Fairy Tale Virus

2013156019 민웅기

2013156049 홍승준

제작일 : 2014.12.13 토

제출일 : 2014.12.

목차

1. **개요 3**

과제 명3

과제 내용3

개발 동기3

1. 개발 내용 **4**

개발 목표 4

활용 기술 4

내용 4

1. 설계 **5**

시스템 설계 5

윈도우 설계 6

객체 설계 6

1. 최종 결과 **7**
2. 실행 **31**
3. 첨부 **33**
4. 개요
   1. 과제 명 :   
      **2D Top view 형식의 슈팅 게임과 결합된 타워 디펜스**
   2. 과제 내용 :  
        
      슈팅 게임과 타워 디펜스를 결합한 게임입니다.  
      플레이어에게 주어진 캐릭터와 오브젝트(성)을 방어해야 하며, 캐릭터를 이용한 슈팅과 타워 디펜스의 전략적인 요소를 더하여 보았습니다.몬스터를 처치하면 주어지는 <마나>를 캐릭터의 업그레이드나 타워, 타워 업그레이드에 어떻게 투자하느냐에 따라 플레이의 형식이 많이 바뀝니다. 기존의 타워 디펜스의 타워만 건설 해놓으면 알아서 되는 방식의 게임을 살짝 변형시켜 컨트롤의 재미 요소를 노려 보았습니다.
   3. 개발 동기 : 자바를 이용한 게임 프로그래밍
5. 개발 내용  
   1. 개발 목표 : 자바의 GUI와 게임 알고리즘을 이용하여 구현하는 활용 능력 증진
   2. 활용 기술 :  
      1. 쓰레드와 타이머
      2. 더블 버퍼링
      3. 메시지 핸들러와 이벤트 리스너 사용
      4. GUI 클래스
      5. 파일 입출력(저장&불러오기)
      6. 객체 지향, 설계  
           
         등등…….
   3. 내용 :   
      몬스터가 길을 따라가다가 최종 도착지에 있는 성에 도착하면 성의 내구도가 떨어집니다. 몬스터가 최종 목적지까지 도달하지 못하게 막는 것이 이 게임의 플레이 목적입니다.  
      시작 시 캐릭터가 주어지고 일정 시간 뒤에 몬스터가 소환됩니다.  
      캐릭터로 직접 공격 할 수도 있고, 몬스터의 길이 아닌 곳에 타워를 설치 할 수도 있습니다. 이 타워는 범위내에 들어오는 몬스터를 자동으로 공격합니다.  
      캐릭터가 몬스터에게 노출되면 몬스터가 캐릭터를 공격하기 시작합니다. 플레이어는 캐릭터를 움직이며 공격을 피할 수 있습니다.  
      몬스터, 캐릭터, 타워가 각각 총을 발사 함으로 총알은 아군, 적군의 총알을 구분하며, 목표 지점을 향해 날아가다가 객체와 부딪히면 (충돌 판정 알고리즘) 타격됩니다.  
      몬스터가 체력이 0이 되면 소멸하면서 캐릭터에게 마나를 줍니다.캐릭터를 강화하거나 체력 회복, 타워 건설, 타워 업그레이드에 마나를 투자하게 됩니다.  
      몬스터를 막으며, 성과 캐릭터의 체력이 0이하로 내려가지 않게 막으면 됩니다.  
      (ps. 추가 사항 : 게임의 모든 스테이지가 끝나면 보스 몬스터 레이드가 무한으로 시작됩니다.)  
      게임 도중에 저장과 불러오기가 가능합니다.
6. 설계
   1. 시스템 설계
      1. 프로그램을 시작하면 게임의 간략한 인터페이스 설명 안내문이 켜지고 확인을 누르면 바로 게임이 시작됩니다.
      2. 기본적으로 시작 시 캐릭터가 주어지고 일정 시간 뒤 몬스터가 등장합니다.
      3. 몬스터들은 일정한 길을 따라 진행하며 아군의 성을 부수러 들어옵니다.
      4. 몬스터를 처치 할 때 마다 ‘마나’(이 게임 안에서 화폐의 개념)가 주어지면 플레이어는 이 것을 이용하여, 캐릭터의 공격력이나 체력에 투자하거나, 건물을 건설할 때 소비하도록 합니다.
      5. 건물의 건설과, 모든 업그레이드는 마우스로 UI를 클릭하여 진행됩니다.
      6. 플레이어 캐릭터가 몬스터에게 노출 될 경우 몬스터는 플레이어를 공격하면서 움직일 것입니다.
      7. 플레이어는 몬스터의 공격을 모두 피해야 하며, 캐릭터로 직접 공격하여 몬스터를 처치 할 수 있습니다.
      8. 캐릭터는 WASD키로 각각 위, 아래, 양 옆으로 움직일 수 있습니다.
      9. 캐릭터의 공격은 스페이스 바로 마우스 포인터의 현재 위치를 방향으로 잡아 공격합니다.
      10. 마우스로 타워를 건설할 수 있습니다. 이 타워는 몬스터가 타워 범위 내에 들어올 경우 공격합니다.
      11. 몬스터의 모든 침공을 막으면 보스 몬스터가 출현하며, 보너스 스테이지로 무한 레이드가 진행 됩니다.
      12. 파일로 결과나 진행 상황을 저장하기 또한 가능합니다.

