헬로, 안드로이드

7주차 – 멀티미디어

강대기
동서대학교 컴퓨터정보공학부

학습 목표

- 오디오를 재생하는 방법에 대해 알아본다.
- 비디오를 재생하는 방법에 대해 알아본다.
- 스도쿠 게임에 음향 효과를 추가해 본다.
- 2D 그래픽을 심화 학습하기 위해, 커스텀 뷰에 대해 학습하고 이벤트 핸들링과 연결해 본다.

차례

- 오디오 재생하기
- 비디오 재생하기
- 스도쿠에 음향 추가하기
- 커스텀뷰
- 이벤트 핸들링
- 요약
- 퀴즈
- 연습문제

오디오 재생하기

- android.media 패키지 안의 MediaPlayer 클래스를 통해 소리와 음악의 출력을 지원함
- 사운드는 가장 쉽게는 윈도우의 녹음기(Sound Recorder)를 통해 생성할 수 있음
- 새 프로젝트 생성
  - Project name – Audio
  - Package name – org.example.audio
  - Activity name – Audio
  - Application name – Audio

오디오 재생하기

- 사운드 파일을 프로젝트의 res/raw 디렉토리 안에 복사함
- res 디렉토리에 파일이 복사되면, 안드로이드 이클립스 플러그인이 자동으로 R 클래스에 자바 기호를 정의함
- Audio 액티비티를 코딩 – 각 사운드마다 새 MediaPlayer 인스턴스를 선언하고, onCreate() 메서드 안에서 초기화함

Audio 액티비티의 오디오 재생 부분

```
@Override
 public void onCreate(Bundle savedInstanceState) {
   super.onCreate(savedInstanceState);
   setContentView(R.layout.main);

   // Native rate is 44.1kHz 16 bit stereo, but
   // to save space we just use MPEG-3 22kHz mono
   up = MediaPlayer.create(this, R.raw.up);
   down = MediaPlayer.create(this, R.raw.down);
   left = MediaPlayer.create(this, R.raw.left);
   right = MediaPlayer.create(this, R.raw.right);
   enter = MediaPlayer.create(this, R.raw.enter);

   a = MediaPlayer.create(this, R.raw.a);
   s = MediaPlayer.create(this, R.raw.s);
   d = MediaPlayer.create(this, R.raw.d);
   f = MediaPlayer.create(this, R.raw.f);
 }
```

오디오 재생하기

- 다른 방법으로는 MediaPlayer 인스턴스를 선언하고 이를 재사용하는 것으로, 서로 다른 사운드들의 겹치기(overlap)을 방지함 (겹치기는 필요할 때도 있고 필요하지 않을 때도 있음)
- 사운드가 필요할 때마다 새로 MediaPlayer 인스턴스를 만드는 방법도 있으나, 실제 안드로이드 시스템에서는 구현 상의 문제로 불가능함. 프로그램의 실행이 지연되거나 프로그램이 크래시(crash)됨
- 참고로, 녹음을 하려면 MediaRecorder 클래스를 사용함

키 눌림을 감지하여 사운드 재생

- Activity.onKeyDown() 메서드를 오버라이드하여 구현함
- 처음의 switch 부분은 사용자가 누른 키에 적합한 미디어 플레이어를 선택하는 부분
- 선택한 후에, seekTo() 메서드를 불러 소리를 되감기(rewind)하고 start() 메서드로 재생함
- 여기서 start()는 비동기 메서드이므로 사운드의 재생 시간에 관계없이 바로 제어가 프로그램으로 반환됨
- setOnCompletionListener() 를 사용하여 사운드 클립이 완료되면 통보받음

Audio 액티비티의 키눌림 감지 부분

```
@Override
 public boolean onKeyDown(int keyCode, KeyEvent event) {
   MediaPlayer mp;

   switch (keyCode) {
   case KeyEvent.KEYCODE_DPAD_UP:        mp = up;        break;
   case KeyEvent.KEYCODE_DPAD_DOWN:        mp = down;        break;
   case KeyEvent.KEYCODE_DPAD_LEFT:        mp = left;        break;
   case KeyEvent.KEYCODE_DPAD_RIGHT:        mp = right;        break;
   case KeyEvent.KEYCODE_DPAD_CENTER:
   case KeyEvent.KEYCODE_ENTER:        mp = enter;        break;
   case KeyEvent.KEYCODE_A:        mp = a;        break;
   case KeyEvent.KEYCODE_S:        mp = s;        break;
   case KeyEvent.KEYCODE_D:        mp = d;        break;
   case KeyEvent.KEYCODE_F:        mp = f;        break;
   default:        return super.onKeyDown(keyCode, event);
   }

   mp.seekTo(0);
   mp.start();
   return true;
 }
```

디버깅 기술

- 안드로이드의 MediaPlayer 는 변하기 쉽고 불안정함 – 메서드 순서가 바뀌거나 인식할 수 없는 형식이 전달되는 등의 사소한 문제로도 크래쉬됨 – 왜냐하면 MediaPlayer 는 Java 층이 얇은 원시 응용 프로그램으로 성능은 빠르나 오류가 많기 때문
- 리눅스의 프로세스 보호 기능으로 에뮬레이터(또는 폰)과 다른 애플리케이션들은 정상 실행됨
- 개발 과정 중에 진단 정보를 얻기 위해서는, 안드로이드 시스템 로그에 출력되는 메시지(message)와 트레이스백(traceback)을 보는 방법이 있음
- 이클립스의 로그캣(LogCat) 뷰나 adb logcat 명령을 사용함

안드로이드가 지원하는 오디오 형식

- WAV (PCM 압축되지 않음)
- AAC (아이팟 포맷, 보호 안됨)
- MP3 (MPEG-3)
- WMA (Windows Media Audio)
- AMR (음성 코덱)
- OGG (Ogg Vorbis)
- MIDI (악기)
- 대부분의 에뮬레이터에서 제대로 동작하는 포맷은 OGG, WAV 와 MP3 포맷임
- 기본 오디오 포맷은 44.1 KHZ, 16 비트 스테레오 오디오
- MP3 의 경우, 음성은 모노, 음악은 스테레오
- OGG 의 경우, 게임 음향 효과와 같은 짧은 클립에 적합함

비디오 재생하기

- 비디오란 결국 여러 장의 사진이 계속 보여지며, 음성도 동기화됨
- MediaPlayer 가 오디오와 유사한 방식으로 비디오에 적용될 수 있음. 다만, 개발자가 이미지를 그릴 surface (표면)이 필요함. start() 와 stop() 메서드로 미디어 재생을 제어함
- MediaPlayer 대신 VideoView 클래스로 더 간단히 비디오를 임베딩할 수 있음

안드로이드가 지원하는 비디오 형식

- MP4 (MPEG-4 낮은 비트 속도)
- H.263
- H.264 (AVC)

- 윈도우 SDK 상에서는 MP4 만이 안정적으로 동작했음

VideoView 클래스로 비디오를 임베딩

- 새로운 프로젝트를 만듦
- 레이아웃 변경
- onCreate() 메서드 수정
- VideoView 클래스의 setVideoPath() 메서드로 파일을 열고 화상 비율을 유지하면서 크기를 컨테이너에 맞춘 후 재생
- 재생할 파일을 업로드
- AndroidManifest.xml 에 적절한 테마 설정

