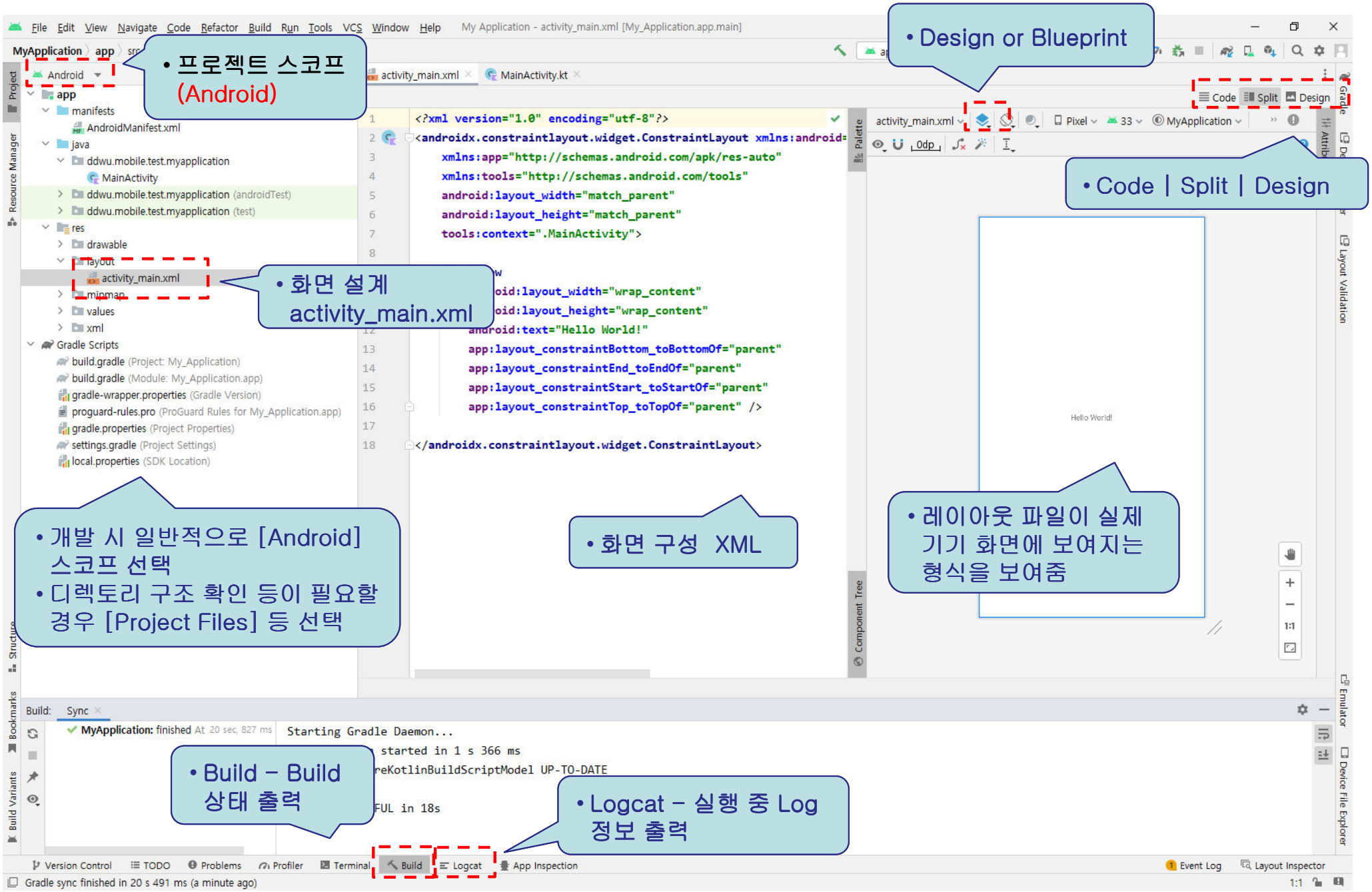
안드로이드 APP 개발 기초

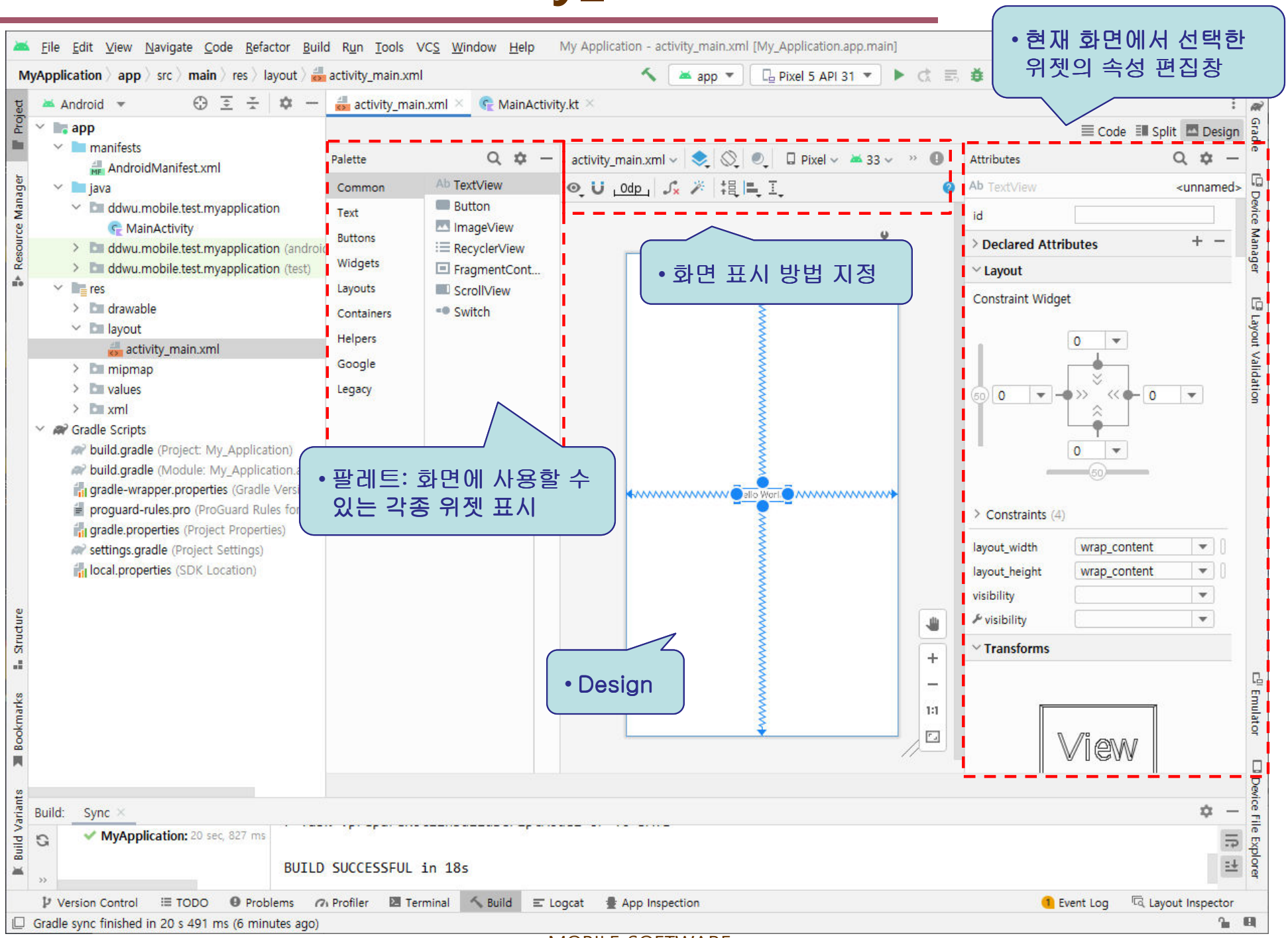