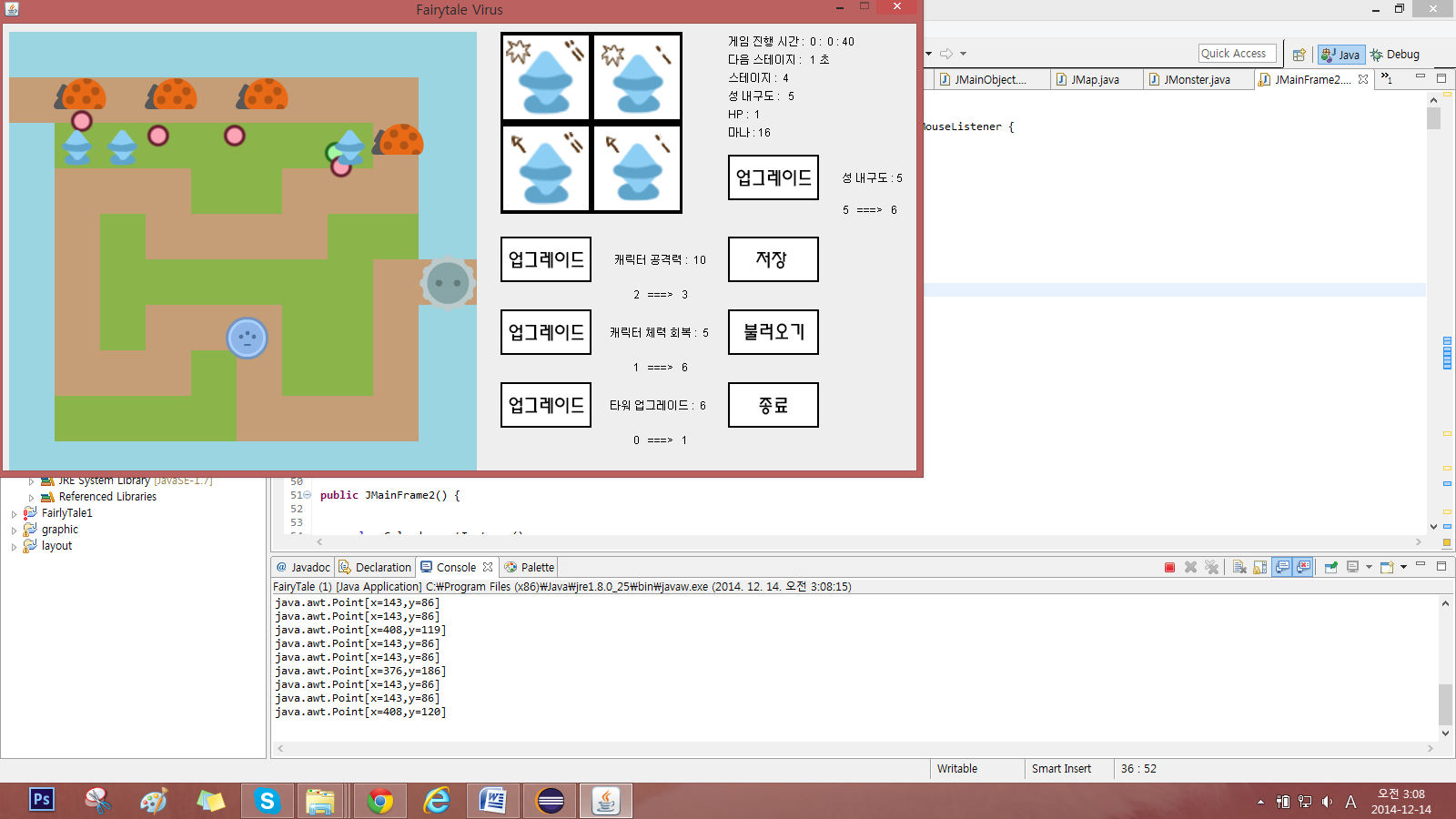
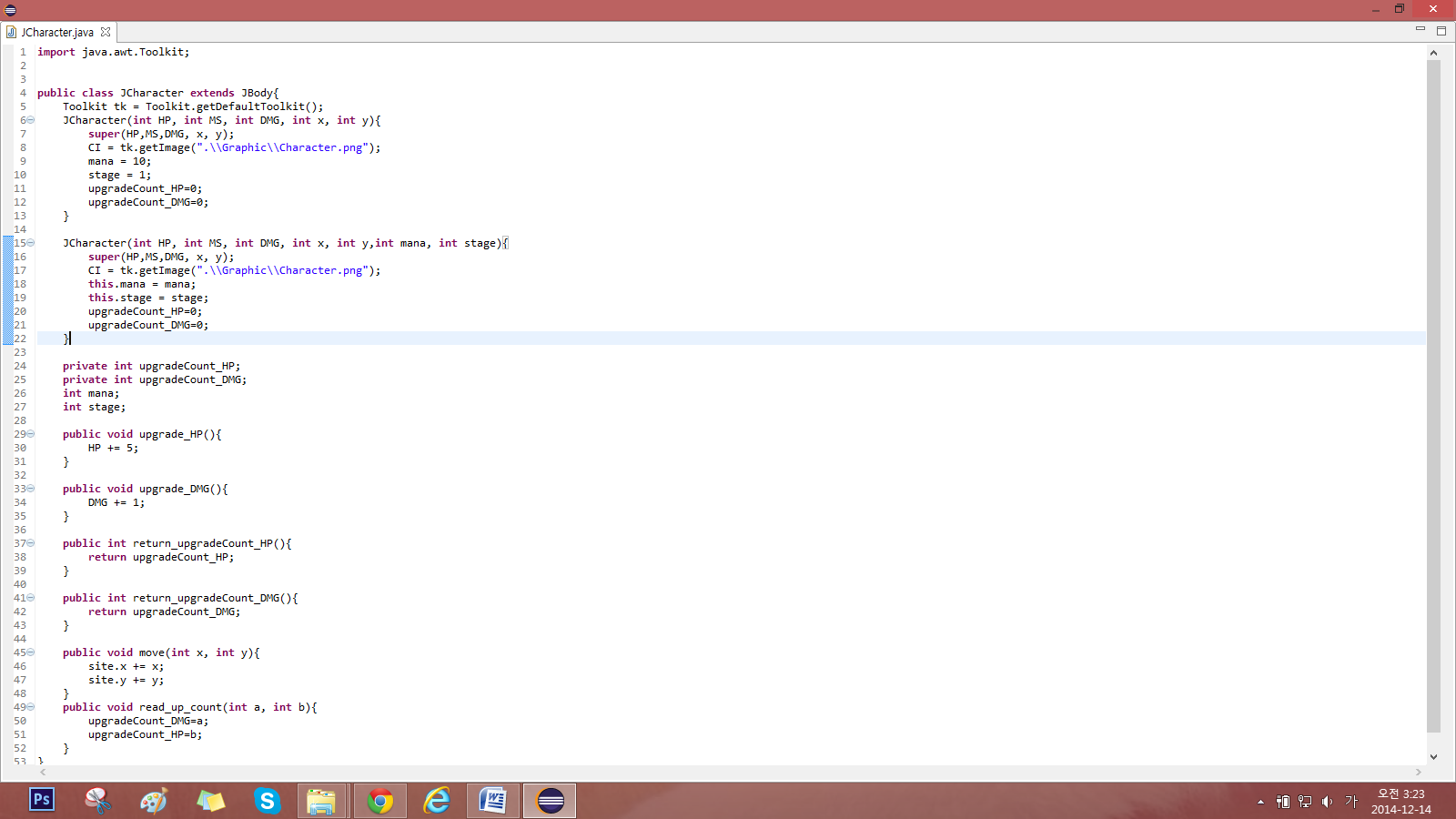
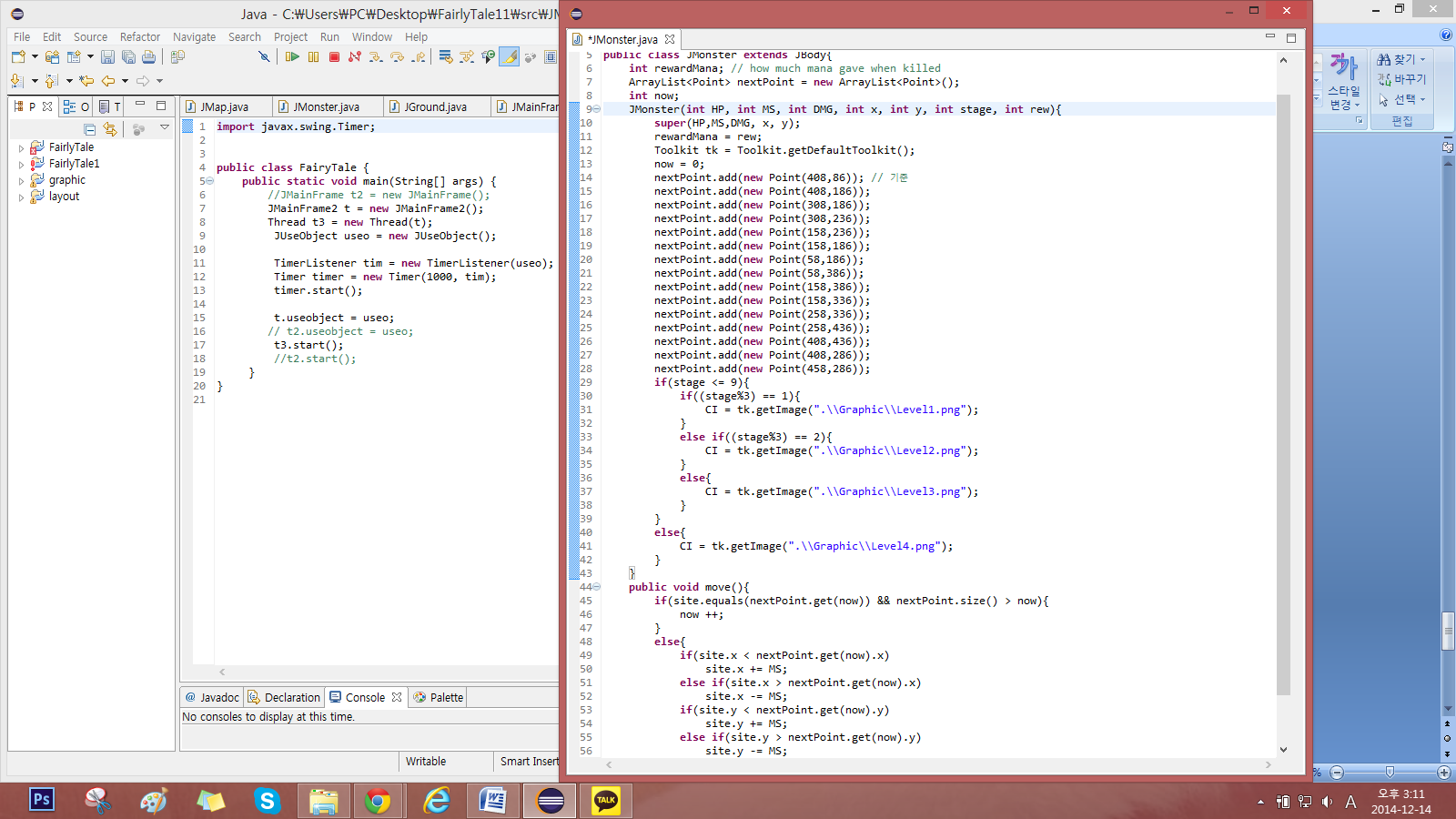
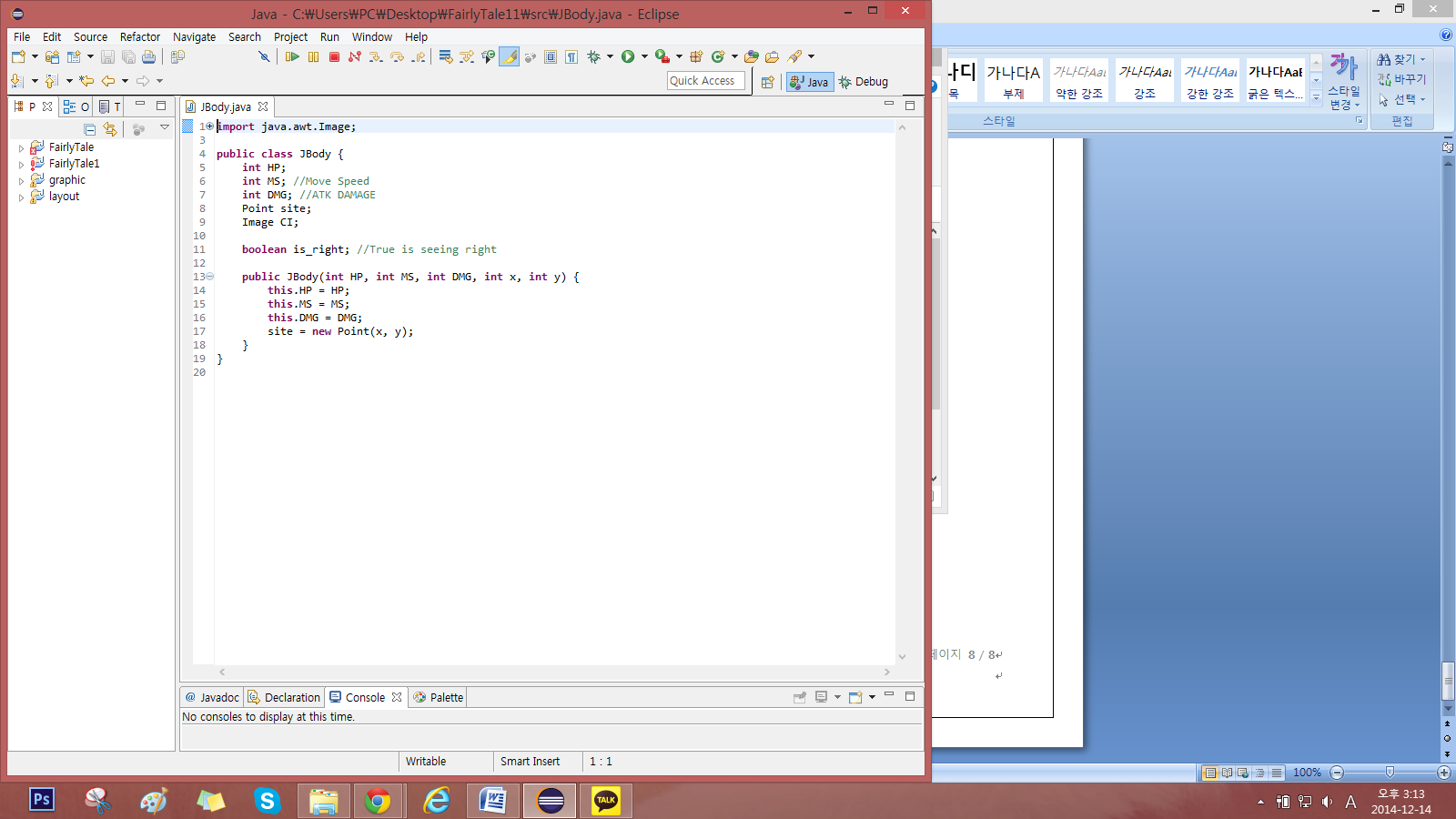
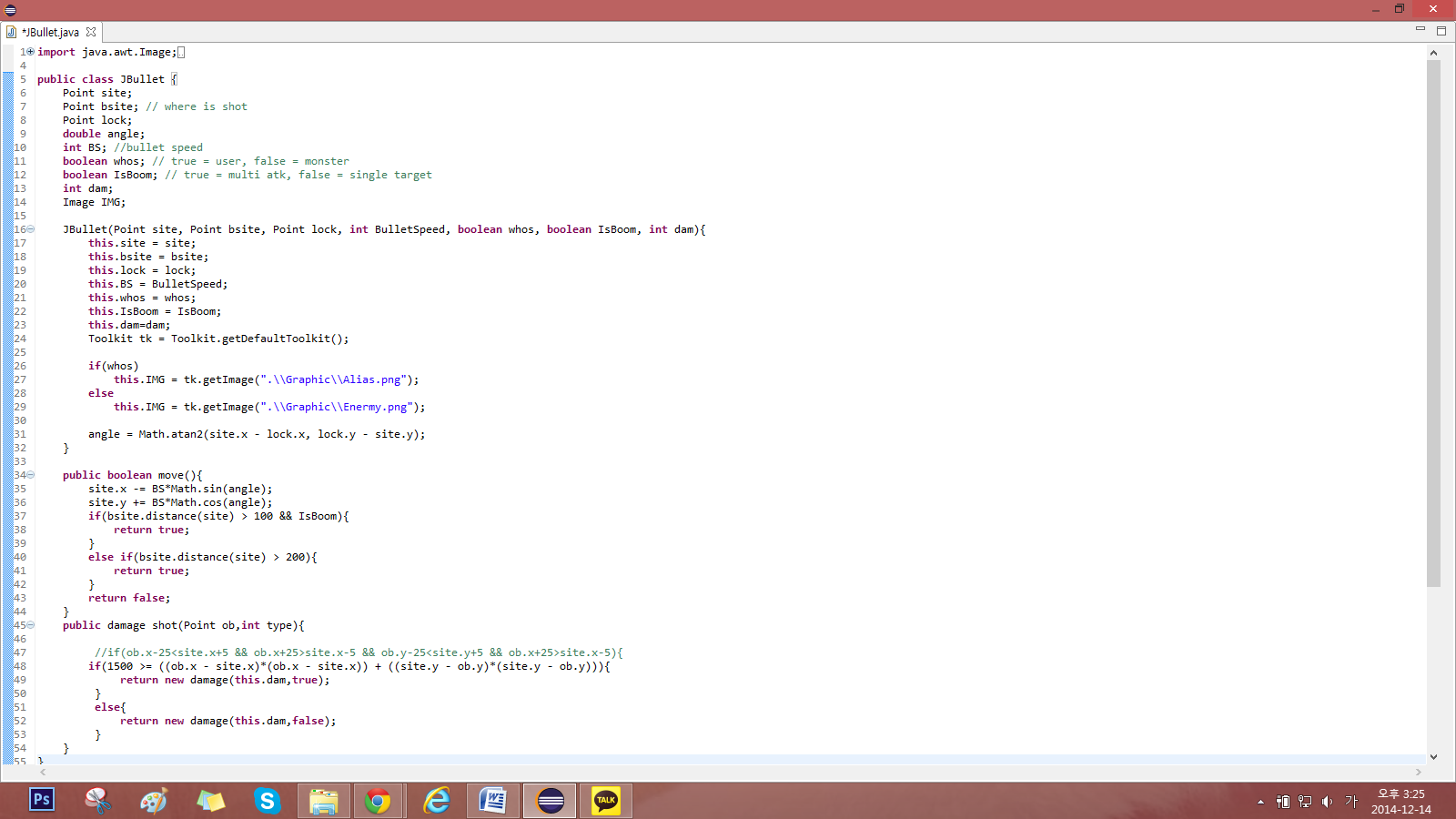
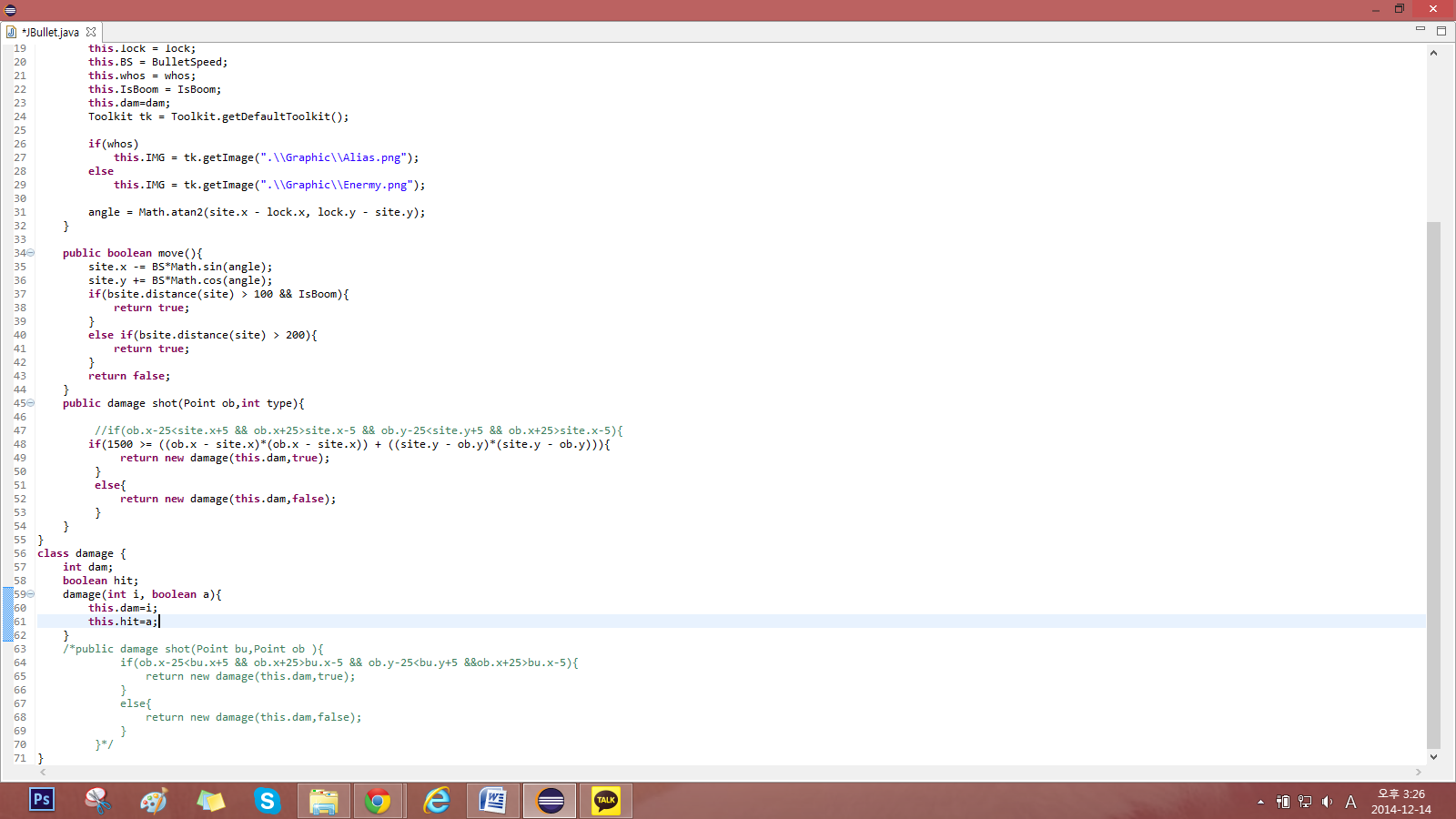
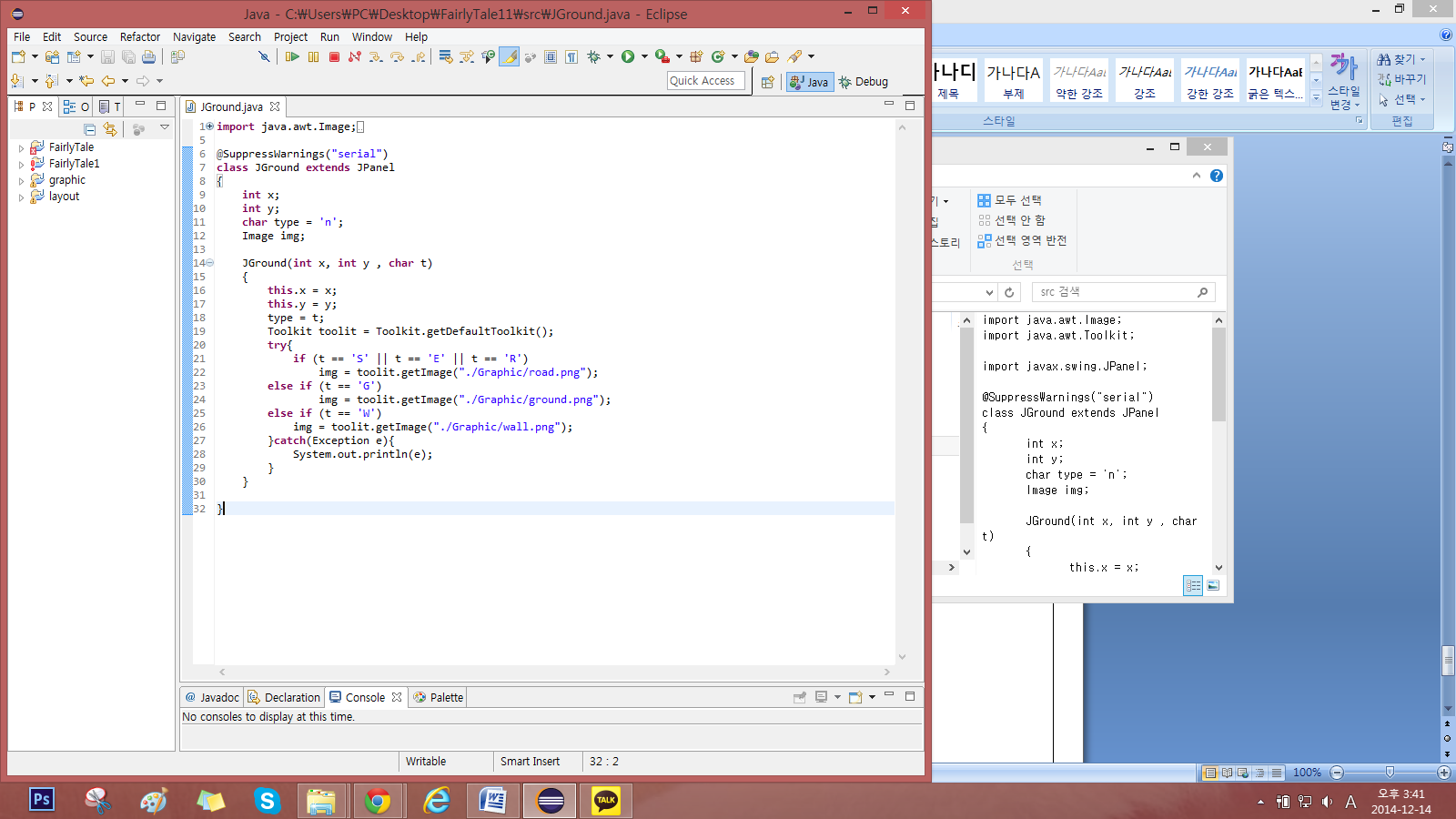
