

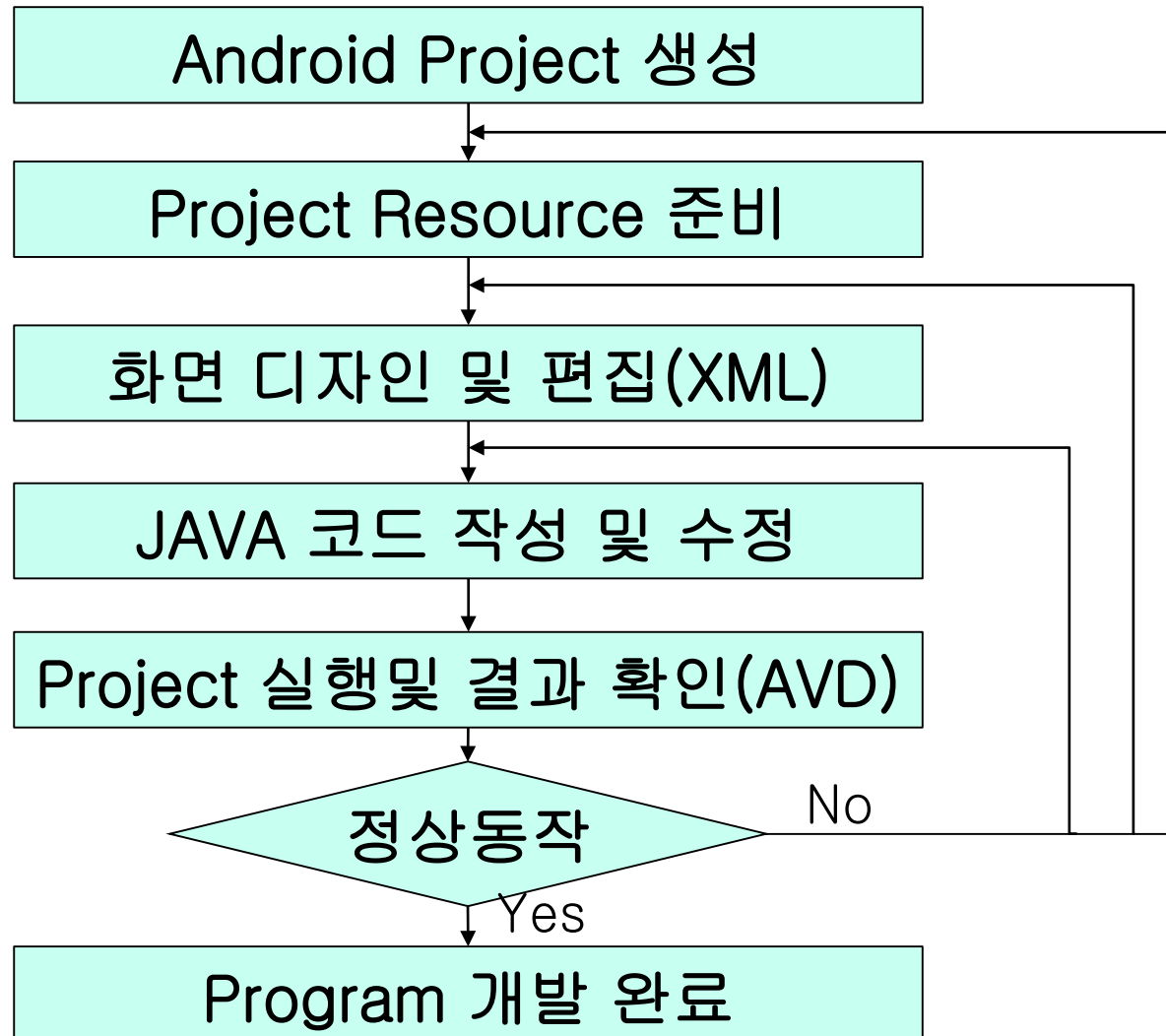


맛보기 프로젝트 Pinwheel(바람개비의 회전)

배 희호 교수
경북대학교
스마트IT과



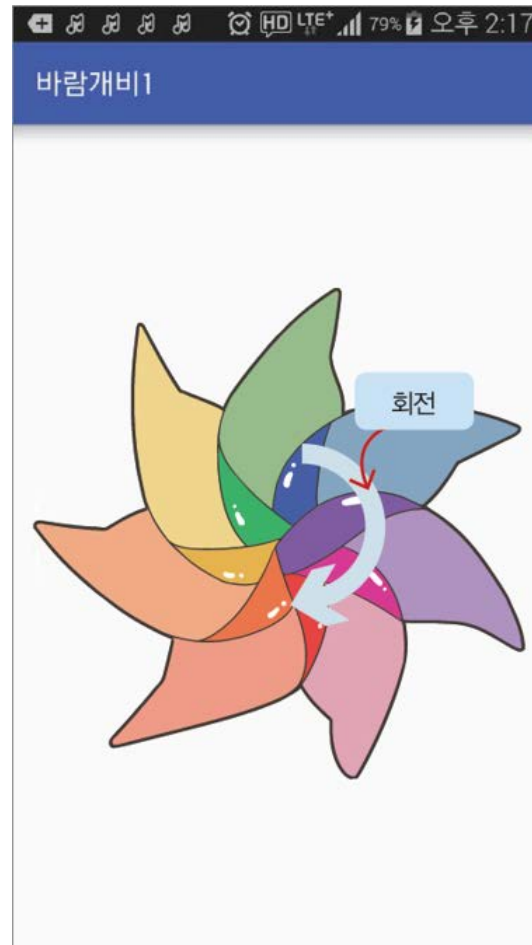
Android 응용 프로그램 개발 단계





바람개비의 회전

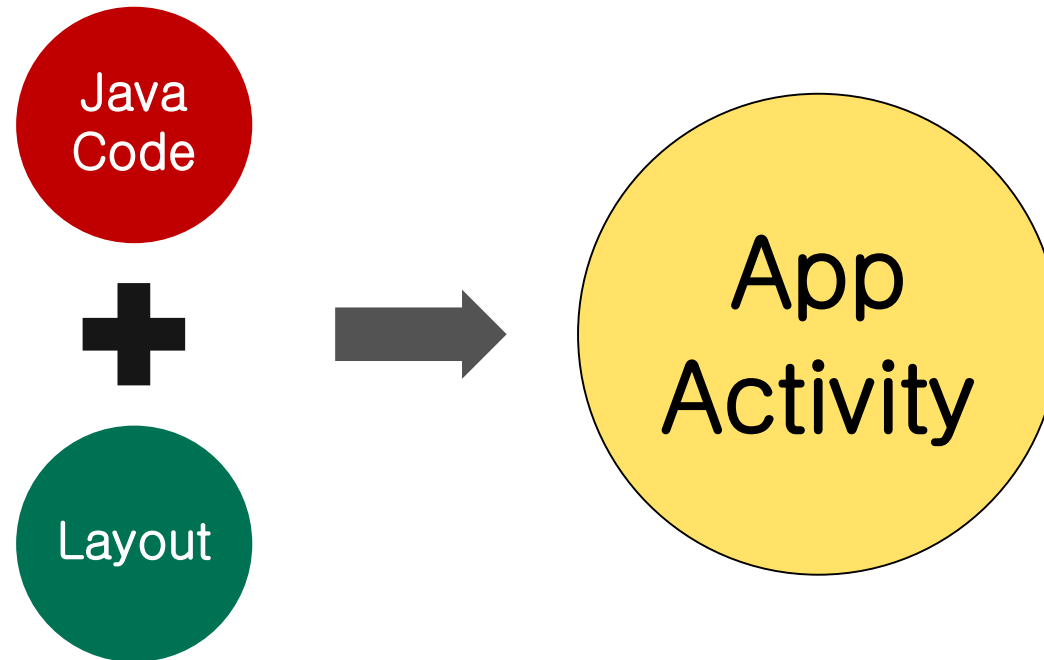
- 바람개비가 2초간 360° 를 일정한 속도로 계속 회전 시키는 간단한 App을 만들어보자





