헬로, 안드로이드

6주차 – 2D 그래픽 배우기 (2)

강대기
동서대학교 컴퓨터정보공학부

학습 목표

- 2D 그래픽에 대해 배운다.
- 수도쿠 키패드를 구현을 통해 GUI 구현을 실습한다.
- 수도쿠 게임 로직의 구현을 통해 안드로이드 프로그래밍을 학습한다.
- 수도쿠 게임의 성능 개선 방법을 고려해 본다.
- 뷰에 대해 체계적으로 학습하고 레이아웃의 활용 방법을 학습한다.

차례

- 키패드 만들기
- 게임 로직 구현하기
- 기타 사항
- 성능 개선하기
- 뷰와 뷰그룹
- 레이아웃
- 액티비티 복습
- 요약
- 퀴즈
- 연습문제

키패드 만들기

- 키패드 – 키보드가 없는 휴대폰에서 유용하게 쓸 수 있는 액티비티의 그리드로 구현함

- 레이아웃 – /res/layout/keypad.xml
- 키패드 클래스 – /src/org/example/sudoku/Keypad.java
- findViews() 메서드 – 모든 키패드의 키와 메인 키패드 창을 불러와 저장
- setListeners() – 모든 키패드의 키를 루프하여 각 키와 주요 키패드 창에 리스너를 설정
- 사용자가 키패드 버튼을 선택하면, 버튼의 인덱스와 retuenResult()가 호출됨
- onKeyDown() – 사용자가 키보드를 사용해 숫자를 입력하면 호출됨
- isValid() – 주어진 숫자가 현재의 위치에 유효한 값인지 확인
- retuenResult() – 호출한 액티비티에 선택된 숫자를 반환

- 게임 클래스 – /src/org/example/sudoku/Game.java
- showKeypadOrError() – 가능한 숫자가 무엇인지 알아보려면 extraData 영역에 있는 이미 사용된 숫자 문자열을 Keypad에 넘김

게임 로직 구현하기

- 게임 클래스 – /src/org/example/sudoku/Game.java
- setTileIfValid() – x,y 위치의 타일에 입력된 새 값이 유효할 때만 타일을 변경
- getUsedTiles() – 선택된 타일의 숫자 목록을 찾아옴
- 사용된 타일의 배열을 캐시하고 필요한 경우에만 calculateUsedTiles()로 계산함

기타 사항

- 게임 구현을 위한 유틸리티 함수와 변수들
- /src/org/example/sudoku/Game.java
- 하드코딩된 수도쿠 퍼즐 – easyPuzzle, mediumPuzzle, hardPuzzle
- getPuzzle() – 난이도 수준값을 취하고 그에 적합한 퍼즐을 반환
- toPuzzleString() – 퍼즐을 정수 배열에서 문자열로 변환
- fromPuzzleString() – 퍼즐을 문자열에서 정수 배열로 변환
- getTile() – x와 y의 위치를 받아 그 타일에 현재 설정된 숫자를 반환함
- getTileString() – 타일을 보여줄 때 사용됨. 타일 값과 함께 문자열을 반환하는 데, 타일이 비었으면 빈 문자열을 반환함