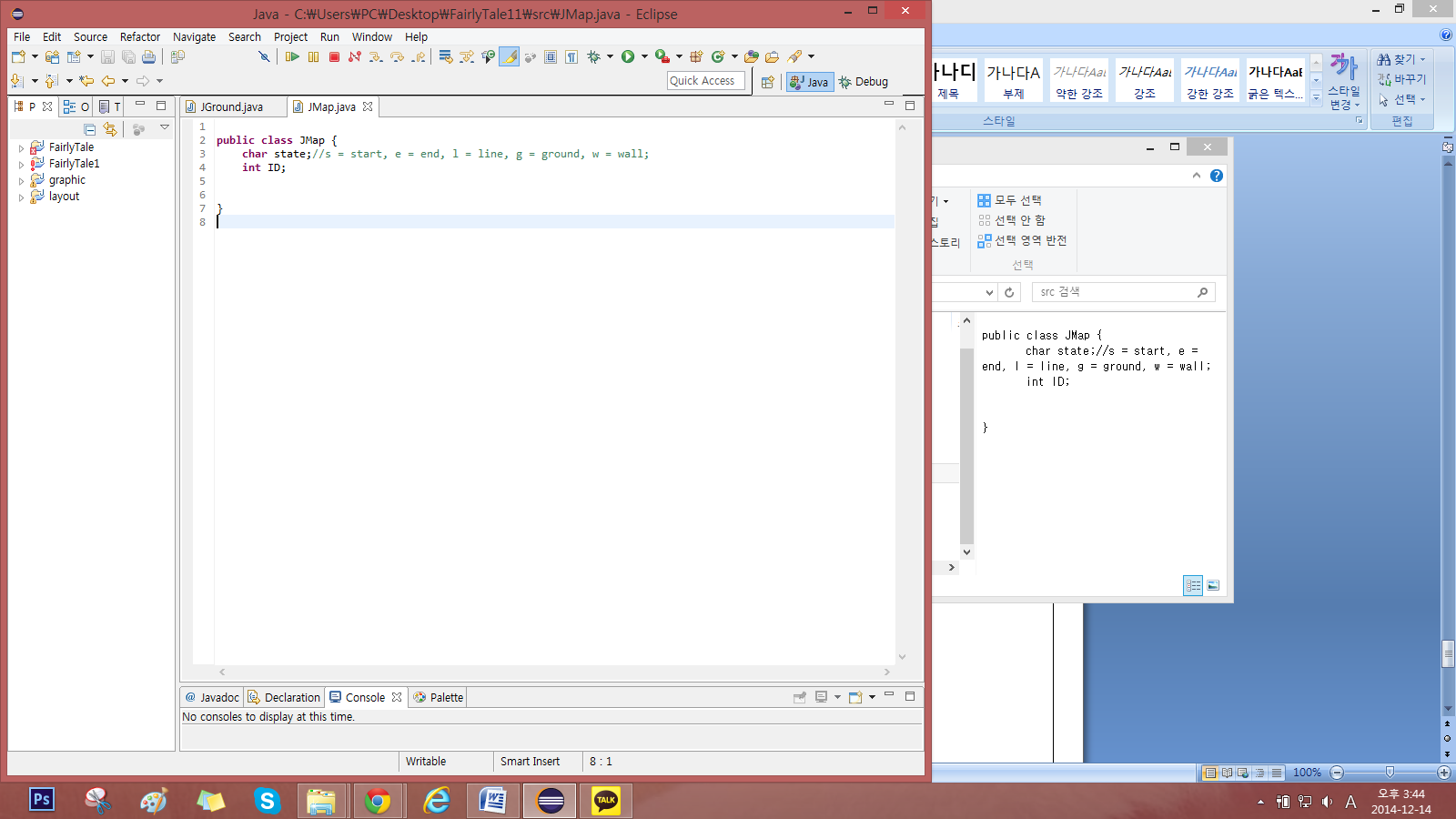
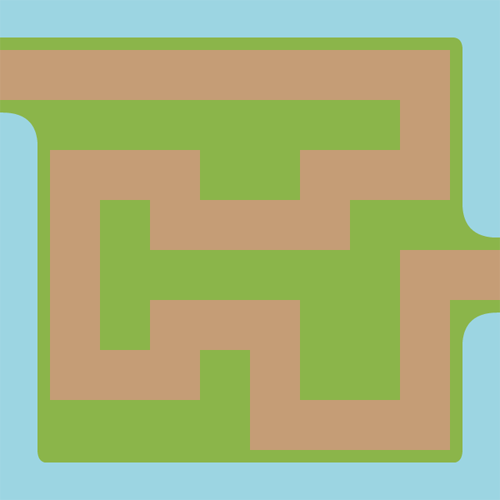
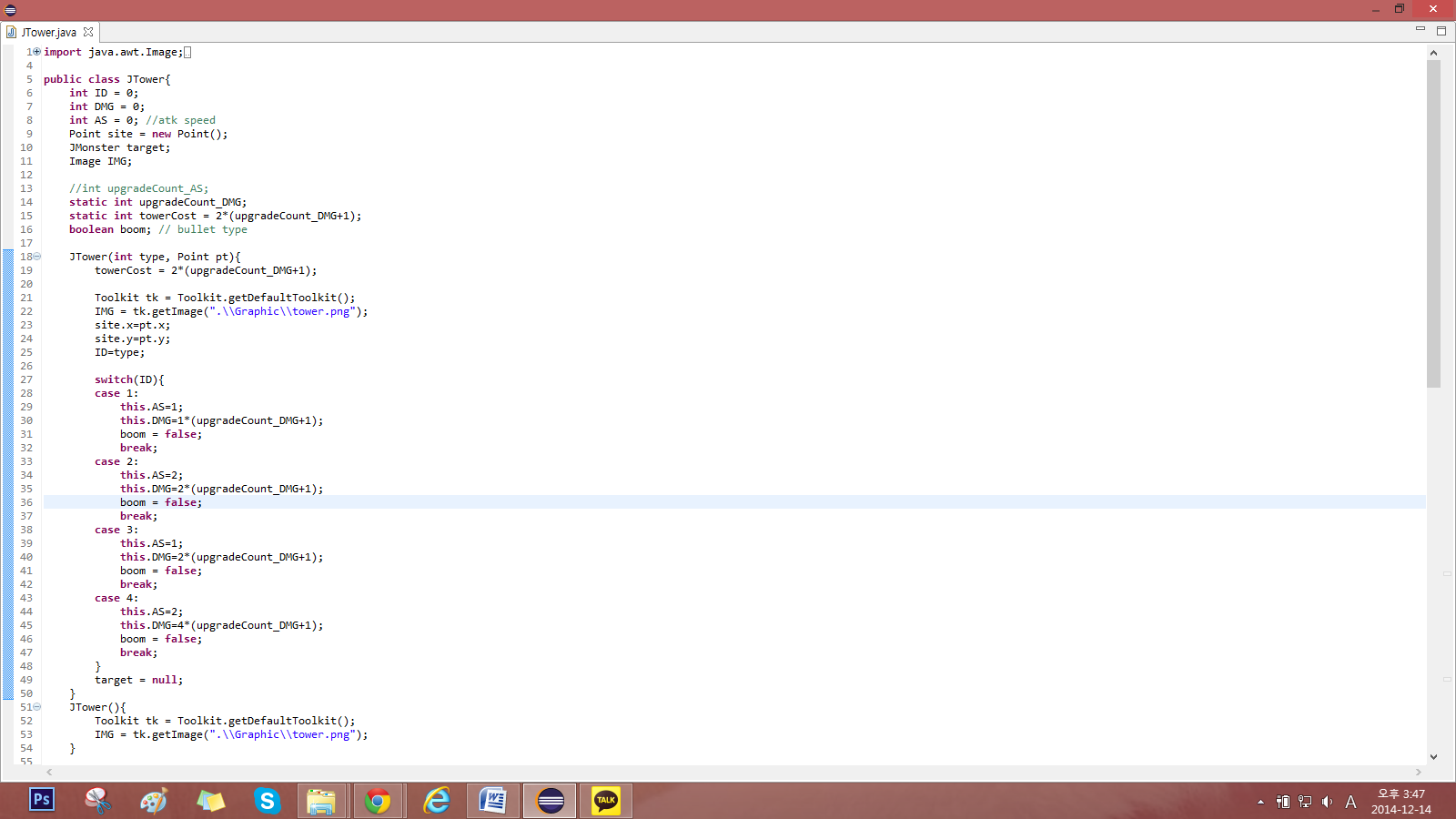
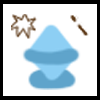
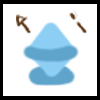
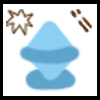