바람개비의 회전

■ Project 구성요소

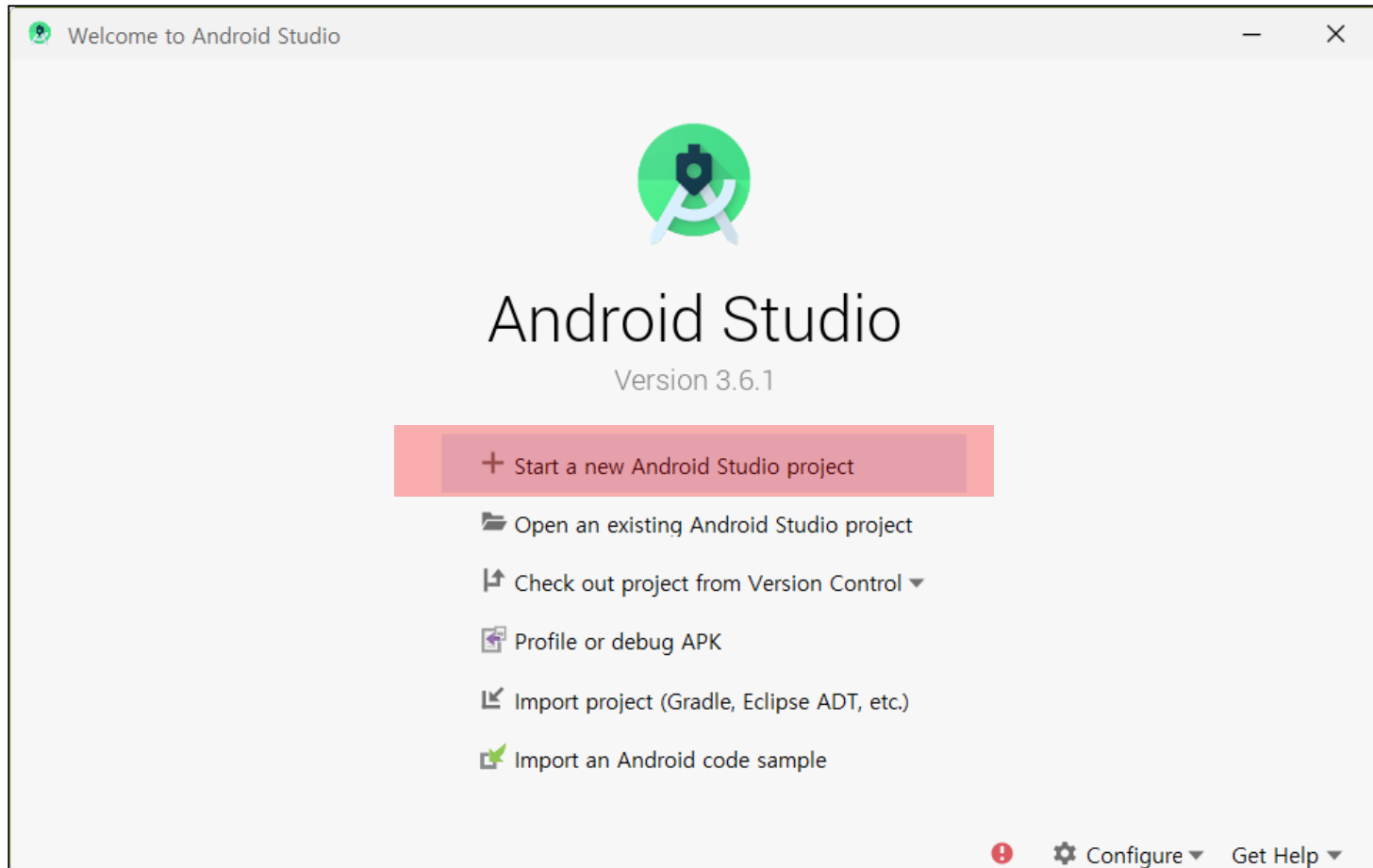




바람개비의 회전

■ Project 생성

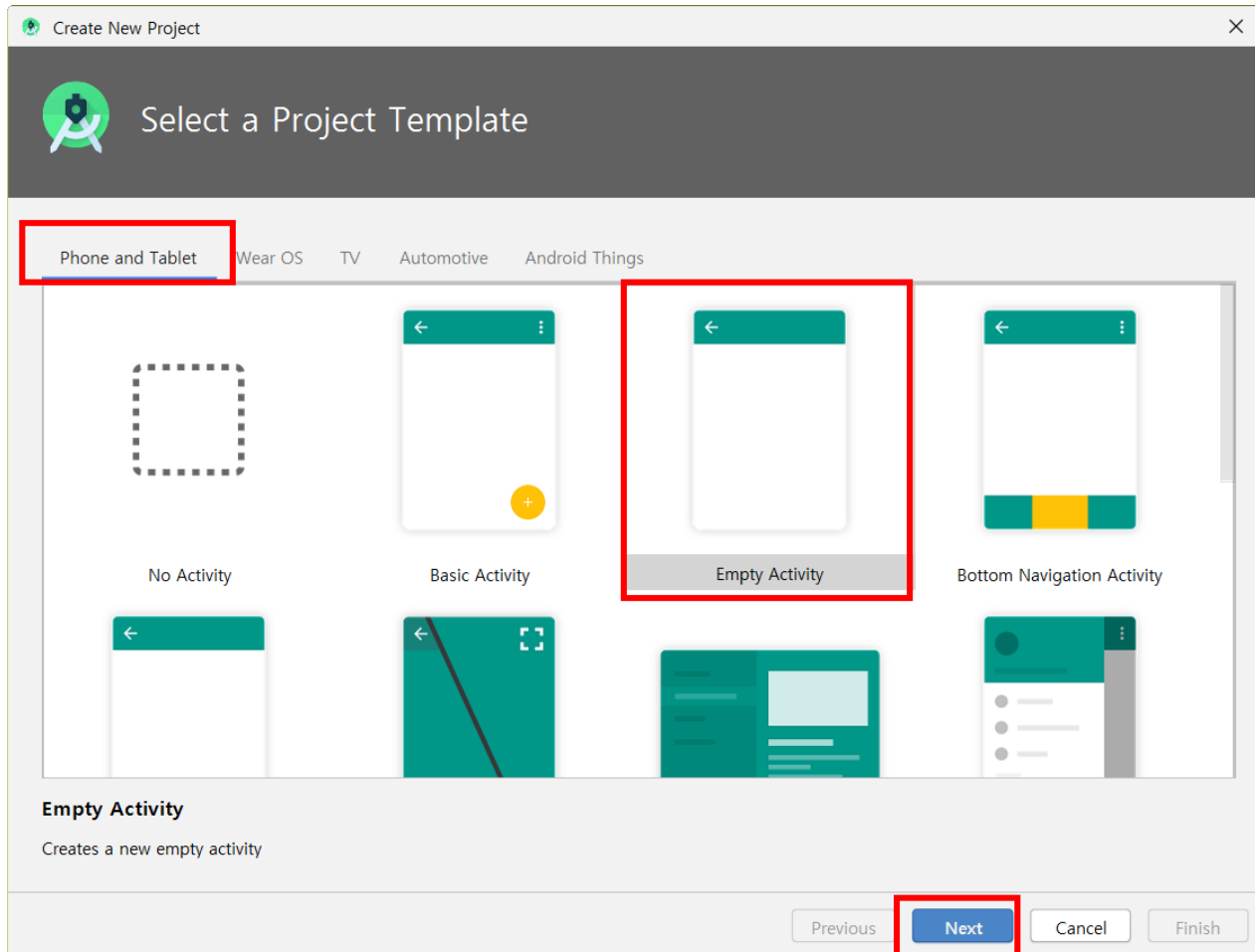
■ [Start a new Android Studio project] 선택