레이아웃 변경

```
<FrameLayout
  xmlns:android="http://schemas.android.com/a
  pk/res/android"
  android:layout_width="fill_parent"
  android:layout_height="fill_parent">
  <VideoView
    android:id="@+id/video"
    android:layout_height="wrap_content"
    android:layout_width="wrap_content"
    android:layout_gravity="center" />
</FrameLayout>
```

Video.java 의 onCreate() 메서드 수정

```
@Override
  public void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);

    // Fill view from resource
    setContentView(R.layout.main);
    VideoView video = (VideoView)
   findViewById(R.id.video);


    // Load and start the movie
    video.setVideoPath("/sdcard/samplevideo.3gp");
    video.start();

  }
```

재생할 파일 업로드

- 커맨드 라인에서 adb 로 올리거나, 안드로이드의 파일 익스플로러(File Explorer)를 이용하여 이클립스에 파일을 업로드 또는 다운로드

```
C:\> adb push c:\code\samplevideo.mp4
  /data/samplevideo.mp4
```

- 안드로이드는 파일 확장명에 상관하지 않음

AndroidManifest.xml 에 적절한 테마 설정

- 비디오가 제목 줄과 상태 줄을 포함한 전체 화면을 차지하게 테마 설정

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
   package="org.example.video"
   android:versionCode="1"
   android:versionName="1.0">
 <application android:icon="@drawable/icon"
     android:label="@string/app_name">
   <activity android:name=".Video"
      android:label="@string/app_name"
      android:theme="@android:style/Theme.NoTitleBar.Fullscreen">
     <intent-filter>
       <action android:name="android.intent.action.MAIN" />
       <category android:name="android.intent.category.LAUNCHER" />
     </intent-filter>
   </activity>
 </application>
   <uses-sdk android:minSdkVersion="3" />
</manifest>
```

화면을 회전하면 비디오가 다시 시작되는 이유

- 기본적으로 안드로이드는 사용자의 프로그램이 화면 회전에 대해 전혀 모른다고 가정함
- 리소스 변경 내용을 감지하기 위해 안드로이드는 액티비티를 전부 제거하고 처음부터 다시 만듦 – 즉, onCreate()가 다시 호출되고 비디오가 다시 시작됨을 의미함
- 이러한 작동 방식은 많은 애플리케이션에서 문제가 없으므로, 개발자들은 신경 쓰지 않아도 되고, 애플리케이션의 생명 주기와 상태 저장/복원을 테스트하는 유용한 방법이 될 수 있음

전환을 부드럽게 최적화하는 방법

- onDestroy()와 onCreate() 에 걸쳐 유지되는 데이터가 저장된 onRetainNonConfigurationInstance() 메서드를 액티비티 안에 구현하여 현재의 인텐트와 실행 중인 스레드의 참조 등을 포함한 모든 것들을 저장. 정보를 되찾으려면 액티비티의 새 인스턴스에 getLastNonConfigurationInstance()를 사용
- AndroidManifest.xml 안의 android:configChange 속성을 사용해서 처리 가능한 변경 내용을 안드로이드에게 알림

스도쿠에 음향 추가하기

- 메인 액티비티(Sudoku.java)에 음악을 추가하려면, onResume()과 onPause() 메서드를 오버라이드
- onResume()는 액티비티와 사용자가 상호작용을 할 때 호출되므로, 여기서 Music 클래스를 start()
- R.raw.main()은 /raw/main.mp3를 참조함
- 새 액티비티를 resume 하기 전에 현 액티비티를 pause 함. 스도쿠에서는 새 게임을 시작시키면, Sudoku 액티비티가 pause되고, Game 액티비티가 시작됨
- 사용자가 Back 버튼이나 Home 버튼을 누를 때도 onPause() 가 호출됨. Sudoku 액티비티에서는, 이 때 타이틀 음악을 멈춰야 하므로, 여기서 Music.stop()
- 게임 액티비티(Game.java)에서도 onResume()과 onPause() 메서드를 오버라이드

Sudoku.java

```
@Override
 protected void onResume() {
    super.onResume();
    Music.play(this, R.raw.main);
 }

 @Override
 protected void onPause() {
    super.onPause();
    Music.stop(this);
 }
```

Game.java

```
@Override
 protected void onResume() {
   super.onResume();
   Music.play(this, R.raw.game);
 }


 @Override
 protected void onPause() {
   super.onPause();
   Log.d(TAG, "onPause");
   Music.stop(this);

   // Save the current puzzle
   getPreferences(MODE_PRIVATE).edit().putString(PREF_PUZZLE,
       toPuzzleString(puzzle)).commit();
 }
```

Music 클래스

- play()
  - play() 메서드는 stop() 메서드를 호출하여 현재 재생되는 음악을 멈춤
  - MediaPlayer.create()를 사용해서 새로운 MediaPlayer 인스턴스를 생성하고 컨택스트와 리소스 ID를 전달
  - 만들어진 인스턴스는 음악을 반복하도록 옵션을 설정한 후 재생함
- stop()
  - MediaPlayer 인스턴스가 있는지 확인하고 stop()과 release() 메서드를 호출함
  - stop() 메서드는 음악을 정지시키며, release() 메서드는 MediaPlayer 인스턴스와 관련된 시스템 리소스를 풀어줌
  - MediaPlayer 는 자바가 아닌 원시(primitive) 코드와 관련이 많은 리소스이므로, GC가 회수할 때까지 기다릴 수 없음, 따라서 release()를 빠뜨리면 에러가 발생할 수 있음

Music.java

```
import android.content.Context;
import android.media.MediaPlayer;

public class Music {
  private static MediaPlayer mp = null;


  /** Stop old song and start new one */
  public static void play(Context context, int resource) {
    stop(context);

    // Start music only if not disabled in preferences
    if (Settings.getMusic(context)) {
      mp = MediaPlayer.create(context, resource);
      mp.setLooping(true);
      mp.start();
    }
  }


