구글 안드로이드 기본 프로그래밍

강대기
동서대학교 컴퓨터정보공학부

학습 목표 (1)

- 선언하여 디자인을 하는 방법을 이해하고, 실행할 수 있다.
- 시작화면을 만드는 방법과 대체 리소스를 사용하는 방법을 이해하고 실행할 수 있다.
- About 과 같은 상자를 구현하고, 테마를 적용하는 법을 이해하고 실행할 수 있다.
- 메뉴를 추가하는 방법을 이해하고 실습할 수 있다.
- 프로그램의 기본 설정(settings)을 정의하는 방법을 알고 실습할 수 있다.
- 대화 상자를 여는 방법을 알고 실습할 수 있다.
- 로그 메시지로 디버깅하는 방법을 이해한다.
- 디버거로 디버깅하는 방법을 이해한다.

학습 목표 (2)

- 2D 그래픽에 대해 배운다.
- Color, Paint, Canvas, Path, Drawable 클래스를 배운다.
- 커스텀 View 클래스를 정의한다.
- 화면 흔들기를 구현해 본다.
- 뷰에 대해 체계적으로 학습하고 레이아웃의 활용 방법을 학습한다.

차례 (1)

- 선언하여 디자인하기
- 시작화면 만들기
- 대체 리소스 사용하기
- About 상자 구현하기
- 테마 적용하기
- 메뉴 추가하기
- Settings 추가하기
- 새 게임 시작하기
- 로그 메시지로 디버깅하기
- 디버거로 디버깅하기
- 프로그램 끝내기
- 요약
- 퀴즈
- 연습문제

차례 (2)

- Color 클래스
- Paint 클래스
- Canvas 클래스
- Path 클래스
- Drawable 클래스
- GUI 보충
  - 뷰와 뷰그룹
  - 레이아웃
  - 액티비티 복습
- 새 액티비티시작하기
- Custom View 클래스 (To Do)
- 화면 흔들기
- 2D GUI 성능 개선하기
- Surface View

선언하여 디자인하기

- 사용자 인터페이스를 디자인하는 방법
  - 절차적 방법 – C나 Java 프로그램 코드로
  - 선언적 방법 – HTML 언어를 통해 표현하여
- 안드로이드의 경우, 둘 다 지원
  - 절차적 방법 – Java 코드
  - 선언적 방법 – XML 표현
- 추천하는 방법은 XML 표현

시작화면 만들기

- Program name – Sudoku
- Package name – org.example.sudoku
- Activity name – Sudoku
- Application name – Sudoku

- 안드로이드 에뮬레이터는 항상 열어놓음
- 게임의 오프닝 화면 구성
- 액티비티 – Sudoku.java
- 리소스 – R.java ← 절대 건들지 말 것
- 레이아웃 – main.xml
  - ADT의 레이아웃 에디터는 별로 좋지 않다

Sudoku.java

```
package org.example.sudoku;

import android.app.Activity;
import android.os.Bundle;

public class Sudoku extends Activity implements OnClickListener {

   /** Called when the activity is first created. */
   @Override
   public void onCreate(Bundle savedInstanceState) {
      super.onCreate(savedInstanceState);
      setContentView(R.layout.main);
  }
}
```

R.java ← 그냥 참고만 하고 절대 건들지 말것

```
/* AUTO-GENERATED FILE.  DO NOT MODIFY.
 *
 * This class was automatically generated by the
 * aapt tool from the resource data it found.  It
 * should not be modified by hand.
 */

package org.example.sudoku;

public final class R {
    public static final class attr {
    }
    public static final class drawable {
        public static final int icon=0x7f020000;
    }
    public static final class layout {
        public static final int main=0x7f030002;
    }
    public static final class string {
        public static final int app_name=0x7f090000;
    }
}
```

레이아웃

- 상위 객체 테두리 안에서 하나 이상의 하위 객체와 이들의 위치를 화면에 설정하는 동작을 포함하는 컨테이너
  - FrameLayout – 모든 하위 객체가 화면의 왼쪽 위에서 시작하도록 정렬 (예: 탭 뷰와 이미지 전환기)
  - LinearLayout – 객체를 한 개의 열 또는 행에 정렬, 가장 흔히 사용됨
  - RelativeLayout – 객체를 서로의 관계를 기준으로 또는 상위 객체와 관계해서 정렬함, 폼에서 자주 사용됨
  - TableLayout – HTML 테이블과 유사하게 하위 객체를 열과 행으로 정렬함
- 컨테이너 – 객체들을 담는 객체

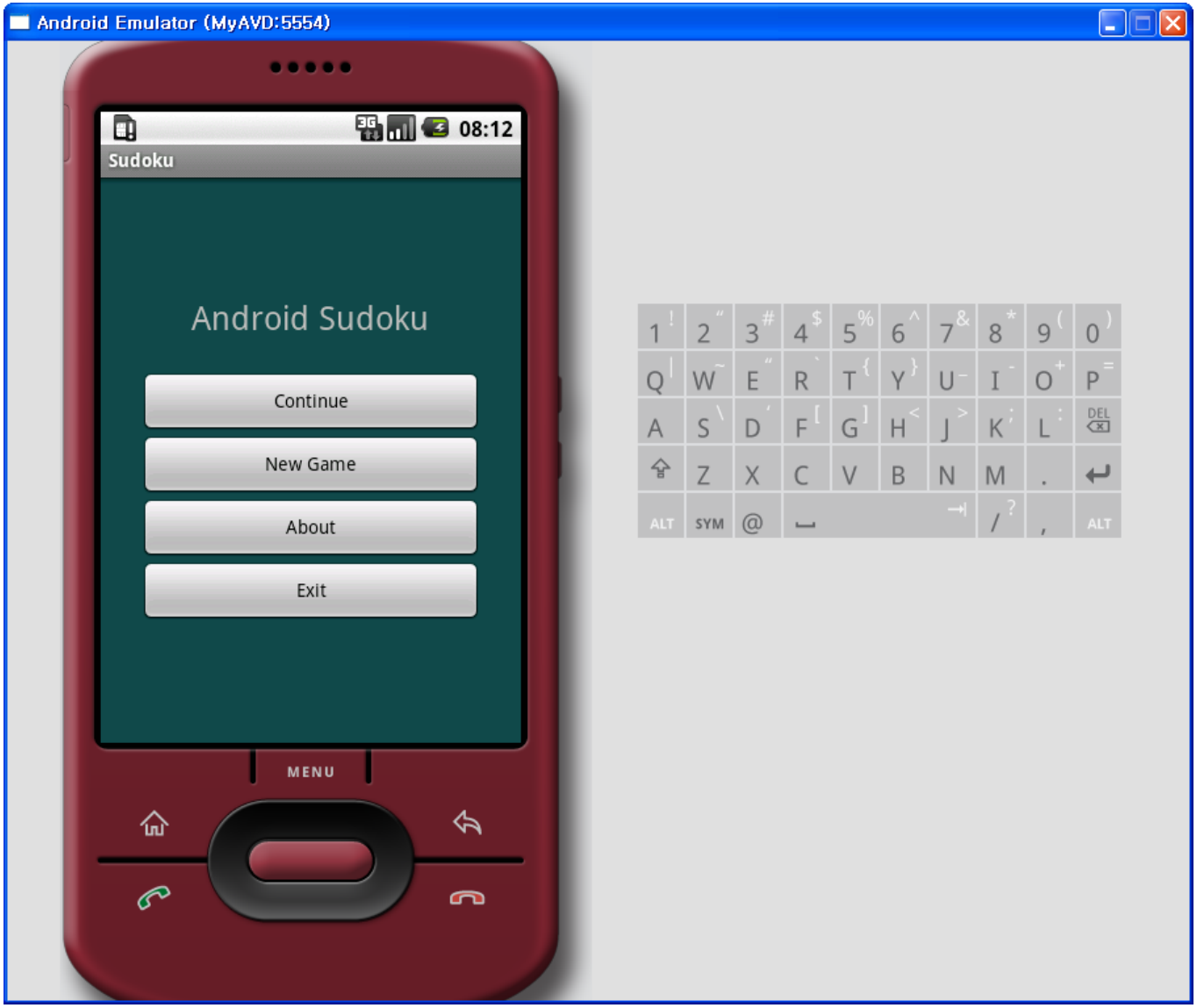
@+id/resid 구문

- 리소스 아이디를 정의하여 이를 통해 참조할 수 있음.
  - android:id="@+id/continue_button "
  - android:id="@+id/new_button"
  - android:id="@+id/about_button"
  - android:id="@+id/exit_button"

main.xml

```
<LinearLayout
  xmlns:android="http://schemas.android.com/apk/res/android"
  android:background="@color/background"
  android:layout_height="fill_parent"    android:layout_width="fill_parent"
  android:padding="30dip"    android:orientation="horizontal">
  <LinearLayout      android:orientation="vertical"
    android:layout_height="wrap_content"      android:layout_width="fill_parent"
    android:layout_gravity="center">
    <TextView        android:text="@string/main_title"
      android:layout_height="wrap_content"       android:layout_width="wrap_content"
      android:layout_gravity="center"       android:layout_marginBottom="25dip"
      android:textSize="24.5sp" />
    <Button        android:id="@+id/continue_button"
      android:layout_width="fill_parent"       android:layout_height="wrap_content"
      android:text="@string/continue_label" />
    <Button        android:id="@+id/new_button"
      android:layout_width="fill_parent"       android:layout_height="wrap_content"
      android:text="@string/new_game_label" />
    <Button        android:id="@+id/about_button"
      android:layout_width="fill_parent"       android:layout_height="wrap_content"
      android:text="@string/about_label" />
    <Button        android:id="@+id/exit_button"
      android:layout_width="fill_parent"       android:layout_height="wrap_content"
      android:text="@string/exit_label" />
  </LinearLayout>
</LinearLayout>
```