바람개비의 회전

- [Select a Project Template] 창에서 [Phone and Tablet] 탭에서 디폴트인 Empty Activity 선택





바람개비의 회전

■ Project 생성

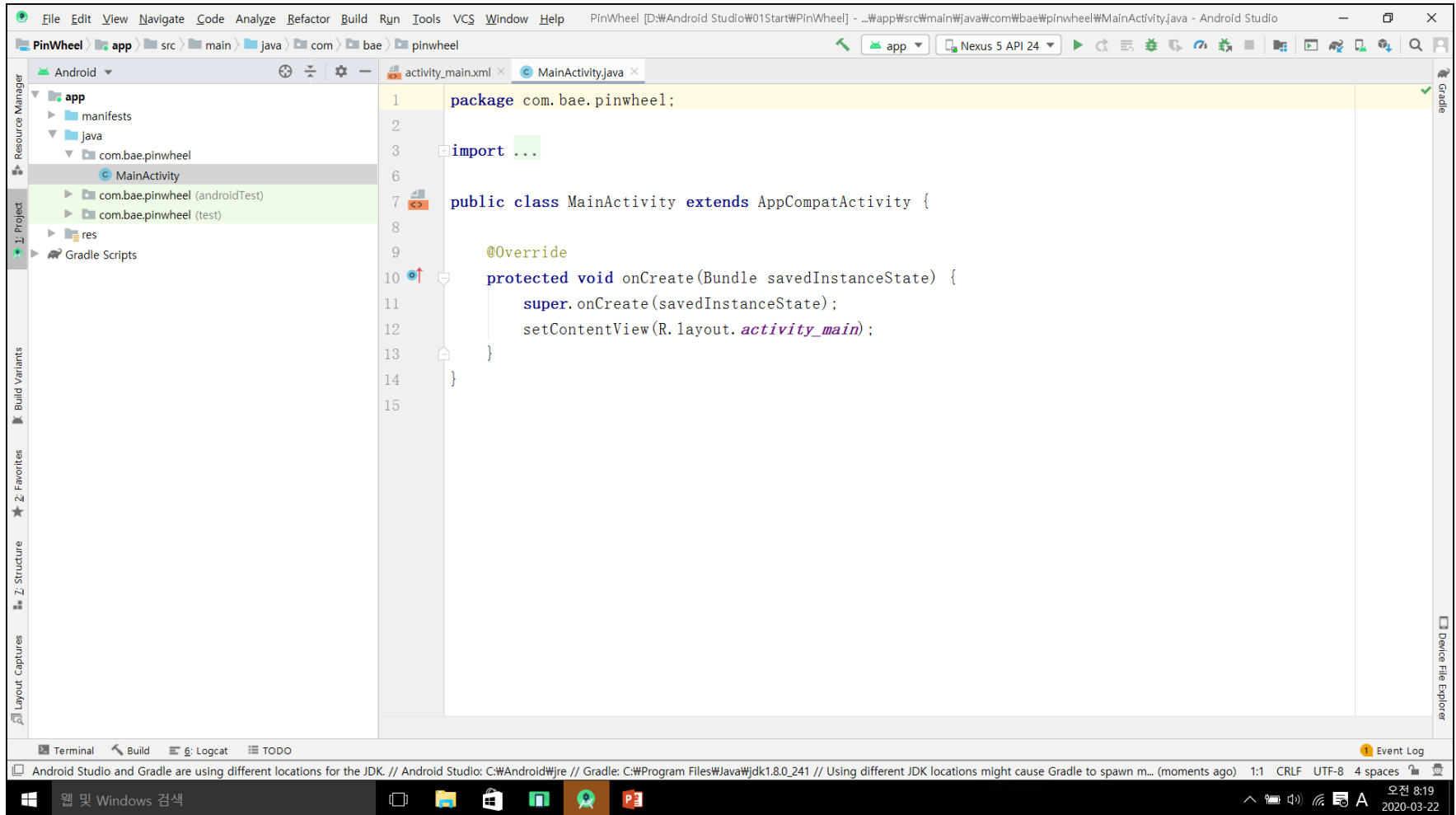
■ Application name : PinWheel 입력





바람개비의 회전

Project 생성 완료



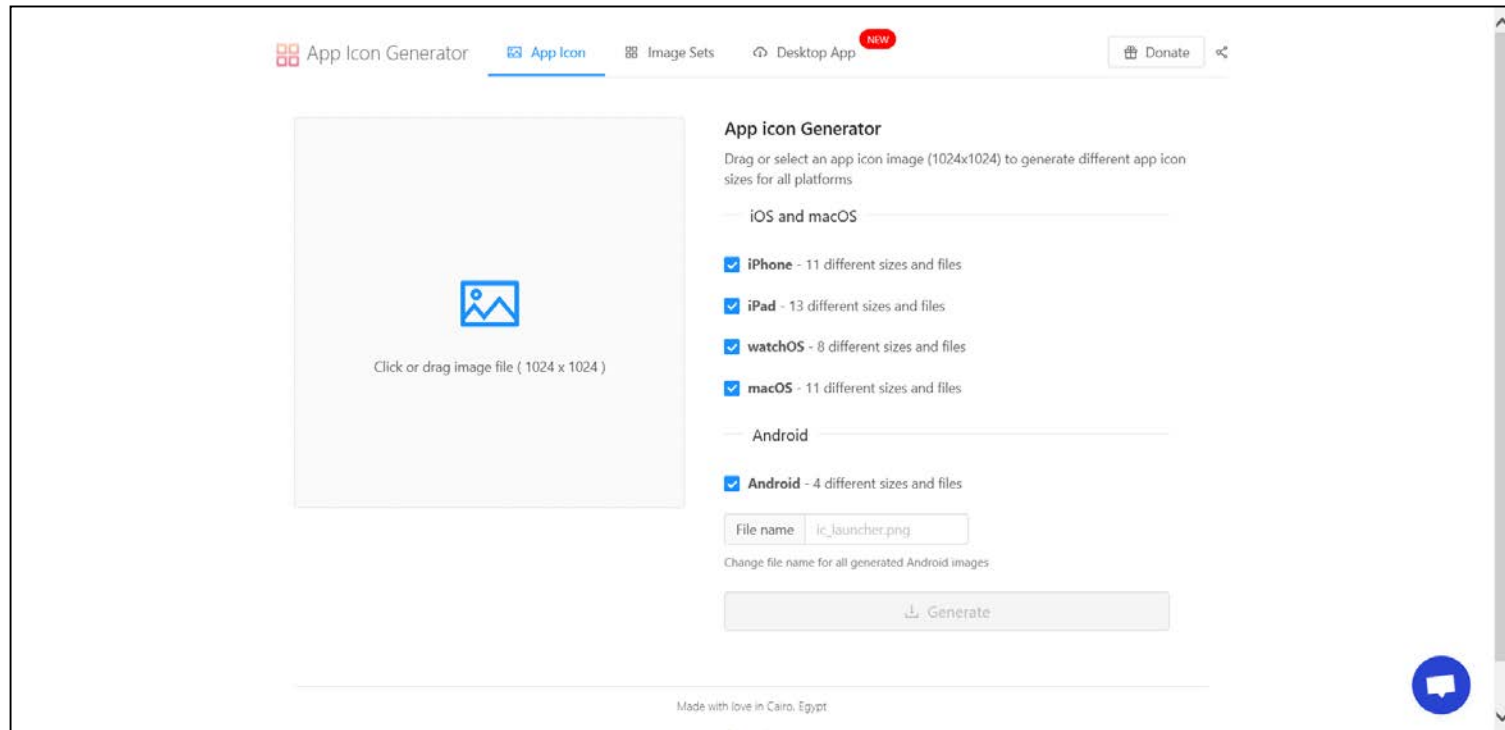


바람개비의 회전

■ 앱 아이콘

■ iOS, Android 애플리케이션 icon을 만들어주는 사이트

■ <https://appicon.co/>



■ 1024x1024 png 파일을 첨부하면 사이즈별 아이콘을 만들어줌



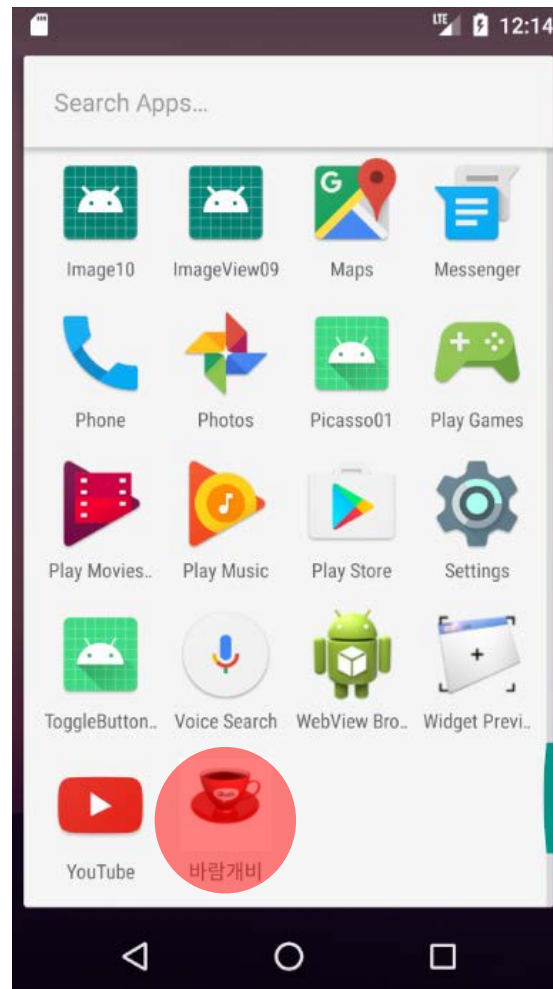
바람개비의 회전

- icon들은 mipmap을 사용해서 또는 전통적인 방식인 drawable을 사용해서 코드와 연결
- App icon
 - App icon 변경은 매니페스트에서
android:icon = “자신이 원하는 사진파일 경로를 설정”
- App 이름 변경
 - label 인데 경로가 string/app_name 이니, string으로 변경하면 됨



바람개비의 회전

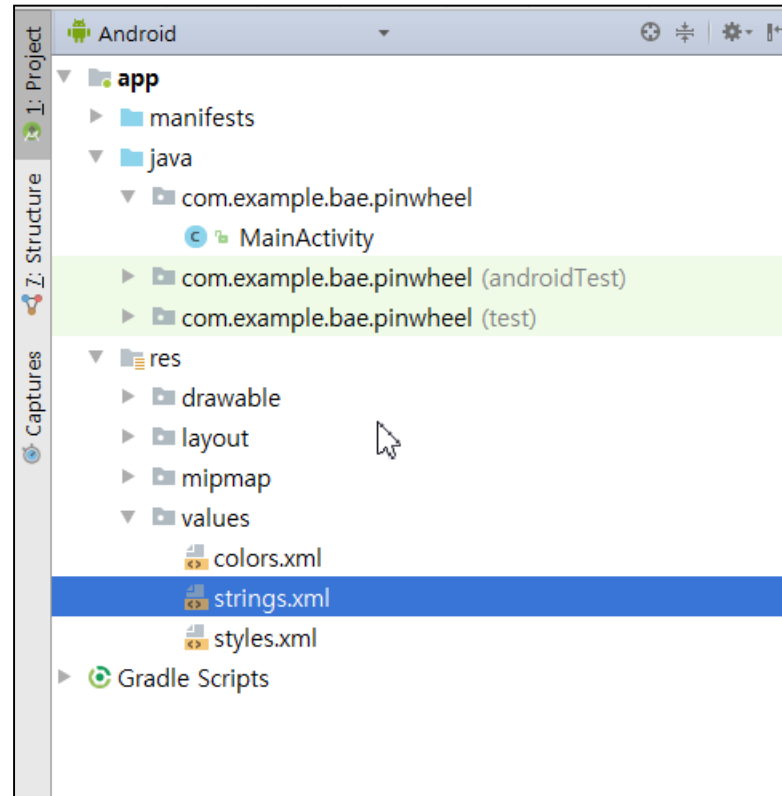
■ App icon 적용





바람개비의 회전

- Application 라벨(이름) 수정
 - [app]-[res]-[values]-[strings.xml] 선택





바람개비의 회전

- Application 라벨 수정
 - name을 “바람개비”로 수정

```
resources>  
  <string name="app_name">PinWheel</string>  
</resources>
```

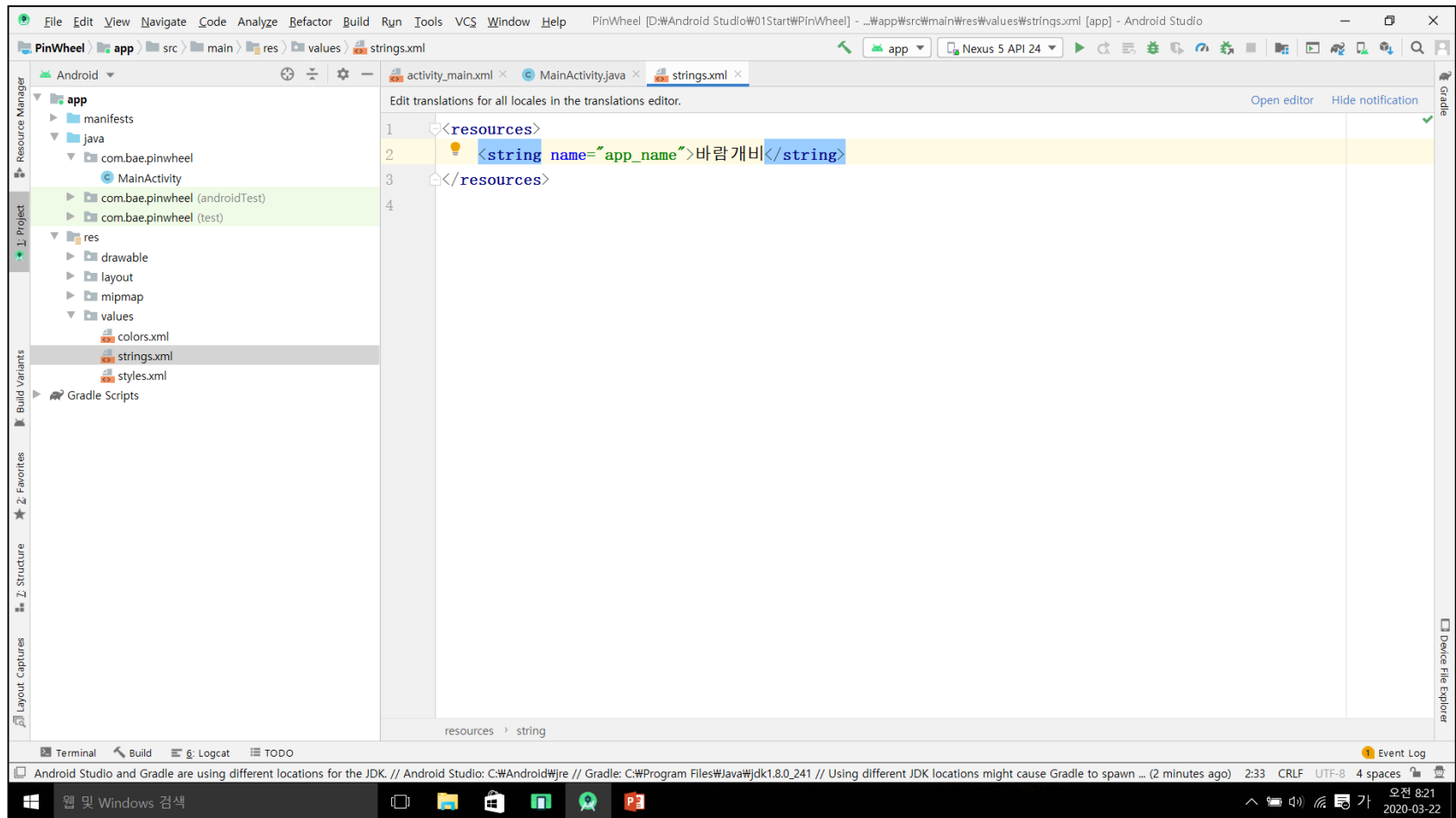


```
<resources>  
  <string name="app_name">바람개비</string>  
</resources>
```



바람개비의 회전

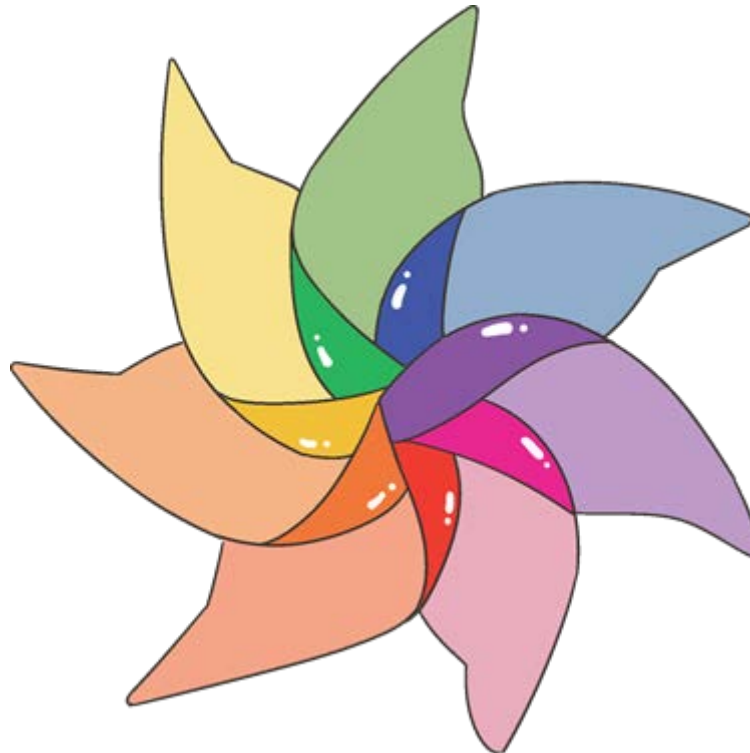
■ Application 라벨 수정





바람개비의 회전

- 바람개비 image 추가
 - 바람개비 image를 확인 (파일 이름 소문자 확인)