  /** Stop the music */
  public static void stop(Context context) {
    if (mp != null) {
      mp.stop();
      mp.release();
      mp = null;
    }
  }
}
```

자신만의 커스텀 뷰 만들기 (여기서는 RobotView)

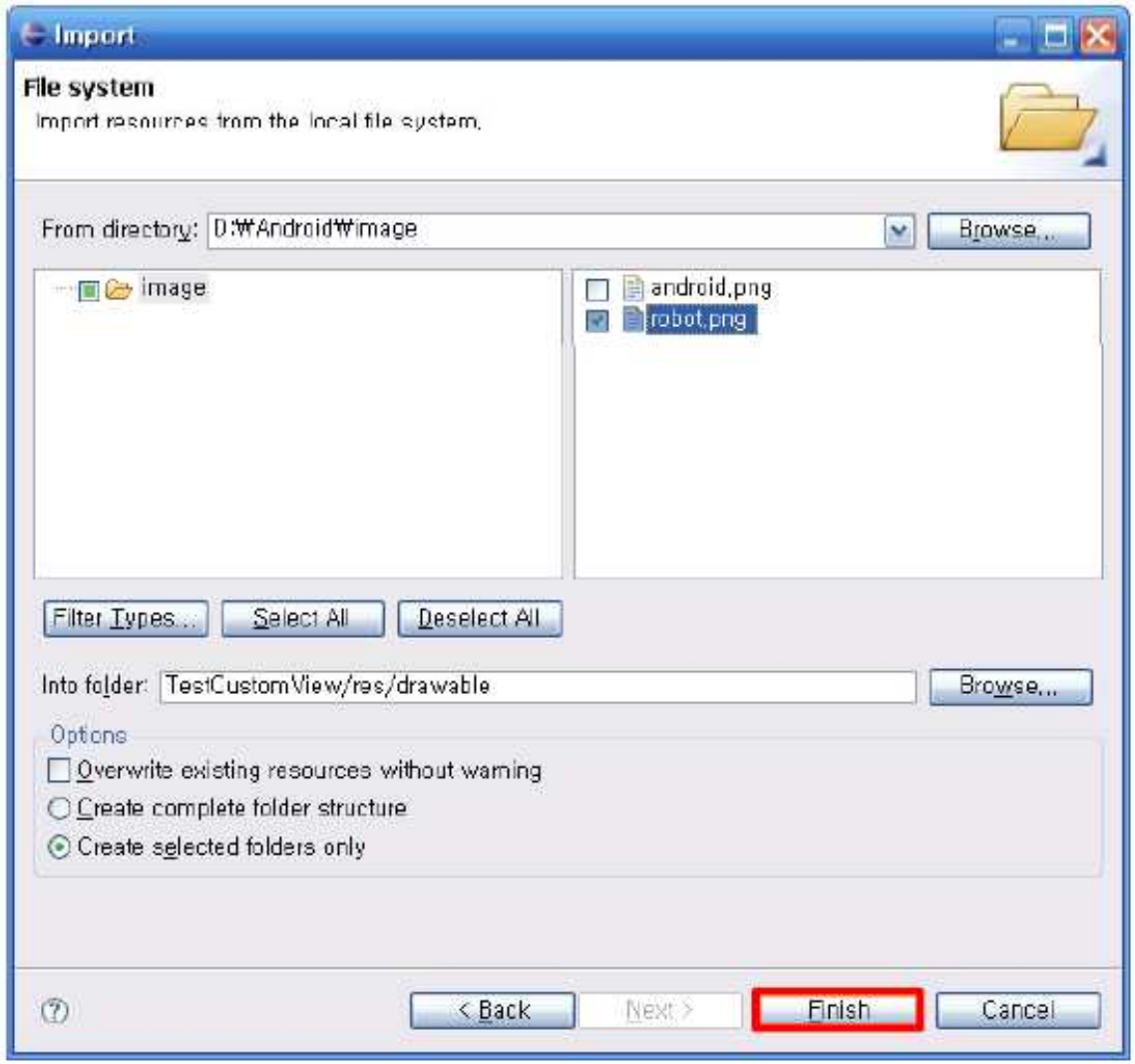
1. 프로젝트 생성
2. 이미지 추가
3. RobotView 클래스 추가
   - android.view.View 확장
4. RobotView() 생성자 추가
5. OnDraw(), OnSizeChanged()
6. 터치 스크린 처리; OnTouchEvent()
7. 키 입력 처리 ; OnKeyDown()

출처 – 한백전자의 Real H/W 기반 안드로이드 응용

새 프로젝트 생성

- Project Name : TestCustomView
- Package Name : hanback.example
- Activity Name : TestCustom
- Application Name : Custom View Test

Import / General /File System으로 파일 추가

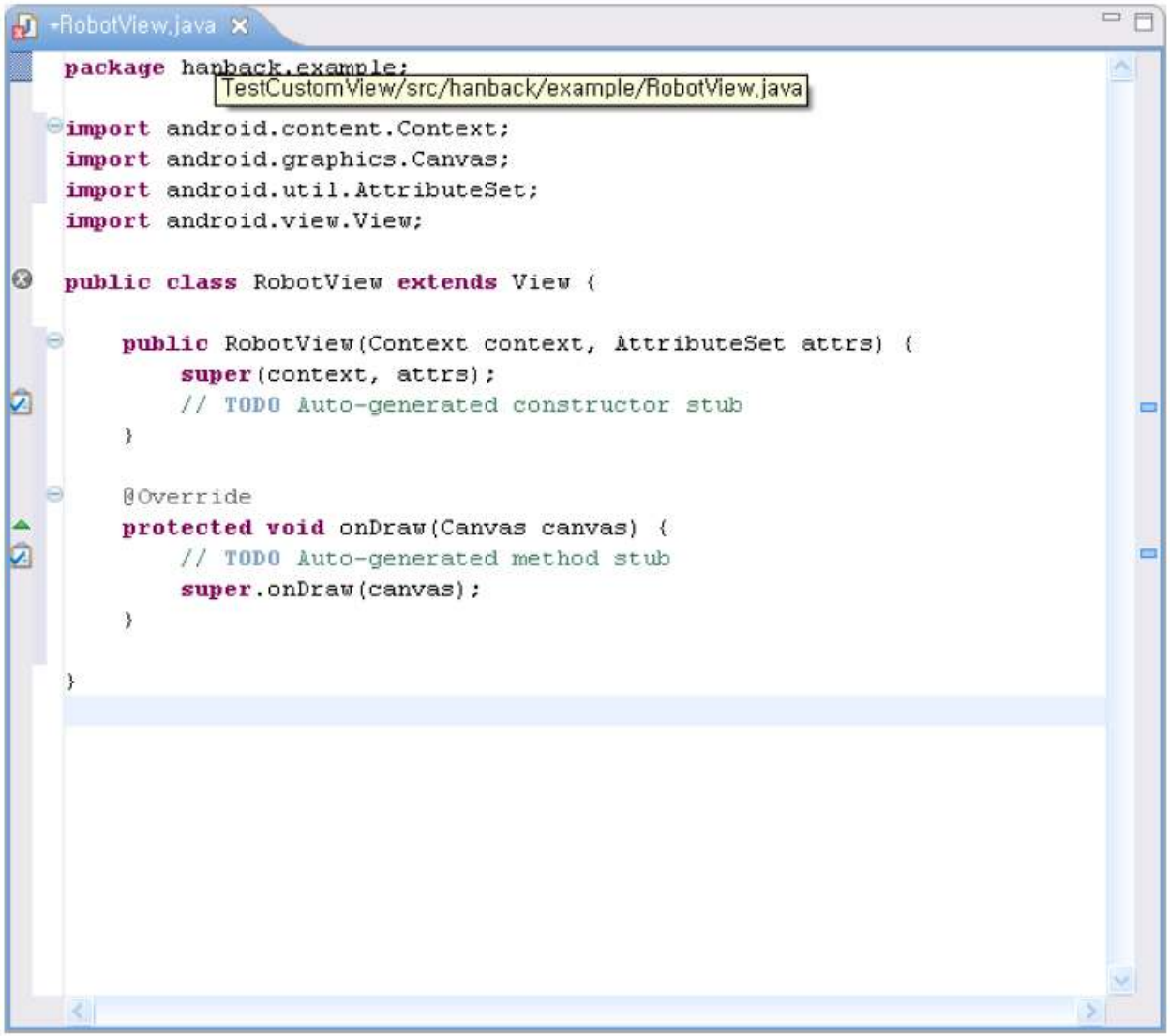


RobotView 클래스 추가

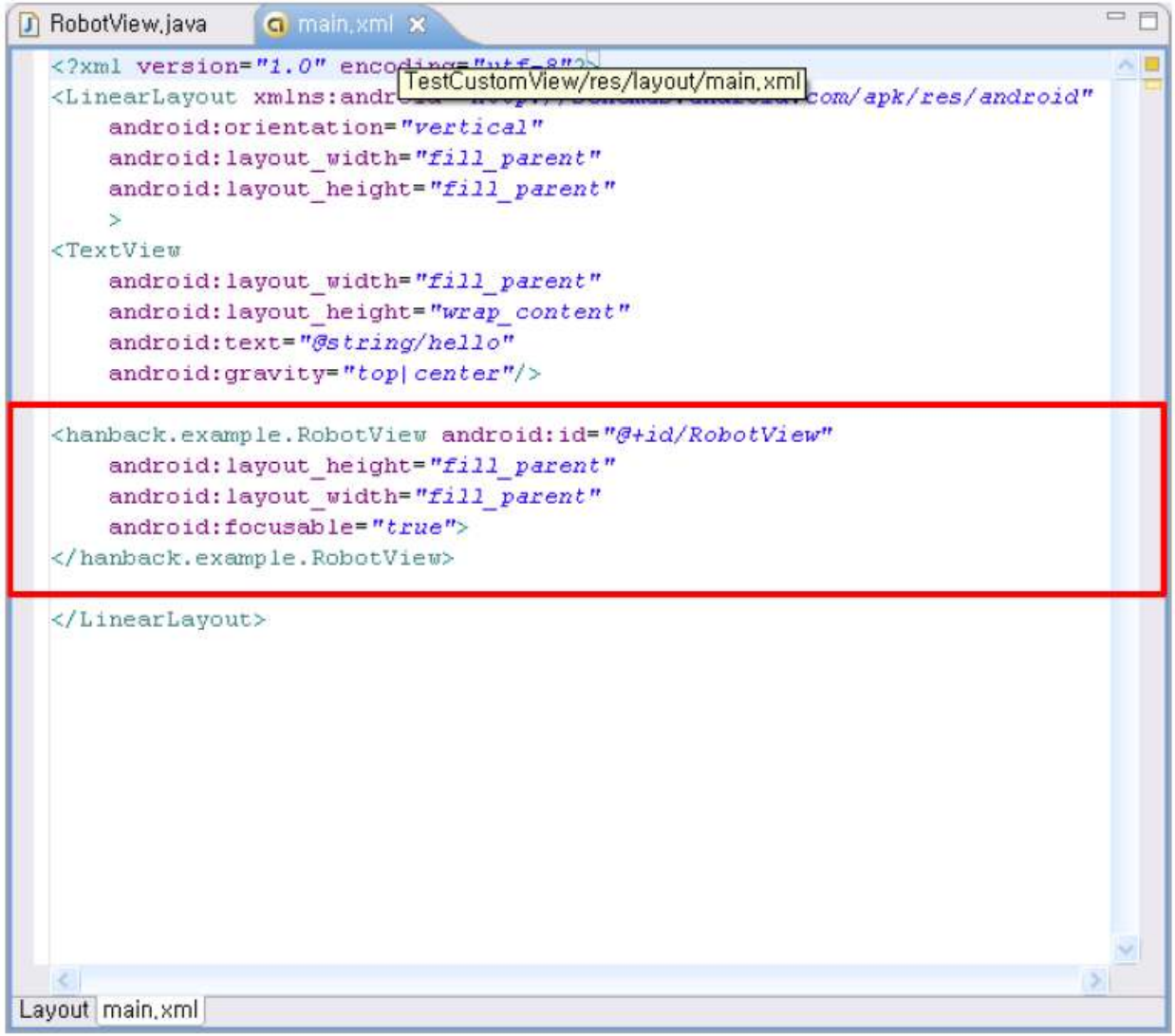
- Package : hanback.example
- Name : RobotView
- Superclass : android.view.View

- Source / Generate Constructors from Superclass 로 생성자 추가
