	src/test/java/edu/colorado/fourdimensionalonedgames/ShipTest.java		
@9 -179,7 +	+179,7 @@ void testSubmarineGenerateCoordinates(){		
	//for the weird submarine protrusion tile	179	//for the weird submarine protrusion tile
	expected.add(new Point3D(8,7,0));	180 181	expected.add(new Point3D(8,7,0));
	for(int i = 0; i < player2Submarine.size; i++){	182 +	<pre>for(int i = 0; i &lt; player2Submarine.getSize(); i++){</pre>
	assertEquals(expected.get(i), results.get(i));		assertEquals(expected.get(i), results.get(i));
1	)	184	)
	+194,7 @@ void testSubmarineGenerateCoordinates(){		
	//damage Submarine's CQ (sub at 2,2 down) (CQ at 2,5) player1.addweapon(spaceLaser);	308 309	//damage Submarine's CQ (sub at 2,2 down) (CQ at 2,5) player1.add/deapon(spaceLaser);
	fireInput1 = new PlayerFireInput("Space Laser", "2", "5");	310	fireInput1 = new PlayerFireInput("Space Laser", "2", "5");
	results = player1.attack(player2.getBoard(), fireInput1);		results = player1.attack(player2.getBoard(), fireInput1);
	result = results.get(0);		result = results.get(0);
	<pre>captainsQ = (CaptainsQuartersTile)result.ship.getShipTiles().get(3);</pre>	313 +	<pre>captainsQ = (CaptainsQuartersTile)result.getShip().getShipTiles().get(3);</pre>
	assertEquals(captainsQ.getHp(), 1); assertFalse(result.ship.destroyed());	315 +	<pre>assertEquals(captainsQ.getHp(), 1); assertFalse(result.getShip().destroyed());</pre>
	assertSame(result.type, AttackResultType.MISS);	316 +	assertSame(result.getType(), AttackResultType.MISS);
	//destroy entire Submarine after hitting CQ one more time	318	//destroy entire Submarine after hitting CQ one more time
	results = player1.attack(player2.getBoard(), fireInput1); result = results.get(0);		results = player1.attack(player2.get8oard(), fireInput1); result = results.get(0);
	assertTrue(result.ship.destroyed());	321 +	assertTrue(result.getShip(),destroyed());
	assertEquals(captainsQ.getHp(), 0);		assertEquals(captainsQ.getHp(), 0);
	assertSame(result.type, AttackResultType.SUNK);	323 +	<pre>assertSame(result.getType(), AttackResultType.SUNK);</pre>
}		324 ) 325	
} 😑		325 326 } $\Theta$	
		_	
- 4	<pre>for(int i = 0; i &lt; player2Submarine.Size; i++){     assertEquals(expected.get(i), results.get(i));</pre>	228 + 229	<pre>for(int i = 0; i &lt; player2Submarine.getSize(); i++){    assertEquals(expected.get(i), results.get(i));</pre>
1	}	230	}
@ -244,7 +	+244,7 @@ void testLinearShipsGenerateCoordinates() {		
	expected.add(new Point3D(9, y, 0));	244	expected.add(new Point3D(9, y, 0));
1	)	245	)
- 1	for (int i = 0, i / nlavar20attlachin rive, iaa) /	246 247 +	for (int i = 0: i / nlavar28attlachin mattita(): iaah /
	<pre>for (int i = 0; i &lt; player2Battleship.size; i++) (     assertEquals(expected.get(i), results.get(i));</pre>	247 +	<pre>for (int i = 0; i &lt; player2Battleship.getSize(); i++) {    assertEquals(expected.get(i), results.get(i));</pre>
1		249	same erdnavs(subacran-Bac(x)), sames-Bac(x)))
}		250	
@0 -266,61	+266,61 @@ void testCaptainQuarters(){		
	fireInput1 = new PlayerFireInput("Single Shot", "1", "1");	266	<pre>fireInput1 = new PlayerFireInput("Single Shot", "1", "1");</pre>
	List <attackresult> results = player1.attack(player2.getBoard(), fireInput1);</attackresult>	267	List(AttackResult) results - player1.attack(player2.getBoard(), fireInput1);
	AttackResult result = results.get(0); assertTrue(result.ship.destroyed());	268 269 +	AttackResult result = results.get(0); assertTrue(result.getShip().destroyed());
	CaptainsQuartersTile captainsQ = (CaptainsQuartersTile)result.ship.getShipTiles().get(0);	278 +	CaptainsQuartersTile captainsQ = (CaptainsQuartersTile)result.getShip().getShipTiles().get(0);
	assertEquals(captainsQ.getHp(), 0);		assertEquals(captainsQ.getHp(), 0);
- 6	assertSame(result.type, AttackResultType.SUNK);	272 +	<pre>assertSame(result.getType(), AttackResultType.SUNK);</pre>
		273 274	
	//damage Destroyer's CQ	275	//damage Destroyer's CQ
	fireInput1 = new PlayerFireInput("Single Shot", "4", "5");	276	fireInput1 = new PlayerFireInput("Single Shot", "4", "5");
	results = player1.attack(player2.getBoard(), fireInput1);		results = player1.attack(player2.get8oard(), fireInput1);
	result = results.get(0); assertFalse(result. <mark>ship</mark> .destroyed());	279 +	result = results.get(0); assertFalse(result.getShip().destroyed());