시작 화면


Android Emulator (MyAVD:5554)
08:12
Sudoku
Android Sudoku
Continue
New Game
About
Exit
MENU

자작 시작 화면 (main.xml)

```
<LinearLayout
  xmlns:android="http://schemas.android.com/apk/res/android"
  android:layout_height="fill_parent"
  android:layout_width="fill_parent"
  android:padding="30dip"
  android:orientation="horizontal">
  <LinearLayout      android:orientation="vertical"
    android:layout_height="wrap_content"      android:layout_width="fill_parent"
    android:layout_gravity="center">
    <TextView        android:text="헬로"
      android:layout_height="wrap_content"       android:layout_width="wrap_content"
      android:layout_gravity="center"       android:layout_marginBottom="25dip"
      android:textSize="24.5sp" />
    <Button        android:id="@+id/continue_button"
      android:layout_width="fill_parent"       android:layout_height="wrap_content"
      android:text="다시 시작" />
    <Button        android:id="@+id/new_button"
      android:layout_width="fill_parent"       android:layout_height="wrap_content"
      android:text="새 게임" />
    <Button        android:id="@+id/about_button"
      android:layout_width="fill_parent"       android:layout_height="wrap_content"
      android:text="...에 관하여" />
    <Button        android:id="@+id/exit_button"
      android:layout_width="fill_parent"       android:layout_height="wrap_content"
      android:text="끝" />
  </LinearLayout>
</LinearLayout>
```

자작 시작 화면 (화면 갈무리)

대체 리소스 사용하기

- 예를 들어, 가로 방향을 위한 레이아웃 지정
  - res/layout-land/main.xml
- 이외에 모든 리소스의 대체 버전을 명시하는 데에도 디렉터리 이름에 리소스 접미사를 사용함
  - 언어, 지역, 화소 밀도, 해상도, 입력 방법 등

dip 와 sp

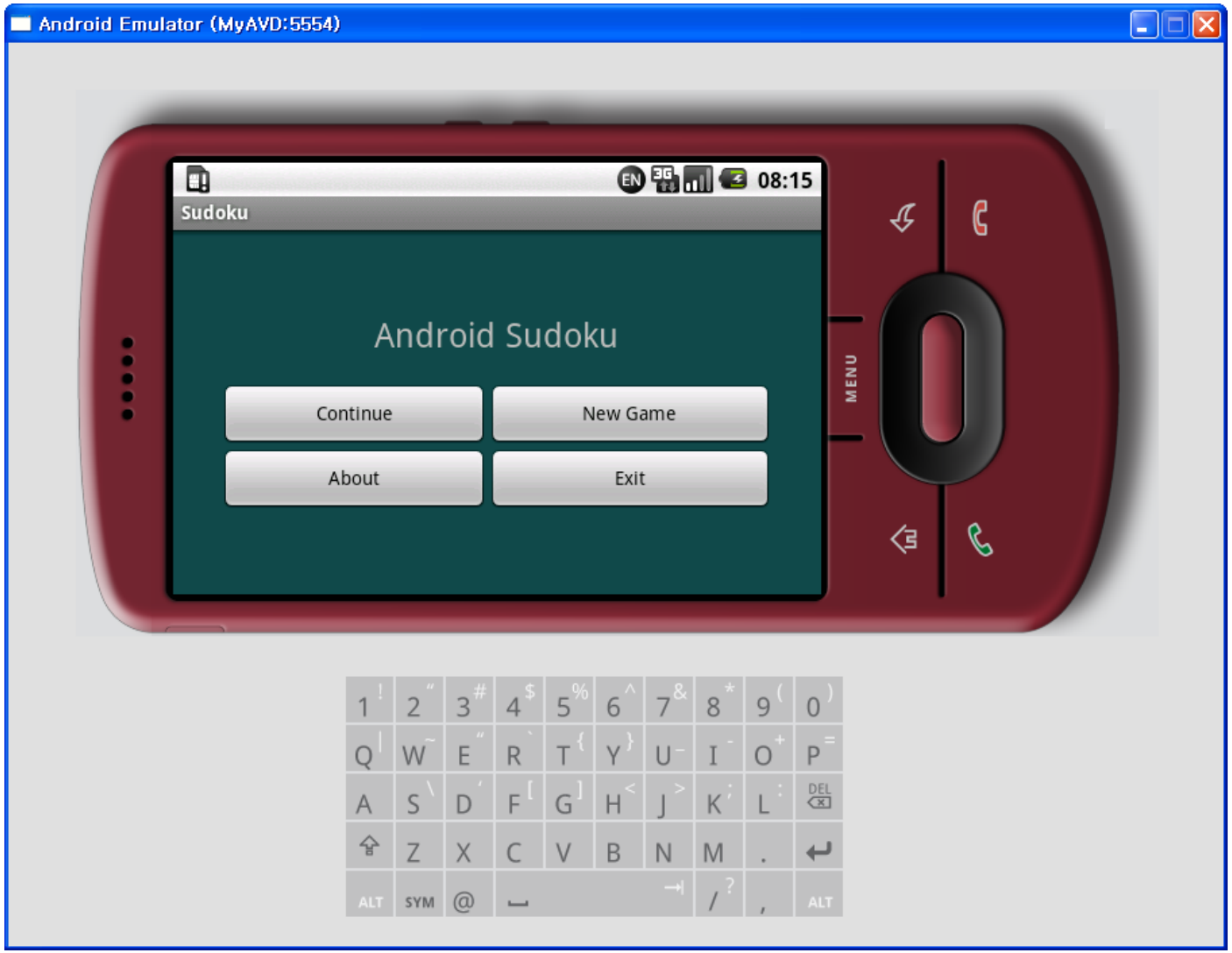
- 단순히 픽셀 단위로 거리를 지정하면, 640x480 화면과 1920x1200 화면에서 다르게 보일 것

- 안드로이드의 단위 지정
  - px – pixel 픽셀
  - in – inch 인치
  - mm – millimeter 밀리미터
  - pt – point (인치의 1/72)
  - dp (density independent pixel) – 밀도에 독립적인 화소, 1 인치 당 160개의 점이 있는 디스플레이에서 1 dp 는 1 px 과 같음
  - dip – dip
  - sp (scale independent pixel) – 스케일에 독립적인 화소, dp와 유사하나 사용자의 글꼴 크기 설정에 의해 측정됨

res/layout-land/main.xml

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout
  xmlns:android="http://schemas.android.com/apk/res/android"    android:background="@color/background"
  android:layout_height="fill_parent"    android:layout_width="fill_parent"    android:padding="15dip"
  android:orientation="horizontal">
  <LinearLayout
    android:orientation="vertical"      android:layout_height="wrap_content"      android:layout_width="fill_parent"
    android:layout_gravity="center"      android:paddingLeft="20dip"      android:paddingRight="20dip">
    <TextView         android:text="@string/main_title"
     android:layout_height="wrap_content“         android:layout_width="wrap_content"
      android:layout_gravity="center"         android:layout_marginBottom="20dip"         android:textSize="24.5sp" />
    <TableLayout         android:layout_height="wrap_content"         android:layout_width="wrap_content"
     android:layout_gravity="center"
      android:stretchColumns="*">
      <TableRow>
        <Button             android:id="@+id/continue_button"             android:text="@string/continue_label" />
        <Button             android:id="@+id/new_button"             android:text="@string/new_game_label" />
      </TableRow>
      <TableRow>
        <Button             android:id="@+id/about_button"             android:text="@string/about_label" />
        <Button             android:id="@+id/exit_button"             android:text="@string/exit_label" />
      </TableRow>
    </TableLayout>
  </LinearLayout>
</LinearLayout>
```

가로 방향 레이아웃


Android Emulator (MyAVD:5554)
08:15
Sudoku
Android Sudoku
Continue
New Game
About
Exit
MENU

About 상자 구현하기

- 스도쿠에 대한 정보가 있는 창이 나옴
- 구현 방법
  - 새 액티비티를 정의하고 시작시킴
  - AlertDialog 클래스를 사용해 보여줌
  - 하위 클래스인 Dialog 클래스를 새 뷰로 팽창시켜 보여줌
- 새 액티비티를 정의함 – About 액티비티
- 이를 위해 새 레이아웃 파일을 정의 – res/layout/about.xml

res/layout/about.xml

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

   xmlns:android="http://schemas.android.com/apk/res/andro
   id"
  android:layout_width="fill_parent"
  android:layout_height="fill_parent"
  android:padding="10dip">
  <TextView
    android:id="@+id/about_content"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="@string/about_text" />
</ScrollView>
```

res/values/strings.xml

```
<string name="about_title">About Android
   Sudoku</string>
   <string name="about_text">\
Sudoku is a logic-based number placement puzzle.
Starting with a partially completed 9x9 grid, the
objective is to fill the grid so that each
row, each column, and each of the 3x3 boxes
(also called <i>blocks</i>) contains the digits
1 to 9 exactly once.
</string>
```