  - View(Context, AttributeSet) 선택
- Source / Override/Implement Methods
  - onDraw(Canvas) 선택

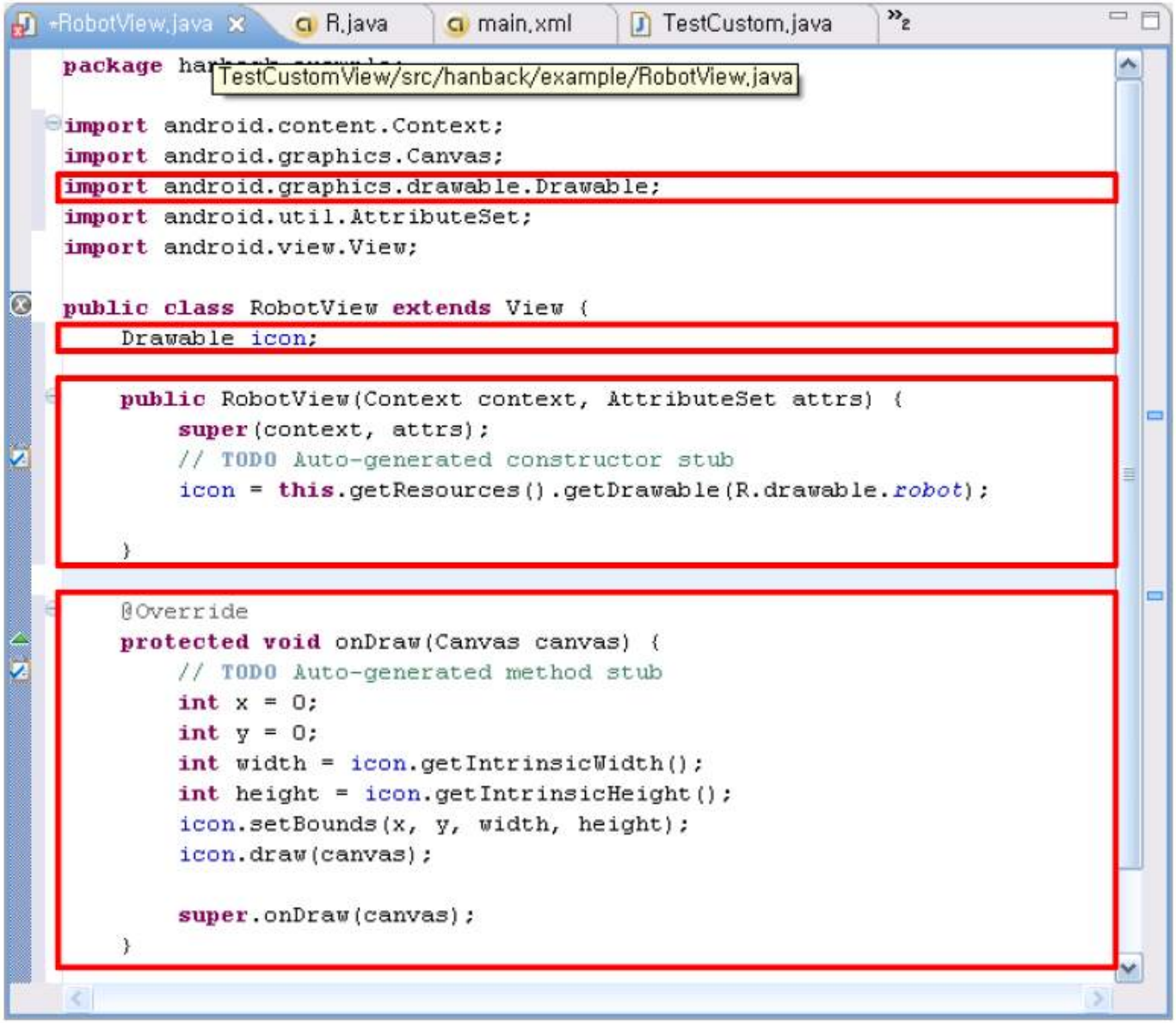
생성된 코드


*RobotView.java
TestCustomView/src/hanback/example/RobotView.java
package hanback.example;
import android.content.Context;
import android.graphics.Canvas;
import android.util.AttributeSet;
import android.view.View;
public class RobotView extends View {
public RobotView(Context context, AttributeSet attrs) {
super(context, attrs);
// TODO Auto-generated constructor stub
}
@Override
protected void onDraw(Canvas canvas) {
// TODO Auto-generated method stub
super.onDraw(canvas);
}
}

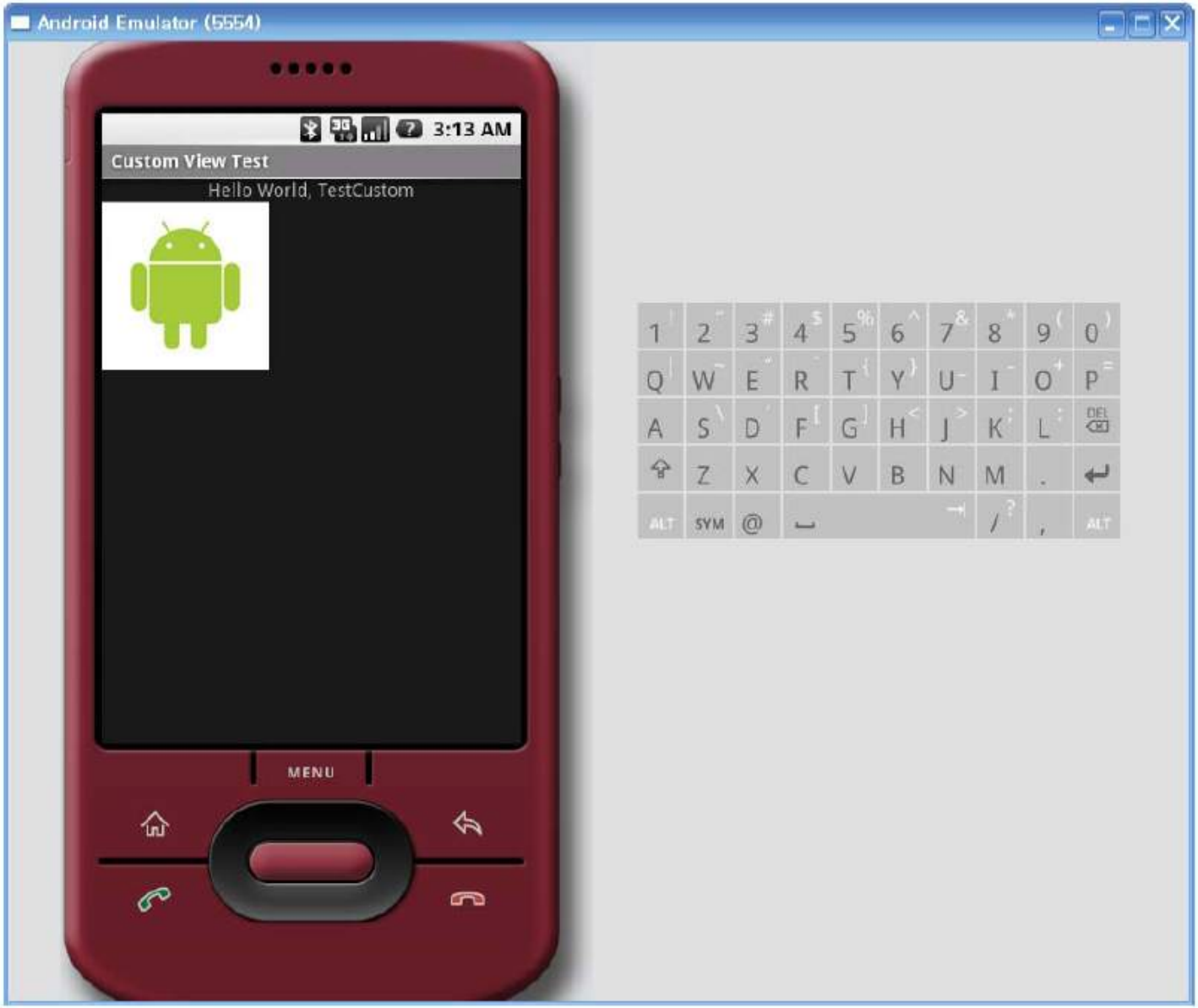
main.xml 에 커스텀 뷰 추가



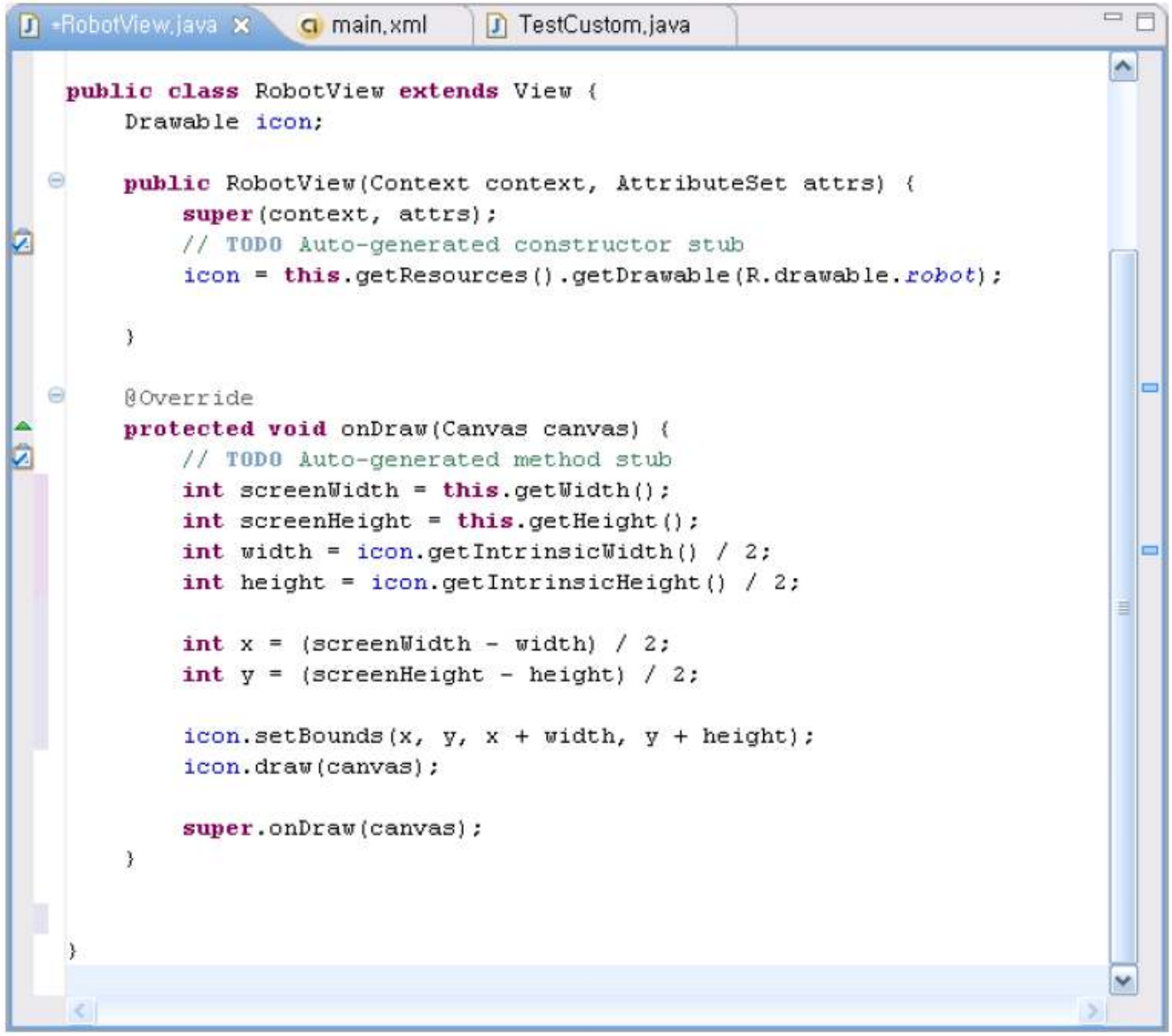
코드 추가


*RobotView.java
R.java
main.xml
TestCustom.java
TestCustomView/src/hanback/example/RobotView.java
package hanback.example;
import android.content.Context;