	captainsQ = (CaptainsQuartersTile)result.ship.getShipTiles().get(1);	280 +	<pre>captainsQ = (CaptainsQuartersTile)result.getShip().getShipTiles().get(1);</pre>
	assertEquals(captainsQ.getHp(), 1);		assertEquals(captainsQ.getHp(), 1);
- 0	assertSame(result.type, AttackResultType.MISS);	282 +	<pre>assertSame(result.getType(), AttackResultType.MISS);</pre>
	//destage anting Destages after hitting CO one was time		//dacknow antino Dacknowno sitan bitting CO and many time
	<pre>//destroy entire Destroyer after hitting CQ one more time results = player1.attack(player2.getBoard(), fireInput1);</pre>	285	<pre>//destroy entire Destroyer after hitting CQ one more time results = player1.attack(player2.getBoard(), fireInput1);</pre>
	result - results.get(0);	286	result = results.get(0);
- 6	assertTrue(result.ship.destroyed());	287 +	assertTrue(result.getShip().destroyed());
	assertEquals(captainsQ.getHp(), 0);	288	assertEquals(captainsQ.getHp(), 0);
	assertSame(result.type, AttackResultType.SUNK);	289 ±	<pre>assertSame(result.getType(), AttackResultType.SUNK);</pre>
	//damage Battleship's CQ		//damage Battleship's CQ
	<pre>fireInput1 = new PlayerFireInput("Single Shot", "5", "7");</pre>		fireInput1 = new PlayerFireInput("Single Shot", "5", "7");
	results = player1.attack(player2.getBoard(), fireInput1); result = results.get(0);	294 295	<pre>results = player1.attack(player2.getBoard(), fireInput1); result = results.get(0);</pre>
	result = results.get(0); captainsQ = (CaptainsQuartersTile)result. <mark>ship</mark> .getShipTiles().get(2);	296 +	result = results.get(0); captainsQ = (CaptainsQuartersTile)result.getShip().getShipTiles().get(2);
	assertEquals(captainsQ.getHp(), 1);		assertEquals(captainsQ.getHp(), 1);
	assertFalse(result.ship.destroyed());	298 •	assertFalse(result.getShip().destroyed());
- 6	assertSame(result.type, AttackResultType.MISS);	299 +	<pre>assertSame(result.getType(), AttackResultType.MISS);</pre>
	//destroy entire Battleship after hitting CQ one more time	300 301	//destroy entire Battleship after hitting CQ one more time
	//destroy entire Battleship after hitting CQ one more time results = player1.attack(player2.getBoard(), fireInput1);	302	<pre>//destroy entire Battleship after hitting CQ one more time results = player1.attack(player2.getBoard(), fireInput1);</pre>
r	result = results.get(0);	303	result = results.get(0);
	assertTrue(result.ship.destroyed());	304 +	<pre>assertTrue(result.getShip().destroyed());</pre>
	assertEquals(captainsQ.getHp(), 0); assertSame(result.type, AttackResultType.SUNK);	305 306 +	<pre>assertEquals(captainsQ.getHp(), 0); assertSame(result.getType(), AttackResultType.SUNK);</pre>
	ourse commercance symmy necessarily personny;	307	opportunity countributy, neconnegately personny,
,	//damage Submarine's CQ (sub at 2,2 down) (CQ at 2,5)	308	//damage Submarine's CQ (sub at 2,2 down) (CQ at 2,5)
	player1.addWeapon(spaceLaser);	309	player1.addWeapon(spaceLaser);
	fireInput1 = new PlayerFireInput("Space Laser", "2", "5"); results = player1.attack(player2.getBoard(), fireInput1);	310 311	fireInputl = new PlayerFireInput("Space Laser", "2", "5");
	results = player1.attack(player2.getBoard(), fireInput1); result = results.get(0);		results = player1.attack(player2.get8oard(), fireInput1); result = results.get(0);
	result = results.get(0); captainsQ = (CaptainsQuartersTile)result.ship.getShipTiles().get(3);	313 +	<pre>captainsQ = (CaptainsQuartersTile)result.getShip().getShipTiles().get(3);</pre>
	assertEquals(captainsQ.getHp(), 1);	314	assertEquals(captainsQ.getHp(), 1);
	assertFalse(result.ship.destroyed());	315 +	assertFalse(result.getShip().destroyed());
- 8	assertSame(result.type, AttackResultType.MISS);	316 +	<pre>assertSame(result.getType(), AttackResultType.MISS);</pre>
- 8	//destroy entire Submarine after hitting CQ one more time		//destroy entire Submarine after hitting CQ one more time
- 8			
- 8			results = player1.attack(player2.getBoard(), fireInput1):
- &	results = player1.attack(player2.getBoard(), fireInput1); result = results.get(0);	320	results = player1.attack(player2.get8oard(), fireInput1); result = results.get(0);
- 8 - 2 / / / /	results = player1.attak(player2.getBoard(), fireInput1); result = results.get(0); assertIrue(result.ämij.destroyed());	320 321 +	result = results.get(0); assertTrue(result.getShip().destroyed());
