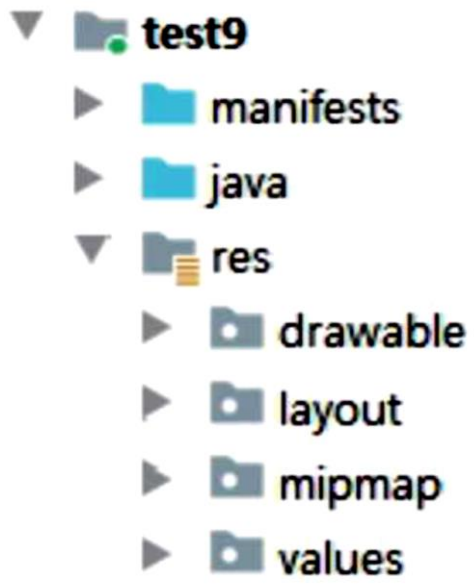
7. 리소스 활용하기

1. 리소프의 종류와 특징
2. 리소스 조건 설정
3. 폰 크기의 호환성
4. 메신저 앱의 인트로 화면 만들기

1. 리소스의 종류와 특징

- **앱 리소스 사용하기**
  - 리소스 디렉터리명은 고정
  - 리소스 파일명은 values에 추가하는 파일을 제외하고는 모두 자바의 이름 작성 규칙 준수
  - 알파벳 대문자를 사용할 수 없음



디렉터리명	리소스 종류
animator	속성 애니메이션 XML
anim	트윈 애니메이션 XML
color	색상 상태 목록 정의 XML
drawable	이미지 리소스
mipmap	앱 실행 아이콘 리소스
layout	레이아웃 XML
menu	메뉴 구성 XML
raw	원시 형태로 이용되는 리소스 파일
values	단순 값으로 이용되는 리소스
xml	특정 디렉터리가 정의되지 않은 나머지 XML 파일
font	글꼴 리소스

1. 리소스의 종류와 특징

앱 리소스 사용하기

- 레이아웃 리소스 — layout 디렉터리
- 이미지 리소스 — drawable 디렉터리
  - 이미지는 PNG, JPG, GIF, 9.PNG 파일
  - XML로 작성한 이미지도 가능

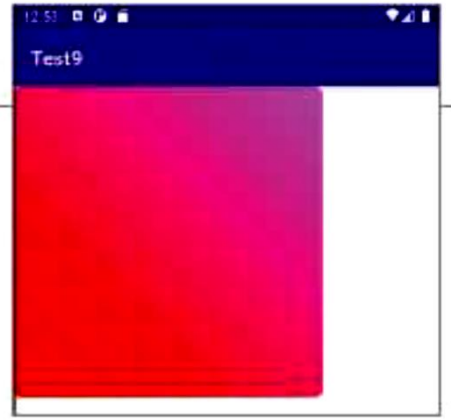
• XML로 작성한 이미지

```
<shape xmlns:android="http://schemas.android.com/apk/res/android"
    android:shape="rectangle">
    <gradient
        android:startColor="#FFFF0000"
        android:endColor="#80FF00FF"
        android:angle="45" />
    <corners android:radius="8dp" />
</shape>
```

• XML 이미지를 사용하는 예

```
<ImageView
    android:layout_width="300dp"
    android:layout_height="300dp"
    android:src="@drawable/gradient_box" />
```

▶ 실행 결과



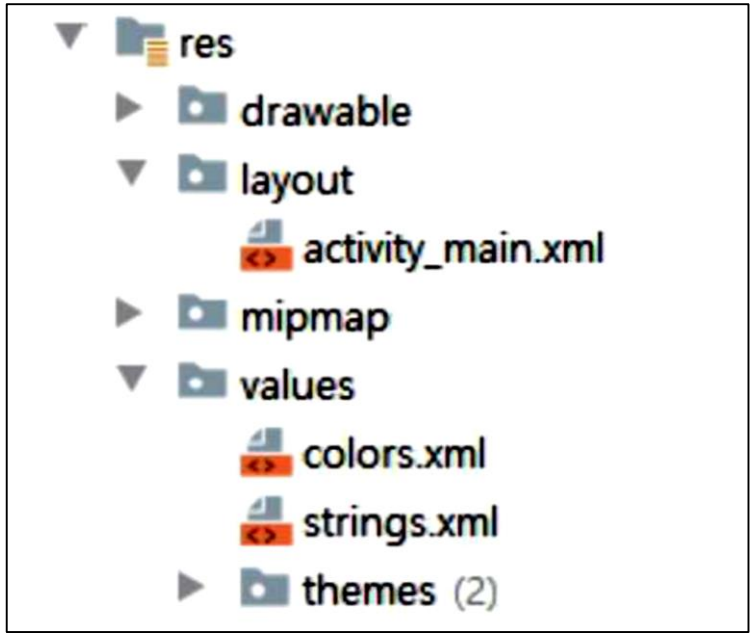
1. 리소스의 종류와 특징

- **리소스의 종류**

태그	설명
<shape>	도형을 의미하며 android:shape="rectangle"처럼 shape 속성을 이용해 도형의 타입을 지정. shape값은 rectangle, oval, line, ring 중에서 선택할 수 있음
<corners>	둥근 모서리를 그리는 데 사용. shape값이 rectangle일 때만 적용
<gradient>	그러데이션 색상 지정
<size>	도형의 크기 지정
<solid>	도형의 색상 지정
<stroke>	도형의 윤곽선 지정