import android.graphics.Canvas;
import android.graphics.drawable.Drawable;
import android.util.AttributeSet;
import android.view.View;
public class RobotView extends View {
Drawable icon;
public RobotView(Context context, AttributeSet attrs) {
super(context, attrs);
// TODO Auto-generated constructor stub
icon = this.getResources().getDrawable(R.drawable.robot);
}
@Override
protected void onDraw(Canvas canvas) {
// TODO Auto-generated method stub
int x = 0;
int y = 0;
int width = icon.getIntrinsicWidth();
int height = icon.getIntrinsicHeight();
icon.setBounds(x, y, width, height);
icon.draw(canvas);
super.onDraw(canvas);
}

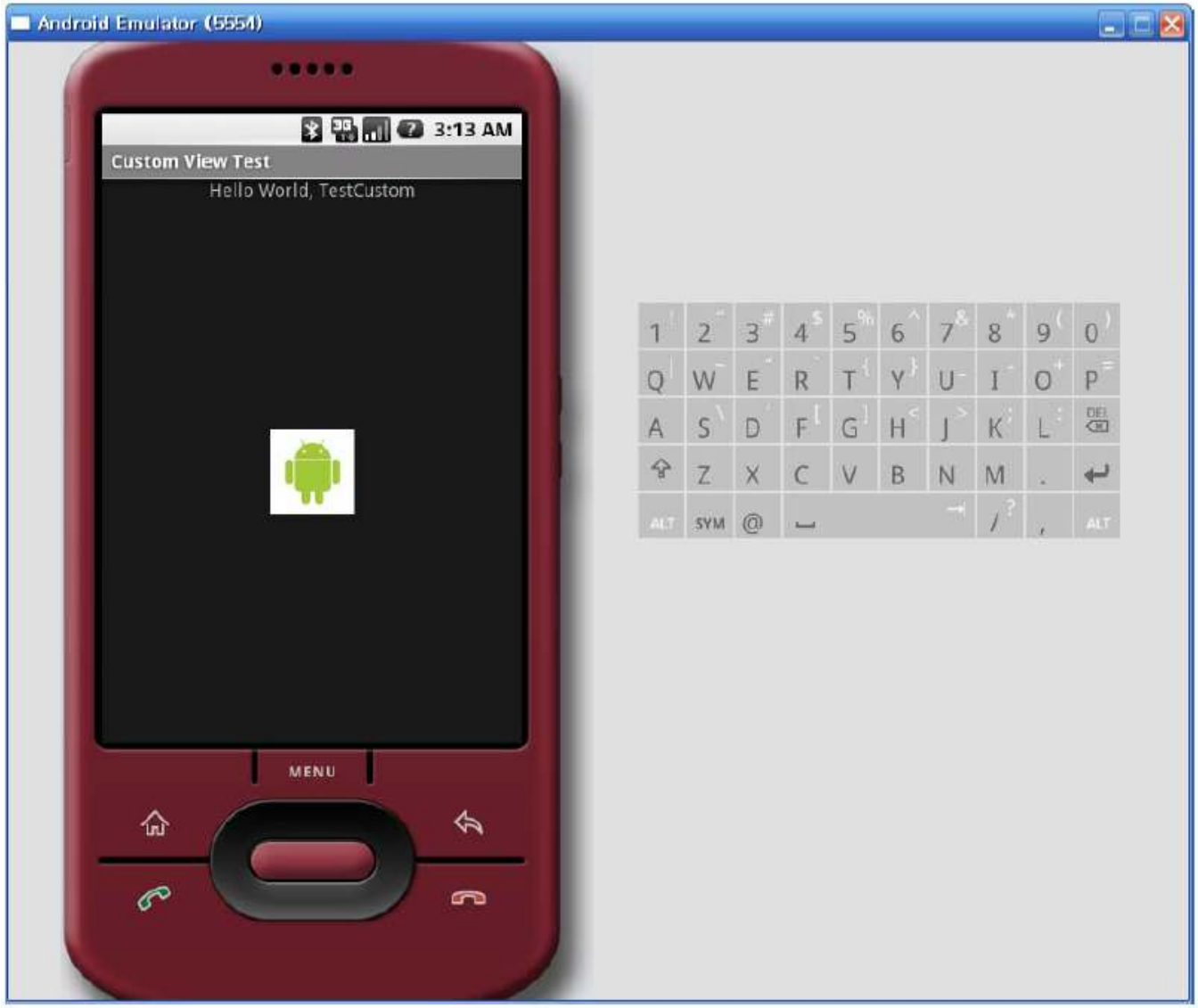
실행 결과


Android Emulator (5554)
3:13 AM
Custom View Test
Hello World, TestCustom
MENU

소스 코드 수정

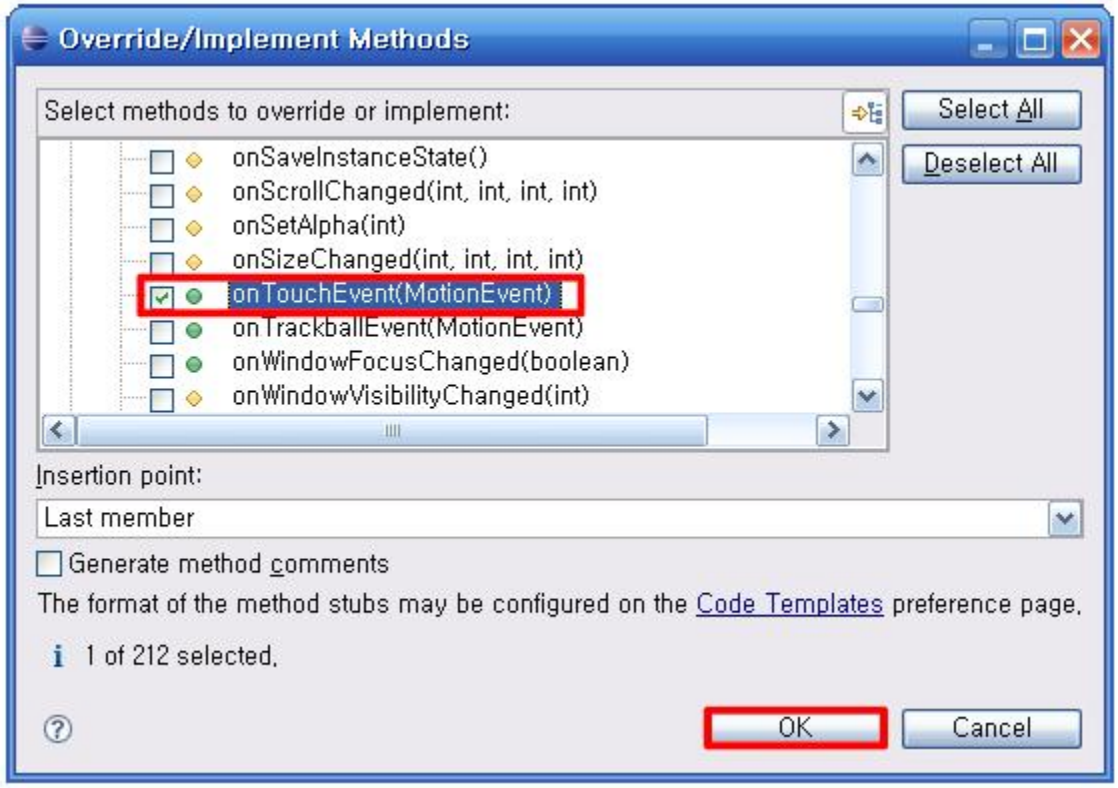

*RobotView.java
main.xml
TestCustom.java