- 8 - 8 // r	results = player1.ettack(player2.getBoard(), fireImput1); result = results.get(0); assertTrue(result.edg.destroyed()); assertEquals(cogtaImQ.getHo(), 0);	320 321 + 322	result = result.get(0): assertTrue(result.getShip(),destroyed()); assertTquals(captship(),etp(), 0);
- 8 - 2 / r	results = player1.attak(player2.getBoard(), fireInput1); result = results.get(0); assertIrue(result.ämij.destroyed());	320 321 +	result = results.get(0); assertInue(result.getShip().destroyee()); assertEnuals.cotalon(0.getWo(), 0); assertShame(result.getType(), AttackResultType.SUN();

```
    → 3 ■■■■■ src/main/java/edu/colorado/fourdimensionalonedgames/game/ship/Ship.java (□)
<u>.</u> @0 -11,7 +11,7 @0
  12 public abstract class Ship implements IRenderable {
                                                                                                                                 public abstract class Ship implements IRenderable {
13 public int size;
15 protected List<ShipTile> shipTiles;
16 private boolean destroyed;
17
                                                                                                                            13

14 + protected int size;
15 protected List<ShipTile> shipTiles;
16 private boolean destroyed;
* @@ -23,6 +23,7 @@ public Ship() {
          public void addTile(ShipT:
     shipTiles.add(tile);
}
                                                                                                                                     public void addTile(ShipTile tile) {
                                                                                                                                        shipTiles.add(tile);
                                                                                                                          26 + public int getSize()(return size;)
Ŧ

→ 2 ■■■■■ src/test/java/edu/colorado/fourdimensionalonedgames/BasicAttacksTest.java □

□
.t. @@ -118,7 +118,7 @@ void attackHit() {
               // sink minesweeper ship
fireInput1 = new PlayerFireInput("Single Shot", "1", "1");
Ŧ
<u>+</u> @0 -6,8 +6,8 @0
                                                                                                                            public FleetControl(Player player){
this.player = player;

this.player = player;
 v ÷ 4 *** src/main/java/edu/colorado/fourdimensionalonedgames/game/ship/MoveFleetCommand.java
<u>+</u> 00 -5,8 +5,8 00
this.player = player;
103 for(AttackResult attackResult : results){
104 - Ship attackedShip = attackResult.ship;
105
106
if (attackedShip == null) {
107
//when missed shot
<u>+</u> @0 -3,14 +3,17 @0
5 public class AttackResult {
6 - public final AttackResultType type;
7 - public final Ship ship;
                                                                                                                               + private final AttackResultType type;
+ private final Ship ship;
                           ult(AttackResultType type, Ship ship) {
                                                                                                                                                 Result(AttackResultType type, Ship ship) {
            this.type = type;
this.ship = ship;
                                                                                                                                        this.type = type;
this.ship = ship;
                                                                                                                            public Ship getShip(){ return ship; }
public AttackResultType getType(){ return type; }
everride

15 public boolean equals(Object o) {

16 if (!(o instanceof AttackResult)) return false;

4
                                                                                                                                     public boolean equals(Object o) {
   if (!(o instanceof AttackResult)) return false;
 .t. @@ -148,7 *148,7 @@ public void fireNeapon(Board opponentBoard, PlayerFireInput userInput){
152

153 if (attackedShip == null) {

154 AlertBox.display("Miss", "Shot missed");
                                                                                                                                    if (attackedShip == null) {