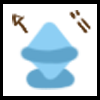
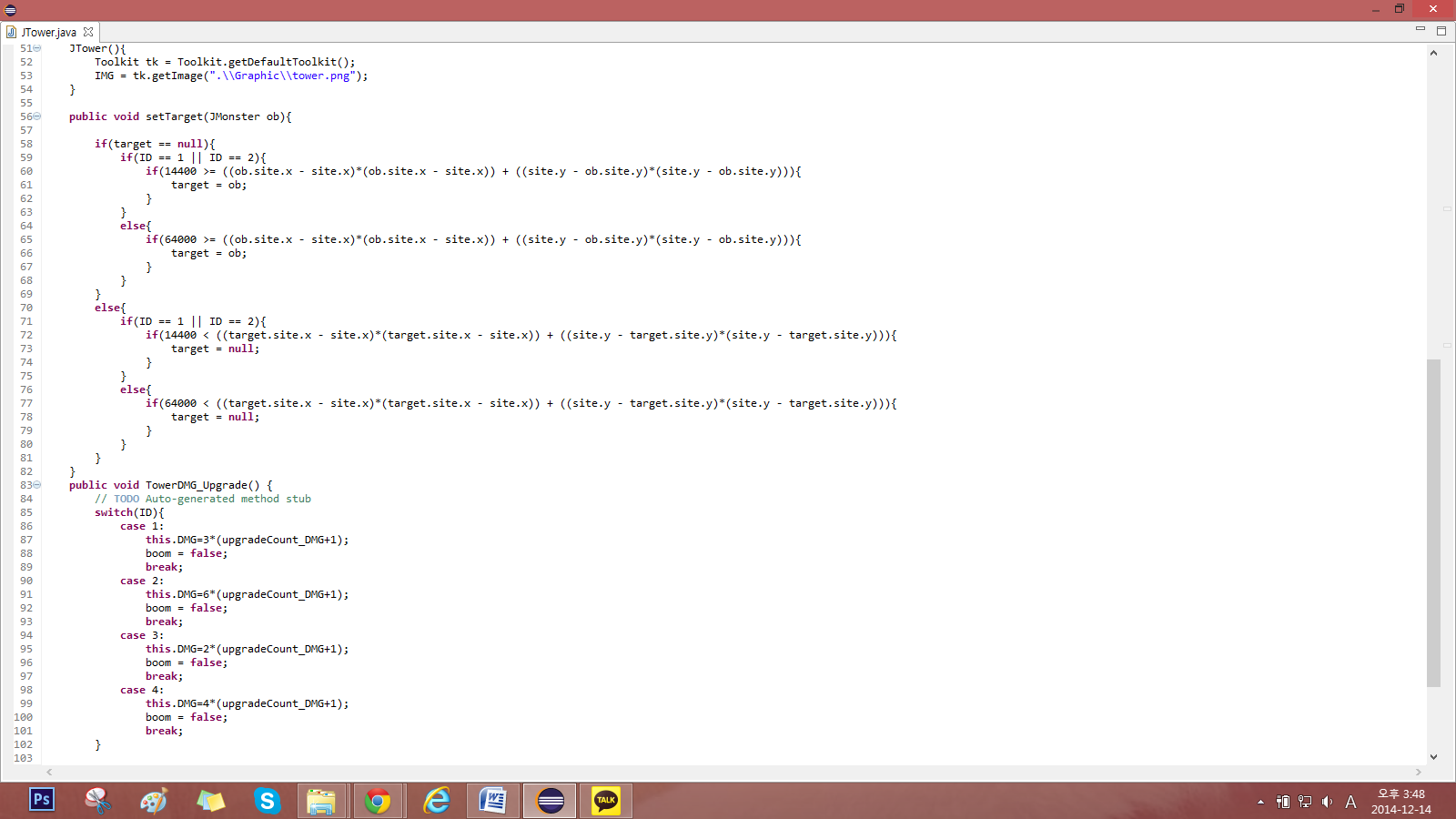
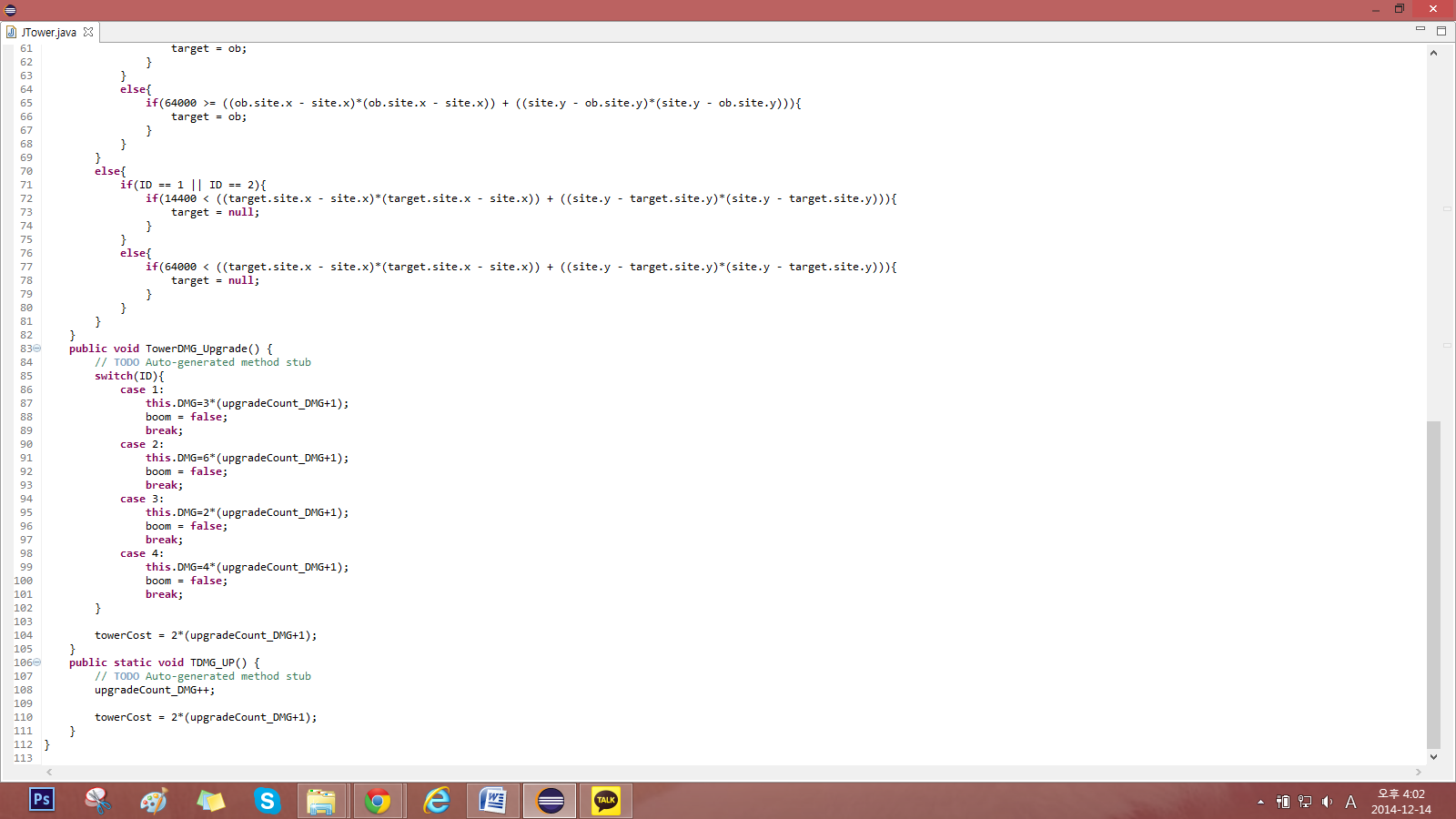
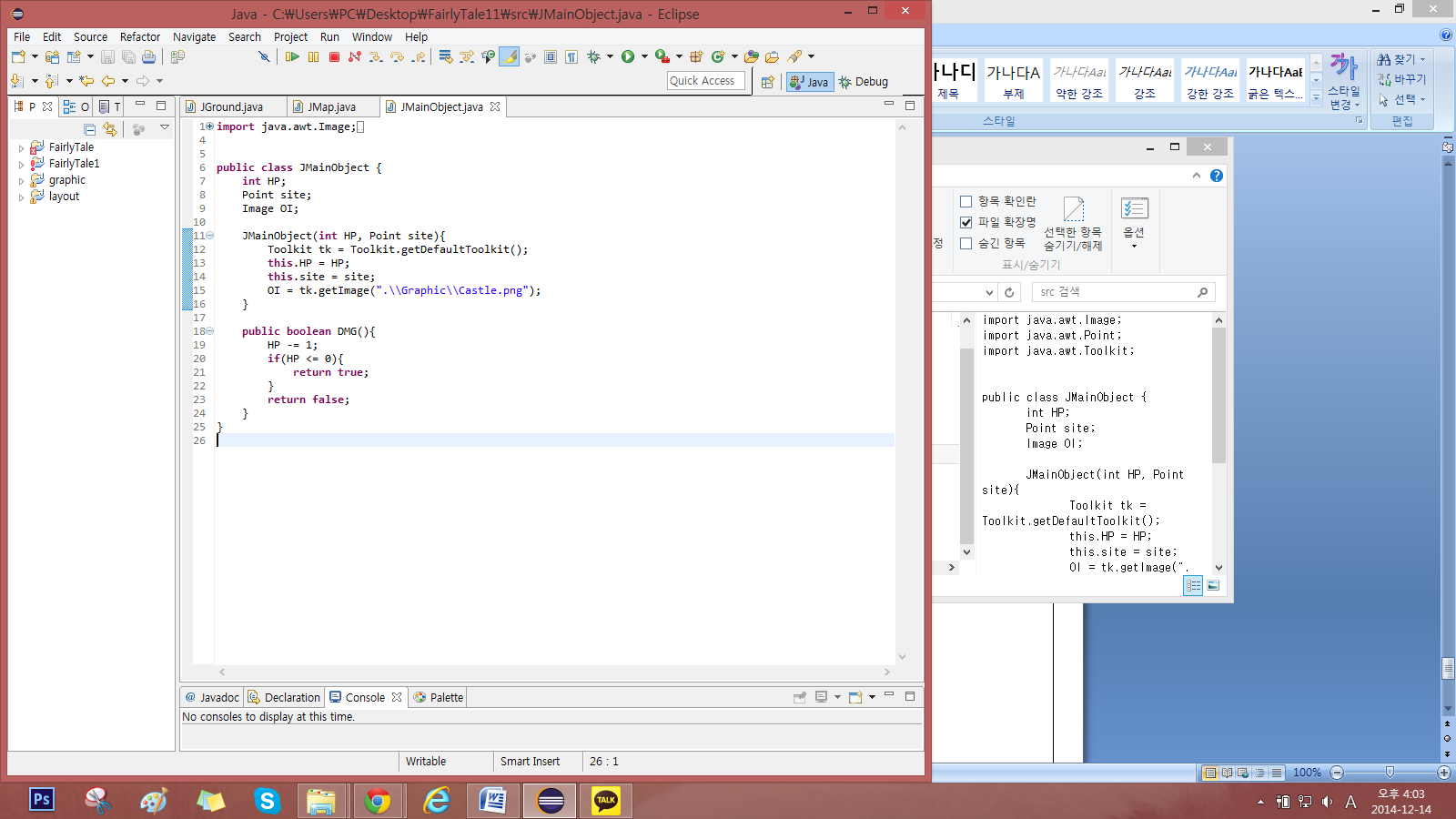
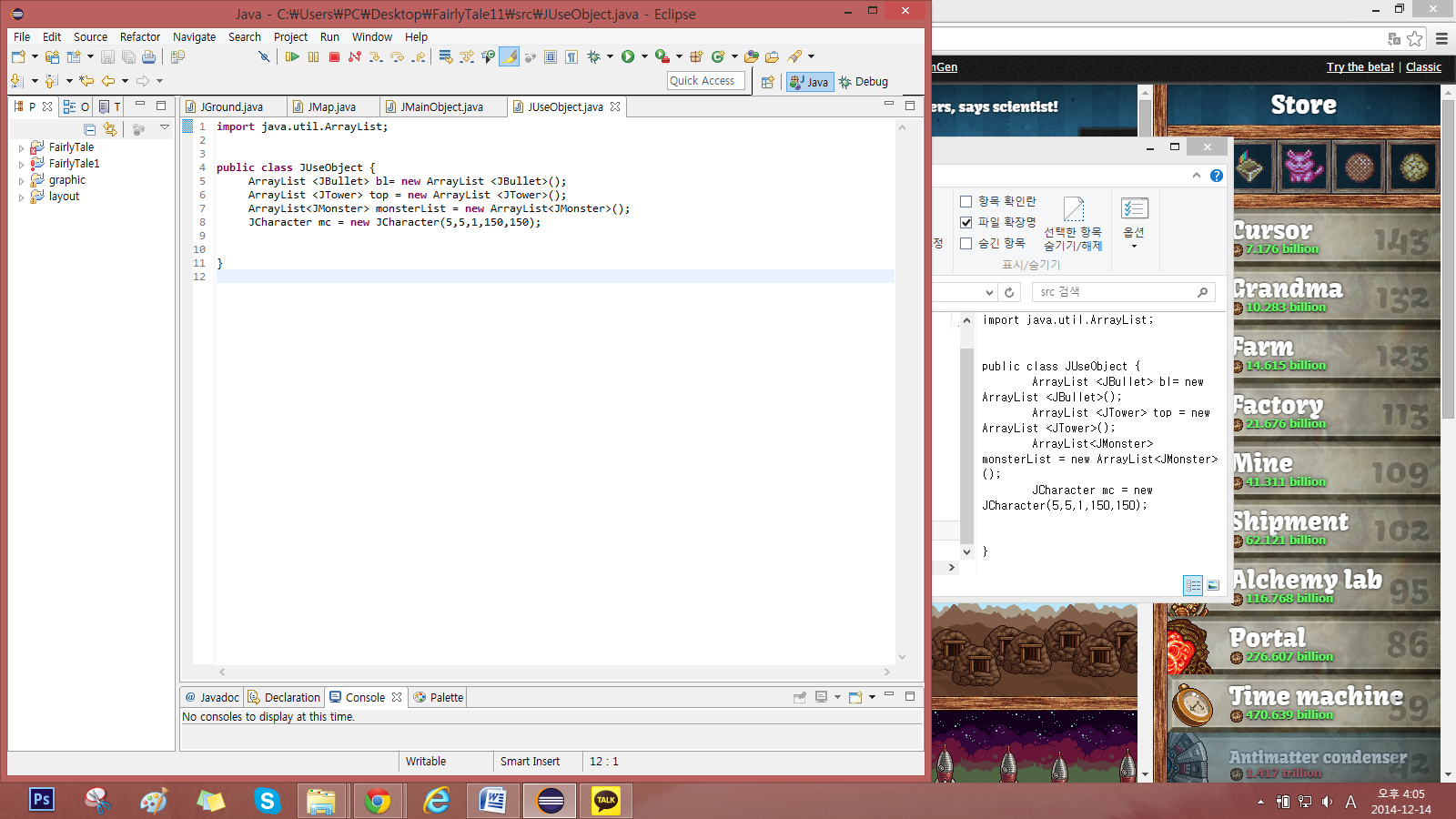
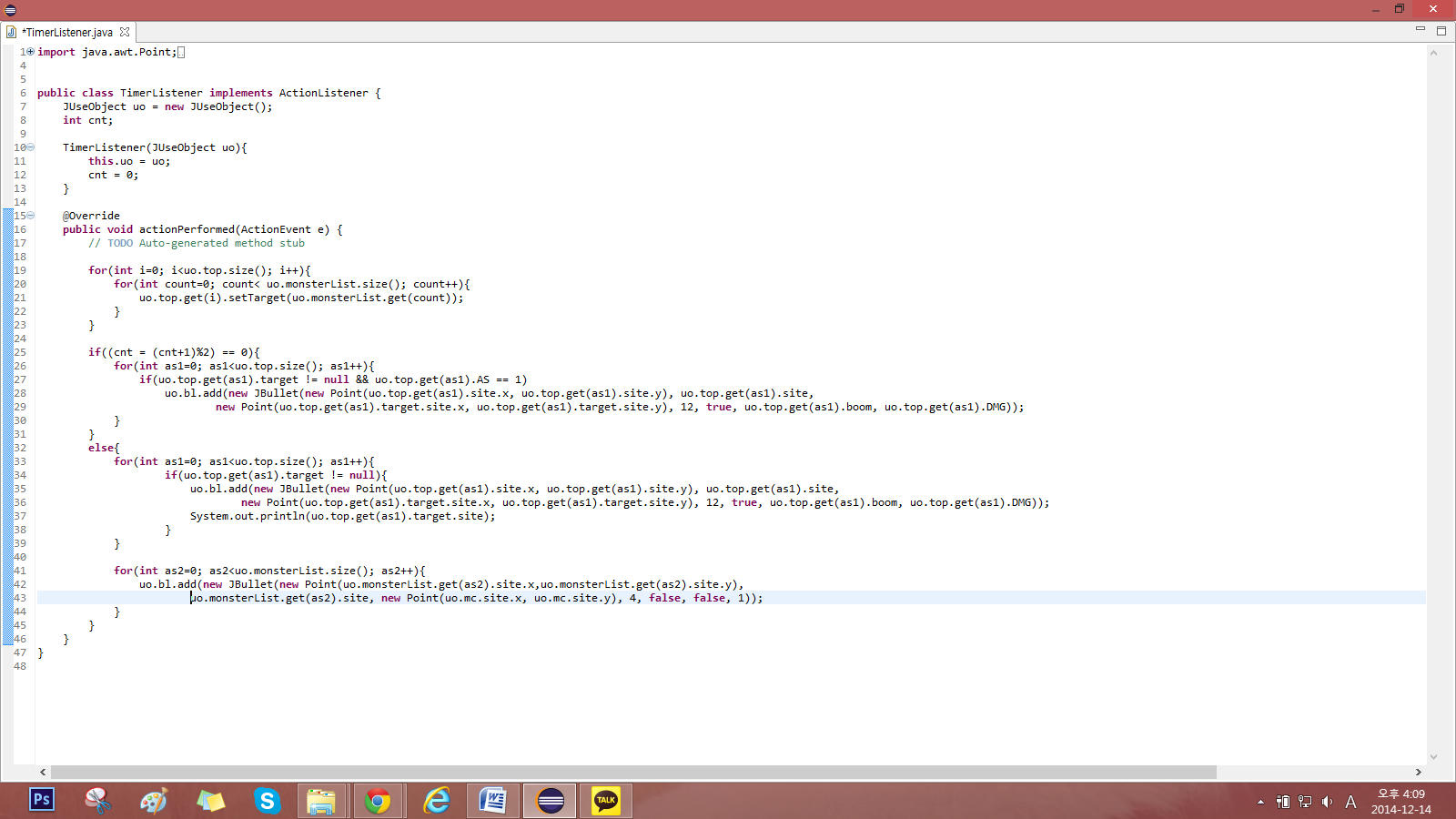
* 1. 윈도우 설계
     1. 화면 구성