- http://developer.android.com

목 차

초기 화면 예 – activity_main.xml 01

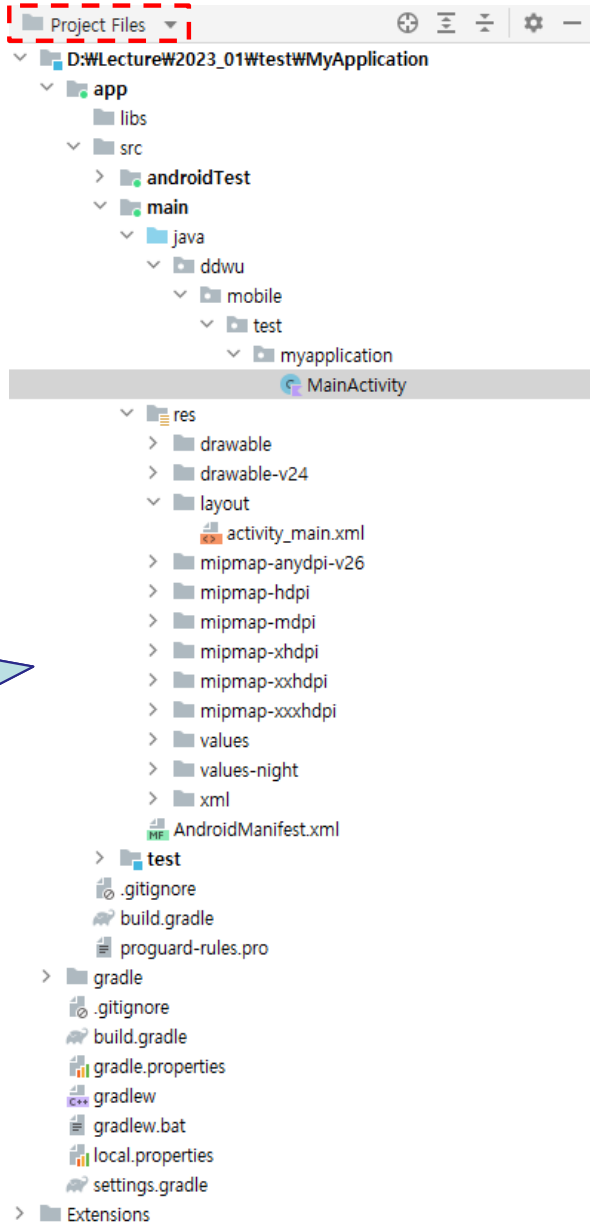


초기 화면 예 – activity_main.xml 02



안드로이드 APP 프로젝트 구조 01

프로젝트 스코프 예: [Android] or [Project Files]