    AlertBox.display("Miss", "Shot missed");
```

## **BEFORE**

```
public class PlayerController implements Initializable {
    private Game game;
    private Player player;
    private Player enemyPlayer;

@FXML
    private GridPane playergpane;

@FXML
    private GridPane enemygpane;

@FXML
    private Button placeShipButton;

@FXML
    private Button passTurnButton;

@FXML
    private Button fireWeaponButton;
```

```
//Refreshes a given gpane with all of this board's tile objects (canvas nodes)
public void linkBoardVisuals(GridPane gpane) {
   Tile tile;
   for (int j = 0; j <= columns; j++) {</pre>
        tile = this.tiles[0][j][0];
        this.renderer.register(tile);
        gpane.add(tile, 0, j);
   }
   for (int i = 1; i <= rows; i++) {
        tile = this.tiles[i][0][0];
        this.renderer.register(tile);
        gpane.add(tile, i, 0);
   }
   for (int i = 1; i <= columns; i++) {
        for (int j = 1; j <= rows; j++) {
           tile = this.tiles[i][j][0];
           this.renderer.register(tile);
           gpane.add(tile, i, j);
        }
   }
}
```

## **AFTER:**

```
package edu.colorado.fourdimensionalonedgames.render.gui;

/ poards

public interface Subject {
    void registerObserver(Observer observer);
    void removeObserver(Observer observer);
    void updateObservers();

}
```

```
package edu.colorado.fourdimensionalonedgames.game;
                                                                                               A4 ×4 A ×
import edu.colorado.fourdimensionalonedgames.game.ship.Orientation;
import edu.colorado.fourdimensionalonedgames.render.Render;
import edu.colorado.fourdimensionalonedgames.game.ship.Ship;
import edu.colorado.fourdimensionalonedgames.render.gui.Observer;
import edu.colorado.fourdimensionalonedgames.render.gui.Subject;
import edu.colorado.fourdimensionalonedgames.render.tile.*;
import javafx.geometry.Point3D;
import java.util.ArrayList;
import java.util.List;
public class Board implements Subject {
   public Tile[][][] tiles;
   public Render renderer;
   private List<Observer> observers;
    public Board(int columns, int rows, int depth, Render renderer) {
        this.rows = rows;
        this.columns = columns;
        this.depth = depth;
```

```
package edu.colorado.fourdimensionalonedgames.render.gui;

import edu.colorado.fourdimensionalonedgames.render.tile.Tile;

//grid panes
public interface Observer {
    void update(Tile[][][] tiles);
}
```

```
package edu.colorado.fourdimensionalonedgames.render.gui;
                                                                                              A2 ×8 ^ ~
 public class Display implements Observer{
     protected GridPane gpane;
     protected Tile[][][] boardState;
     protected Render renderer;
     //Send initial board to display
     public Display(GridPane gpane, Tile[][][] board, Render renderer){
          this.boardState = board;
         this.renderer = renderer;
         this.gpane = gpane;
     @Override
     public void update(Tile[][][] newBoardState) {
         for(Tile[][] tileColumn : newBoardState){
             for(Tile[] tileRow : tileColumn){
                  swapTile(tileRow[0]);
         boardState = newBoardState;
                                                                                              A1 x3 ^ v
import edu.colorado.fourdimensionalonedgames.render.tile.SeaTile;
import edu.colorado.fourdimensionalonedgames.render.tile.ShipTile;
import edu.colorado.fourdimensionalonedgames.render.tile.Tile;
import javafx.scene.layout.GridPane;
public class EnemyDisplay extends Display implements Observer{
   public EnemyDisplay(GridPane gpane, Tile[][][] board, Render renderer) { super(gpane, board, renderer)
   @Override
   public void update(Tile[][][] newBoardState) {
        //remove and add children to gpane
        for(Tile[][] tileColumn : newBoardState){
            for(Tile[] tileRow : tileColumn){
                Tile newTile = tileRow[0];
                if(newTile instanceof ShipTile){
                    if(newTile.revealed || newTile.shot){
                        swapTile(newTile);
                    else{
                else if(newTile instanceof SeaTile || newTile instanceof LetterTile){
                    swapTile(newTile);
```