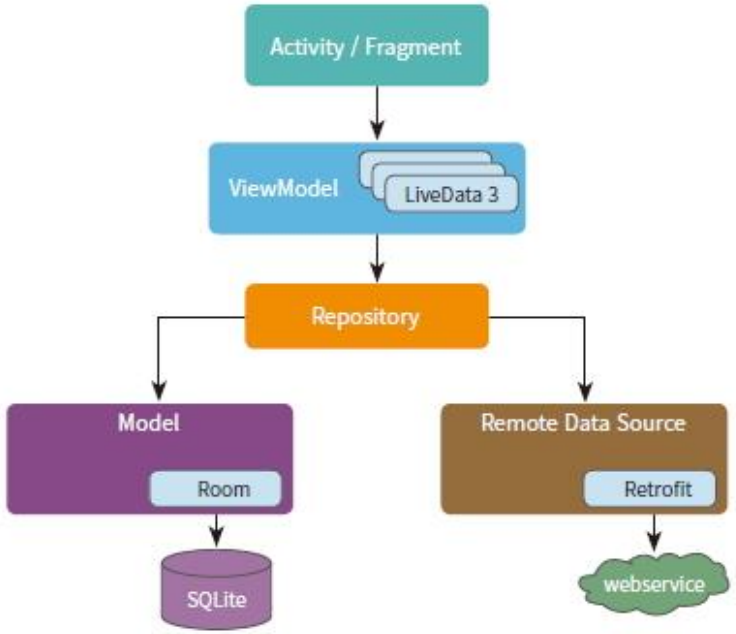
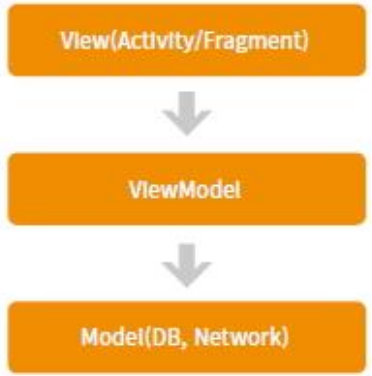
28장. 뷰모델과 라이브데이터

28.1 뷰모델

28.1.1 AAC 와 뷰모델

업무 처리와 화면을 분리해 개발
모델과 뷰를 직접 연결하지 말고 중간에 뷰모델을 두어 뷰에서 뷰모델에 일을 시키면 뷰모델은 모델을 이용하는 구조





28.1 뷰모델

28.1.2 ViewModel 활용

뷰모델은 ViewModel 클래스를 상속받아 작성

```
public class Test1ViewModel extends ViewModel {
MutableLiveData<User1> user;
public MutableLiveData<User1> getUser() {
if (user == null) {
user = new MutableLiveData<User1>();
user.postValue(new User1("gildong", "hong"));
}
return user;
}
}
```

뷰모델을 액티비티에서 이용

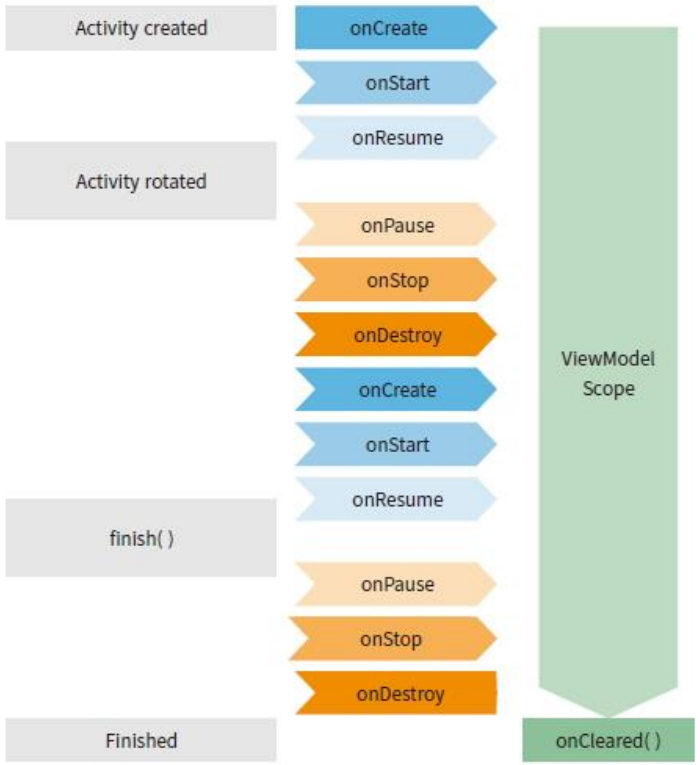
```
Test1ViewModel model = ViewModelProviders.of(this).get(Test1ViewModel.class);
model.getUser().observe(this, user -> {
//do something…
});
```