pinwheel.png



바람개비의 회전

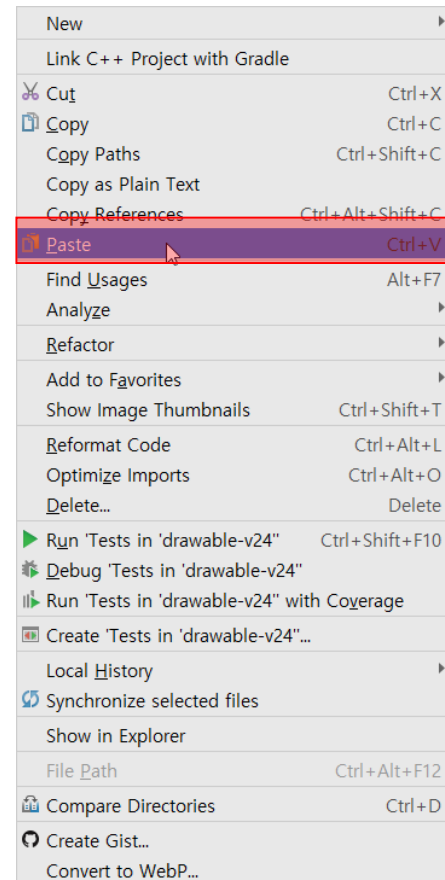
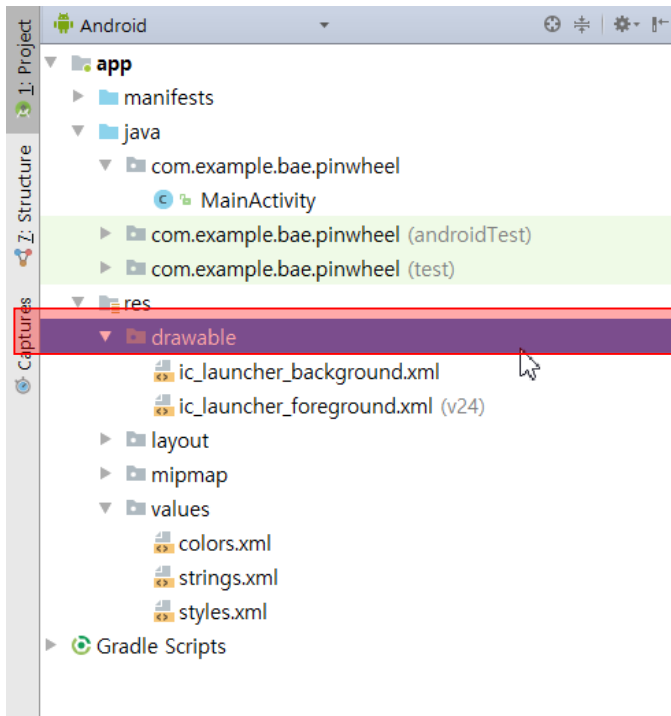
- 바람개비 image 추가
- 바람개비 image를 복사





바람개비의 회전

- 바람개비 image 추가
 - [app]-[drawable] 선택
 - Paste를 선택

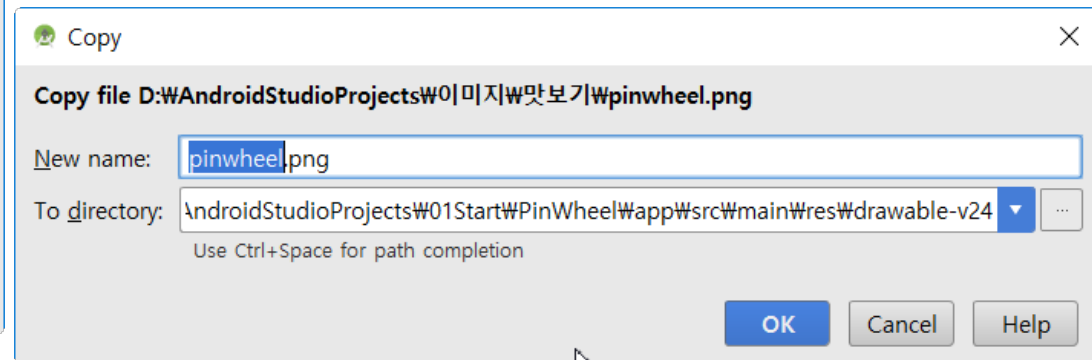
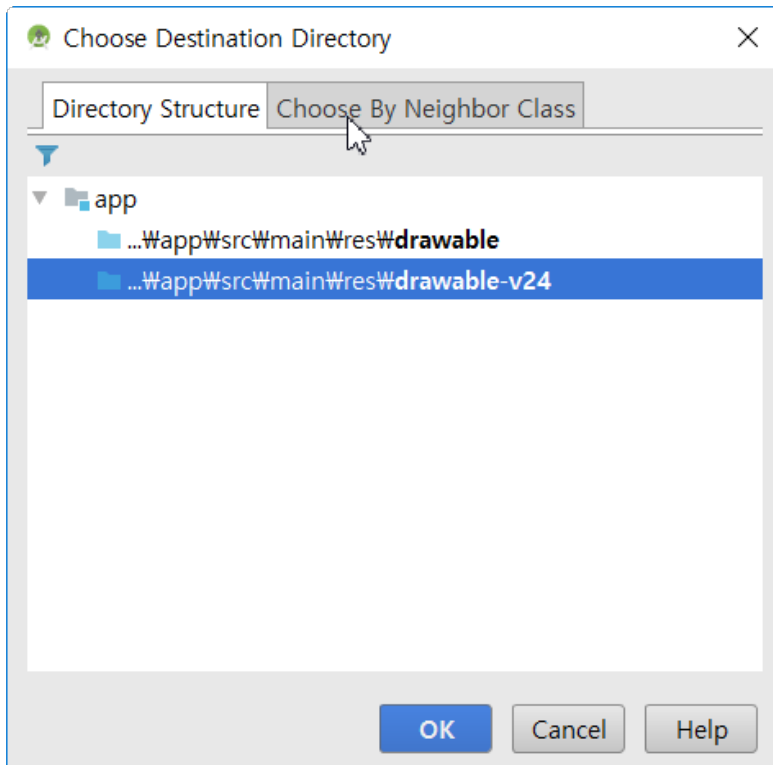




바람개비의 회전

■ 바람개비 image 추가

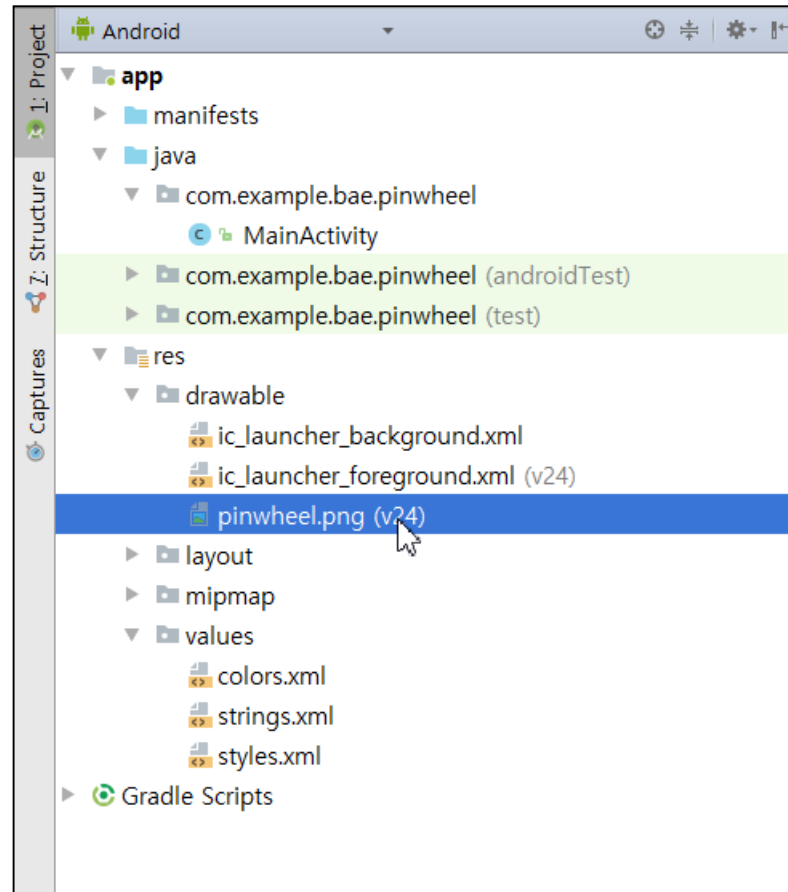
■ [Choose Destination Directory]





바람개비의 회전

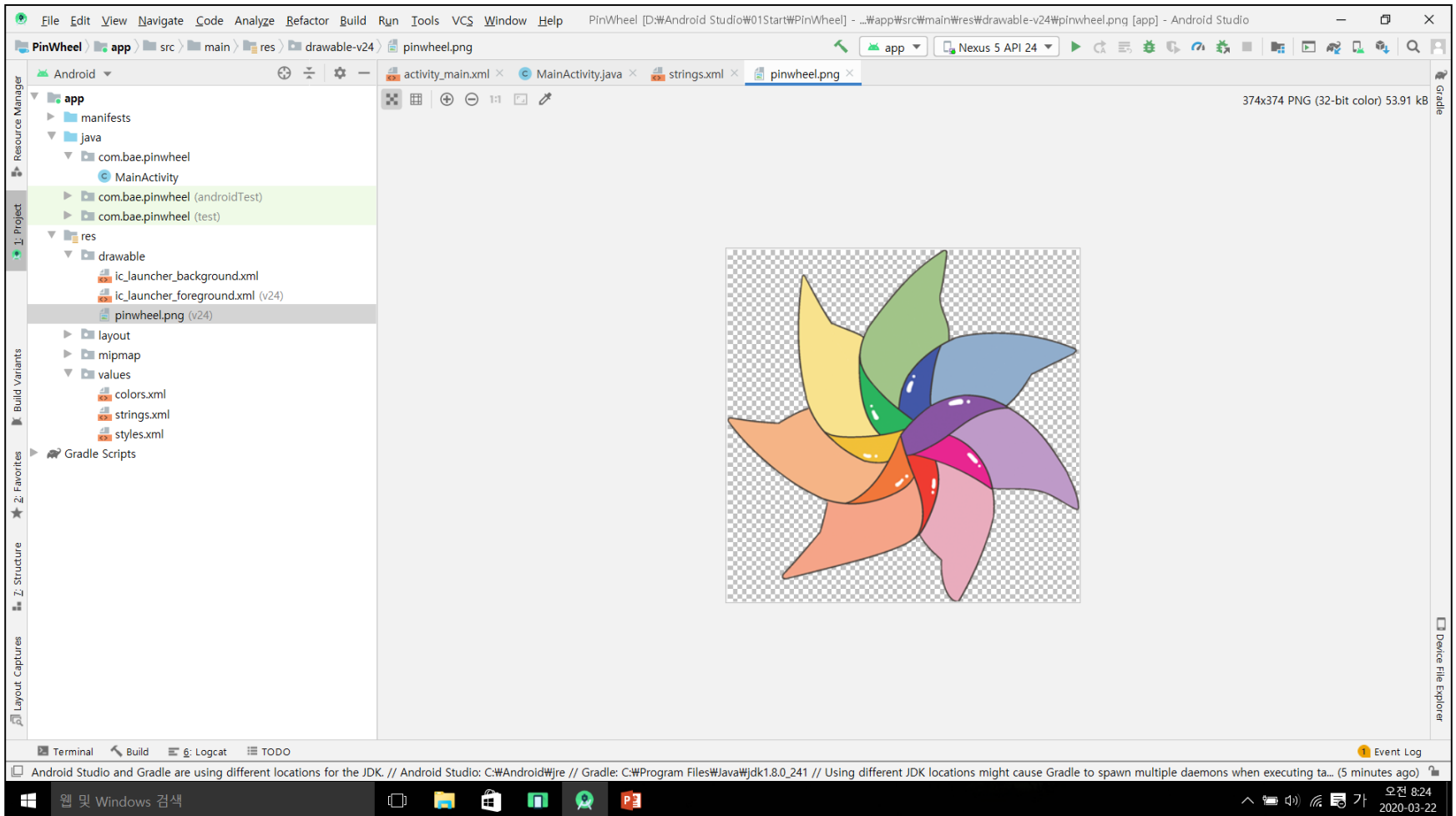
- 바람개비 image 추가
 - drawable 폴더에 image가 추가된 것을 확인