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

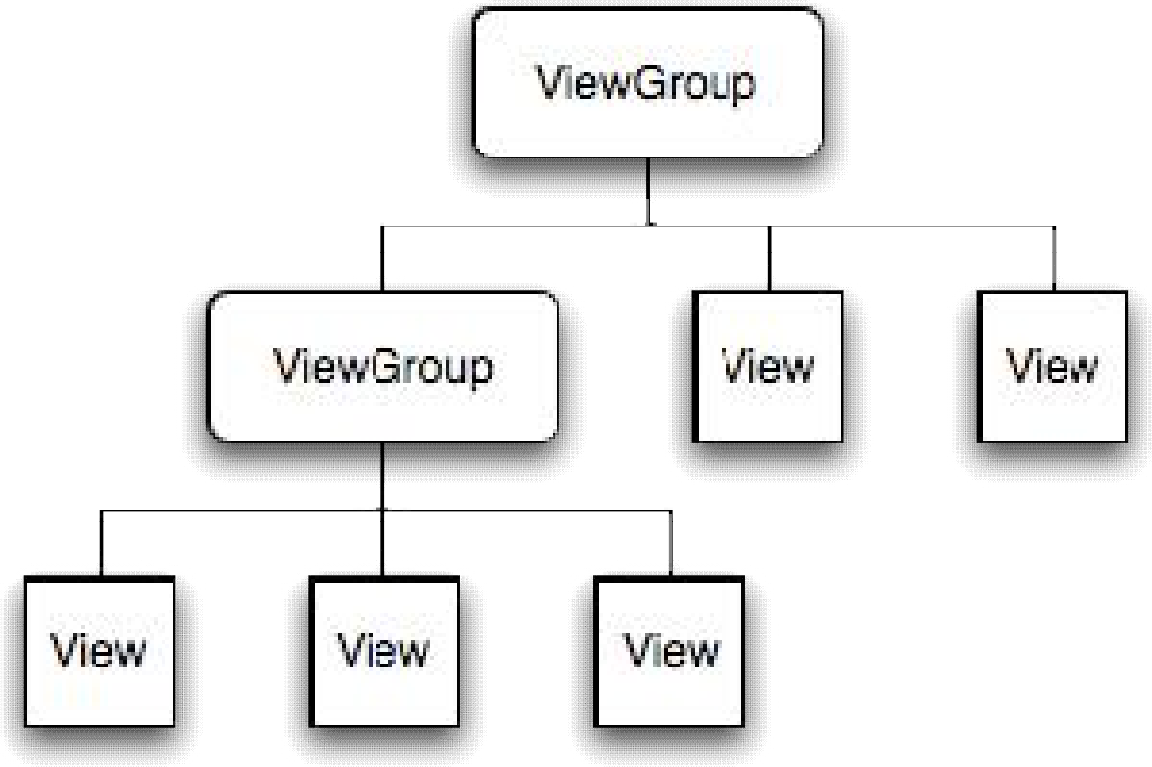
차례

- 키패드 만들기
- 게임 로직 구현하기
- 기타 사항
- 성능 개선하기
- 뷰와 뷰그룹
- 레이아웃
- 액티비티 복습
- 요약
- 퀴즈
- 연습문제