1. 리소스의 종류와 특징

- 실행 아이콘 리소스 — mipmap 디렉터리
- 값 리소스 — values 디렉터리
- 문자열, 색상, 크기, 스타일, 배열 등의 값을 XML로 저장
- values 디렉터리의 리소스 파일은 파일명이 R인 파일에 식별자로 등록되지 않고 리소스 파일에 값을 지정한 태그의 name 속성값이 등록



• 문자열 리소스 등록

```
<resources>
    <string name="app_name">Test9</string>
    <string name="txt_data1">Hello</string>
    <string name="txt_data2">World</string>
</resources>
```

• XML에서 문자열 리소스 사용

```
<TextView
    android:id="@+id/textView"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="@string/txt_data1" />
```

• 코드에서 문자열 리소스 사용

```
binding.textView.text = getString(R.string.txt_data2)
```

1. 리소스의 종류와 특징

• 색상 리소스 등록

```
<resources>
    <color name="txt_color">#FFFF00</color>
    <color name="txt_bg_color">#FF0000</color>
</resources>
```

• 크기 리소스 등록

```
<resources>
    <dimen name="txt_size">20dp</dimen>
</resources>
```

• XML에서 색상과 크기 리소스 사용

```
<TextView
    android:id="@+id/textView"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:text="@string/txt_data1"
    android:textColor="@color/txt_color"
    android:background="@color/txt_bg_color"
    android:textSize="@dimen/txt_size" />
```

• 코드에서 색상과 크기 리소스 사용

```
binding.textView.text = getString(R.string.txt_data2)
binding.textView.setTextColor(ResourcesCompat.getColor(resources, R.color.txt_color, null))
binding.textView.textSize = resources.getDimension(R.dimen.txt_size)
```

1. 리소스의 종류와 특징

- 스타일 리소스는 style 태그로 등록
- 스타일 속성은 뷰에 설정되는 여러 속성을 스타일에 등록하여 한꺼번에 적용

• 스타일 등록

```
<resources>
    <style name="MyTextStyle">
        <item name="android:textSize">@dimen/txt_size</item>
        <item name="android:textColor">@color/txt_color</item>
    </style>
    <style name="MyTextStyleSub" parent="MyTextStyle">
        <item name="android:textColor">#0000FF</item>
        <item name="android:background">@color/txt_bg_color</item>
    </style>
</resources>
```

• 스타일 리소스 사용

```
<TextView
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    style="@style/MyTextStyleSub"
    android:text="Hello World" />
```

1. 리소스의 종류와 특징

- 색상 리소스 — color 디렉터리
  - color 디렉터리의 리소스는 특정 뷰의 상태를 표현하고 그 상태에 적용되는 색상을 등록

• 색상 리소스 등록

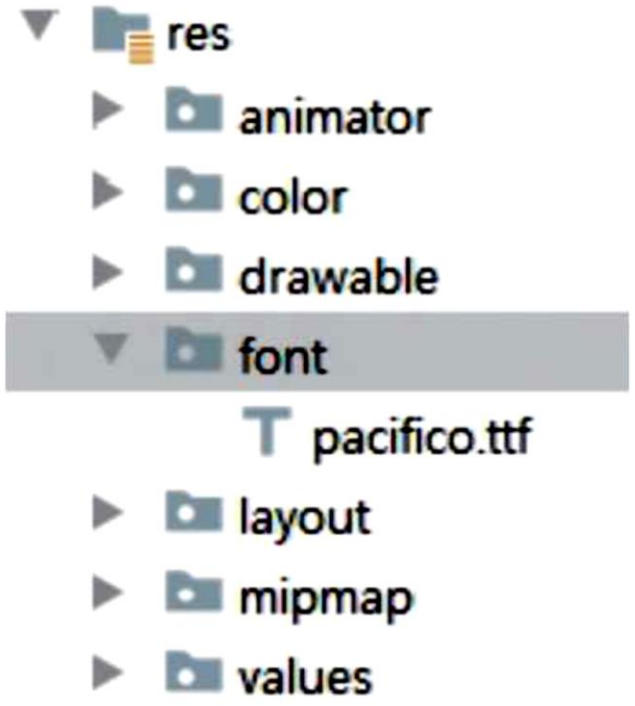
```
<?xml version="1.0" encoding="utf-8"?>
<selector xmlns:android="http://schemas.android.com/apk/res/android">
    <item android:state_pressed="true"
        android:color="#ffff0000" />
    <item android:state_focused="true"
        android:color="#ff0000ff" />
    <item android:color="#ff000000" />
</selector>
```