About.java

```
package org.example.sudoku;

import android.app.Activity;
import android.os.Bundle;

public class About extends Activity {
   @Override
   protected void onCreate(Bundle savedInstanceState)
   {
      super.onCreate(savedInstanceState);
      setContentView(R.layout.about);
   }
}
```

Sudoku.java

```
import android.content.Intent;
import android.view.View;
import android.view.View.OnClickListener;

...

  /** Called when the activity is first created. */
  @Override
  public void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.main);

    // Set up click listeners for all the buttons
    View continueButton = findViewById(R.id.continue_button);
    continueButton.setOnClickListener(this);
    View newButton = findViewById(R.id.new_button);
    newButton.setOnClickListener(this);
    View aboutButton = findViewById(R.id.about_button);
    aboutButton.setOnClickListener(this);
    View exitButton = findViewById(R.id.exit_button);
    exitButton.setOnClickListener(this);
  }
```

Sudoku.java

```
public class Sudoku extends Activity implements OnClickListener {

...

  public void onClick(View v) {
    switch (v.getId()) {
    case R.id.continue_button:
      startGame(Game.DIFFICULTY_CONTINUE);
      break;
      // ...

    case R.id.about_button:
      Intent i = new Intent(this, About.class);
      startActivity(i);
      break;
    // More buttons go here (if any) ...
    case R.id.new_button:
      openNewGameDialog();
      break;
    case R.id.exit_button:
      finish();
      break;

    }
  }
```

AndroidManifest.xml에 다음을 추가

```
<activity android:name=".About"
        android:label="@string/about_title" >
    </activity>
```

테마 적용하기

```
<activity android:name=".About"
        android:label="@string/about_title"

  android:theme="@android:style/Theme.Dialog"
  >
     </activity>
```

메뉴 추가하기

- 안드로이드가 지원하는 두 가지 메뉴
  - 메뉴 버튼을 눌렀을 때 나오는 메뉴
  - 화면을 손가락으로 누를 때 나오는 상황적 (contextual) 메뉴
- 메뉴 버튼을 눌렀을 때 나오는 메뉴를 만들려면
  1. 사용할 문자열 정의
  2. XML을 사용하여 메뉴 정의
  3. import 추가
  4. 메서드 오버라이드

사용할 문자열 정의 /res/values/strings.xml

```
<string
  name="settings_label">Settings...</string>
  <string name="settings_title">Sudoku
  settings</string>
  <string name="settings_shortcut">s</string>
  <string name="music_title">Music</string>
  <string name="music_summary">Play
  background music</string>
  <string name="hints_title">Hints</string>
  <string name="hints_summary">Show hints
  during play</string>
```

XML을 사용하여 메뉴 정의 /res/menu/menu.xml

```
<?xml version="1.0" encoding="utf-8"?>
<menu
  xmlns:android="http://schemas.android.com/apk/res/android">
  <item android:id="@+id/settings"
    android:title="@string/settings_label"

  android:alphabeticShortcut="@string/settings_shortcut" />
</menu>
```

import 추가
Sudoku.java

```
import android.view.Menu;
import android.view.MenuInflater;
import android.view.MenuItem;
```

메서드 오버라이드 (onCreateOptionsMenu in Sudoku.java)

```
@Override
  public boolean onCreateOptionsMenu(Menu
  menu) {
    super.onCreateOptionsMenu(menu);
    MenuInflater inflater = getMenuInflater();
    inflater.inflate(R.menu.menu, menu);
    return true;
  }
```

메서드 오버라이드 (onOptionsItemSelected in Sudoku.java)

```
@Override
  public boolean onOptionsItemSelected(MenuItem
  item) {
    switch (item.getItemId()) {
    case R.id.settings:
      startActivity(new Intent(this, Prefs.class));
      return true;
    // More items go here (if any) ...
    }
    return false;
  }
```

Settings 추가하기

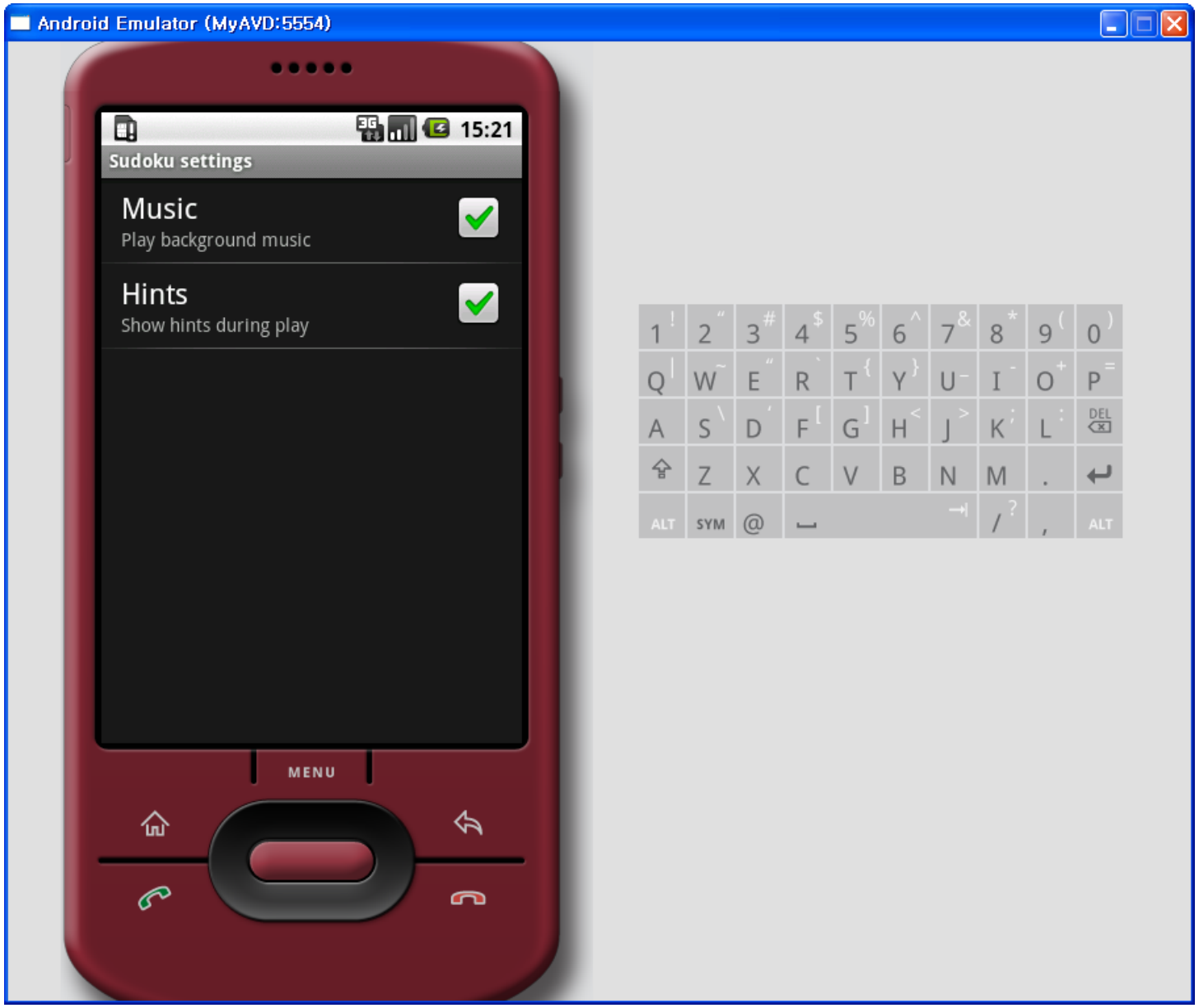
- 기본 설정 정의
  - 예: res/xml/settings.xml
- Prefs 클래스 정의하여 PreferenceActivity 확장
  - /src/org/example/sudoku/Prefs.java
- Prefs 액티비티를 AndroidManifest.xml에 등록
  - <activity android:name=".Prefs"
  - android:label="@string/settings_title">
  - </activity>

res/xml/settings.xml

```
<?xml version="1.0" encoding="utf-8"?>
<PreferenceScreen
  xmlns:android="http://schemas.android.com/apk/res/android">
  <CheckBoxPreference
    android:key="music"
    android:title="@string/music_title"
    android:summary="@string/music_summary"
    android:defaultValue="true" />
  <CheckBoxPreference
    android:key="hints"
    android:title="@string/hints_title"
    android:summary="@string/hints_summary"
    android:defaultValue="true" />
</PreferenceScreen>
```