바람개비의 회전

바람개비 image 추가





바람개비의 회전



- 일반적인 애플리케이션 작성 절차
 - 사용자 인터페이스 작성(XML)
 - Layout XML 작성
 - String XML 작성
 - Color XML 작성
 - Dim XML 작성
 - 자바 코드 작성(JAVA)



바람개비의 회전

■ 바람개비 image 중앙 배치

■ [app]-[res]-[layout]-[activity_main.xml] 파일 선택

■ 마우스를 편집창에서 두 번째 줄에 위치 시킴

```
<?xml version="1.0" encoding="utf-8"?>
<androidx.constraintlayout.widget.ConstraintLayout xmlns:android="http://schemas
  xmlns:app="http://schemas.android.com/apk/res-auto"
  xmlns:tools="http://schemas.android.com/tools"
  android:layout_width="match_parent"
  android:layout_height="match_parent"
  tools:context=".MainActivity">

  <TextView
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="Hello World!"
    app:layout_constraintBottom_toBottomOf="parent"
    app:layout_constraintLeft_toLeftOf="parent"
    app:layout_constraintRight_toRightOf="parent"
    app:layout_constraintTop_toTopOf="parent" />

</androidx.constraintlayout.widget.ConstraintLayout>
```



바람개비의 회전

■ 바람개비 image 중앙 배치

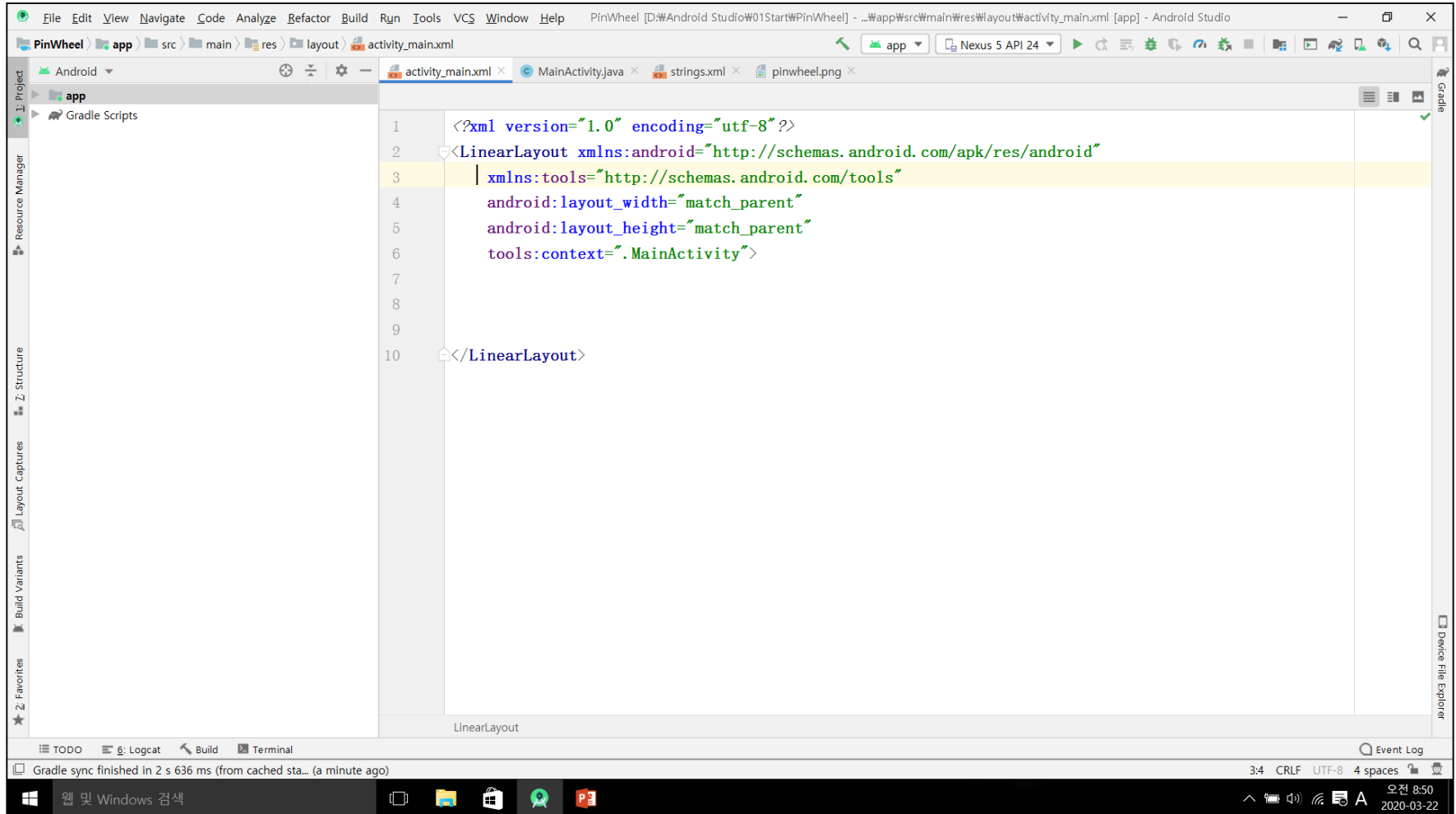
- android.support.constraint.ConstraintLayout을 선택
- “LinearLayout”으로 수정함

```
1 <?xml version="1.0" encoding="utf-8"?>
2 <androidx.constraintlayout.widget.ConstraintLayout xmlns:android="http://
3     LinearLayout
4     ListView
5     androidx.appcompat.widget.LinearLayoutCompat
6     FrameLayout
7     androidx.constraintlayout.widget.ConstraintLa...
8     GridLayout
9     TwoLineListItem
10    androidx.core.widget.ContentLoadingProgressBar
11    androidx.drawerlayout.widget.DrawerLayout
12    android.opengl.GLSurfaceView
13    AbsoluteLayout
14    app:layout_constraintLeft_toLeftOf="parent"
15    app:layout_constraintRight_toRightOf="parent"
16    app:layout_constraintTop_toTopOf="parent" />
17
18 </androidx.constraintlayout.widget.ConstraintLayout>
```