• 색상 리소스 사용

```
<Button
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:text="Click Me!!"
    android:textColor="@color/button_text" />
```

1. 리소스의 종류와 특징

- 글꼴 리소스 — font 디렉터리
  - font 디렉터리는 글꼴 리소스를 저장



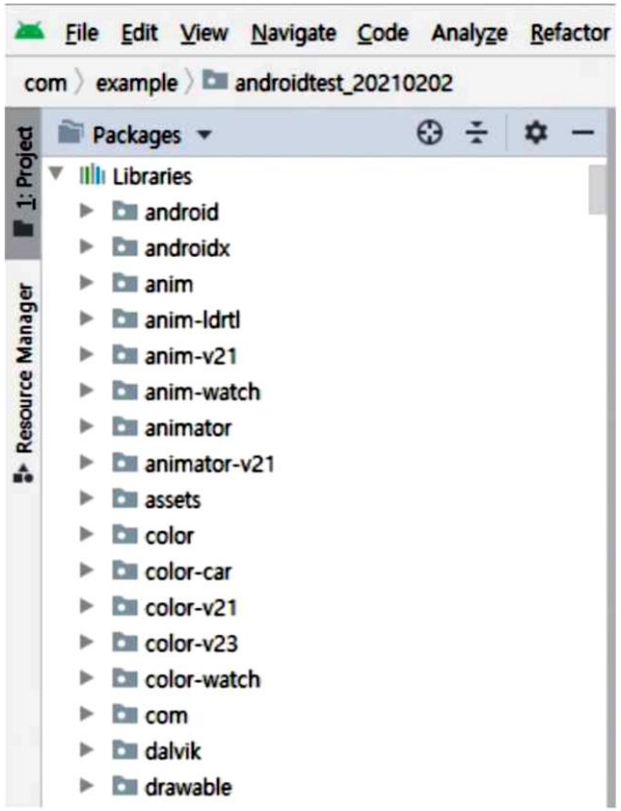
• 글꼴 리소스 사용

```
<TextView
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="HelloWorld"
    android:textSize="20dp"
    android:fontFamily="@font/pacifico" />
```

▶ 실행 결과

1. 리소스의 종류와 특징

- 플랫폼 리소스 사용하기
  - 안드로이드 플랫폼이 제공하는 리소스
  - android.R이라는 플랫폼 라이브러리의 R 파일에 등록



• 코드에서 플랫폼 리소스 사용

```
binding.imageView.setImageDrawable(ResourcesCompat.getDrawable(resources,
    android.R.drawable.alert_dark_frame, null))
binding.textView.text=getString(android.R.string.emptyPhoneNumber)
```

• XML에서 플랫폼 리소스 사용

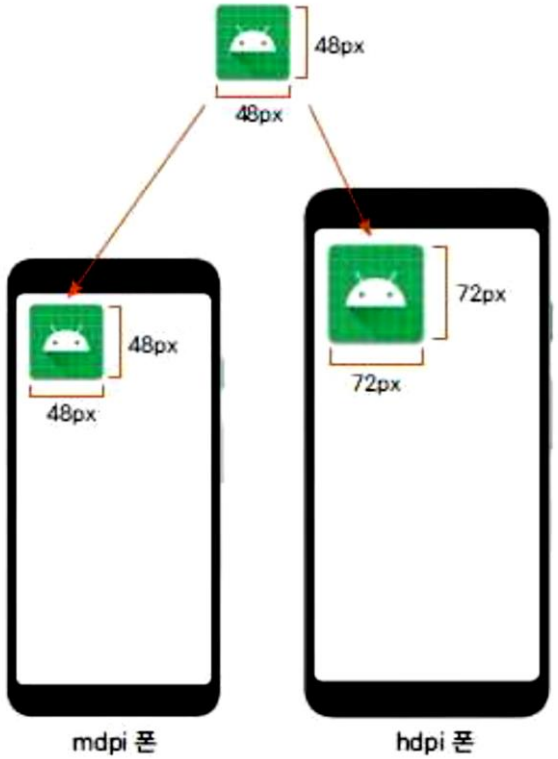
```
<ImageView
    android:id="@+id/imageView"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:src="@android:drawable/alert_dark_frame"/>

<TextView
    android:id="@+id/textView"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="@android:string/emptyPhoneNumber"/>
```