/src/org/example/sudoku/Prefs.java

```
package org.example.sudoku;

import android.os.Bundle;
import android.preference.PreferenceActivity;

public class Prefs extends PreferenceActivity {
  @Override
  protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    addPreferencesFromResource(R.xml.settings);
  }
}
```


Android Emulator (MyAVD:5554)
15:21
Sudoku settings
Music
Play background music
Hints
Show hints during play
MENU

새 게임 시작하기

- 문자열 추가
- 배열 리소스 만듦
- import 추가
- “News Game” 버튼 클릭 처리
- 난이도 목록 사용자 인터페이스 만듦

/res/values/strings.xml

```
<string name="new_game_title">Difficulty</string>
<string name="easy_label">Easy</string>
<string name="medium_label">Medium</string>
<string name="hard_label">Hard</string>
```

/res/values/arrays.xml

```
<resources>
  <array name="difficulty">
    <item>@string/easy_label</item>
    <item>@string/medium_label</item>
    <item>@string/hard_label</item>
  </array>
</resources>
```

/src/org/example/sudoku/Sudoku.java import 추가

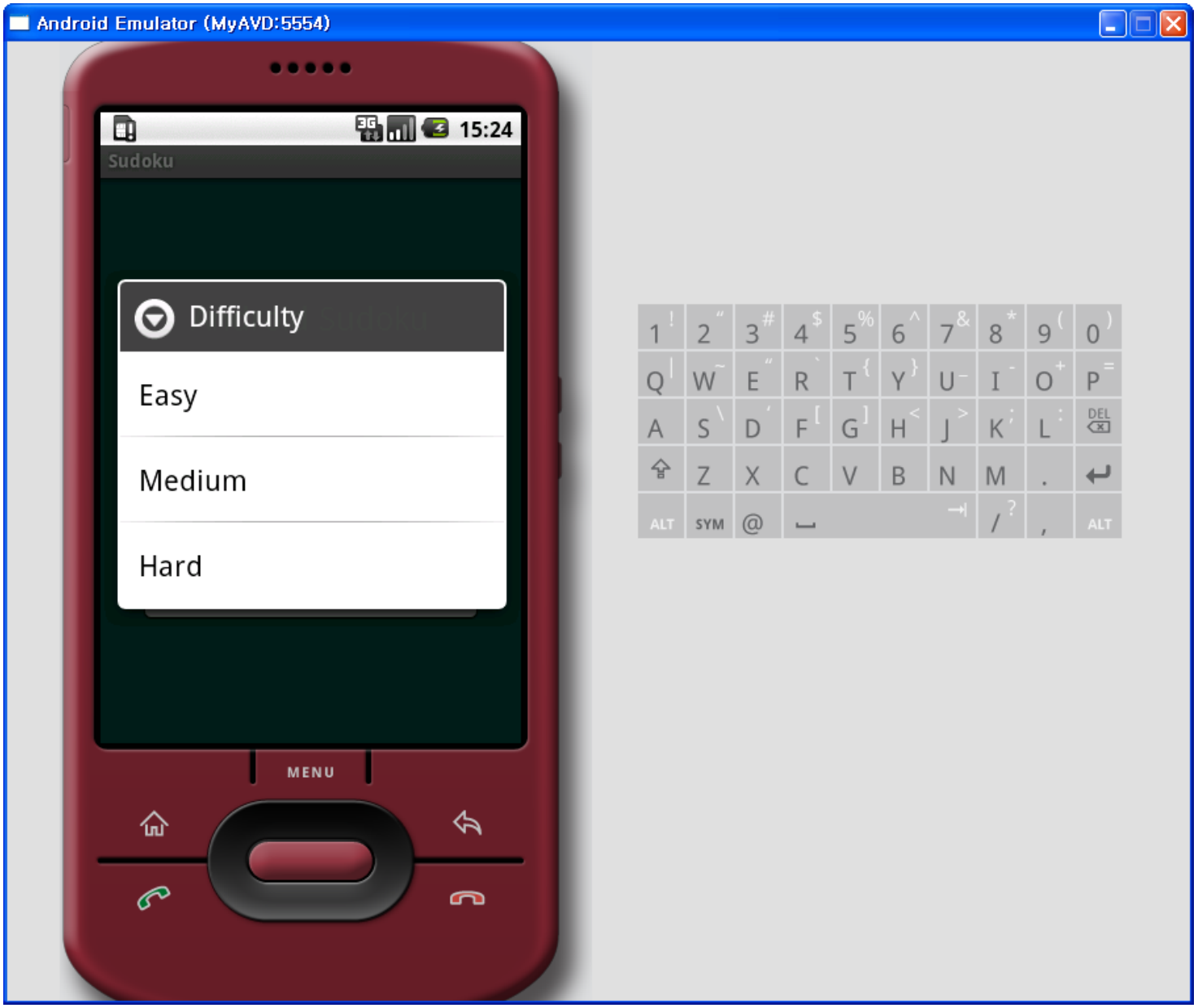
```
import android.app.AlertDialog;
import android.content.DialogInterface;
import android.util.Log;
```

/src/org/example/sudoku/Sudoku.java “News Game” 버튼 클릭 처리

```
public void onClick(View v) {
    switch (v.getId()) {
    case R.id.continue_button:
      startGame(Game.DIFFICULTY_CONTINUE);
      break;
      // ...

    case R.id.about_button:
      Intent i = new Intent(this, About.class);
      startActivity(i);
      break;
    // More buttons go here (if any) ...
    case R.id.new_button:
      openNewGameDialog();
      break;
    case R.id.exit_button:
      finish();
      break;

    }
}
```

/src/org/example/sudoku/Sudoku.java 난이도 목록 사용자 인터페이스 만듦

```
/** Ask the user what difficulty level they want */
  private void openNewGameDialog() {
    new AlertDialog.Builder(this)
        .setTitle(R.string.new_game_title)
        .setItems(R.array.difficulty,
         new DialogInterface.OnClickListener() {
           public void onClick(DialogInterface dialoginterface,
               int i) {
             startGame(i);
           }
         })
        .show();
  }

  /** Start a new game with the given difficulty level */
  private void startGame(int i) {
    Log.d(TAG, "clicked on " + i);
    Intent intent = new Intent(Sudoku.this, Game.class);
    intent.putExtra(Game.KEY_DIFFICULTY, i);
    startActivity(intent);
  }
```


Android Emulator (MyAVD:5554)
15:24
Sudoku
Difficulty
Easy
Medium
Hard
MENU

로그 메시지로 디버깅하기

- Log.e() – 오류 (error)
- Log.w() – 경고 (warning)
- Log.i() – 정보 (information)
- Log.d() – 디버깅 (debugging)
- Log.v() – 세부 정보 (verbose)

- Window > Show View > Other > Android > LogCat

디버거로 디버깅하기

- 프로젝트 디버깅 활성화
- AndroidManifest.xml 에서 android:debuggable="true"
  - <application android:icon="*@drawable/icon*" android:label="*@string/app_name*" android:debuggable="*true*">
- Right click > Debug As > Android Application

프로그램 끝내기

- 대부분의 스마트폰 응용 프로그램에서 끝내기 버튼은 필요치 않음 – 안드로이드의 경우, 이전 버튼이나 홈 키를 누르면 됨
- Activity 를 종료하는 방법

```
public void onClick(View v) {
    switch (v.getId()) {
    case R.id.exit_button:
      finish();
      break;

    }
  }
```

Color 클래스

- 색 – 알파(alpha), 레드(red), 그린(green), 블루(blue) – 32 비트 정수
- 알파 – 투명도
  - int color = Color.BLUE; // 파란 색
  - color = Color.argb(127,255,0,255); // 반투명 보라색
- XML 리소스에서 색상 정의
  - <resources>
  - <color name="mycolor">#3500ffff</color>
  - </resources>
- 자바 코드 내에서 사용
  - color = getResources().getColor(R.color.mycolor);

Paint 클래스

- 비트맵, 텍스트, 기하학적 모양의 그래픽을 그리는 데 필요한 스타일과 색상 정보를 가짐
- 화면에 칠을 할 때 필요한 색상 설정
  - cPaint.setColor(Color.LTGRAY);