public class RobotView extends View {
Drawable icon;
public RobotView(Context context, AttributeSet attrs) {
super(context, attrs);
// TODO Auto-generated constructor stub
icon = this.getResources().getDrawable(R.drawable.robot);
}
@Override
protected void onDraw(Canvas canvas) {
// TODO Auto-generated method stub
int screenWidth = this.getWidth();
int screenHeight = this.getHeight();
int width = icon.getIntrinsicWidth() / 2;
int height = icon.getIntrinsicHeight() / 2;
int x = (screenWidth - width) / 2;
int y = (screenHeight - height) / 2;
icon.setBounds(x, y, x + width, y + height);
icon.draw(canvas);
super.onDraw(canvas);
}
}

실행 결과


Android Emulator (5554)
3:13 AM
Custom View Test
Hello World, TestCustom
MENU

터치 이벤트 처리

- onTouchEvent(MotionEvent) 추가



터치 이벤트를 위한 코드 수정

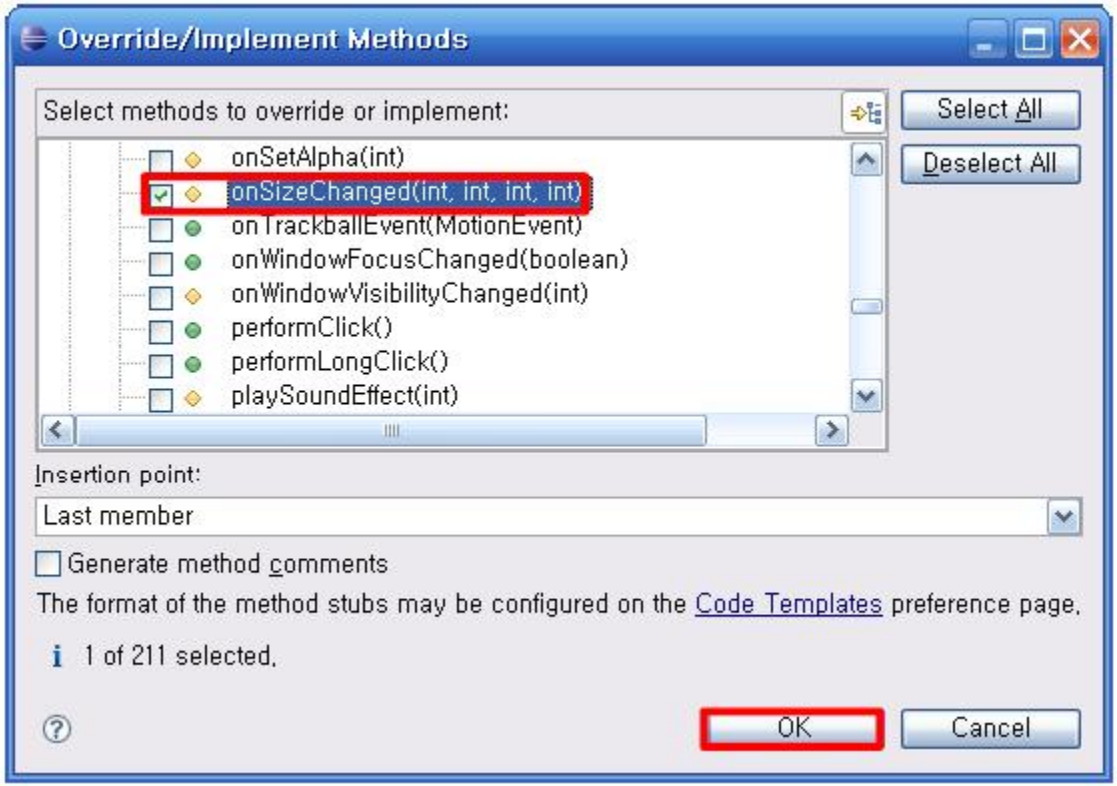
- x, y 변수를 RobotView의 멤버 변수로 뺌
- onTouchEvent() 함수 작성 & 실행 확인