**1**

**2**

**3**

* + - 1. 1번의 전체 화면에서 화면을 크게 두 가지 영역으로 나눌 것 입니다. 왼쪽에는 게임 화면을 오른쪽에는 UI를 배치 할 것입니다.
      2. 2번의 화면에서는 실질적인 게임 플레이가 이루어 지는 공간으로, 맵과 지형, 캐릭터, 몬스터, 건물이 그려질 공간입니다.
      3. 3번의 화면에서는 건물을 건설하거나 캐릭터의 현재 상태, 건물과 캐릭터의 업그레이드, 게임 종료, 저장, 불러오기 등 UI 를 배치할 것입니다. 건물을 건설하는 UI는 게임플레이와 직관됨으로 게임 2번의 화면에 최대한 가깝게 배치할 것 입니다.
  1. 객체 설계
     1. 메인 : FairyTale.java
     2. Body(체력을 가진 객체) : Character(캐릭터), Monster(몬스터) 가 상속
     3. MaintObject(성) : 성에 대한 클래스
     4. Bullet (탄환) : 탄환에 대한 클래스
     5. Ground, Map : 맵과 맵 이미지 등에 대한 클래스
     6. UseObject, damage(데미지 전달) : 공유 클래스 및 메시지 패싱 용 구조체
     7. MainFrame : GUI 구성 클래스
     8. Tower : 타워 클래스
     9. 타이머 클래스

1. 최종 결과  
   1. C:\Users\PC\Desktop\FairlyTale11\Graphic\Character.png JCharacter(캐릭터)
   2. C:\Users\PC\Desktop\FairlyTale11\Graphic\Level4.pngC:\Users\PC\Desktop\FairlyTale11\Graphic\Level2.pngC:\Users\PC\Desktop\FairlyTale11\Graphic\Level3.pngC:\Users\PC\Desktop\FairlyTale11\Graphic\Level1.pngJMonster
   3. JBody
   4. C:\Users\PC\Desktop\FairlyTale11\Graphic\Alias.pngC:\Users\PC\Desktop\FairlyTale11\Graphic\Enermy.pngJBullet
   5. damage
   6. JGround
   7. JMap
   8. JTower 1
   9. JTower 2
   10. C:\Users\PC\Desktop\FairlyTale11\Graphic\Castle.pngJMainObject(성)
   11. JUseObject
   12. TimerListener
   13. JMainFrame2(GUI)

import java.awt.Color;

import java.awt.Container;

import java.awt.Graphics;

import java.awt.GridLayout;

import java.awt.Image;

import java.awt.Point;

import java.awt.Toolkit;

import java.awt.event.KeyEvent;

import java.awt.event.KeyListener;

import java.awt.event.MouseEvent;

import java.awt.event.MouseListener;

import java.awt.event.MouseMotionListener;

import java.awt.event.WindowEvent;

import java.awt.event.WindowListener;

import java.io.DataInputStream;

import java.io.DataOutputStream;

import java.io.FileInputStream;

import java.io.FileOutputStream;

import java.io.IOException;

import java.util.Calendar;

import javax.swing.JFrame;

import javax.swing.JOptionPane;

import javax.swing.JPanel;

@SuppressWarnings("serial")

public class JMainFrame2 extends JFrame implements Runnable, KeyListener, MouseMotionListener, MouseListener, WindowListener {

JUseObject useobject;

private int keyStatus = 0;

char map\_type[][] = {

{'W','W','W','W','W','W','W','W','W','W'},

{'S','R','R','R','R','R','R','R','R','W'},

{'W','G','G','G','G','G','G','G','R','W'},

{'W','R','R','R','G','G','R','R','R','W'},

{'W','R','G','R','R','R','R','G','G','W'},

{'W','R','G','G','G','G','G','G','R','E'},

{'W','R','G','R','R','R','G','G','R','W'},

{'W','R','R','R','G','R','G','G','R','W'},

{'W','G','G','G','G','R','R','R','R','W'},

{'W','W','W','W','W','W','W','W','W','W'}

};

JGround map[][] = new JGround[10][10];

int map\_id[][] = new int[10][10];

BackGroundImage BGI = new BackGroundImage();

Image tower1, tower2, tower3, tower4, upgrade, savei, loadi, exiti;

int times, timem, timeh, timest, monstime;

Calendar cal, cal2;

JMainObject castle;

Point mouseP = new Point();

int mousemode=0;//마우스 이벤트 상태

int createtower=0;//타워 건설 타입

boolean gameover;

public JMainFrame2() {

setLocation(500,200);

gameover = false;

cal = Calendar.getInstance();

castle = new JMainObject(5, new Point(461,280));

monstime = 0;

times = 0;

timem = 0;

timeh = 0;

timest = 5;

Toolkit tk = this.getToolkit();

tower1 = tk.getImage(".\\Graphic\\buy tower1.png");

tower2 = tk.getImage(".\\Graphic\\buy tower2.png");

tower3 = tk.getImage(".\\Graphic\\buy tower3.png");

tower4 = tk.getImage(".\\Graphic\\buy tower4.png");

upgrade = tk.getImage(".\\Graphic\\Upgrade.png");

savei = tk.getImage(".\\Graphic\\Save.png");

exiti = tk.getImage(".\\Graphic\\Exit.png");

loadi = tk.getImage(".\\Graphic\\Load.png");

this.setSize(1010, 520);

this.setTitle("Fairytale Virus");

this.setResizable(false);

this.setLayout(null);