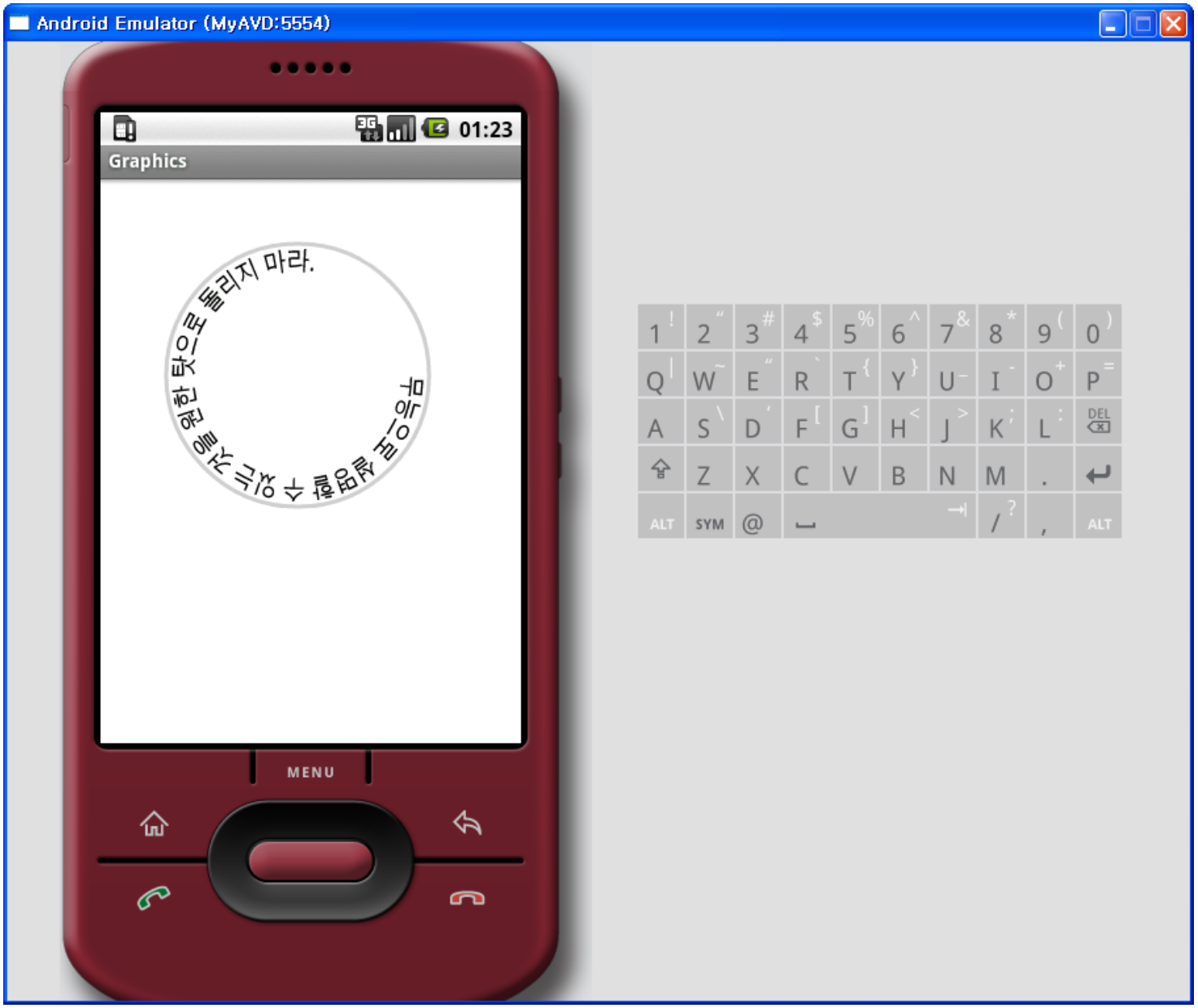
Canvas 클래스

- 그림이 그려지는 표면
- Activity > View > Canvas
- 캔버스에 그리려면, View.onDraw() 를 오버라이드

```
public class Graphics extends Activity {
  @Override
  public void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.main);
  }
}

static public class GraphicsView extends View {
  public GraphicsView(Context context) {
    super(context);
  }

  @Override
  protected void onDraw(Canvas canvas) {
    // 사용자의 그리기 코드가 여기에 추가됨
  }
}
```

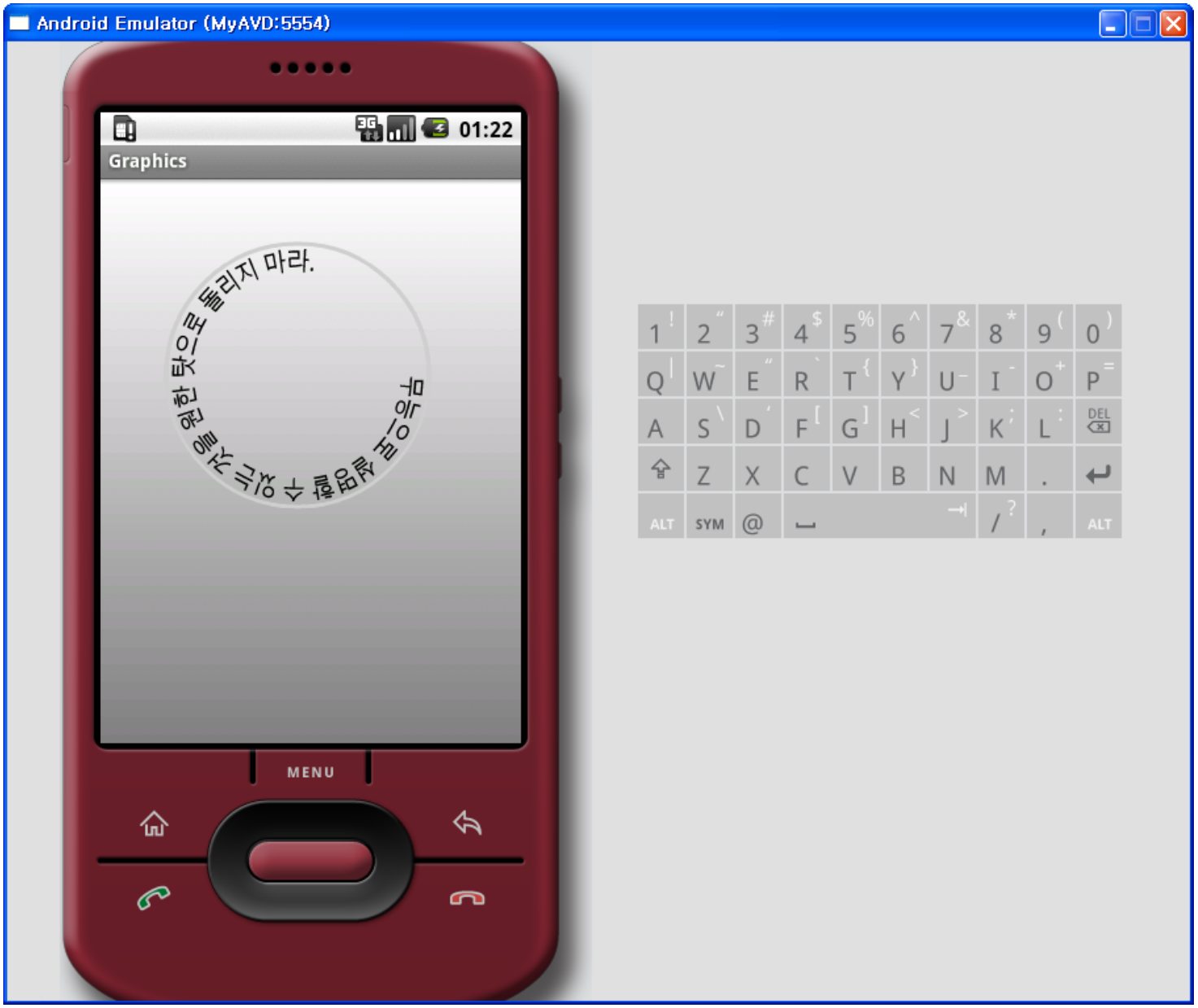
Path 클래스

- 선, 사각형, 커브 등 벡터 그리기 명령어를 포함
- 원형 경로 정의 – CW는 시계 방향
  - circle = new Path();
  - circle.addCircle(150,150,100,Direction.CW);
- 텍스트 추가
  - private static final String QUOTE = “무능으로 설명할 수 있는 것을 원한 탓으로 돌리지 마라.”;
  - canvas.drawPath(circle, cPaint);
  - canvas.drawTextOnPath(QUOTE, circle, 0, 20, cPaint);