```
public class RobotView extends View {
    Drawable icon;
    int x, y;

    public RobotView(Context context, AttributeSet attrs) {
        super(context, attrs);
        // TODO Auto-generated constructor stub
        icon = this.getResources().getDrawable(R.drawable.robot);
    }
```

```
    @Override
    public boolean onTouchEvent(MotionEvent event) {
        // TODO Auto-generated method stub
        x = (int)event.getX();
        y = (int)event.getY();
        this.invalidate();

        return super.onTouchEvent(event);
    }
```

초기화 시 화면 가운데 이미지 위치

- onSizeChanged(int, int , int ,int) 추가



코드 수정 1

- screenWidth, screenHeight 멤버 변수로.
- width, height 멤버 변수로
- 멤버 함수 수정

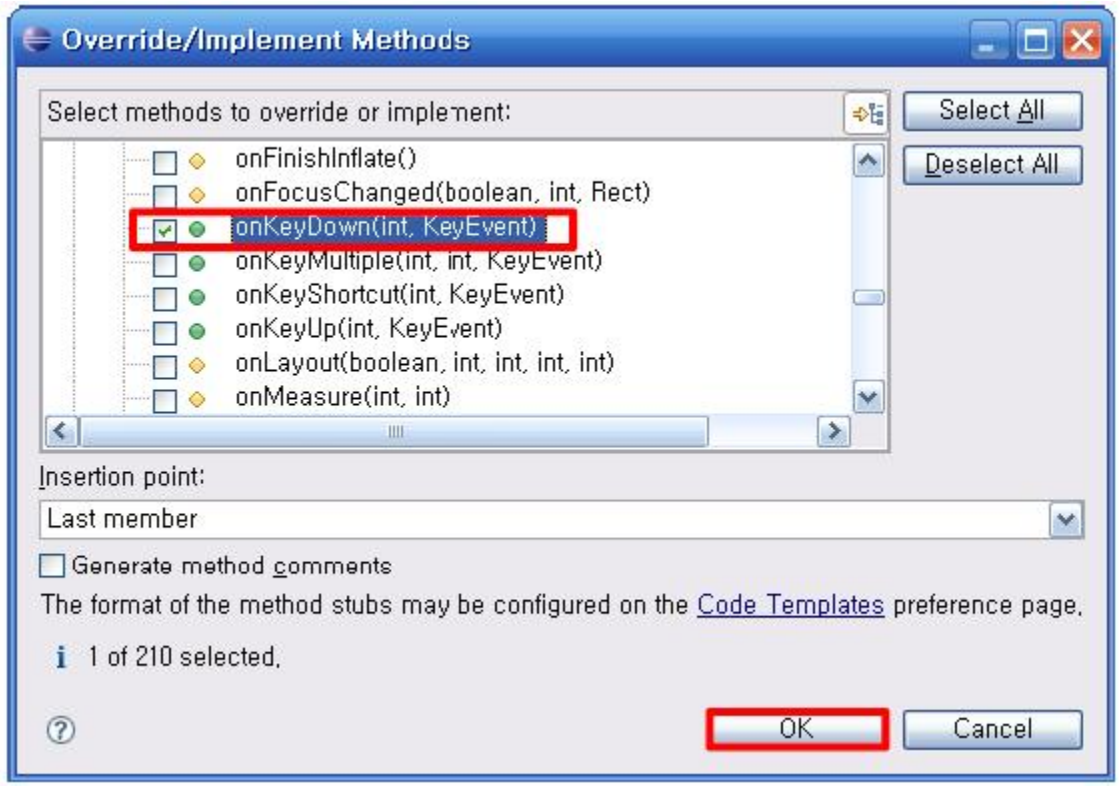
```
public class RobotView extends View {
    Drawable icon;
    int x, y;
    int screenWidth, screenHeight;
    int width, height;

