commit_20.png
      |--- commit_21.png
      |--- commit_22.png
      |--- commit_23.png
      |--- commit_24.png
      |--- commit_25.png
      |--- commit_26.png
      |--- commit_27.png
      |--- commit_28.png
      |--- commit_29.png
      |--- commit_30.png
      |--- commit_31.png
      |--- commit_32.png
      |--- logo_abet.png
      |--- logo_unsa.jpg
      |--- logo_episunsa.png
      |--- uml.png
   |--- commits.bash
   |--- Informe.pdf
   |--- Informe.tex
|--- SERVER
   |--- .vscode
   |--- bin
   |--- lib
   |--- src
      |--- FX
         |--- MainGame
```

```
            |--- Classes
               |--- Archer.java
               |--- Knight.java
               |--- Soldier.java
               |--- Spearman.java
               |--- Swordsman.java
            |--- Board.java
      |--- Utils
         |--- BetterColor.java
         |--- DBConnector.java
         |--- MainGameOperation.java
         |--- MainMenuOperation.java
         |--- ServerConnection.java
         |--- Utils.java
         |--- VideogameConstants.java
      |--- MainServer.java
   |--- SERVER.jar
|--- VIDEOGAME
   |--- .vscode
   |--- bin
   |--- JavaFX
   |--- lib
   |--- src
      |--- FX
         |--- MainGame
            |--- Classes
               |--- Archer.java
               |--- Knight.java
               |--- Soldier.java
               |--- Spearman.java
               |--- Swordsman.java
            |--- Board.java
            |--- MainGame.fxml
            |--- MainGameController.java
         |--- MainMenu
            |--- MainMenu.fxml
            |--- MainMenuController.java
      |--- img
         |--- action_attack.png
         |--- action_move.png
         |--- background_beach.png
         |--- background_data.png
         |--- background_desert.png
         |--- background_forest.png
         |--- background_meadow.png
         |--- background_mountain.png
         |--- settings.png
         |--- tile_archer.png
         |--- tile_knight.png
         |--- tile_spearman.png
         |--- tile_swordsman.png
         |--- tile_tile.png
         |--- waiting.png
      |--- Utils
         |--- BetterColor.java
         |--- DBConnector.java
```

```
        |--- MainGameOperation.java
        |--- MainMenuOperation.java
        |--- Resolution.java
        |--- Tile.java
        |--- Utils.java
        |--- VideogameConstants.java
     |--- Videogame.java
  |--- VIDEOGAME.jar
```

9. Rúbricas

9.1. Entregable Informe

Tipo de Informe

Informe	
Latex	El informe está en formato PDF desde Latex, con un formato limpio (buena presentación) y facil de leer.

9.2. Rúbrica para el contenido del Informe y demostración

- **El alumno debe marcar o dejar en blanco en celdas de la columna Checklist si cumplio con el ítem correspondiente.**
- **Si un alumno supera la fecha de entrega, su calificación será sobre la nota mínima aprobatoria, siempre y cuando cumpla con todos los items.**
- **El alumno debe autocalificarse en la columna Estudiante de acuerdo a la siguiente tabla:**

Niveles de desempeño

	Nivel			
Puntos	Insatisfactorio 25 %	En Proceso 50 %	Satisfactorio 75 %	Sobresaliente 100 %
2.0	0.5	1.0	1.5	2.0
4.0	1.0	2.0	3.0	4.0

Rúbrica para contenido del Informe y demostración

Contenido y demostración		Puntos	Checklist	Estudiante	Profesor
1. GitHub	Hay enlace URL activo del directorio para el laboratorio hacia su repositorio GitHub con código fuente terminado y fácil de revisar.	2	X	2	
2. Commits	Hay capturas de pantalla de los commits más importantes con sus explicaciones detalladas. (El profesor puede preguntar para refrendar calificación).	4	X	4	
3. Código fuente	Hay porciones de código fuente importantes con numeración y explicaciones detalladas de sus funciones.	2	X	1.5	
4. Ejecución	Se incluyen ejecuciones/pruebas del código fuente explicadas gradualmente.	2	X	1.5	
5. Pregunta	Se responde con completitud a la pregunta formulada en la tarea. (El profesor puede preguntar para refrendar calificación).	2	X	2	
6. Fechas	Las fechas de modificación del código fuente estan dentro de los plazos de fecha de entrega establecidos.	2	X	2	
7. Ortografía	El documento no muestra errores ortográficos.	2	X	1.5	
8. Madurez	El Informe muestra de manera general una evolución de la madurez del código fuente, explicaciones puntuales pero precisas y un acabado impecable. (El profesor puede preguntar para refrendar calificación).	4	X	4	
Total		20		18.5	

10. Referencias

- **Aedo, M. y Castro, E. (2021). FUNDAMENTOS DE PROGRAMACIÓN 2 - Tópicos de Programación Orientada a Objetos. Editorial UNSA.**

- **JavaFX (2023). Getting Started with JavaFX.** `https://openjfx.io/openjfx-docs/`