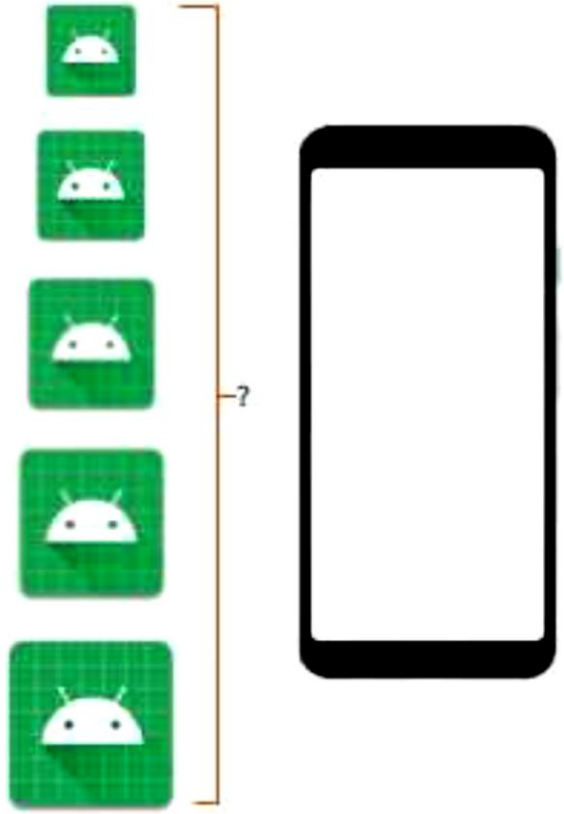
2. 리소스 조건 설정

리소스 조건 설정이란?