바람개비의 회전

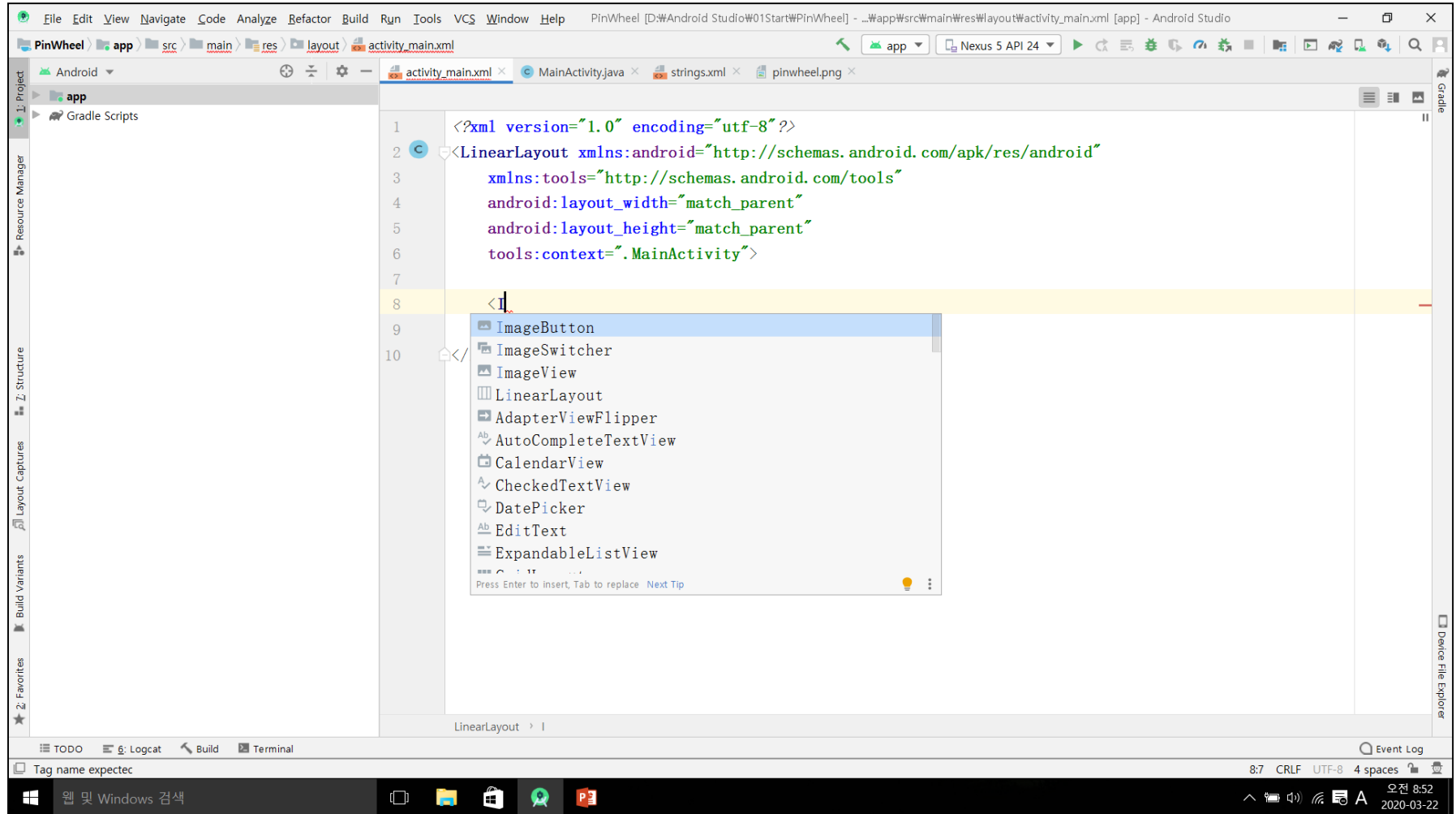
<TextView> 지우기





바람개비의 회전

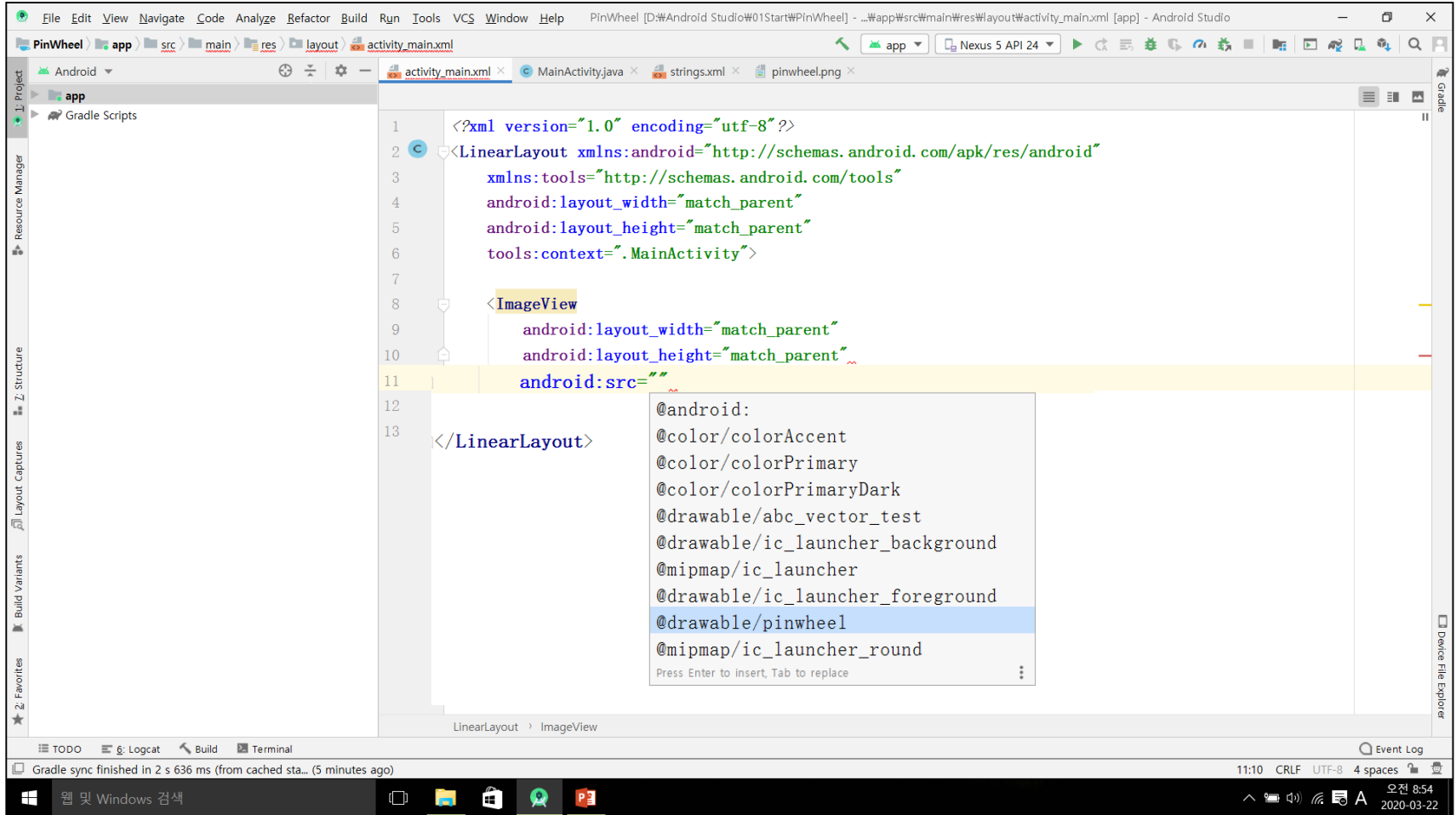
<ImageView> 추가





바람개비의 회전

Image src 속성 입력





바람개비의 회전



■ 바람개비 image 중앙 배치

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context=".MainActivity">

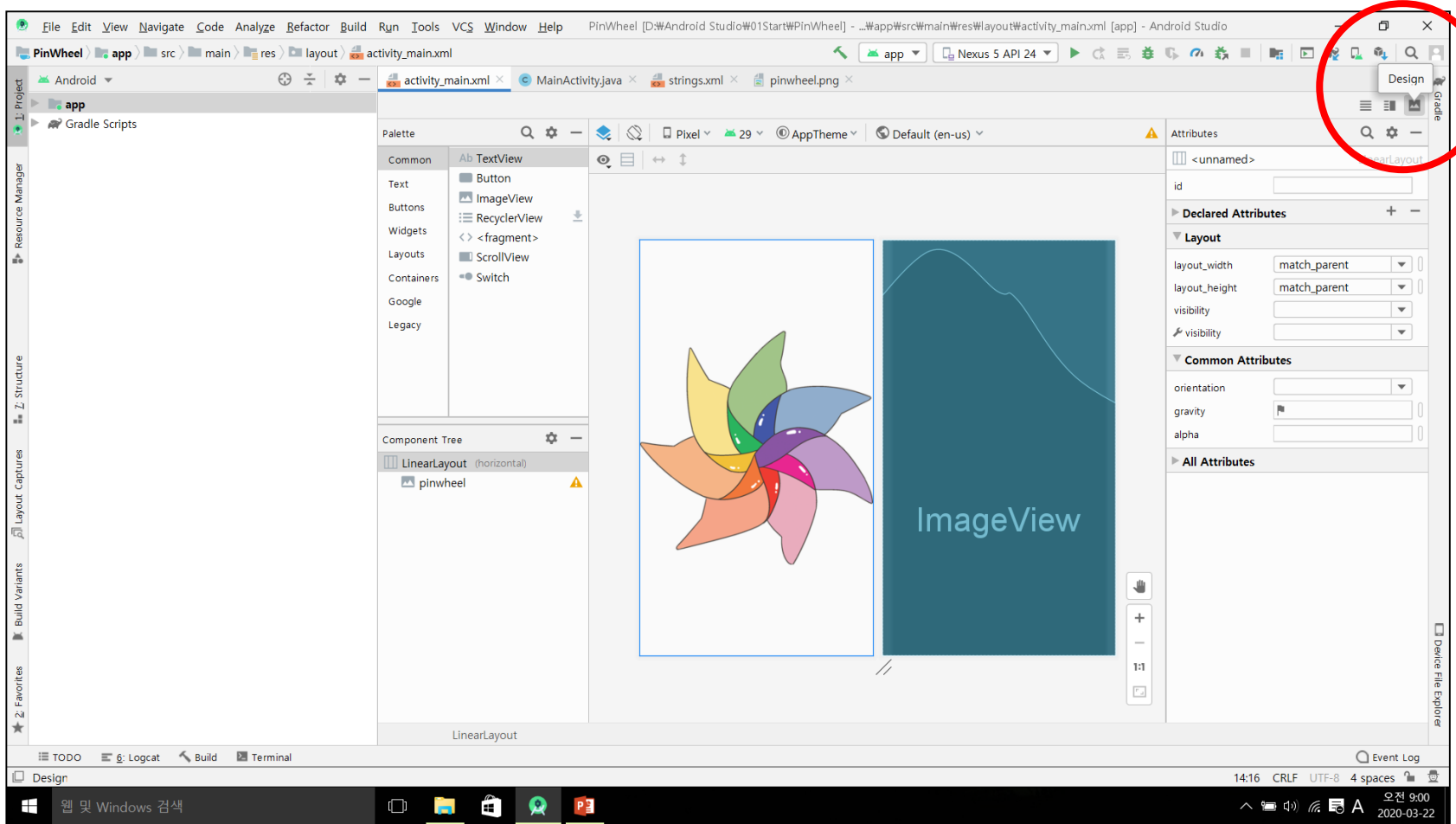
    <ImageView
        android:id="@+id/pinwheel"
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:src="@drawable/pinwheel" />

</LinearLayout>
```



바람개비의 회전

- 바람개비 image 중앙 배치
- [Design] 탭 선택

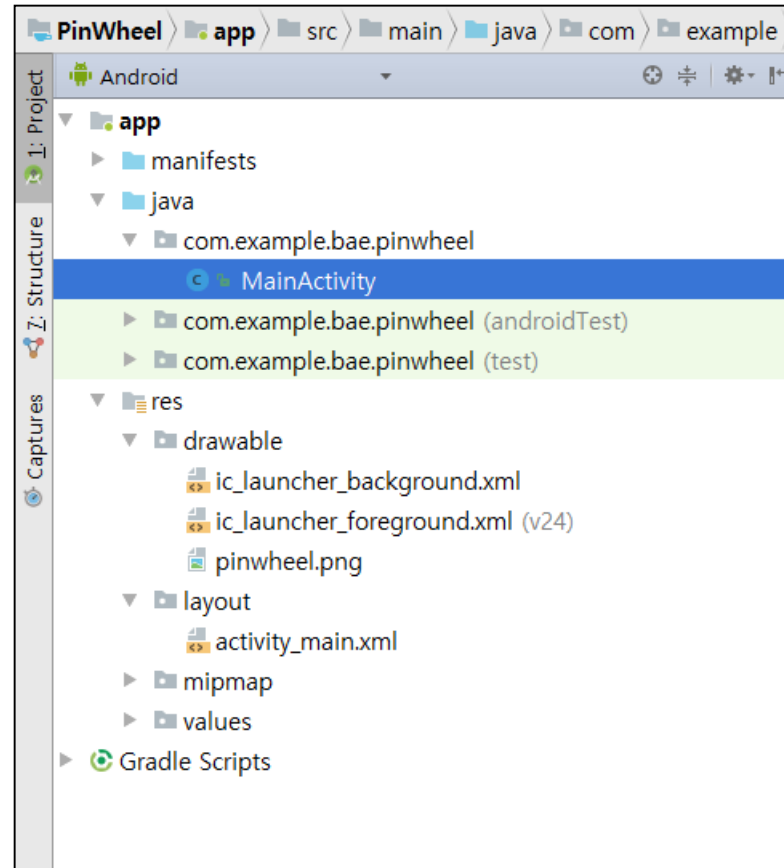




바람개비의 회전

■ JAVA 프로그램 작성

■ [app]-[java]-[MainActivity] 파일 선택





바람개비의 회전

■ MainActivity 클래스

MainActivity 자바 클래스가 호출될 때 처음 실행되는 메소드

```
public class MainActivity extends AppCompatActivity {  
  
    @Override  
    protected void onCreate(Bundle savedInstanceState) {  
        super.onCreate(savedInstanceState);  
        setContentView(R.layout.activity_main);  
    }  
}
```

액티비티 생성

activity_main.xml에서 정의된 화면 레이아웃을 액티비티에 출력



바람개비의 회전



- MainActivity 클래스
- 다음 코드 추가