    public RobotView(Context context, AttributeSet attrs) {
        super(context, attrs);
        // TODO Auto-generated constructor stub
        icon = this.getResources().getDrawable(R.drawable.robot);

        width = icon.getIntrinsicWidth() / 2;
        height = icon.getIntrinsicHeight() / 2;
    }

    @Override
    protected void onDraw(Canvas canvas) {
        // TODO Auto-generated method stub
        icon.setBounds(x, y, x + width, y + height);
        icon.draw(canvas);

        super.onDraw(canvas);
    }
```

코드 수정 2

```
@Override
public boolean onTouchEvent(MotionEvent event) {
    // TODO Auto-generated method stub
    x = (int)event.getX();
    y = (int)event.getY();
    this.invalidate();

    return super.onTouchEvent(event);
}

@Override
protected void onSizeChanged(int w, int h, int oldw, int oldh) {
    // TODO Auto-generated method stub
    screenWidth = this.getWidth();
    screenHeight = this.getHeight();


    x = (screenWidth - width) / 2;
    y = (screenHeight - height) / 2;

    super.onSizeChanged(w, h, oldw, oldh);
}
```

키 이벤트 처리하기

- onKeyDown(int, KeyEvent) 추가



소스 코드 추가

```
@Override
public boolean onKeyDown(int keyCode, KeyEvent event) {
    // TODO Auto-generated method stub
    switch (keyCode)
    {
    case KeyEvent.KEYCODE_DPAD_UP:
        y -= 10;
        break;
    case KeyEvent.KEYCODE_DPAD_DOWN:
        y += 10;
        break;
    case KeyEvent.KEYCODE_DPAD_LEFT:
        x -= 10;
        break;
    case KeyEvent.KEYCODE_DPAD_RIGHT:
        x += 10;
        break;
    }
    this.invalidate();

    return super.onKeyDown(keyCode, event);
}
```

Java 코드(Code)에서 뷰(View) 다루기 (http://www.mobileplace.co.kr/2353)

- id

```
1.  <TextView
2.      android:id="@+id/text"
3.      android:layout_width="fill_parent"
4.      android:layout_height="wrap_content"
5.      />
```

- Code

```
1.  TextView t = (TextView)findViewById(R.id.text);
2.   t.setText("Hello");
3.   t.setBackgroundColor(0xFFFF0000);
4.   t.setGravity(Gravity.LEFT);
```

View.onClickListener

```
1. Button button =
   (Button)findViewById(R.id.button);
2.  button.setOnClickListener(new
   View.OnClickListener() {
3.      public void onClick(View v) {
4.         //Code
5.      }
6.   });
```

요약

- 오디오를 재생하는 방법에 대해 알아보았다.
- 비디오를 재생하는 방법에 대해 알아보았다.
- 스도쿠 게임에 음향 효과를 추가해 보았다.
- 2D 그래픽을 심화 학습하기 위해, 커스텀 뷰에 대해 체계적으로 학습하고 이벤트 핸들링과 연결했다.

퀴즈

- 오디오를 재생하려면 어떤 클래스를 사용해야 하는가?
- 비디오를 재생하려면 어떤 클래스를 사용해야 하는가? 다른 방법은 뭐가 있는가?
- 앞의 슬라이드에서 View의 onClickListener()를 구현하는 방법을 보면, setOnClickListener 를 이용하여 직접 코드를 집어넣는 방법을 사용하고 있다. 새로운 클래스를 만들어서 Listener 메서드들만 따로 관리하는 방법은 없을까?
- 동영상 변환 툴(예, Badak)을 이용하여 기존의 동영상을 변환해서 안드로이드에서 실행시켜 보자.

연습문제

- 자신이 가지고 있는 MP3 파일을 재생하는 프로그램을 작성해 보자.
- 여러 MP3 파일 리스트들을 보여주면서, 선택할 경우, 재생시켜 주는 프로그램을 작성해 보자.
- 화면에 특정 이미지를 보여주면서 그 이미지에 알맞은 음악을 재생하다가, 키를 누르면 다른 이미지를 보여주면서 그 이미지에 맞는 음악을 재생해 주는 프로그램을 작성하라.
- 동영상 변환 툴(예, Badak)을 이용하여 기존의 동영상을 변환해서 안드로이드에서 실행시켜 보자.

심화 연습문제 및 프로젝트 아이디어

- 대부분의 응용 프로그램들은 서비스를 이용하여 음악을 재생한다. 서비스를 이용하여 음악을 재생하는 프로그램을 작성해 보라.
- 다음의 스도쿠 변종 게임들을 조사해 보고, 어떤 게임들인지 간단히 설명해 보라.
  - Kakuro
  - Killer sudoku
  - Jigsaw Sudoku puzzle
  - nonomino
  - Mini Sudoku
  - Hypersudoku
  - Gattai 5
  - Greater Than Sudoku