Android Emulator (MyAVD:5554)
01:23
Graphics
뉴턴은 절멈할 수 있는 것에 원한 탓으로 돌리지 마라.
MENU

Drawable 클래스

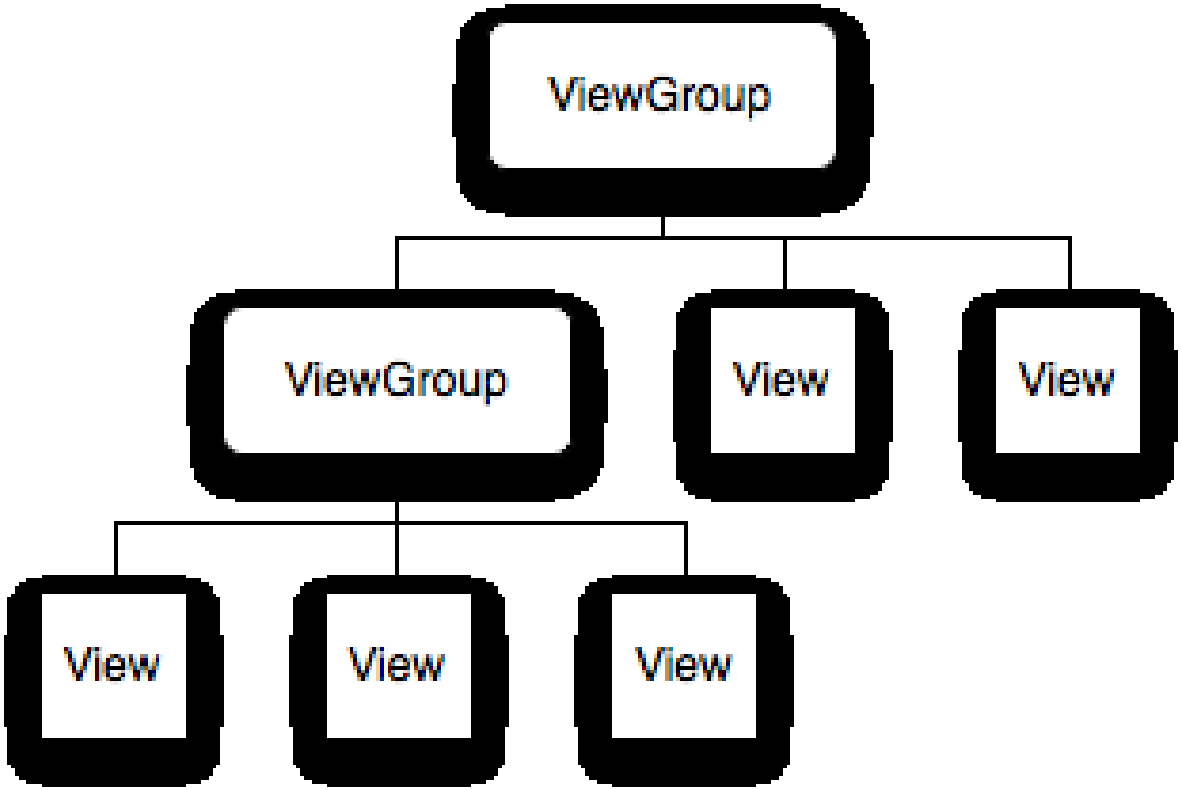
- 비트맵, 단색 등과 같이 화면 표시용 시각적 요소에 사용됨
- 다른 그래픽에 결합시킬 수 있으며, 사용자 인터페이스 위젯(버튼이나 뷰의 배경)에 사용됨
- Drawable 의 형태
  - 비트맵(Bitmap) – PNG 또는 JPG 이미지
  - 나인패치(NinePatch) – 늘어나는 PNG 이미지
  - 모양(Shape) – 경로에 기초한 벡터 그리기 명령어, Scalable Vector Graphics (SVG) 의 축소판
  - 레이어 (Layer) – Z-order 순서에 따라 서로 덮어 그리는 하위 drawable을 포함하는 컨테이너
  - 상태 (State) – 상태(비트 마스크; bit mask)에 맞는 하나의 하위 drawable을 보여주는 컨테이너
  - 레벨 (Level) – 컨테이너
  - 스케일 (Scale) – 컨테이너


Android Emulator (MyAVD:5554)
01:22
Graphics
음악을 틀면 학수 있는 곳에 원한 탓으로 돌리지 마라.
MENU

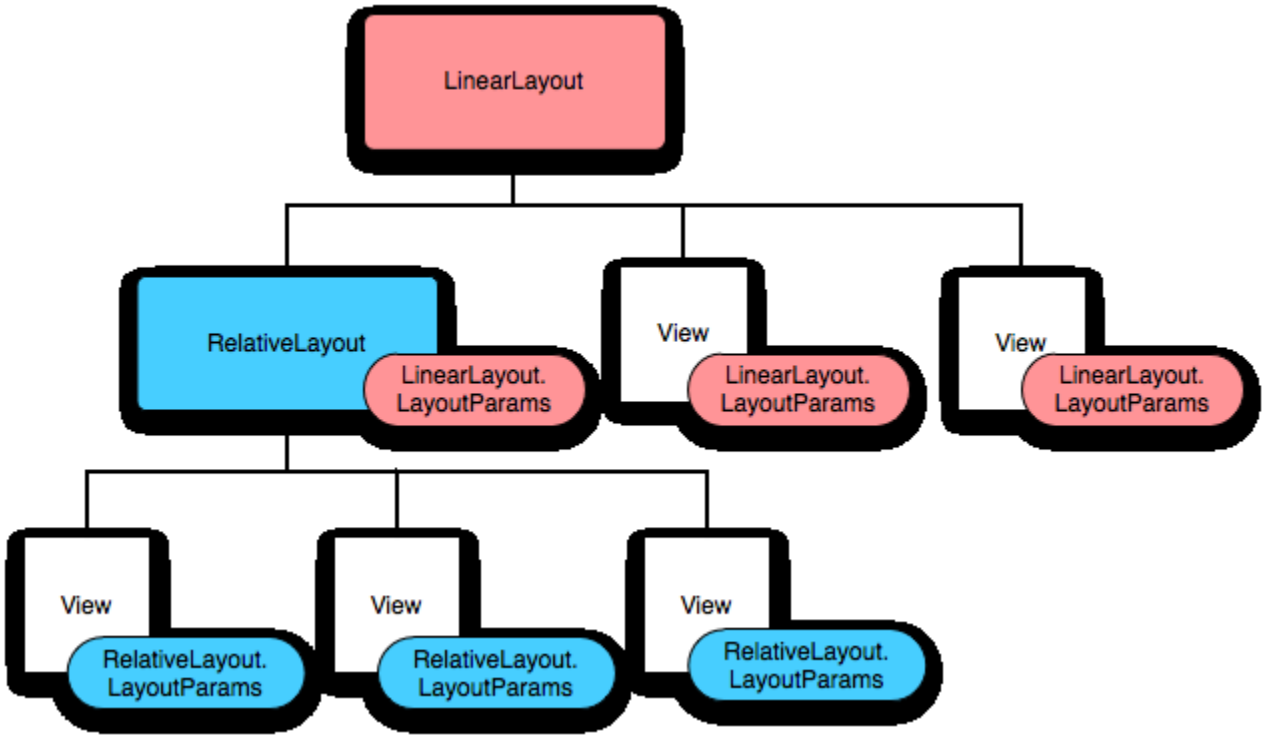
사용자 인터페이스 구현 Implementing a UI

- View
  - 뷰는 android.view.View 를 기본 클래스로 가지는 객체
  - 화면의 특정한 사각형 영역에 대한 레이아웃과 속성들을 저장하고 있는 자료 구조
- Viewgroups
  - 뷰그룹(viewgroup)은 android.view.Viewgroup 클래스의 객체
- 뷰그룹과 뷰를 저장하는 컨테이너
  - 뷰그룹은 특별한 타입의 뷰 객체로서, 그 기능은 그 뷰그룹에 종속적인 뷰들이나 다른 뷰그룹들의 집합을 저장하고 관리하는 것