- 리소스를 특정 환경에서만 적용되도록 설정
- 기기별 실행 아이콘 크기

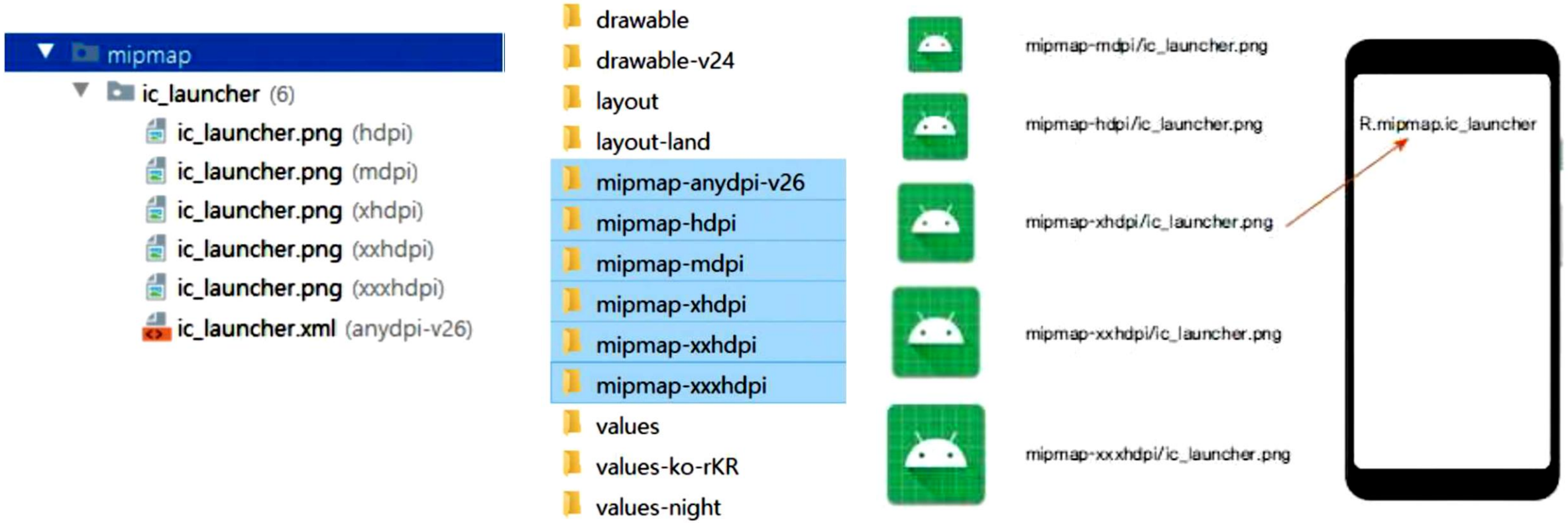


화면 밀도	크기
XXXHDPI	192×192
XXHDPI	144×144
XHDPI	96×96
HDPI	48×48
MDPI	36×36



2. 리소스 조건 설정

- 리소스 디렉터리 이름에서 붙임표(-) 뒤의 단어가 리소스의 조건
- mipmap-mdpi 디렉터리는 mdpi라는 조건



2. 리소스 조건 설정

조건	예시	설명
MCC 및 MNC	mcc310 mcc310-mnc004	이동통신 국가 코드(MCC)와 선택적으로 이동통신 네트워크 코드(MNC) 추가 가능. mcc310은 미국이며 mcc310-mnc004는 버라이즌을 이용하는 미국을 의미
언어 및 지역	en ko-rKR	ISO639-1 언어 코드이며 선택적으로 뒤에 소문자 r을 추가해 ISO3166-1-alpha-2의 지역 코드가 나올 수 있음
레이아웃 방향	ldrtl ldltr	히브리어처럼 오른쪽에서 왼쪽으로 쓰는 언어의 레이아웃에서 유용하게 이용 가능. ldrtl은 오른쪽에서 왼쪽, ldltr은 왼쪽에서 오른쪽 방향 레이아웃
더 작은 쪽	sw320dp	화면 크기 중 더 작은 쪽에 대한 조건. 화면 방향과 상관없이 화면의 높이와 너비 중 작은 쪽에 대한 조건을 의미. sw320dp이면 너비든 높이든 상관없이 작은 쪽의 치수가 320dp인 경우를 의미
이용 가능한 너비	w720dp	화면 너비에 대한 조건. w720dp이면 너비가 720인 기기
이용 가능한 높이	h720dp	화면 높이에 대한 조건. h720dp이면 높이가 720인 기기
화면 크기	small, normal, large, xlarge	화면 크기를 small, normal, large, xlarge로 판단해 조건 명시. small은 320×428, normal은 320×470, large는 480×640, xlarge는 720×960 정도의 크기

2. 리소스 조건 설정

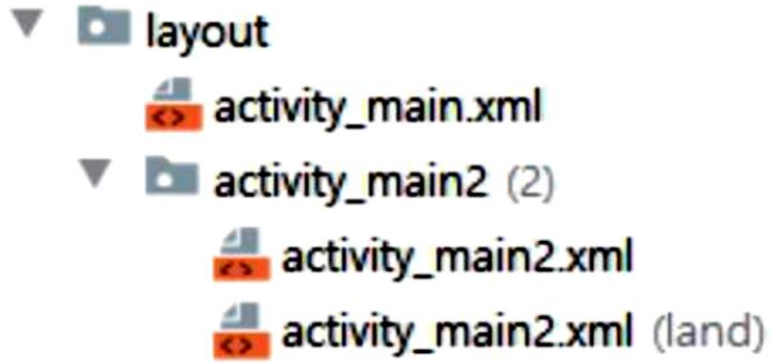
화면 비율	long, notlong	화면의 종횡비 조건. long은 WQVGA, WVGA, FWVGA 등의 긴 화면, notlong은 QVGA, HVGA, VGA 등의 길지 않은 화면
원형 화면	round, notround	원형 화면인지 판단. round는 웨어러블 기기처럼 둥근 화면을 가지는 기기. notround는 폰이나 태블릿처럼 사각형 화면의 기기
화면 방향	port, land	화면의 방향에 대한 조건. port는 세로 방향, land는 가로 방향
UI 모드	car, desk, television, application, watch, vrheadset	기기가 도크에 삽입되거나 제거될 때 대응을 위한 조건. car는 자동차, desk는 데스크, television은 TV, application은 표시되지 않은 제품
야간 모드	night, notnight	야간 모드에 대응하기 위한 조건. night는 야간, nonight는 주간
화면 픽셀 밀도	ldpi, mdpi, hdpi, xhdpi, xxhdpi, xxxhdpi, nodpi, tvdpi, anydpi, nnndpi	화면 밀도에 대한 조건. ldpi는 120dpi, mdpi는 160dpi, hdpi는 240dpi, xhdpi는 320dpi, xxhdpi는 480dpi, xxxdpi는 640dpi, nodpi는 크기를 조정하지 않을 리소스에 사용
터치 스크린 유형	notouch, finger	터치 스크린을 제공하는지 판단. notouch는 터치 스크린이 없는 기기