Container backgroundPane = new Container();

this.add(backgroundPane);

backgroundPane.setLayout(null);

backgroundPane.setBounds(0,0,1020,560);

backgroundPane.add(BGI);

Container groundPane = new Container();

this.add(groundPane);

GridLayout layout = new GridLayout(10,10);

groundPane.setLayout(layout);

groundPane.setBounds(10, 10, 500, 500);

for(int j=0; j<10; j++){

for(int i=0; i<10; i++){

map\_id[i][j]=-10;

map[i][j] = new JGround(10+50\*i, 35+50\*j, map\_type[j][i]);

groundPane.add(map[i][j]);

}

}

Container towerbuyPane = new Container();

this.add(towerbuyPane);

towerbuyPane.setBounds(450, 5, 570, 555);

towerbuyPane.setLayout(null);

this.addKeyListener(this);

this.addMouseMotionListener(this);

this.addMouseListener(this);

this.addWindowListener(this);

setDefaultCloseOperation(JFrame.EXIT\_ON\_CLOSE);

this.setVisible(true);

}

public void run() {

try {

while(true) {

//Move

if((keyStatus & 0x0001) == 0x0001 ) {

if ( useobject.mc.site.x < 413 ) useobject.mc.site.x +=2;

}

if((keyStatus & 0x0010) == 0x0010 ){

if ( useobject.mc.site.x > 60 ) useobject.mc.site.x -=2;

}

if((keyStatus & 0x0100) == 0x0100 ){

if ( useobject.mc.site.y > 85 ) useobject.mc.site.y -=2;

}

if((keyStatus & 0x1000) == 0x1000 ){

if ( useobject.mc.site.y < 438 ) useobject.mc.site.y +=2;

}

//MonsterMove

if(!useobject.monsterList.isEmpty()){

for(int i=0; i<useobject.monsterList.size(); i++){

useobject.monsterList.get(i).move();

if(useobject.monsterList.get(i).nextPoint.size() == useobject.monsterList.get(i).now){

if(castle != null){

if(castle.DMG()){

gameover = true;

windowClosing(null);

}

}

for(int j=0; j<useobject.top.size(); j++){

if(useobject.top.get(j).target == useobject.monsterList.get(i)){

useobject.top.get(j).target = null;

}

}

useobject.monsterList.remove(i);

}

}

}

//OutRangeBullet Delete

for(int ThreadCount=0; ThreadCount<useobject.bl.size(); ThreadCount++){

if(useobject.bl.get(ThreadCount).move()){

useobject.bl.remove(ThreadCount);

ThreadCount--;

}

}

//check Bullet - Monster or Player

for(int ThreadCount1=0; ThreadCount1<useobject.bl.size(); ThreadCount1++){

if(useobject.bl.get(ThreadCount1).whos){ // true

for(int count=0; count< useobject.monsterList.size(); count++){

damage pr = useobject.bl.get(ThreadCount1).shot(useobject.monsterList.get(count).site, 1);

if(pr.hit){

useobject.bl.remove(ThreadCount1);

ThreadCount1--;

useobject.monsterList.get(count).HP-=pr.dam;

if(useobject.monsterList.get(count).HP<=0){

for(int i=0; i<useobject.top.size(); i++){

if(useobject.top.get(i).target == useobject.monsterList.get(count)){

useobject.top.get(i).target = null;

}

}

useobject.mc.mana += useobject.monsterList.get(count).rewardMana;

useobject.monsterList.remove(count);

}

else {

//크앙크앙

}

break;

}

else{

continue;

}

}

}

else{

damage pr = useobject.bl.get(ThreadCount1).shot(useobject.mc.site,1);

if(pr.hit){

useobject.bl.remove(ThreadCount1);

ThreadCount1--;

useobject.mc.HP-= 1;

if(useobject.mc.HP <= 0){

gameover = true;

windowClosing(null);

}

}

}

}

cal2 = Calendar.getInstance();

if(cal2.get(Calendar.SECOND) > cal.get(Calendar.SECOND)){

times += cal2.get(Calendar.SECOND) - cal.get(Calendar.SECOND);

if(times >= 60){

times = 0;

timem += 1;

if(timem >= 60){

timem = 0;

timeh += 1;

}

}

timest -= cal2.get(Calendar.SECOND) - cal.get(Calendar.SECOND);

if(timest < 0){

if(monstime < 5){

timest = 1;

if(useobject.mc.stage < 10){

useobject.monsterList.add(new JMonster(((4\*useobject.mc.stage)+(useobject.mc.stage\*(useobject.mc.stage-1))), 1, useobject.mc.stage, -40, 86, useobject.mc.stage, 2));

monstime ++;

}

else{

useobject.monsterList.add(new JMonster(((4\*useobject.mc.stage)+(useobject.mc.stage\*(useobject.mc.stage-1))\*4), 2, 10, -40, 86, useobject.mc.stage, useobject.mc.stage));

monstime += 5;

}

}

else{

useobject.mc.stage += 1;

monstime = 0;

timest = 5;

}

}

/\*if(useobject.AS1T < 0){

useobject.AS1T = 1;

for(int as1=0; as1<useobject.top.size(); as1++){

if(useobject.top.get(as1).target != null){

useobject.bl.add(new JBullet(new Point(useobject.top.get(as1).site.x, useobject.top.get(as1).site.y), useobject.top.get(as1).site, useobject.top.get(as1).target.site, 8, true, useobject.top.get(as1).boom, useobject.top.get(as1).DMG));

if(useobject.top.get(as1).AS == 1)

useobject.bl.add(new JBullet(new Point(useobject.top.get(as1).site.x, useobject.top.get(as1).site.y), useobject.top.get(as1).site, useobject.top.get(as1).target.site, 8, true, useobject.top.get(as1).boom, useobject.top.get(as1).DMG));

}

}

}

}

for(int ThreadCount2=0; ThreadCount2<useobject.top.size(); ThreadCount2++){

for(int count=0; count< useobject.monsterList.size(); count++){

useobject.top.get(ThreadCount2).setTarget(useobject.monsterList.get(count));

}

}\*/

}

cal = Calendar.getInstance();

repaint();

Thread.sleep(20);

}

}catch (Exception e){}

}

public void paint(Graphics g) {

Image buf = createImage(1020, 600);

Graphics back = buf.getGraphics();

back.clearRect(10, 35, 1020, 600);

for(int i=0; i<10; i++){

for(int j=0; j<10; j++){

back.drawImage(map[i][j].img, map[i][j].x, map[i][j].y, this);

}

}

back.drawImage(tower1, 550, 35, tower1.getWidth(this), tower1.getHeight(this), Color.WHITE, this);

back.drawImage(tower2, 650, 35, tower1.getWidth(this), tower1.getHeight(this), Color.WHITE, this);

back.drawImage(tower3, 550, 135, tower1.getWidth(this), tower1.getHeight(this), Color.WHITE, this);

back.drawImage(tower4, 650, 135, tower1.getWidth(this), tower1.getHeight(this), Color.WHITE, this);

//back.drawImage(upgrade, )

String str = null;

if(castle != null){

back.drawImage(castle.OI, castle.site.x, castle.site.y, castle.OI.getWidth(this), castle.OI.getHeight(this),this);

str = String.format("성 내구도 : %3d", castle.HP);

back.drawString(str, 800, 110);

}

else{

str = String.format("성 내구도 : %3d", 0);

back.drawString(str, 800, 110);

}

str = String.format("게임 진행 시간 : %2d : %2d : %2d", timeh,timem,times);

back.drawString(str, 800, 50);

str = String.format("다음 스테이지 : %3d 초", timest);

back.drawString(str, 800, 70);

str = String.format("스테이지 : %2d", useobject.mc.stage);

back.drawString(str, 800, 90);

if(castle != null){

back.drawImage(upgrade, 800, 170, upgrade.getWidth(this), upgrade.getHeight(this), this);

str = String.format("성 내구도 : %1d", 5);

back.drawString(str, 925, 200);

str = String.format("%3d ===> %3d", castle.HP, castle.HP+1);

back.drawString(str, 920, 235);

}

str = String.format("HP : %2d", useobject.mc.HP);

back.drawString(str, 800, 130);

str = String.format("마나 : %2d", useobject.mc.mana);

back.drawString(str, 800, 150);

back.drawImage(upgrade, 550, 260, upgrade.getWidth(this), upgrade.getHeight(this), this);

str = String.format("캐릭터 공격력 : %3d", 10);

back.drawString(str, 675, 290);

str = String.format("%3d ===> %3d", useobject.mc.DMG, useobject.mc.DMG+1);

back.drawString(str, 690, 328);

back.drawImage(upgrade, 550, 340, upgrade.getWidth(this), upgrade.getHeight(this), this);

str = String.format("캐릭터 체력 회복 : %2d", 5);

back.drawString(str, 670, 370);

str = String.format("%3d ===> %3d", useobject.mc.HP, useobject.mc.HP+5);

back.drawString(str, 690, 408);

back.drawImage(upgrade, 550, 420, upgrade.getWidth(this), upgrade.getHeight(this), this);

str = String.format("타워 업그레이드 : %2d", useobject.top.size()\*(JTower.upgradeCount\_DMG+1)\*2);

back.drawString(str, 670, 450);

str = String.format("%3d ===> %3d", JTower.upgradeCount\_DMG, JTower.upgradeCount\_DMG+1);

back.drawString(str, 690, 488);

back.drawImage(savei, 800, 260, upgrade.getWidth(this), upgrade.getHeight(this), this);

back.drawImage(loadi, 800, 340, upgrade.getWidth(this), upgrade.getHeight(this), this);

back.drawImage(exiti, 800, 420, upgrade.getWidth(this), upgrade.getHeight(this), this);

for(int j=0; j<useobject.bl.size(); j++){

back.drawImage(useobject.bl.get(j).IMG, useobject.bl.get(j).site.x, useobject.bl.get(j).site.y, useobject.bl.get(j).IMG.getWidth(this), useobject.bl.get(j).IMG.getHeight(this), this);

}

for(int i=0; i<useobject.monsterList.size(); i++){

back.drawImage(useobject.monsterList.get(i).CI, useobject.monsterList.get(i).site.x, useobject.monsterList.get(i).site.y, useobject.monsterList.get(i).CI.getWidth(this), useobject.monsterList.get(i).CI.getHeight(this), this);

}

for(int i=0; i<useobject.top.size(); i++){

back.drawImage(useobject.top.get(i).IMG, useobject.top.get(i).site.x, useobject.top.get(i).site.y, useobject.top.get(i).IMG.getWidth(this), useobject.top.get(i).IMG.getHeight(this), this);

}

back.drawImage(useobject.mc.CI, useobject.mc.site.x, useobject.mc.site.y, useobject.mc.CI.getWidth(this), useobject.mc.CI.getHeight(this), this);

g.drawImage(buf, 0, 0, this);

}

public void keyPressed(KeyEvent e){

if(e.getKeyCode()==KeyEvent.VK\_D) keyStatus |= 0x0001;

if(e.getKeyCode()==KeyEvent.VK\_A) keyStatus |= 0x0010;

if(e.getKeyCode()==KeyEvent.VK\_W) keyStatus |= 0x0100;

if(e.getKeyCode()==KeyEvent.VK\_S) keyStatus |= 0x1000;

}

public void keyReleased(KeyEvent e)

{

if(e.getKeyCode()==KeyEvent.VK\_D) keyStatus -= 0x0001;

if(e.getKeyCode()==KeyEvent.VK\_A) keyStatus -= 0x0010;

if(e.getKeyCode()==KeyEvent.VK\_W) keyStatus -= 0x0100;

if(e.getKeyCode()==KeyEvent.VK\_S) keyStatus -= 0x1000;

if(e.getKeyCode()==KeyEvent.VK\_SPACE) useobject.bl.add(new JBullet(new Point(useobject.mc.site.x, useobject.mc.site.y), new Point(useobject.mc.site.x, useobject.mc.site.y), new Point(mouseP.x, mouseP.y), 3, true, false, useobject.mc.DMG));

//미사일 발싸 !