```
ImageView imageView = findViewById(R.id.pinwheel);  
ObjectAnimator animator = ObjectAnimator.ofFloat(imageView, "rotation",360);  
animator.setInterpolator(new LinearInterpolator());  
animator.setDuration(2000);  
animator.setRepeatCount(ValueAnimator.INFINITE);  
animator.start();
```

- 오류 확인
- 실시간으로 확인 가능 (편집창 우측 상단)



바람개비의 회전



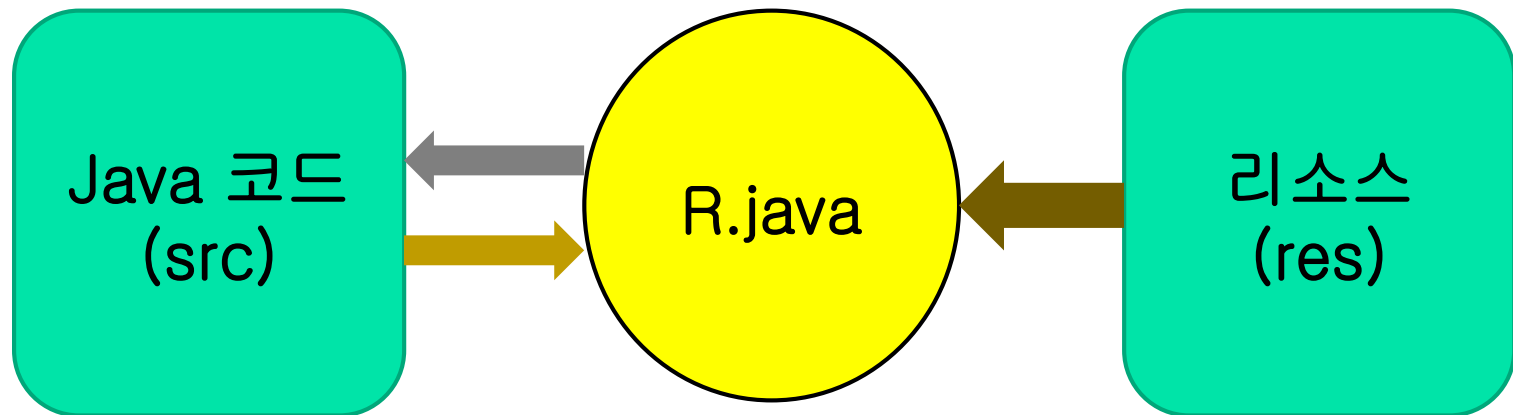
■ 화면 리소스 정의

- JAVA 코드에서 호출해 사용하려는 모든 element에는 반드시 “android:id” 속성으로 **id를 지정해야 함**
 - 반대로 JAVA 코드에서 사용하지 않으면 이름을 정할 필요가 없음
- 일반적으로 “@+id/...”와 같은 형태로 id를 지정하고, 중복되지 않는 유일한 문자열을 지정해야 함



바람개비의 회전

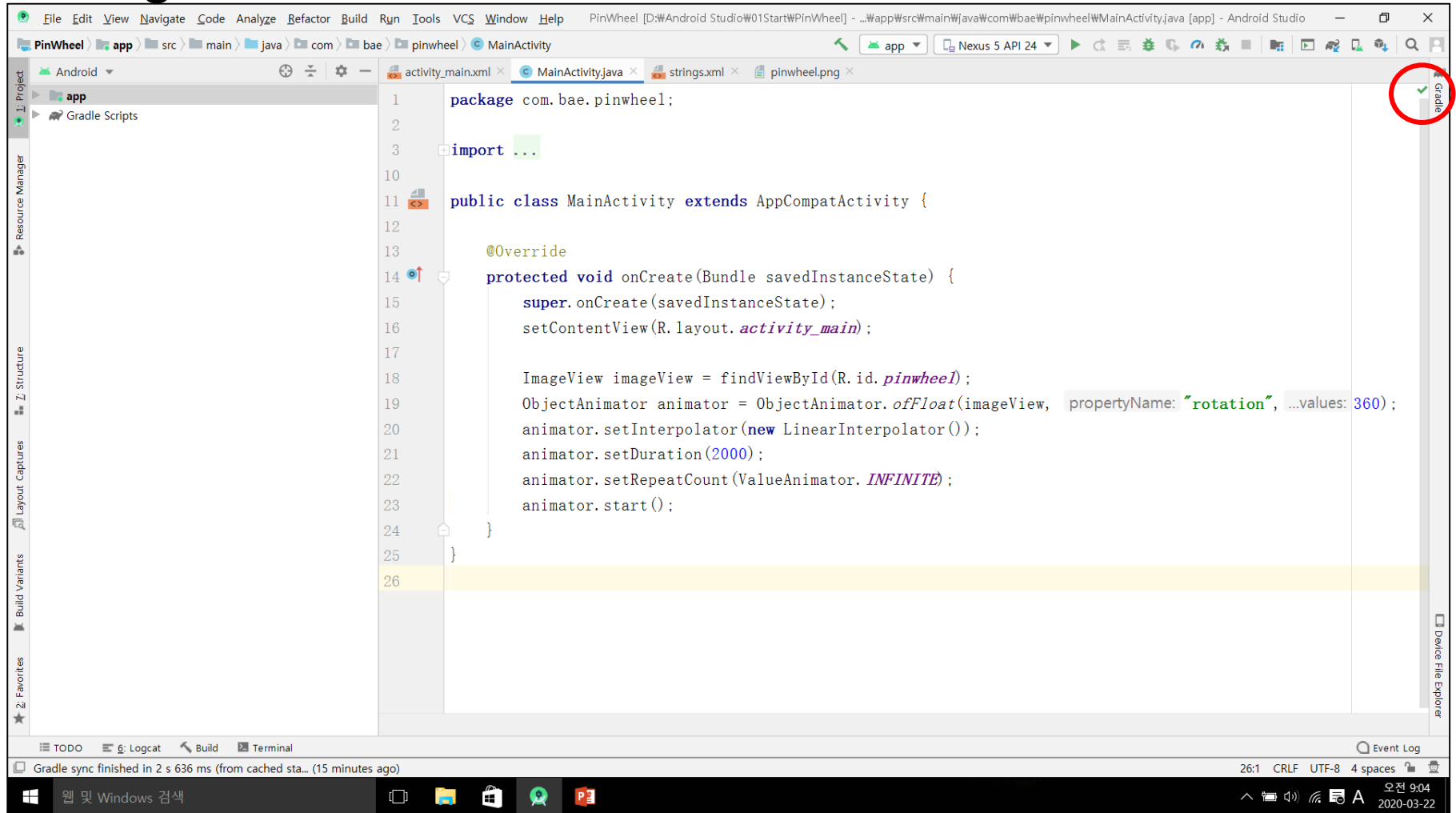
- 화면 리소스 JAVA 코드와 연결하는 방법
 - @+id로 지정한 항목을 찾아 오려면 JAVA 코드에서 findViewById() 메소드를 사용
 - XML 레이아웃 파일을 안드로이드 빌드 시스템이 분석해 JAVA 코드에서 호출해 쓸 수 있도록 자동으로 생성한 자바 코드 파일이 바로 R.java다
 - 모든 레이아웃 정보는 R.layout 변수를 통해 접근





바람개비의 회전

■ Program 작성 완성





바람개비의 회전

■ 자동 완성 기능 특징을 이용할 것

- `setOnClickListener` 메소드의 내용을 다 쳤다면
- 안드로이드 스튜디오의 장점은 자동완성에 있음
- `test_Button`(버튼 변수) 다음 `.`(점)을 치고 기다리면 자동 완성이 기능이 실행. 그러면 `setOnClickListener` 클릭 !! 하지만 기능오류로 덜 완성 되는데, 이때는 괄호 안에 `new`를 치고 `View`의 `V`를 치면 다시 자동완성 기능이 실행 되는데 맨 위 `OnClickListener`를 누르면 기본들이 완성되어 있음

```
super.onCreate(savedInstanceState);
setContentView(R.layout.activity_main);
test_textView.setText("안드로이드");
test_Button.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View v) {
        test_textView.append("안드로이드");
    }
});
```


```
super.onCreate(savedInstanceState);
setContentView(R.layout.activity_main);
test_textView=(TextView)findViewById(R.id.test_textView);
test_Button=(Button)findViewById(R.id.test_Button);