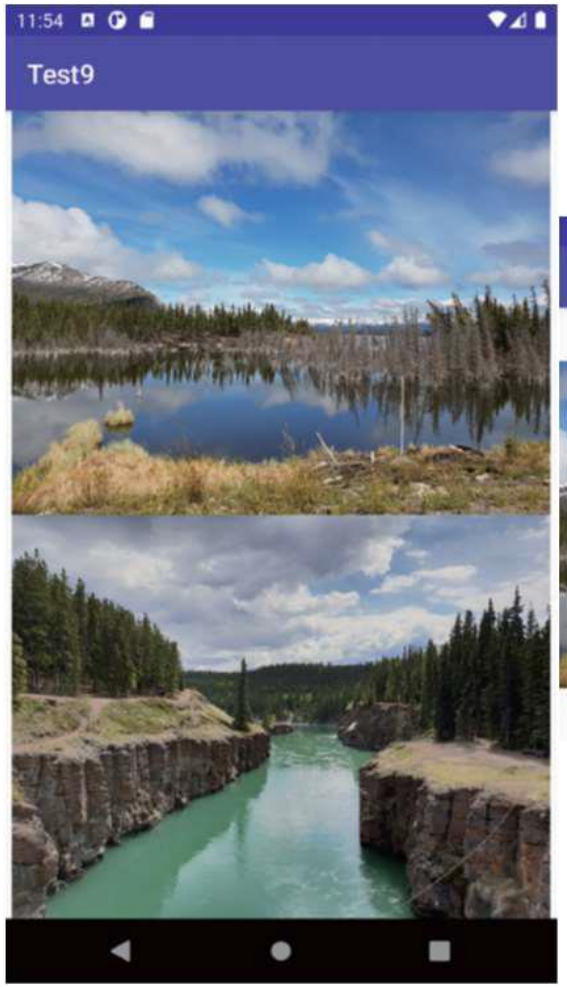
2. 리소스 조건 설정

키보드 가용성	keysexposed, keyshidden, keyssoft	키보드의 유형. keyssoft는 소프트 키보드, keysexposed는 키보드가 노출되어 있는 기기, keyshidden은 키보드가 있으나 숨길 수 있는 기기
기본 텍스트 입력방법	nokeys, qwerty, 12key	nokeys는 하드웨어 키보드가 없는 경우. qwerty는 하드웨어 쿼티 키보드가 있는 경우. 12key는 하드웨어 12키가 있는 경우
탐색 키 가용성	navexposed, navhidden	탐색 키 사용 가능 조건. navexposed는 탐색 키 사용 가능, navhidden은 탐색 키 사용 불가
기본 비터치 탐색방법	nanav, dpad, trackball, wheel	터치하지 않고 탐색이 가능한 기기가 있는지 판단
플랫폼 버전	v21	기기의 API 레벨

2. 리소스 조건 설정

- 화면 회전에 대응하기
  - 방향에 따라 화면을 다르게 제공
  - 리소스 조건으로 설정하여 처리





2. 리소스 조건 설정

국제 언어 제공하기

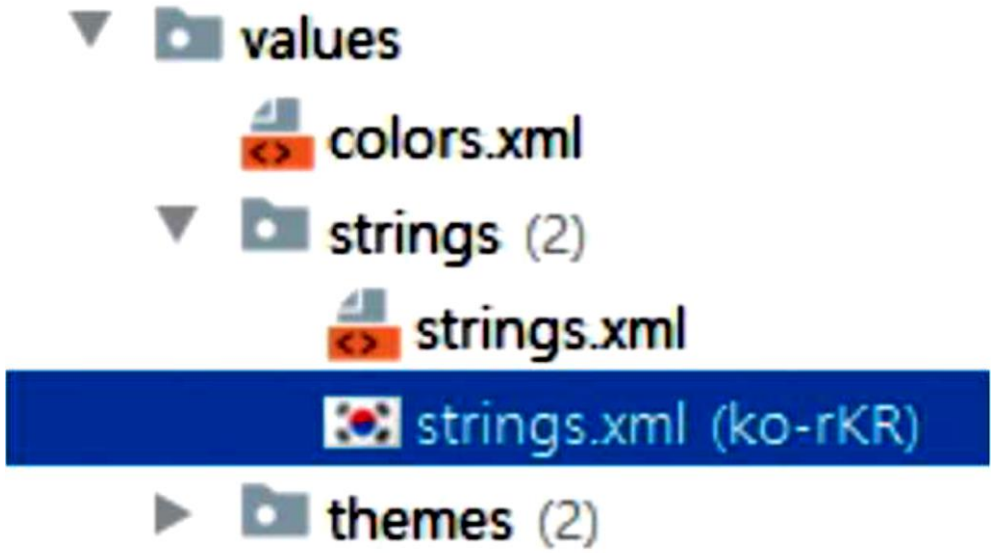
- 리소스 문자열을 각국 언어로 제공
- 파일을 여러 개 만들어서 각 언어에 맞는 리소스 문자열을 담고 어느 XML 파일을 적용해야 하는지를 리소스 디렉터리명으로 지정

• 영어 문자열 리소스 예

```
<resources>
    <string name="app_name">Test9</string>
    <string name="intro">Hello</string>
</resources>
```

• 한국어 문자열 리소스 예

```
<resources>
    <string name="app_name">테스트9</string>
    <string name="intro">안녕하세요</string>
</resources>
```



3. 폰 크기의 호환성

- 논리적인 단위 알아보기
  - dpi는 dots per inch의 줄임말로 1인치 안에 있는 도트의 개수
  - 크기 지정에 사용할 수 있는 단위
    - dp(dip: density-independent pixels): 스크린의 물리적 밀도에 기반을 둔 단위
    - sp(sip: scale-independent pixels): dp와 유사하며 글꼴 크기에 적용
    - pt(points): 스크린 크기의 1/72을 1pt로 함
    - px: 픽셀
    - mm: 밀리미터
    - in: 인치
  - mdpi 폰에서 1dp는 1px입니다.
  - xxhdpi 폰은 개발자가 지정한 크기보다 3배 정도 크게 출력