• 개발 중 필요에 따라 변경하며 구성 확인

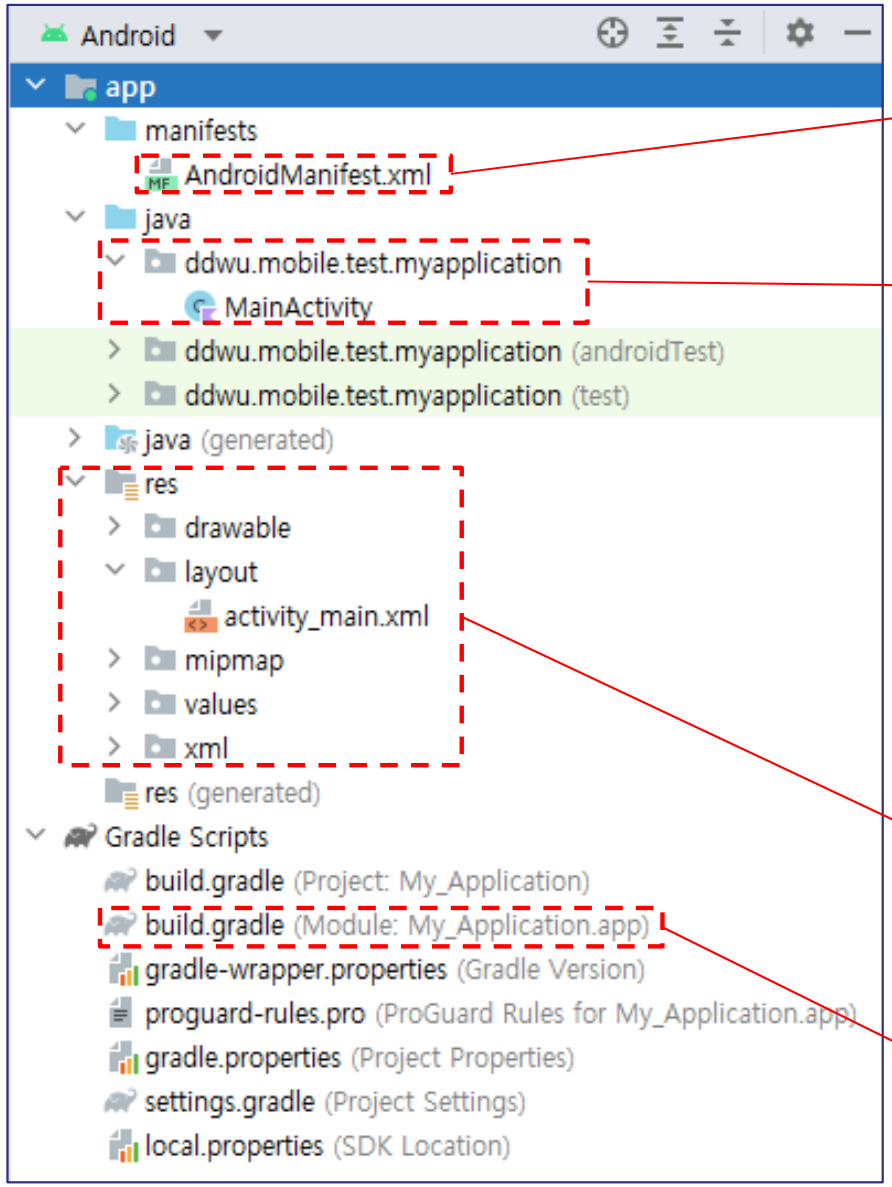
안드로이드 APP 프로젝트 구조 02

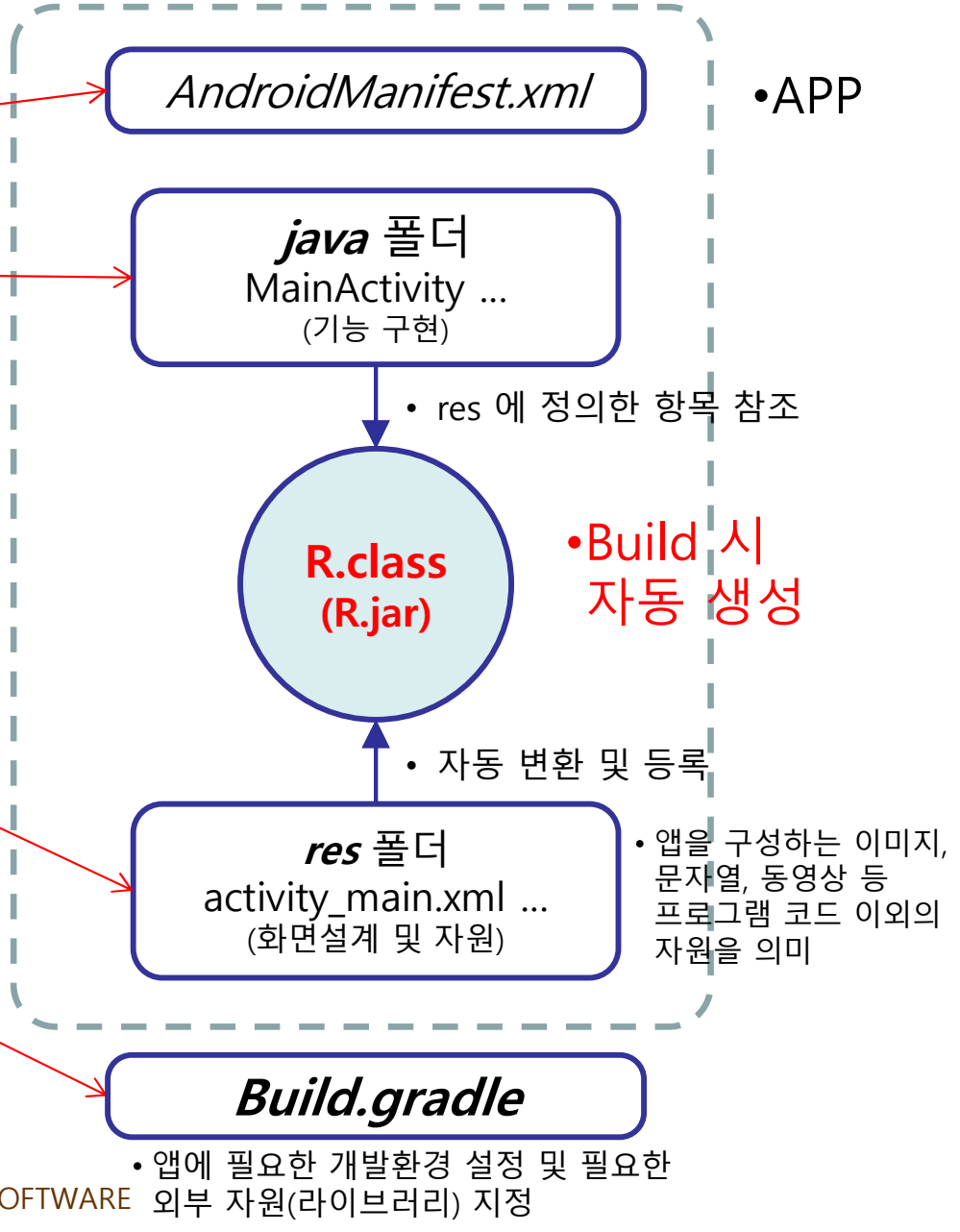
주요 구성 요소

파일 또는 폴더명		설명
AndroidManifest.xml		안드로이드 앱 설정 정보 파일 앱의 구성요소 및 속성 등을 정의 (시작 액티비티 지정)
패키지/MainActivity (.kt)		자동으로 추가하는 기본 액티비티의 구현(코틀린) 파일 → 앱 화면의 기능을 구현
res 폴더 (앱 사용 자원 지정)	**layout/activity_main.xml**	자동으로 추가하는 기본 액티비티의 레이아웃 XML 파일 → 앱 화면의 모양을 표현
	menu 폴더	앱에서 사용할 메뉴의 구성 정의
	mipmap 폴더	앱에서 사용하는 아이콘 이미지 파일을 화면 밀도(Density)별로 생성 (새로 추가됨)
	drawable 폴더	앱에서 사용할 이미지 파일을 저장
	values 폴더	앱에서 사용할 값(문자열, 수치값, 색상값 등)을 XML 파일로 저장
build.gradle (Module: ...)		앱을 빌드하기 위한 설정정보 파일 필요한 외부 라이브러리 등의 정보를 기록