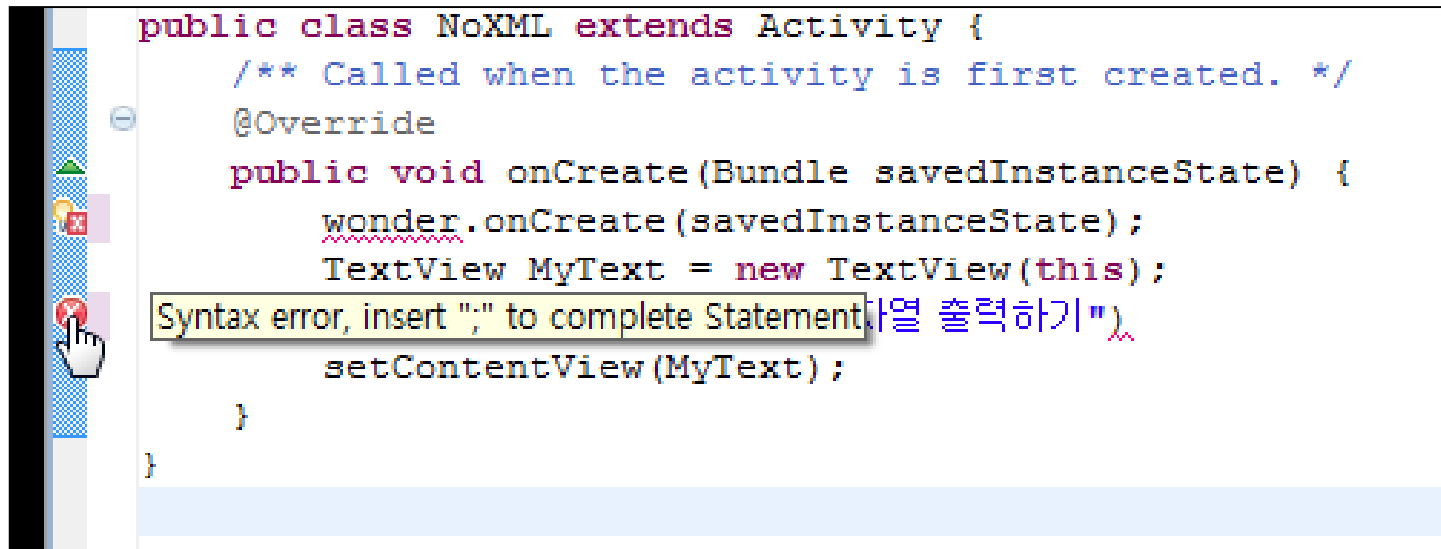
test_Button.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View v) {
        test_Button.setText("안드로이드");
    }
});
```



바람개비의 회전

■ Program Compile

- Android Studio는 소스가 편집될 때마다 주기적으로 백그라운드 컴파일을 수행하고, 구문의 타당성 점검, 그 결과를 편집창에 표시함
- 에러 발생 시 편집기 왼쪽에  표시가 나타나며, mouse over 시 에러에 대한 상세 설명을 나타냄



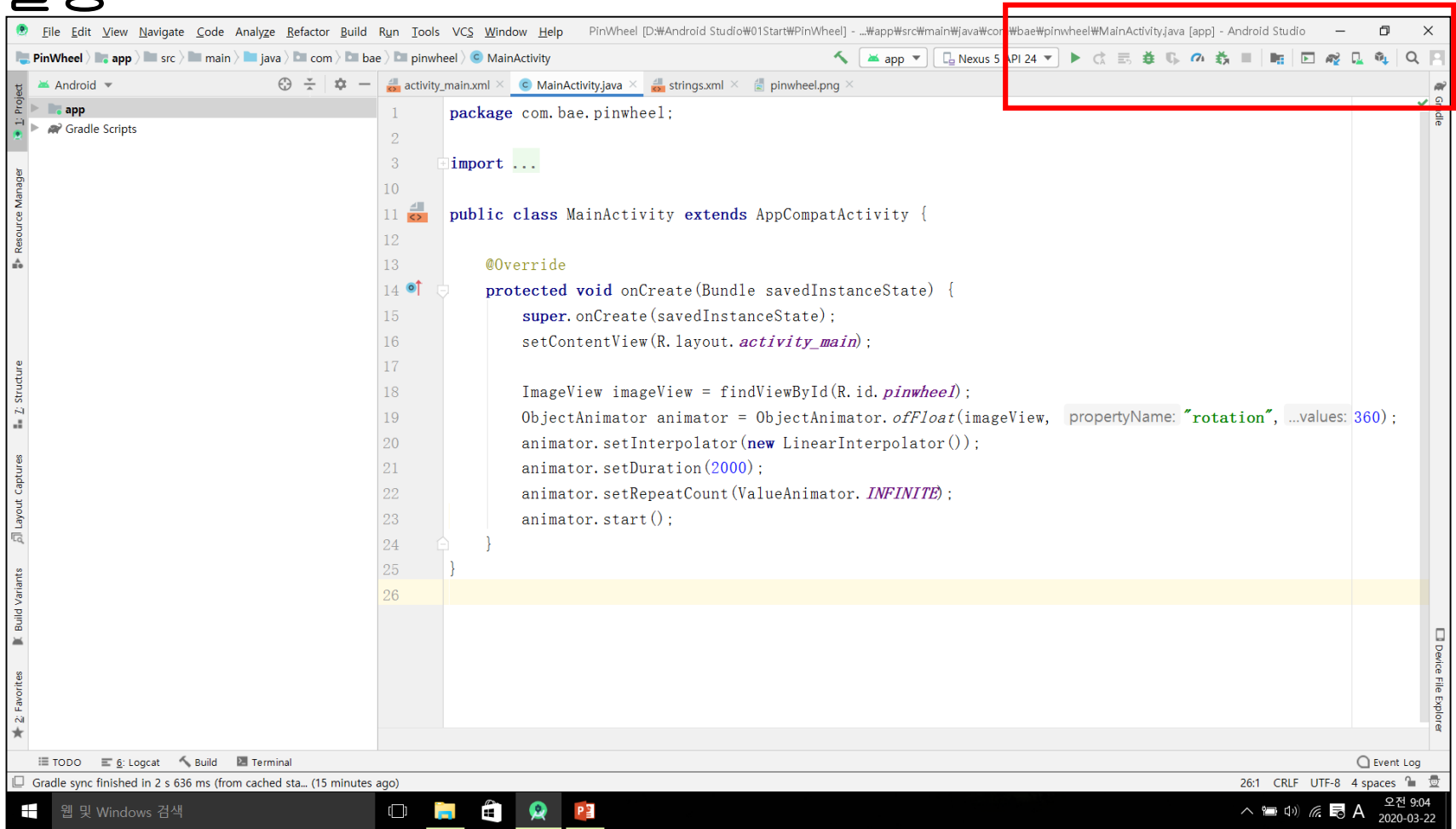
```
public class NoXML extends Activity {  
    /** Called when the activity is first created. */  
    @Override  
    public void onCreate(Bundle savedInstanceState) {  
        wonder.onCreate(savedInstanceState);  
        TextView MyText = new TextView(this);  
        setContentView(MyText);  
    }  
}
```

Syntax error, insert ";" to complete Statement



바람개비의 회전

■ 실행





바람개비의 회전



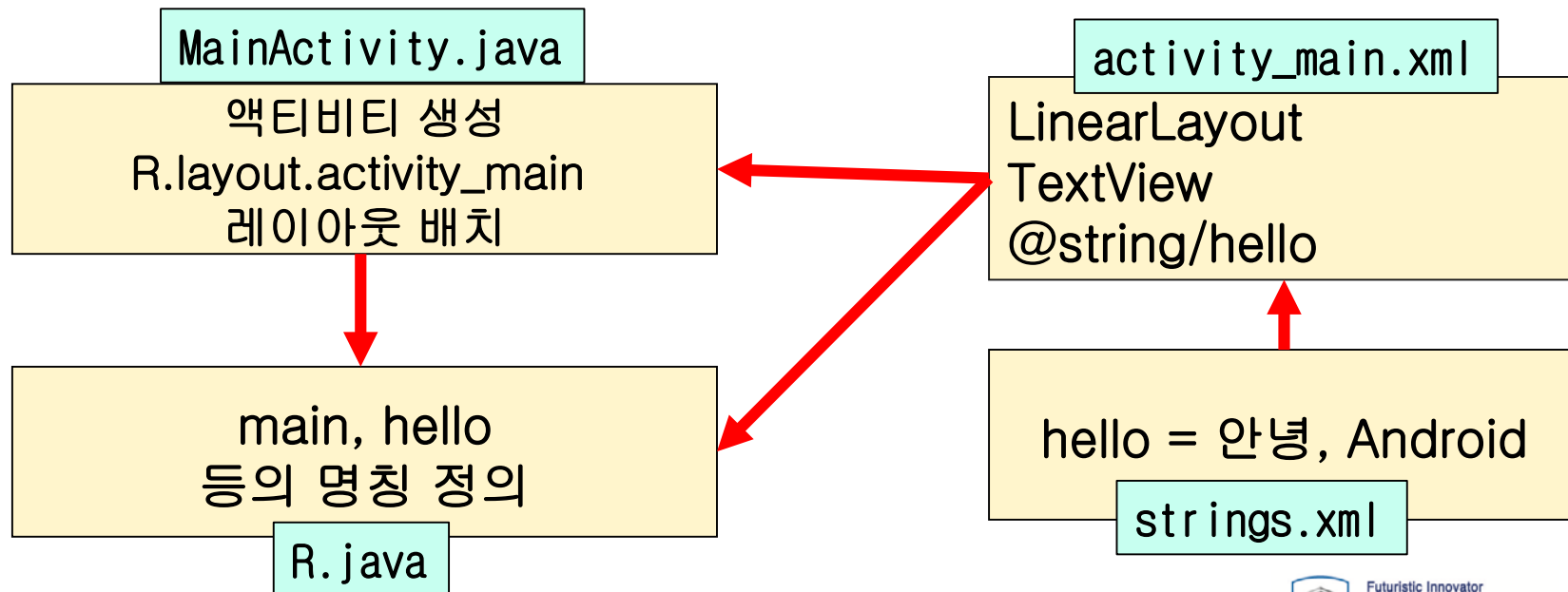
■ 프로그램 실행

- Android에는 main()이 없음
- Activity별로 실행됨
- Android 애플리케이션이 실행되면 MainActivity.java가 실행 (이것은 AndroidManifest.xml에 정의되어 있음)
 - Activity를 상속받은 MainActivity 클래스가 실행되면서 onCreate() 메소드가 가장 먼저 실행
 - onCreate() 메소드에서 setContentView() 메소드는 화면에 보여줄 View를 등록함
 - 여기서는 R.layout.activity_main
 - R.layout.activity_main이므로, res/layout 폴더의 activity_main.xml을 MainActivity 클래스의 View로 보여짐



바람개비의 회전

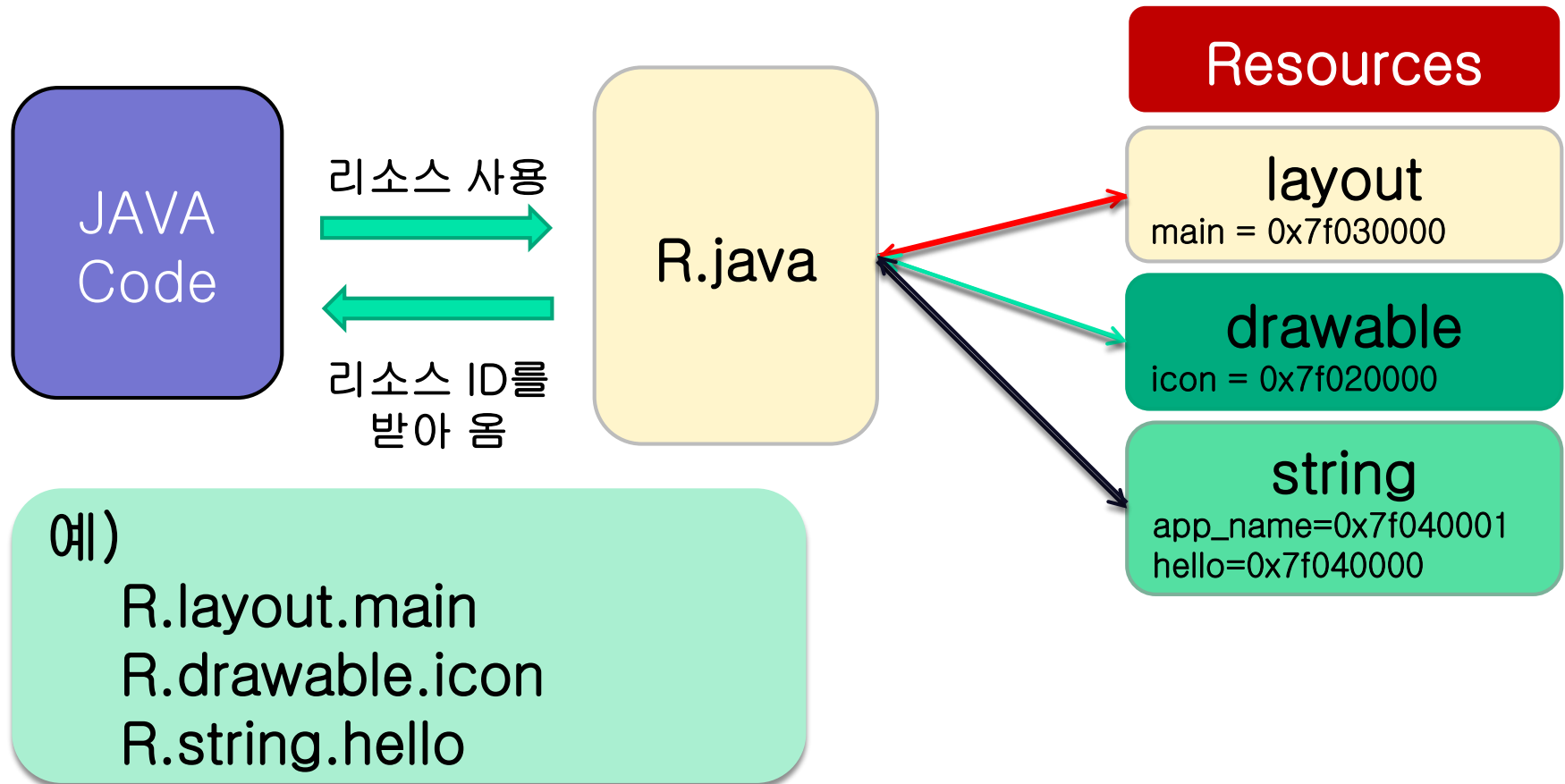
- strings.xml : hello 문자열 정의
- activity_main.xml : 정의된 문자열을 Linear Layout의 Text View에 넣음
- MainActivity.java : setContentView 메서드 호출, 레이아웃을 액티비티에 배치
- R.java : 각 파일들이 참조하는 객체의 ID 정의





바람개비의 회전

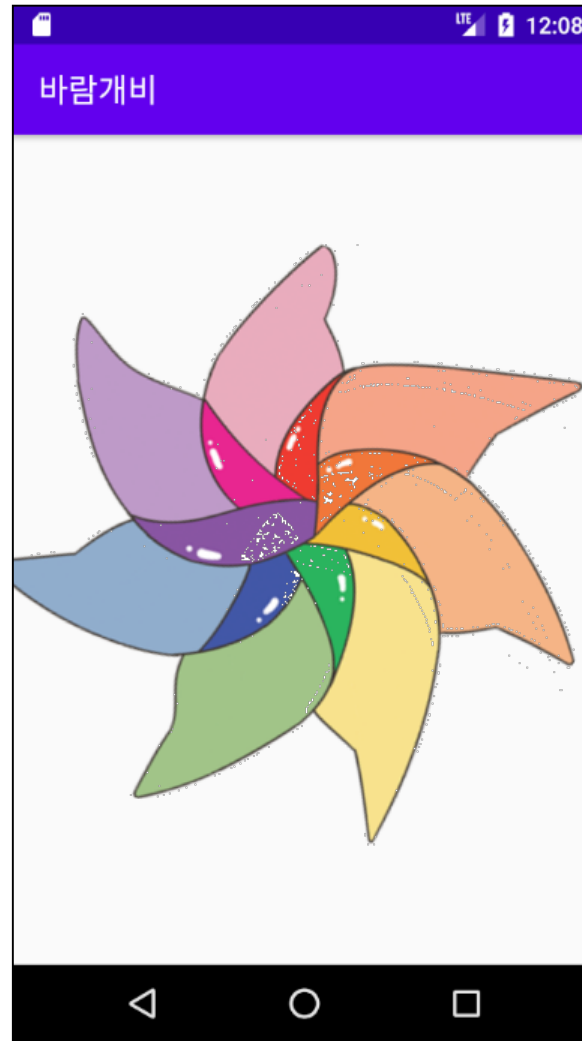
■ R.java를 이용한 리소스 접근





바람개비의 회전

■ 실행





추가



■ Android Title Bar 없애기

```
requestWindowFeature(Window.FEATURE_NO_TITLE);
```

■ Android Status Bar 없애기

```
getWindow().setFlags(WindowManager.LayoutParams.FLAG_FULLSCREEN, WindowManager.LayoutParams.FLAG_FULLSCREEN);
```