크기	설명
ldpi	저밀도 화면이며 ~120dpi
mdpi	중밀도 화면이며 ~160dpi
hdpi	고밀도 화면이며 ~240dpi
xhdpi	초고밀도 화면이며 ~320dpi
xxhdpi	초초고밀도 화면이며 ~480dpi
xxxhdpi	초초초고밀도 화면이며 ~640dpi

크기	배율
ldpi	0.75
mdpi	1.0
hdpi	1.5
xhdpi	2.0
xxhdpi	3.0
xxxhdpi	4.0

3. 폰 크기의 호환성

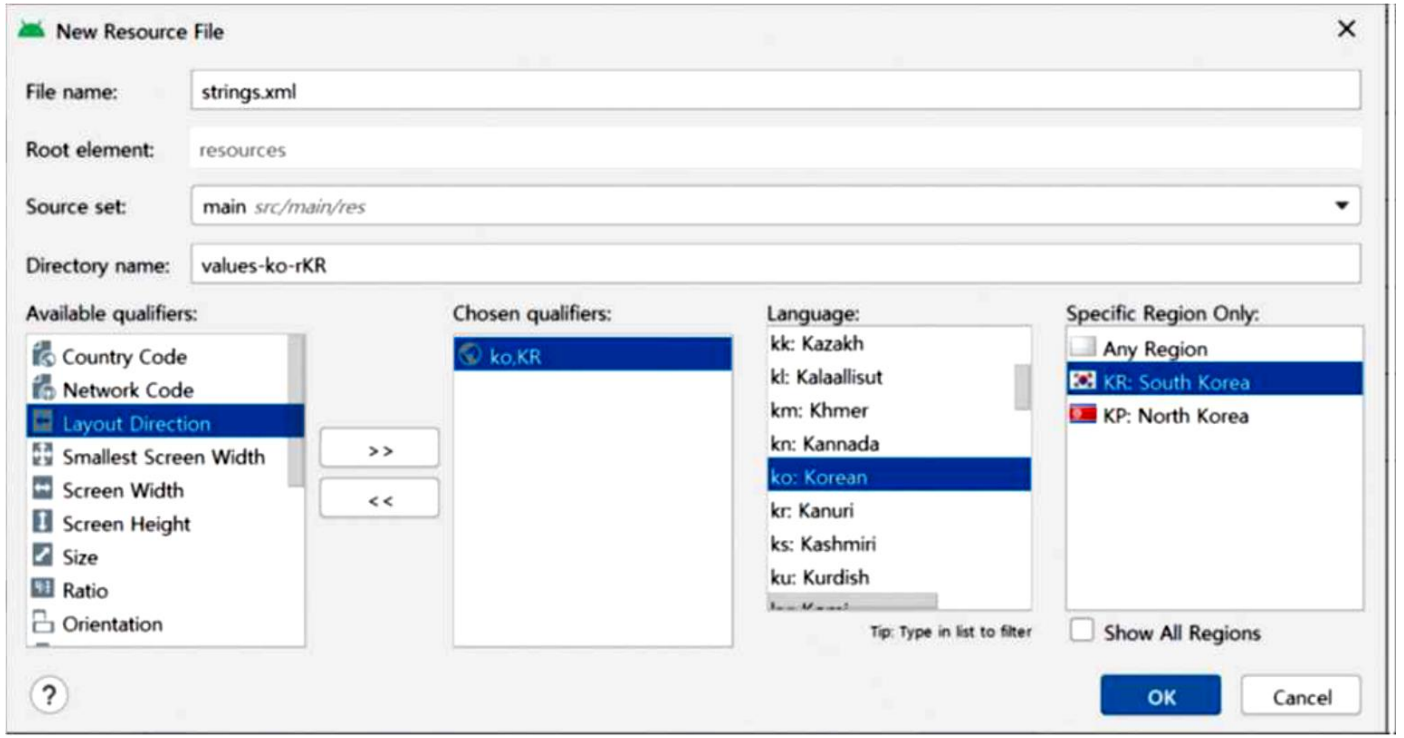
- 화면 정보 가져오기
  - 30 이전 버전에서는 DisplayMetrics로 크기 정보 활용
  - 30 버전부터는 이 방법을 지원하지 않고(deprecation), WindowMetrics를 이용

• 기기의 가로, 세로 크기 가져오기

```
if (Build.VERSION.SDK_INT >= Build.VERSION_CODES.R) {
    val windowMetrics: WindowMetrics = windowManager.currentWindowMetrics
    binding.textView.text = "width : ${windowMetrics.bounds.width()},
                            height : ${windowMetrics.bounds.height()}"
} else {
    val display = windowManager.defaultDisplay
    val displayMetrics = DisplayMetrics()
    display?.getRealMetrics(displayMetrics)
    binding.textView.text = "width : ${displayMetrics.widthPixels},
                            height : ${displayMetrics.heightPixels}"
```