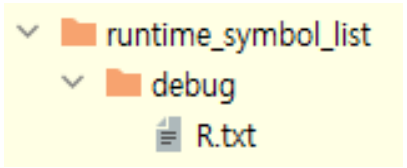
프로젝트 주요 항목들 사이의 관계

Build 시 R.class 생성



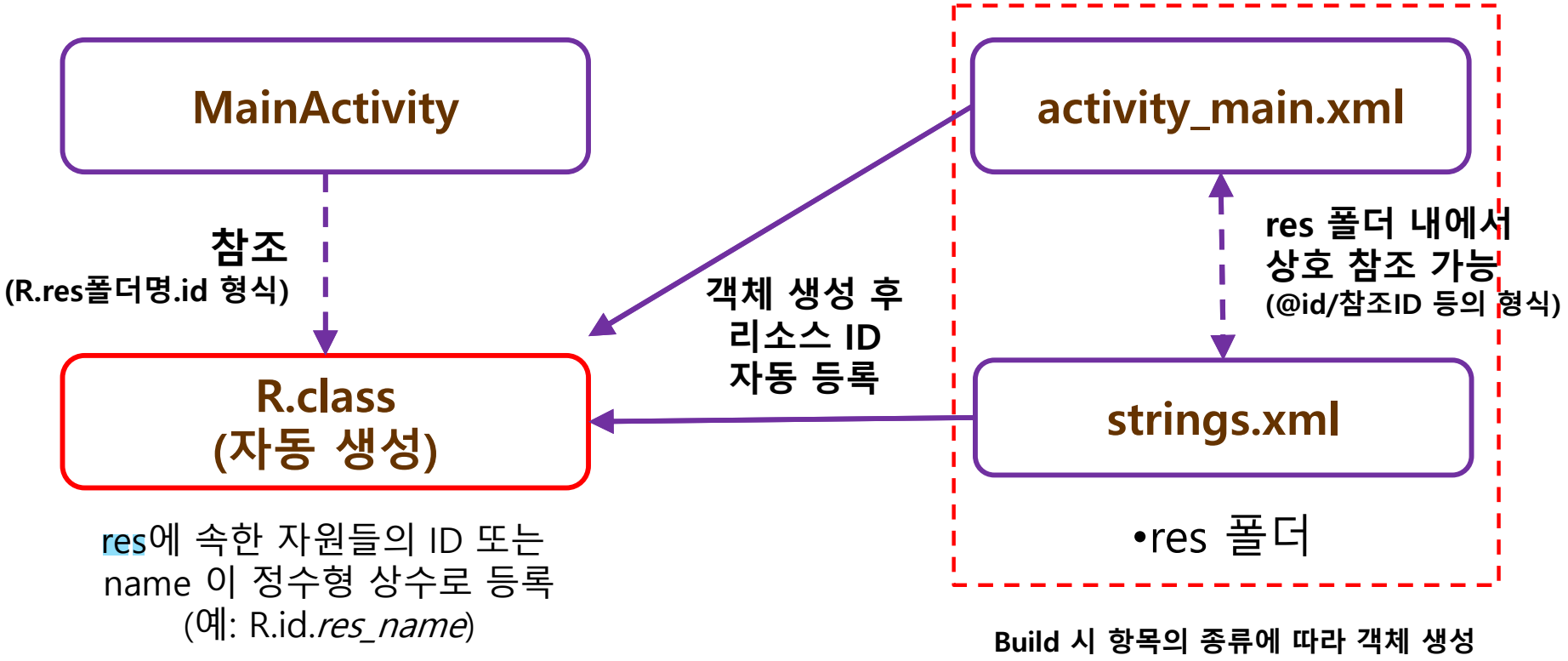


Resource 의 사용



R.class

- 화면에 지정한 자원 등은 소스코드에서 참조하여야 함
- 빌드 시 res 폴더의 요소들은 적절한 Java 클래스의 객체로 자동 변환 및 생성 (raw 폴더의 이미지 등 일부 제외)
- 생성한 Java 객체의 ID(정수형 상수)는 R.class 에 등록
- 소스코드(MainActivity 등)에서 R.class 의 ID 를 사용하여 res 의 항목에 접근



XML을 활용한 화면 구현의 장점

- 코드만을 사용해서 화면 구현도 가능

```
override fun onCreate(savedInstanceState: Bundle?) {
    super.onCreate(savedInstanceState)
    // setContentView(R.layout.activity_main)
    val textView : TextView = TextView( context: this)
    textView.setText("코드로 문자열 출력")
    setContentView(textView)
}
```

- XML 화면 구현의 장점
  - 개발자와 디자이너의 분담 작업 용이
  - 교체 가능하므로 호환성 확보, 국제화에 유리
  - 변경된 모듈만 컴파일하므로 개발 속도 개선
  - 레이아웃 재활용 가능

- 정적인 레이아웃은 XML로, 동적인 레이아웃은 코드로 구현

실습

- 실제 기기를 연결하여 앱 디버깅을 활성화 하시오.

- 실습 공통
  - 패키지명: ddwucom.mobile.week01.test03
  - 저장위치: c:₩work₩class01₩week01₩test03₩프로젝트명

- 다음과 같은 프로젝트를 생성한 후 레이아웃에 위젯들을 자유로이 배치하는 앱을 개발 및 실행 하시오.
  - 프로젝트명: exam01

- 프로젝트를 레이아웃 없이 생성한 후 코드에서 레이아웃을 작성하는 방식으로 앱을 개발 및 실행 하시오.
  - 프로젝트명: exam02