}

public void keyTyped(KeyEvent ke) {}

/\*public static void main(String[] args) {

Thread t = new Thread(new JMainFrame2());

t.start();

//t2.start();

}\*/

@Override

public void mouseDragged(MouseEvent arg0) {

// TODO Auto-generated method stub

}

@Override

public void mouseMoved(MouseEvent arg0) {

// TODO Auto-generated method stub

mouseP = arg0.getPoint();

/\*String str;

str = String.format("%d", mousemode);

System.out.println(str);\*/

}

@Override

public void mouseEntered(MouseEvent arg0) {

// TODO Auto-generated method stub

}

@Override

public void mouseExited(MouseEvent arg0) {

// TODO Auto-generated method stub

}

@Override

public void mousePressed(MouseEvent arg0) {

// TODO Auto-generated method stub

}

@Override

public void mouseReleased(MouseEvent arg0) {

// TODO Auto-generated method stub

}

@Override

public void mouseClicked(MouseEvent e) {

// TODO Auto-generated method stub

// TODO Auto-generated method stub

int x=e.getX();

int y=e.getY();

if(e.getButton() == MouseEvent.BUTTON2){

mousemode = 0;

createtower = 0;

}

switch(mousemode){

case 0 ://초기 상태

if(useobject.mc.mana>=JTower.towerCost){

if(mcheck(x,y,new Point(550,35),tower1.getWidth(this),tower1.getHeight(this))){

this.mousemode=1;

this.createtower=1;

}

if(mcheck(x,y,new Point(650,35),tower1.getWidth(this),tower1.getHeight(this))){

this.mousemode=1;

this.createtower=2;

}

if(mcheck(x,y,new Point(550,135),tower1.getWidth(this),tower1.getHeight(this))){

this.mousemode=1;

this.createtower=3;

}

if(mcheck(x,y,new Point(650,135),tower1.getWidth(this),tower1.getHeight(this))){

this.mousemode=1;

this.createtower=4;

}

}

if( mcheck(x,y, new Point(550,260), upgrade.getWidth(this), upgrade.getHeight(this)) ){

if(useobject.mc.mana>=((useobject.mc.return\_upgradeCount\_DMG()+1))\*10){

useobject.mc.mana-=((useobject.mc.return\_upgradeCount\_DMG()+1))\*10;

useobject.mc.upgrade\_DMG();

}

}

else if( mcheck(x,y, new Point(550,340), upgrade.getWidth(this), upgrade.getHeight(this)) ){

if(useobject.mc.mana>=5){

useobject.mc.mana-=5;

useobject.mc.upgrade\_HP();

}

}

else if( mcheck(x,y, new Point(550,420), upgrade.getWidth(this), upgrade.getHeight(this)) ){

if(useobject.mc.mana>=useobject.top.size()\*((JTower.upgradeCount\_DMG+1)\*2)){

if(useobject.top.size()>0){

useobject.mc.mana-=useobject.top.size()\*((JTower.upgradeCount\_DMG+1)\*2);

JTower.TDMG\_UP();

for(int toco =0; toco <useobject.top.size(); toco++){

useobject.top.get(toco).TowerDMG\_Upgrade();

}

}

}

}

else if( mcheck(x,y, new Point(800,170), upgrade.getWidth(this), upgrade.getHeight(this)) ){

if(useobject.mc.mana>=5 && castle != null){

useobject.mc.mana-=5;

castle.HP += 1;

}

}

else if( mcheck(x,y, new Point(800,260), upgrade.getWidth(this), upgrade.getHeight(this)) ){

save();

}

else if( mcheck(x,y, new Point(800,340), upgrade.getWidth(this), upgrade.getHeight(this)) ){

open();

}

else if( mcheck(x,y, new Point(800,420), upgrade.getWidth(this), upgrade.getHeight(this)) ){

windowClosing(null);

}

break;

case 1 ://타워 생성

if(10<x && 35<y && x<10+500 && y<35+500){//맵 영역 부터 확인

for(int gi=0; gi<10; gi++){

if(mousemode == 0) break;

for(int gj=0; gj<10; gj++){

if(mousemode == 0) break;

if(mcheck(x-25,y-25,32+50\*gi, 60+50\*gj,50,50) && map\_type[gj][gi]=='G'){

boolean is = false;

for(int i=0;i<useobject.top.size();i++){

is=mcheck(10+50\*gi, 35+50\*gj,useobject.top.get(i).site);

if(is){

break;

}

else{

continue;

}

}

if(!is){

useobject.mc.mana-=JTower.towerCost;

mousemode=0;

useobject.top.add(new JTower(createtower, new Point(10+(50\*gi), 35+(50\*gj))));

createtower =0;

}

}

}

}

}

else{

//잠시 보류

}

}

}

class BackGroundImage extends JPanel {

public void paint(Graphics g){

Toolkit tk = this.getToolkit();

Image image = tk.getImage(".\\Graphic\\bgi.png");

g.drawImage(image, 0, 0, this);

}

}

public boolean mcheck(int x, int y, Point site, int width, int hegiht){

if(site.x<x&&site.y<y&&site.x+width>x&&site.y+hegiht>y)

return true;

else

return false;

}

public boolean mcheck(int x, int y, Point site){

if(x==site.x&&y==site.y){

return true;

}

else{

return false;

}

}

public boolean mcheck(int x, int y,int x0, int y0, int wid, int hegiht){

if(x<x0&&y<y0&&x+wid>x0&&y+hegiht>y0)

return true;

else

return false;

}

@SuppressWarnings("static-access")

public void save(){

DataOutputStream wr=null;

int tower\_count=useobject.top.size();

try {

wr = new DataOutputStream(new FileOutputStream("FairySave.dat"));

wr.writeInt(tower\_count);

//wr.writeInt(JTower.upgradeCount\_AS);

wr.writeInt(JTower.upgradeCount\_DMG);

for(int ct=0; ct<tower\_count; ct++){

wr.writeInt(useobject.top.get(ct).DMG);

wr.writeInt(useobject.top.get(ct).AS);

wr.writeInt(useobject.top.get(ct).site.x);

wr.writeInt(useobject.top.get(ct).site.y);

wr.writeInt(useobject.top.get(ct).ID);

wr.writeInt(useobject.top.get(ct).towerCost);

wr.writeBoolean(useobject.top.get(ct).boom);

//wr.writeInt(top.get(ct).upgradeCount\_AS);

//wr.writeInt(top.get(ct).upgradeCount\_DMG);

}

wr.writeInt(useobject.mc.DMG);

wr.writeInt(useobject.mc.HP);

wr.writeInt(useobject.mc.mana);

wr.writeInt(useobject.mc.stage);

wr.writeInt(useobject.mc.site.x);

wr.writeInt(useobject.mc.site.y);

wr.writeInt(useobject.mc.MS);

wr.writeInt(useobject.mc.return\_upgradeCount\_DMG());

wr.writeInt(useobject.mc.return\_upgradeCount\_HP());

wr.writeInt(times);

wr.writeInt(timem);

wr.writeInt(timeh);

} catch (IOException ioe) {

// TODO Auto-generated catch block

System.out.println("저장 실패");

}

finally{

try{

wr.close();

}catch(Exception e1){

}

}

}

@SuppressWarnings("static-access")

public void open(){

DataInputStream wr=null;

try {

wr = new DataInputStream(new FileInputStream("FairySave.dat"));

int tower\_count=wr.readInt();

useobject.top.clear();

useobject.bl.clear();

useobject.monsterList.clear();

timest = 5;

monstime = 0;

for(int read=0; read<tower\_count;read++){

useobject.top.add(new JTower());

}

//JTower.upgradeCount\_AS=wr.readInt();

JTower.upgradeCount\_DMG=wr.readInt();

for(int ct=0; ct<tower\_count; ct++){

useobject.top.get(ct).DMG= wr.readInt();

useobject.top.get(ct).AS= wr.readInt();

useobject.top.get(ct).site.x= wr.readInt();

useobject.top.get(ct).site.y= wr.readInt();

useobject.top.get(ct).ID= wr.readInt();

useobject.top.get(ct).towerCost=wr.readInt();

useobject.top.get(ct).boom =wr.readBoolean();

//top.get(ct).upgradeCount\_AS=wr.readInt();

//top.get(ct).upgradeCount\_DMG=wr.readInt();

}

useobject.mc.DMG=wr.readInt();

useobject.mc.HP=wr.readInt();

useobject.mc.mana=wr.readInt();

useobject.mc.stage=wr.readInt();

useobject.mc.site.x=wr.readInt();

useobject.mc.site.y=wr.readInt();

useobject.mc.MS=wr.readInt();

useobject.mc.read\_up\_count(wr.readInt(), wr.readInt());

times=wr.readInt();

timem=wr.readInt();

timeh=wr.readInt();

} catch (IOException ioe) {

// TODO Auto-generated catch block

System.out.println("읽기 오류");

}

finally{

try{

wr.close();

}catch(Exception e1){

}

}

}

@Override

public void windowActivated(WindowEvent arg0) {

// TODO Auto-generated method stub

}

@Override

public void windowClosed(WindowEvent e) {

// TODO Auto-generated method stub

}

@Override

public void windowClosing(WindowEvent e) {

// TODO Auto-generated method stub

if(gameover){

JOptionPane.showMessageDialog(null, "GAME OVER" );

System.exit(0);

}

else

if(JOptionPane.showConfirmDialog(null, "종료하시겠습니까?", "확인", JOptionPane.YES\_NO\_OPTION, JOptionPane.PLAIN\_MESSAGE) == JOptionPane.OK\_OPTION){

System.exit(0);

