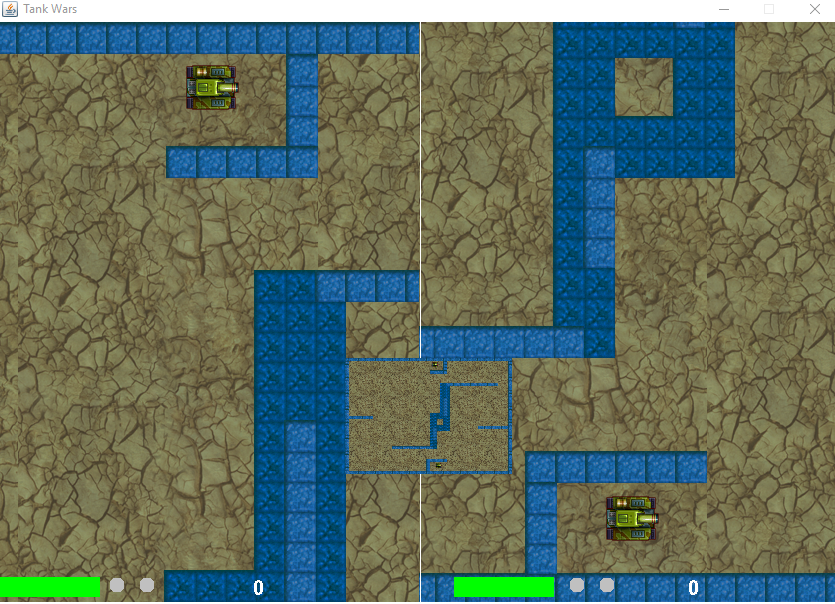
Game Project Documentation

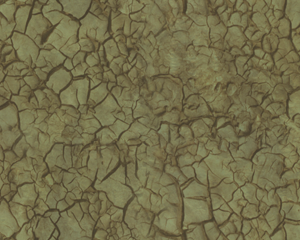
**This is output of game program**:



**Game Map**

The game screen is divided into two sections: the left hand side, which renders the area of the map immediately around Player One, and the right hand side, which renders the area of the map immediately around Player Two.

The map is composed of a background tile (*background\_tile.png*), onto which various obstructions may be placed.



The two obstructions that you must create for the Tank Game are a wall (*wall.png*),



and an indestructible wall (*wall\_indestructible.png*).



The indestructible wall obstruction is placed surround the map, creating a boundary past which tanks will not travel. Indestructible walls also appear on the map, preventing movement by tanks. Regular wall obstructions is destroyed using the tank weapons.

A minimap is rendered in the lower portion of the screen that shows the entire map,zoomed out, with markers for walls and the player locations.

**Player Controls**:

Tank Game is a two player game. Both players play from the same computer. Player one represents left tank. Player Two represents right tank.

Player One controls are:

\* A: Rotate left

\* D: Rotate right

\* W: Move forward

\* S: Move in reverse

\* Space: Fire weapon

Player Two controls are:

\* Left Arrow: Rotate left

\* Right Arrow: Rotate right

\* Up Arrow: Move forward

\* Down Arrow: Move in reverse

\* Return: Fire weapon

**Tanks**

The two tanks are smoothly rendered as they move across the map. The music will be played when the game is begun. The both players can use keyboard to control two tanks. The bullets are two tanks’ weapons. The both tanks launch bullets to fire each other. A sound is played when a bullet is fired. A sound is played when a bullet hits a target (wall or tank). Please see above game screen. The two green rectangles represent each tank’s hp. The small gray ball represents tank’s lives numbers. Each tank has two gray balls which represent tank’s lives numbers. It means that each tank has two lives opportunities. Each zero number represent each tank’s score. When left tank’s bullet hits right tank for each time, left tank’s score will be increased to ten and right tank’s hp will be decreased until it dies. Then right tank’s lives numbers will become one after it dies. Then the game will be restarted automatically and right tank will have only one remaining life. If right tank’s lives numbers become empty, the left tank will win this game. Then the game is finished and the music is stopped. The game screen also displays the left tank’s scores.