28.1 뷰모델

뷰 모델의 생명주기

ViewModelProviders.of (this).get ()으로 생성한 뷰모델 객체는 액티비티 스코프 내에서 싱글턴 (singleton)으로 유지

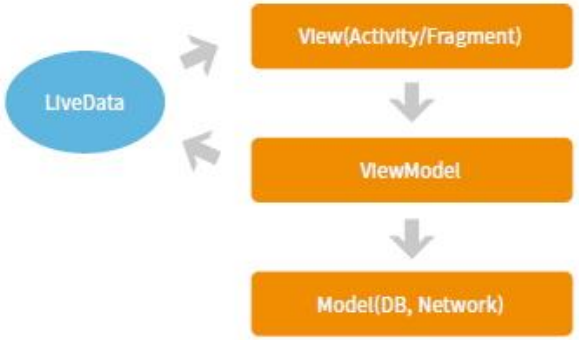
ViewModelProviders.of (this).get ()에 의해 뷰모델 객체가 생성되면 액티비티의 화면 회전과 같은 상황에서도 뷰모델 객체는 소멸되지 않고 그대로 유지



28.2 라이브데이터

28.2.1 라이브데이터와 옵서버

뷰모델이 데 이터를 액티비티에 전달할 때 라이브데이터를 사용



뷰모델

```
public MutableLiveData<Integer> getLiveData(){
MutableLiveData<Integer> liveData = new MutableLiveData<Integer>();
liveData.setValue(Integer.valueOf(10));
return liveData;
}
```

액티비티

```
viewModel.getLiveData().observe(this, result -> {
Log.d("kkang","observe....." + result);
});
```

28.2 라이브데이터

28.2.2 커스텀 라이브데이터

LiveData 클래스를 상속받아 직접 커스텀 라이브데이터 클래스를 만들어 이용 가능

```
public class MyLiveData extends LiveData<Integer> {
public void setData(int data){
postValue(data);
}
}
```

```
MyLiveData liveData= new MyLiveData();
liveData.observe(this, result -> {
Log.d("kkang", "result : " + result);
});
```

28.2 라이브데이터

28.2.3 postValue() vs setValue()

postValue ( ), setValue ( ) 두 함수 모두 뷰모델에서 결과 데이터를 라이브데이터에 담는 역 할
postValue ( ) 함수는 백그라운드 스레드에 의해 동작하 고 setValue ( ) 함수는 메인 스레드에 의해 동작
액티비티에서 라이브데이터를 백그라운드 스레드에서 이용하게 되면 차이 발생

```
new Thread(new Runnable() {
@Override
public void run() {
viewModel.get_postValue().observe(Main4Activity.this, result -> {
Log.d("kkang","background thread.. postValue :" + result);
});
}
}).start();
// 에러
new Thread(new Runnable() {
@Override
public void run() {
viewModel.get_setValue().observe(Main4Activity.this, result -> {
Log.d("kkang","background thread.. setValue :" + result);
});
}
}).start();
```

postValue( ) 함수를 이용한 뷰모델의 함수 호출에는 별문 제가 없지만 setValue ( ) 함수를 이용한 뷰모델 함수는 런타임 때 에러가 발생
액티비티에서 옵서버를 이용하지 않고 뷰모델의 데이터를 직접 받는다면 postValue( ) 함수에 의한 결과는 받을 수 없

Step by Step 실습 – 28-1. 뷰모델, 라이브데이터

- 그레이들 작업
- AndroidManifest.xml
- MyViewModel.java
- MainActivity
- 실행