트리 구조의 사용자 인터페이스
A Tree-Structured UI



레이아웃 패러미터
LayoutParams: How a Child Specifies Its Position and Size

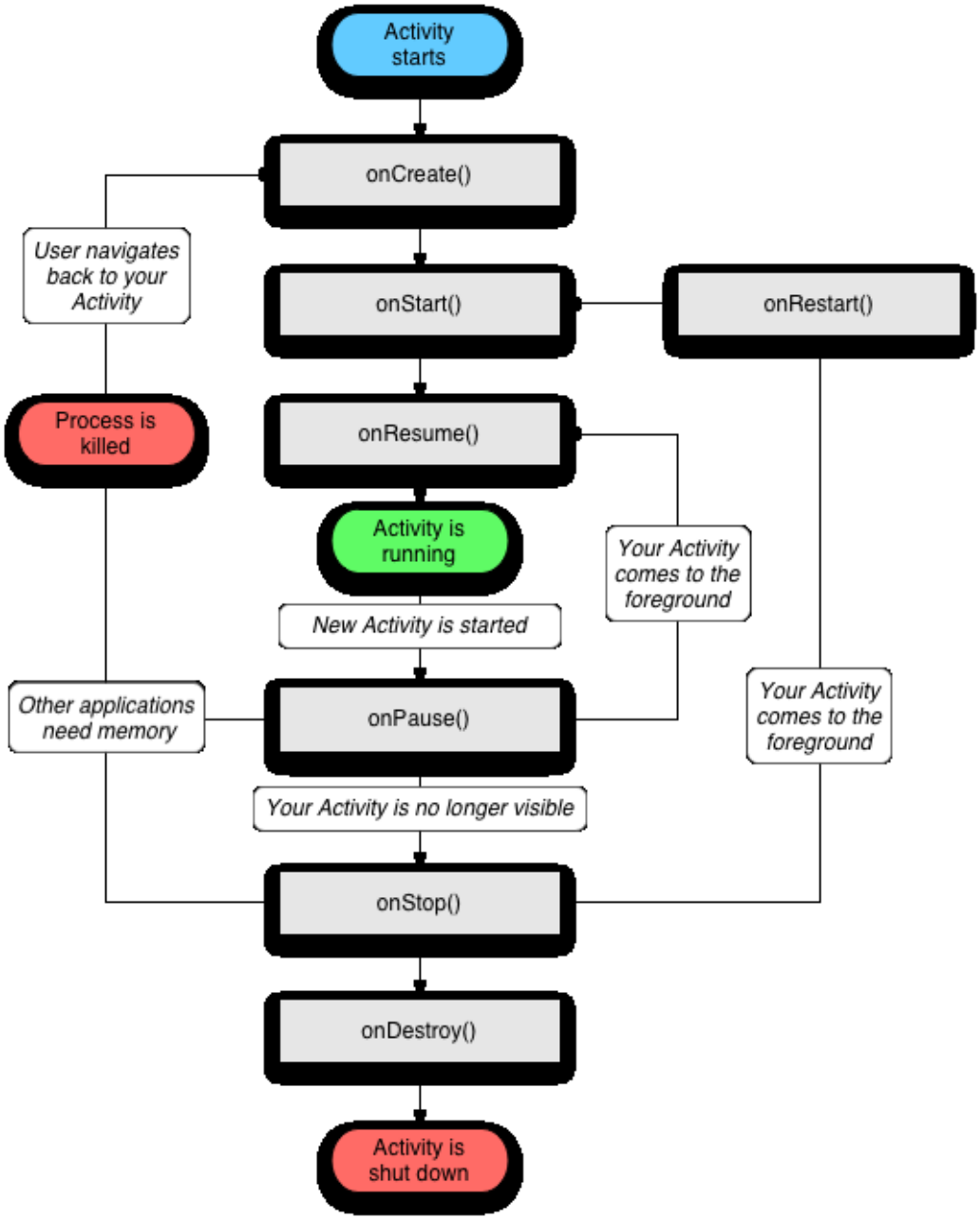


Building Blocks

- AndroidManifest.xml (안드로이드 응용 프로그램 제어 파일)
  - The control file that tells the system what to do with all the top-level components you've created – 탑-레벨 요소들을 어떻게 사용하고 연결할 것인지를 지정하는 제어 파일
  - http://developer.android.com/guide/topics/manifest/manifest-intro.html
- Activities (액티비티)
  - An Activity is, fundamentally, an object that has a life cycle – 수명 주기를 가지고 있는 객체로 하나의 화면과 대응됨
- Views (뷰)
  - A View is an object that knows how to draw itself to the screen. – 화면에 쓰여지는 그래픽 객체로, 윈도우 프로그래밍에서 콘트롤과 동일함
- Intents (인텐트 – 의도라는 뜻)
  - An Intent is a simple message object that represents an “intention” to do something. – 액티비티들 간에 주고받는 메시지 객체
- Services (서비스)
  - A Service is a body of code that runs in the background – 유닉스의 데몬같이 운영체제의 배경에서 오랫동안 작동되는 프로그램
- Notifications (통지)
  - A Notification is a small icon that appears in the status bar – 상태 바에 나타나는 조그만 아이콘 (스마트폰 운영체제들은 대부분 필수적으로 가지고 있음)
  - Users can interact with this icon to receive information.
- ContentProviders (컨텐트 프로바이더)
  - A ContentProvider is a data storehouse that provides access to data on the device – 디바이스의 데이터에 접근하기 위해 사용되는 요소

Activities (액티

- http://developer.android.com/reference/android/app/Activity.html



새 프로젝트
New Project

- Menu → File → New → Project
- Android 선택
- Project Name – Hello Android
- Package Name – dsu.android
- Activity Name – Main
- Application Name – HelloAndroid

프로젝트 실행
Run the Project

- Menu → Run → Run Configuration
- Right mouse click → New
- Name → AndroidConfiguration1
- Browse HelloAndroid Project
- Apply and Run

새 액티비티
New Activity

- Add New Class
- MyActivity
- Subclass of Activity → Browse
- android.app.Activity

레이아웃 XML
Layout XML

- /res/layout folder
- New File
- 새로운 파일을 myactivity.xml (소문자) 라고 정함
- main.xml 을 myactivity.xml 로 복사함
- 다음과 같이 바꿈
  - android:text="*@string/hello*“
  - to whatever you like!

AndroidManifest 변경 (Update AndroidManifest)

- 새로운 액티비티가 나올 때마다 AndroidManifest 변경

- AndroidManifest.xml 선택
- Activity tag 을 하나 더 복사
- 다음과 같이 변경
  - android:name, android:label (타이틀바!)

메쏘드 오버라이드
Override Methods

- MyActivity.java
- 마우스 오른쪽 클릭 (Right-click)
- Source 선택
- Override/Implement Methods
- onCreate(Bundle) 선택
  - setContentView(R.layout.myactivity);
  - import android.util.Log;
  - Log.d("MyTag", "Print Test Log");

실행 컨피규레이션
Run Configuration

- Run Menu
- Run Configuration
- Launch smartphone.android.MyActivity
- Apply
- Run

- Click Menu on the emulator

액티비티(Activity), 뷰(View), 레이아웃(Layout) 실습

- 참고 - http://www.mobileplace.co.kr/1050

레이아웃 XML
Layout XML

```
1.  <?xml version="1.0" encoding="utf-8"?>
2.  <LinearLayout
    xmlns:android="http://schemas.android.com/apk/res/and
    roid"
3.      android:orientation="vertical"
4.      android:layout_width="fill_parent"
5.      android:layout_height="fill_parent"
6.      >
7.  <TextView
8.      android:layout_width="fill_parent"
9.      android:layout_height="wrap_content"
10.     android:text= "Hello, Oman"
11.     />
12. </LinearLayout>
```

뷰와 레이아웃
Views and Layouts

- View : TextView, Button, ImageView, ListView, EditText, etc.
- Layout : LinearLayout, RelativeLayout, FrameLayout, AbsoluteLayout, etc.

- View Attribute
  - layout_width, layout_height, background, visibility, id
  - http://code.google.com/intl/ko-KR/android/reference/android/view/View.html

뷰와 레이아웃
Views and Layouts

```
1.  <?xml version="1.0" encoding="utf-8"?>
2.  <LinearLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
3.      android:orientation="vertical"
4.      android:layout_width="fill_parent"
5.      android:layout_height="fill_parent"
6.      android:background="#FF888888"
7.      >
8.  <TextView
9.      android:layout_width="fill_parent"
10.     android:layout_height="wrap_content"
11.     android:text="파랑"
12.     android:background="#FF0000FF"
13.     />
14. </LinearLayout>
```

뷰와 레이아웃
Views and Layouts

```
1.  <?xml version="1.0" encoding="utf-8"?>
2.  <LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
3.     android:orientation="vertical"
4.     android:layout_width="fill_parent"
5.     android:layout_height="fill_parent"
6.     >
7.  <TextView
8.     android:layout_width="fill_parent"
9.     android:layout_height="wrap_content"
10.    android:text="Red"
11.    android:background="#FFFF0000"
12.    />
13. <TextView
14.    android:layout_width="wrap_content"
15.    android:layout_height="50dp"
16.    android:text="Green"
17.    android:background="#FF00FF00"
18.    />
19. <TextView
20.    android:layout_width="fill_parent"
21.    android:layout_height="wrap_content"
22.    android:text="Blue"
23.    android:background="#FF0000FF"
24.    />
25. </LinearLayout>
```

XML에서의 TextView, ImageView, LinearLayout 실습

- 참고 – http://www.mobileplace.co.kr/2198

TextView attribute

- 그 전의 예에서 계속
- TextView
  - width – wrap_content
  - height – 50dp
  - android:visibility="gone"
    - 화면에 보이고 공간을 차지함
  - android:visibility="invisible"
    - 화면에 보이지 않으나 공간을 차지함
  - android:visibility="gone"
    - 화면에 보이지도 않고, 공간을 차지하지도 않음

TextView attribute

```
1.  <TextView
2.      android:layout_width="fill_parent"
3.      android:layout_height="fill_parent"
4.      android:text = "Hello, Oman"
5.      android:textColor = "#FF0000FF"
6.      android:textSize = "30sp"
7.      android:textStyle = "italic"
8.      android:gravity = "right|center_vertical"
9.      android:singleLine = "true"
10. />
```

TextView attribute

- android:text
- android:textColor
- android:textSize
- android:textStyle – bold, italic, etc.
- android:gravity – top, bottom, left, right, center, center_vertical, center_horizontal
- android:singleLine

ImageView

```
1. <ImageView
2.     android:layout_width="fill_parent"
3.     android:layout_height="fill_parent"
4.     android:src="@drawable/icon"
5.     android:scaleType="center"
6.     />
7. android:scaleType="fillCenter"
```