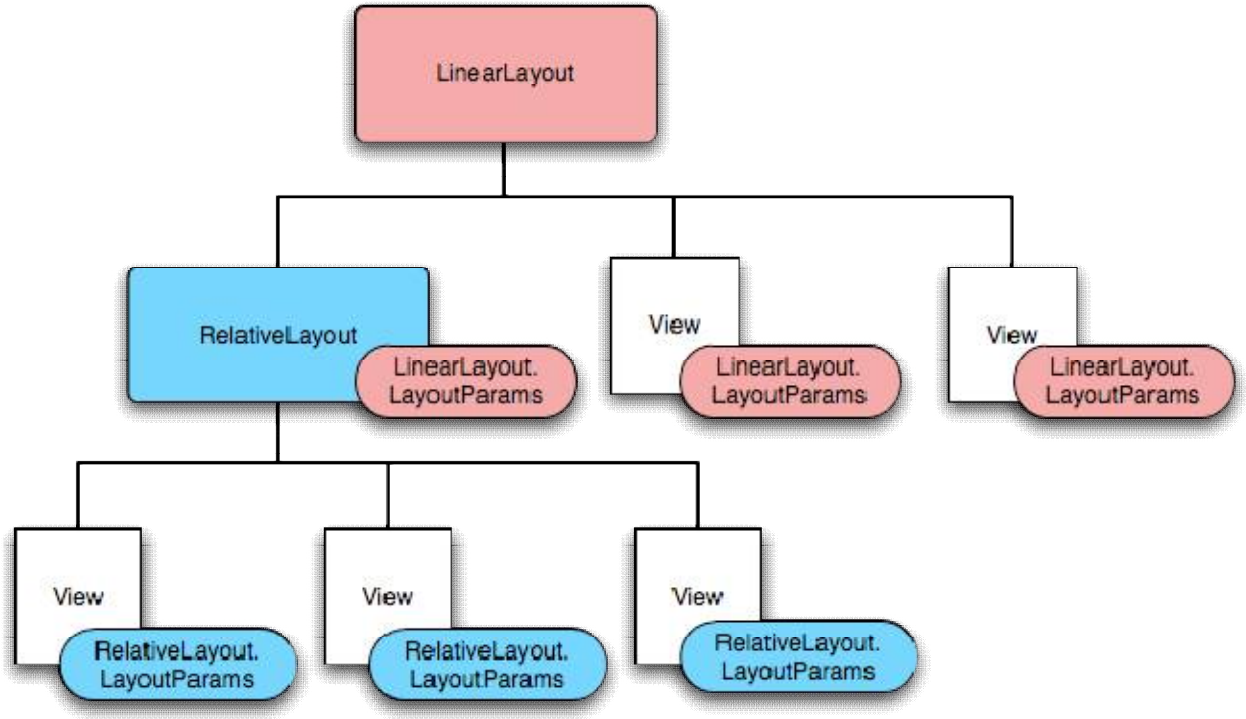
사용자 인터페이스 구현
Implementing a UI

- 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

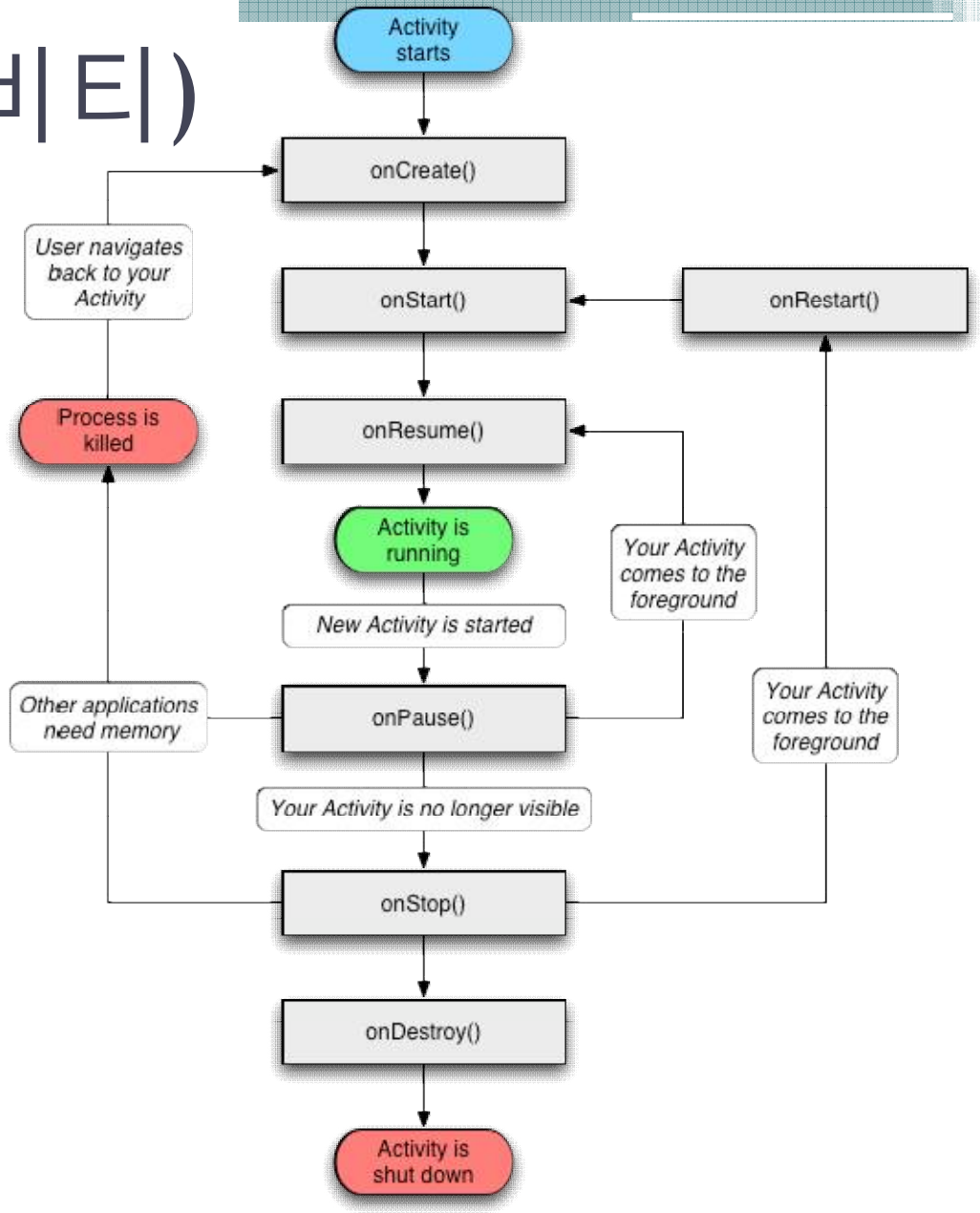


Building Blocks

- 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 (액티비티)

- 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

```
<TextView
    android:layout_width="fill_parent"
    android:layout_height="fill_parent"
    android:text = "Hello, Oman"
    android:textColor = "#FF0000FF"
    android:textSize = "30sp"
    android:textStyle = "italic"
    android:gravity = "right|center_vertical"
    android:singleLine = "true"
/>
```

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

```
<ImageView
    android:layout_width="fill_parent"
    android:layout_height="fill_parent"
    android:src="@drawable/icon"
    android:scaleType="center"
    />
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

요약

- 2D 그래픽에 대해 배웠다.
- 키패드 구현에 대해 알아보였다.
- 수도쿠 게임 로직을 더 상세히 알아보았다.
- 수도쿠 게임의 성능 개선에 대한 고려 사항을 알아보였다.
- 뷰와 레이아웃의 활용 방법을 체계적으로 학습했다.

퀴즈

- 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.

연습문제

- 안드로이드 SDK를 분석하여 어떠한 View 들이 있는지 전부 찾아보고 정리해 보자.
- 플레이어가 퍼즐을 맞추면 불꽃놀이 효과를 주도록 프로그램을 고쳐보자.
- 플레이어가 퍼즐을 맞추면 타일이 회전하도록 프로그램을 고쳐보자.
- 퍼즐을 푸는 동안, 배경 화면이 움직이도록 해보자.

심화 연습문제 및 프로젝트 아이디어

- 컴퓨터 프로그램으로 스도쿠 문제를 푸는 방법 중 하나인 Knuth의 dancing links 방법에 대해서 알아보고 이를 요약하여 설명해 보라.
- 책에 예제로 있는 스도쿠 문제는 하드코딩되어 있다. 자동으로 스도쿠 문제를 생성하려면 어떻게 해야 할까? (예: QQWing 프로그램)
- 자동으로 스도쿠 문제를 생성할 수 있다면, 퍼즐의 난이도는 어떻게 설정할 수 있을까?