**Code’s introduction and overview**:

In the github repository, I provided six java files which were called TankBullet.java, MapBackground.java, Walls.java, DrawWalls.java, Tank.java, and tanksGame.java. These six java files were saved in the TanksGame folder. The tanksGame.java provided main function which can compile the program for the tank game. There were many png files and wav files which were saved in the TanksGame/Pictures folder. These png files and wav files were used in the tanksGame code. There were other two java files, which were called KeyboardControl.java and KeyboardEvents.java,were saved in the TanksGame/GameKeyControl folder. The implementation of this game project was to upload png files, upload wav files, place images on the game screen, play music when the game is begun, and allow players to control two tanks from keyboard.

|  |
| --- |
| **tanksGame** |
| main()  CompileProgram()  UploadResources()  getControls()  GetAudioResource()  GameMusicStart()  run()  paint()  GetLeftTankViewWindow()  GetRightTankViewWindow()  BuildOuterGraphicPage()  BuildGraphicPage()  RenewPlayGame()  FinishGame()  GetResource()  GetBufferedImageResource()  openWindow() |

**Classes Diagram**:

superclass

|  |
| --- |
| **MapBackground** |
| MapBackground()  DrawBackground() |

|  |
| --- |
| **TankBullet** |
| TankBullet()  BulletCollision()  BulletMovement()  DrawBullet() |

|  |
| --- |
| **Tank** |
| Tank()  DrawTank()  TankExplosion()  TankCollision()  TankMovement()  UseControls()  update() |

|  |
| --- |
| **Walls** |
| Walls()  PlaceWallsOnMapBackground()  drawWalls()  regularWallCollision()  indestructibleWallCollision() |

|  |
| --- |
| **DrawWalls** |
| DrawWalls()  RenewRegularWall()  RenewIndestructibleWall()  DrawRegularWall()  DrawIndestructibleWall()  RegularWallCollision()  IndestructibleWallCollision() |

|  |
| --- |
| **KeyboardControl** |
| KeyboardControl()  keyReleased()  keyPressed() |

|  |
| --- |
| **KeyboardEvents** |
| SetKeyEvents() |

subclasses

* MapBackground.java

The MapBackground is one of subclasses of tanksGame. The class works with the TanksGame package and two methods. The purpose of implementation of this class is to place a background image on the game map.

* TankBullet.java

The TankBullet is one of subclasses of tanksGame. The class works with the TanksGame package and four methods. The purpose of this class is to provide bullets for tank’s weapon.

* DrawWalls.java

The DrawWalls is one of subclasses of tanksGame. The class works with the TanksGame package and seven methods. The purpose of this class is to place two different colored walls obstructions on game map.

* Walls.java

The Walls is one of subclasses of tanksGame. The class works with the TanksGame package and five methods. The purpose of this class is to provide information about two different colored walls’ positions and provide information how many walls are placed on game map. This class uses all methods of the DrawWalls class to draw all walls.

* KeyboardEvents.java

The KeyboardEvents is one of subclasses of tanksGame. It is also a subclass of the Tank and a subclass of the KeyboardControl. The class works with the TanksGame.GameKeyControls package and one method. The purpose of this class is to set each key typed event. The key typed event is generated by a single key press, such as “c”.

* KeyboardControl.java

The KeyboardControl is one of subclasses of tanksGame. It is a superclass of KeyboardEvents. The class works with the TanksGame.GameKeyControl package and three methods. The method of the KeyboardEvents is used in the KeyboardControl. The purpose of this class is to get the key pressed and key released events. Pressing and releasing a key on the keyboard will result in the generating the following key events.

* Tank.java

The Tank is one of subclasses of tanksGame. The class works with the TanksGame package and seven methods. The purpose of this class is to place tank on game map and allow player controls tank’s movement from keyboard. The UseControls method is one of methods in this class. Its purpose is to allow player to control tank’s movement on computer keyboard. The method of the KeyboardEvents class is used in the UseControls method. The purpose is to set a key typed event after a player press a key on the keyboard. The UseControls method also provides getKeyCode method. The getKeyCode method is to return keyCode of event for key pressed events and key released events. Then the tank can be moved by player’s controls. The UseControls method is used in the another method update. The update method is an Observer function. This function is called when a control’s change is happened in the state of observable.

* tanksGame.java

The tanksGame is a superclass of all classes. The class works with the TanksGame package and seventeen methods. The purpose of this class is to execute a program for the tank game. The seventeen methods of the tanksGame are to upload image source files, to upload audio source files, to use HashMap to store all key typed events for players’ controls, to use methods of subclasses to get image source files, place images on game map, and control two tanks’ movement from keyboard, to paint a game screen which divides into two windows that display two areas of two tanks, to create an outer graphical page which shows a big game screen, a minimap, tanks’ scores, and tanks’ lives’ numbers, to create an inner graphical page which shows background image, two tanks, walls, and bullets, to open the window of this graphical display, to update playing game again, to design how to finish the game, to open music when a game is started, and to execute a program for the game.