8. For your pic, copy it to res/drawable

LinearLayout

```
1.  <LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
2.    android:orientation="vertical"
3.    android:layout_width="fill_parent"
4.    android:layout_height="fill_parent"
5.    >
6.  <TextView
7.    android:layout_width="fill_parent"
8.    android:layout_height="0dp"
9.    android:layout_weight="1"
10.   android:background="#FF880000"
11.   />
12. <TextView
13.   android:layout_width="fill_parent"
14.   android:layout_height="0dp"
15.   android:layout_weight="1"
16.   android:background="#FF008800"
17.   />
18. <TextView
19.   android:layout_width="fill_parent"
20.   android:layout_height="0dp"
21.   android:layout_weight="1"
22.   android:background="#FF000088"
23.   />
24. </LinearLayout>
```

LinearLayout

- 그 전의 예에서 다음과 같이 변경
- orientation → vertical
- android:layout_width="0 dp"
- android:layout_height="fill_parent“
- layout_weight → 1, 2, 1

새로운 액티비티 시작하기

- startGame() – Game 이라는 액티비티를 위해 새 인텐트를 만들고 extraData 영역에 난이도 수준 숫자를 넣고 startActivity() 호출로 액티비티 시작

```
/** Start a new game with the given difficulty level
 */
private void startGame(int i) {
    Log.d(TAG, "clicked on " + i);
    Intent intent = new Intent(Sudoku.this,
Game.class);
    intent.putExtra(Game.KEY_DIFFICULTY, i);
    startActivity(intent);
}
```

Custom View 클래스

입력 다루기

- 아이폰 프로그래밍의 장점은 폰이 하나로 고정되어 있어서, 화면의 가로 세로 크기를 계산하여 그래픽을 처리할 필요가 없다는 것 – 따라서, 아예 애플 측에서는 직접 고정된 상수로 입력하는 것을 선호한다는 소문조차도 있음
- 안드로이드의 경우, 폰의 크기가 다양하고, 넷북일 수도 있으며, 디패드(키패드), 터치 스크린, 트랙볼 등의 입력도 있을 수 있음 – 따라서, 화면의 다양한 해상도는 물론 여러 입력 장치까지 지원해야 함

화면 흔들기

- 사용자가 이미 사용된 숫자를 집어넣는 것과 같은 명백한 실수를 하는 경우, 화면을 앞뒤로 흔들어 줌
- /res/anim/shake.xml 에 정의된 R.anim.shake 리소스를 불러와 실행해서 화면을 1 초에 10픽셀씩 좌우로 흔들어 줌
- 이 애니메이션의 실행 횟수와 속력, 가속도 등은 XML 내에 정의된 애니메이션 인터폴레이터(interpolator)에 의해 조정됨
  - 인터폴레이터 – /res/anim/cycle_7.xml

2D GUI 성능 개선하기

- onDraw() 메서드는 성능이 매우 중시되는 코드
  - 최소한의 작업만 담당하게 함
- onDraw() 메서드의 속도를 높이는 방법
  - onDraw() 메서드에선 되도록 객체 할당을 안함
  - 색상 상수 등은 다른 곳(예를 들면 뷰의 생성자)에서 프리페치
  - Paint 객체를 미리 만들어 놓고, onDraw()에서는 그 객체의 인스턴스를 사용
  - getWidth()의 반환 값인 폭(width)처럼 여러 번 사용되는 값은 메서드의 앞부분에서 로컬 사본에 저장해 놓고 엑세스함

Surface View 클래스

요약 (1)

- 선언하여 디자인을 하는 방법을 설명하고, 실제 예를 보였다.
- 시작화면을 만드는 방법과 대체 리소스를 사용하는 방법을 설명하고, 실제 예를 보였다.
- About 과 같은 상자를 구현하고, 테마를 적용하는 법을 설명하고, 실제 예를 보였다.
- 메뉴를 추가하는 방법을 설명하였다.
- 프로그램의 기본 설정(settings)을 정의하는 방법을 설명하였다.
- 대화 상자를 여는 방법을 설명하였다.
- 로그 메시지로 디버깅하는 방법을 소개하였다.
- 디버거로 디버깅하는 방법을 소개하였다.

요약 (2)

- 2D 그래픽에 대해 배웠다.
- Color, Paint, Canvas, Path, Drawable 클래스를 배웠다.
- Custom View 클래스를 정의해 보았다.
- 화면 흔들기를 구현하였다.
- 뷰와 레이아웃의 활용 방법을 체계적으로 학습했다.
- 2D 그래픽의 성능 개선에 대한 고려 사항을 알아보였다.

퀴즈 (1)

- 사용자 인터페이스를 디자인하는 방법들은 무엇이 있는가?
- 자동으로 생성되며 리소스와 관련이 있는 클래스는 무엇인가?
- 레이아웃은 무엇인가?
- 컨테이너란 무엇인가?
- 리소스 아이디는 어떻게 정의되는가?
- 대체 리소스를 사용해야 하는 경우는 언제인가?
- 새로운 액티비티는 어디에 등록해야 하는가?

퀴즈 (2)

- 안드로이드가 지원하는 두 가지 메뉴는 무엇인가?
- 기본 설정을 정의하기 위해서는 무슨 클래스를 확장해야 하는가?
- New Game 버튼을 클릭하면 나오는 대화 상자는 어떤 클래스인가?
- 로그 메시지로 디버깅하는 종류는 몇 가지가 있고 무엇 무엇인가?
- 프로젝트의 디버깅을 활성화하려면 어떻게 해야 하는가?
- 액티비티를 종료하려면 어떻게 하면 되는가?

퀴즈 (3)

- 화면에 칠할 때 필요한 색상을 설정하려면 어떤 클래스들을 이용해야 하는가?
- 화면에 그림이 그려지는 표면은 어떤 클래스인가?
- Path 클래스의 역할을 설명하라.
- Drawable 한 것들을 아는 대로 열거하라.
- XML로 콘텐츠를 설정하지 않고, 코드로 설정하는 경우의 장점은 무엇인가?
- 생성자에서 뷰의 폭과 높이를 사용하지 말아야 할 이유는 무엇인가?
- Activity 클래스의 onSizeChanged() 메서드에서 할 수 있는 일들은 무엇인가? 아는대로 쓰라.
- Activity 클래스의 onDraw() 메서드는 어떤 일을 하는가?
- Canvas 클래스의 drawLine() 메서드를 설명하라.
- Canvas 클래스의 drawText() 메서드를 설명하라.
- FontMetrics 클래스에 대해 설명하라.
- Activity 클래스의 onKeyDown() 메서드는 어떤 일을 하는가?
- Activity 클래스의 onTrackBallEvent() 메서드는 어떤 일을 하는가?
- invalidate() 메서드에서 매개 변수가 있을 때와 없을 때의 차이는 무엇인가?
- 애니메이션 인터폴레이터란 무엇인가?

연습문제 (1)

- 프로그램을 실행했을 때, 초기 화면을 선언해 보자. 이를 위해 xml 파일을 정의하고, 액티비티의 메서드를 수정하자.
- 선언한 초기 화면에 대해, 가로로 회전했을 때를 위한 레이아웃을 정의해 보자.
- 새로운 액티비티를 만들어 보자. 어떠한 일들을 해야 하는가?
- 안드로이드 SDK를 분석하여 어떠한 View 들이 있는지 전부 찾아보고 정리해 보자.

연습문제 (2)

- 메뉴 버튼을 눌렀을 때 나오는 메뉴에 자신의 이름이 나오도록 바꾸어 보라.
- 기본 설정에 체크 박스를 3 개 더 정의해 보라.
- HelloMenu 라는 안드로이드 프로그램 프로젝트를 만들어서 메뉴 버튼 눌렀을 때 나오는 메뉴 아이템을 "설정"이라고 하고, "설정"이라는 아이템을 누르면 "음악"과 "힌트"에 대한 설정 화면(액티비티)를 실행하는 프로그램을 만들어 보자.
- openNewGameDialog 을 보면 한 줄로 구성되어 있다. 이를 세 줄 이상으로 나누어 보라.
- openNewGameDialog 메서드의 DialogInterface.OnClickListener 를 매개변수가 아닌 외부 클래스로 정의해 보라.

연습문제 (3)

- 화면 흔들기를 1초에 20 픽셀씩 좌우로 흔드는 것을 10회 반복하도록 고쳐보자
- 자신의 Hello Android 프로그램에서 화면 흔들기를 1초에 10 픽셀씩 좌우로 흔드는 것을 7회 반복하도록 해보자
  - 힌트: XML 파일에서 최상위 Layout에 대해 ID를 부여한 뒤, findViewById()로 해당 Layout의 View 참조를 가져와 멤버 변수로 저장한다
- 몇 개의 이미지들을 미리 준비해 놓고, 버튼을 누를 때마다 화면의 이미지가 바뀌게 하고 싶다. 어떻게 구현해야 할까?
- 몇 개의 이미지들을 미리 준비해 놓고, 화면의 이미지를 빠르게 바꾸어 나가다가, 버튼을 누르면 그 때의 이미지로 멈추게 하고 싶다. 어떻게 구현해야 할까?
- 아주 간단한 3x3 틱택토(Tic-Tac-Toe) 게임을 만들어 보자. 화면 디자인에는 크게 신경쓰지 말고, 프로그램의 로직 구현에만 집중해서 만들어 보자.