}

}

@Override

public void windowDeactivated(WindowEvent e) {

// TODO Auto-generated method stub

}

@Override

public void windowDeiconified(WindowEvent e) {

// TODO Auto-generated method stub

}

@Override

public void windowIconified(WindowEvent e) {

// TODO Auto-generated method stub

}

@Override

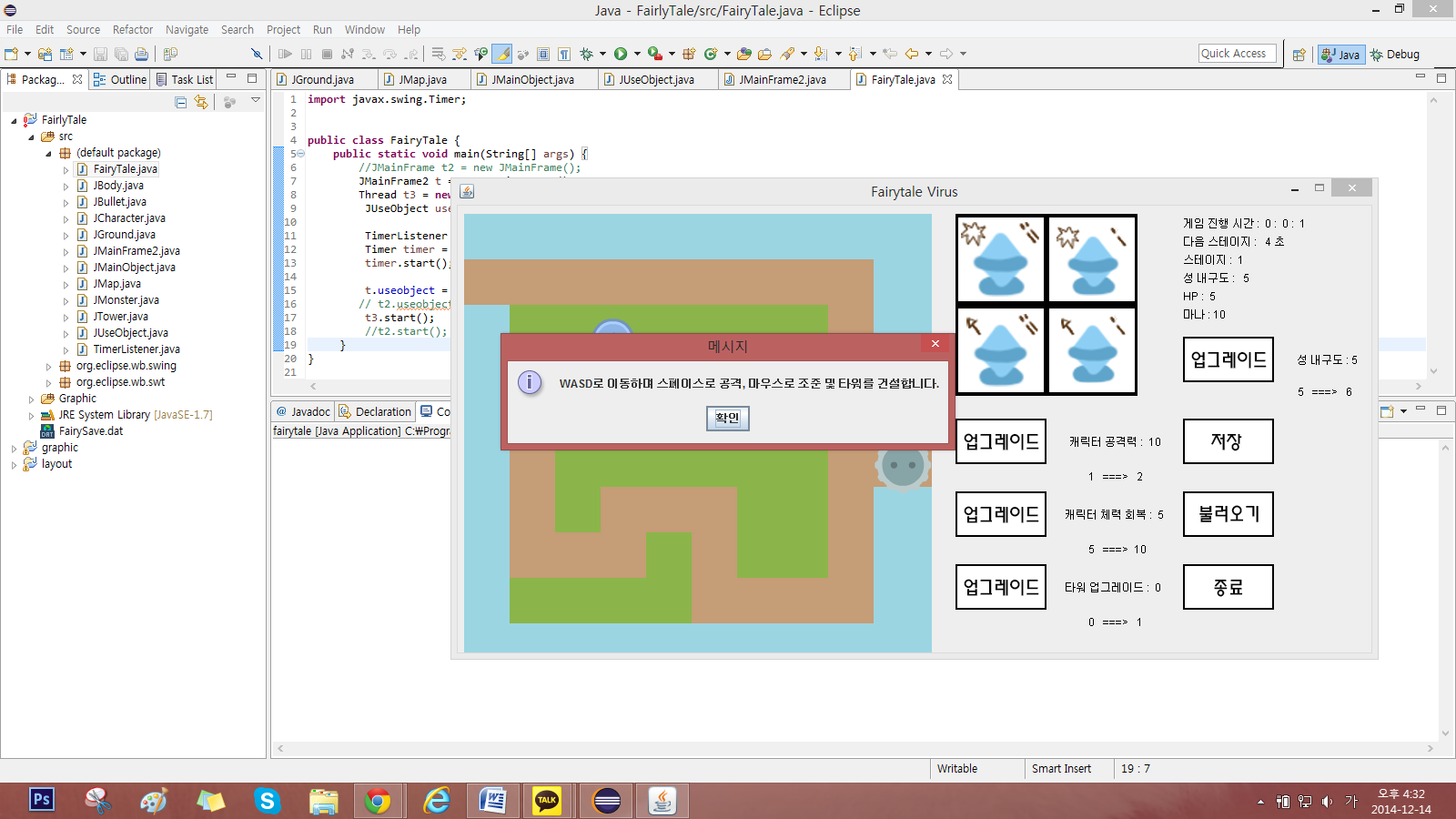
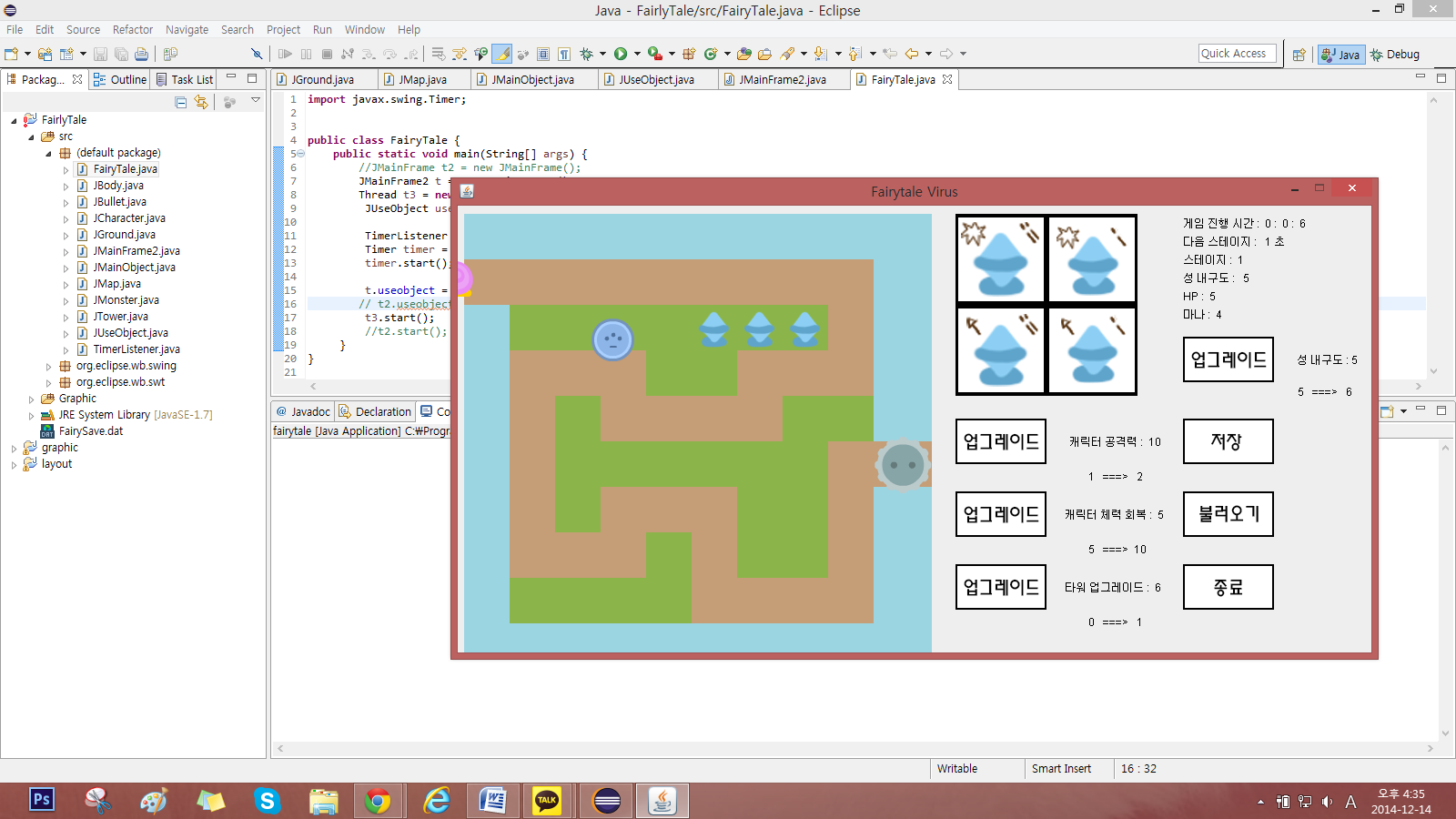
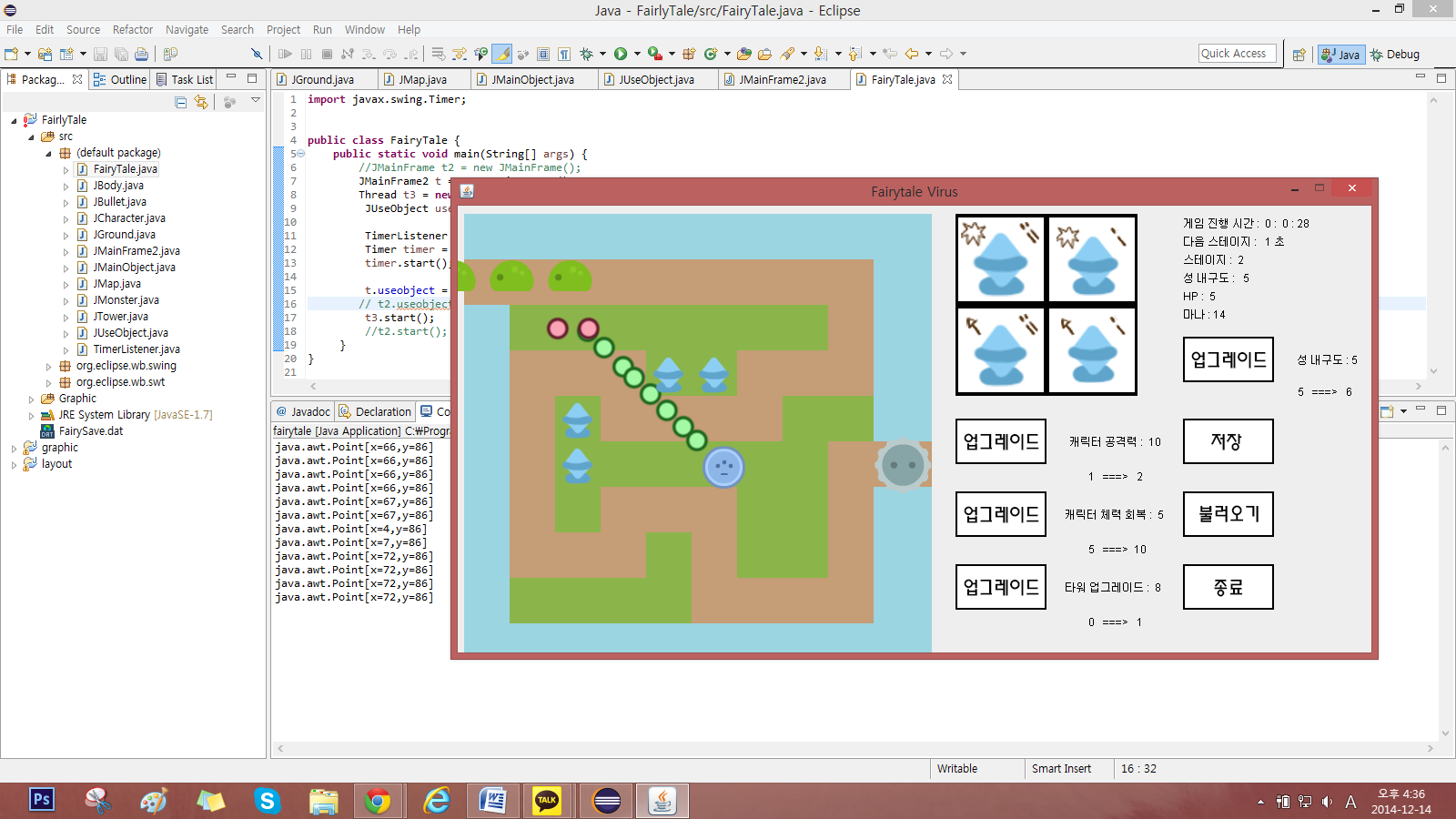
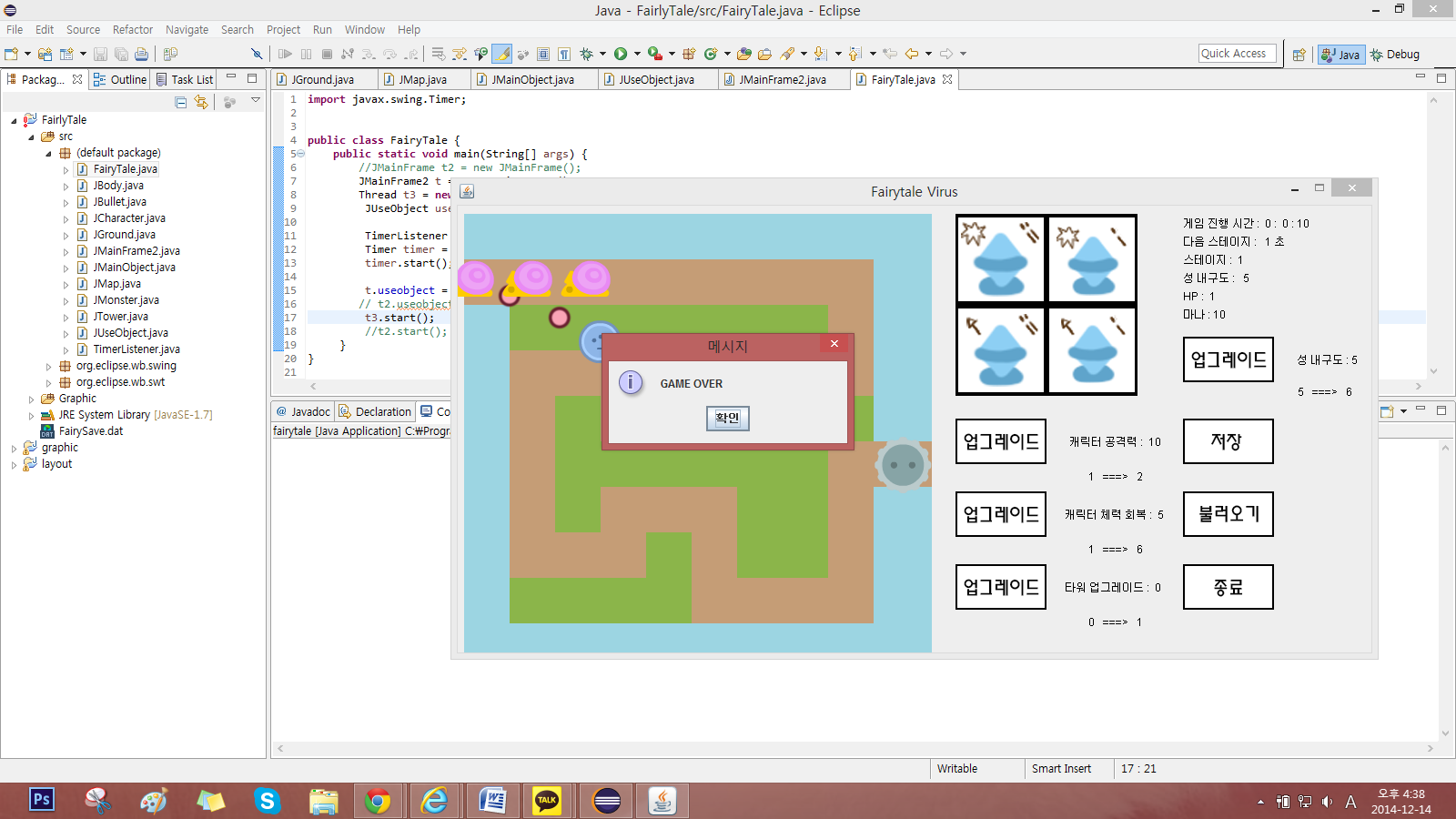
public void windowOpened(WindowEvent e) {

// TODO Auto-generated method stub

JOptionPane.showMessageDialog(null, "WASD로 이동하며 스페이스로 공격, 마우스로 조준 및 타워를 건설합니다.");

}

}

1. 실행   
     
     
     
     
     
   
2. 첨부
   1. 어려웠던 점 :
      1. 슈팅 게임에서 타워와 몬스터의 자동 공격(타겟 팅)과 총알들의 충돌 판정 알고리즘 : 게임 매니악스 슈팅게임 알고리즘이라는 책을 참고하며 해결
      2. 우리가 생각한 화면 상의 좌표와 실제 프레임에서의 좌표의 엇갈림 : 프레임에서의 좌표를 콘솔 창에 띄워가며 정확한 좌표 확인
      3. GUI의 이해 부족 : 오라클 홈페이지의 GUI 관련 클래스를 찾아보고, 프로그래밍 사이트에서 클래스 사용법에 대해 숙지 후 사용, 주로 교재 참고.
      4. 그래픽 소스의 부재 : 공동 구매한 그래픽 소스 사용, 일부 그래픽 소스는 포기
      5. 스레드를 20ms마다 돌려서 새 화면을 구성하는데 그 시간에 타워 판정과 총알 발사 등 객체들의 행동들과 그 계산을 처리하는 것이 불가능하여 타이머를 추가로 공부하여 해결.
   2. 후기  
      알고리즘이 많이 어렵다거나 사용하는 클래스나 기술들이 어렵지 않았으며, 복잡하고 양이 많을 뿐 중간고사 이후부터 꾸준히 해오면 충분히 가능한 프로젝트. 만약 그래픽 소스와 어느 정도 실력이 더 있었으면 더 완성도 높은 게임을 제작했을 듯