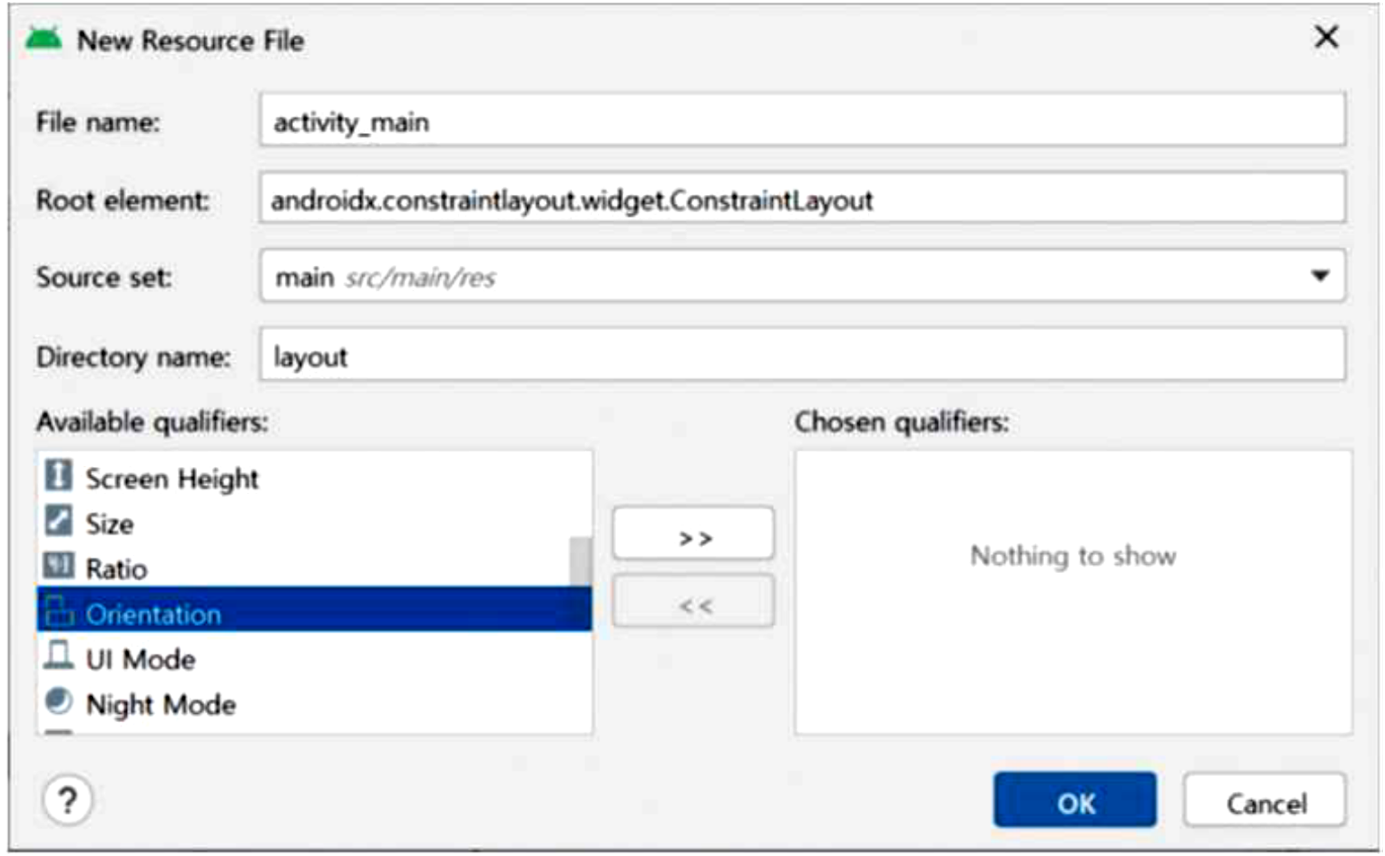
실습: 메신저 앱의 인트로 화면 만들기

- 1단계. 새 모듈 생성하기
  - Ch9_Resource라는 이름으로 새로운 모듈을 작성.
- 2단계. 리소스 파일 준비하기
  - round_button.xml 파일과 intro.png 파일 drawable 디렉터리에 복사
- 3단계. 언어별 문자열 리소스 작성하기
  - strings.xml 파일 작성
  - values-ko-rKR/values.xml 파일 생성, 작성
- 4단계. 세로 방향 화면 구성하기
  - activity_main.xml 파일 작성



실습: 메신저 앱의 인트로 화면 만들기

- 5단계. 기로방향 화면 구성하기
  - layout-land/activity_main.xml 파일 생성, 작성



실습: 메신저 앱의 인트로 화면 만들기

- 6단계. 앱 실행하기