Android LiveData library

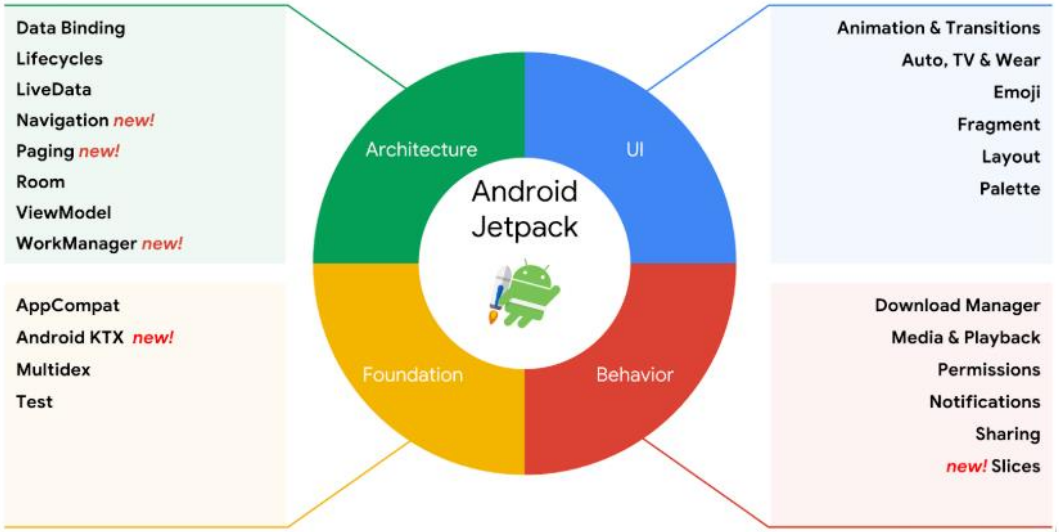
JEON YONGTAE
https://github.com/yongtaii/yongapps

1

Android LiveData

LiveData

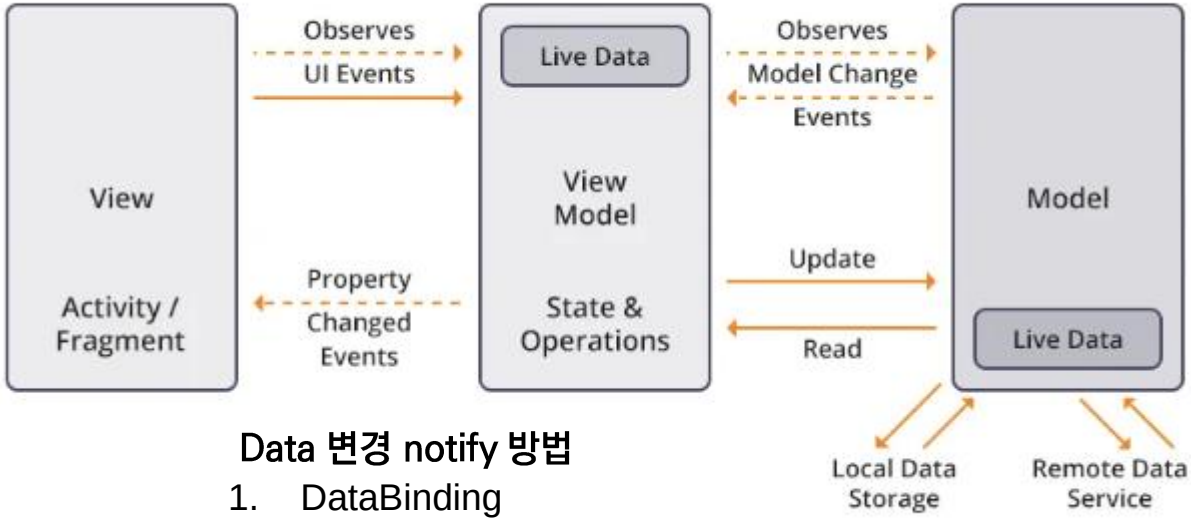
JetPack component란?



JetPack 이란 ?

- 구글 IO에서 62개정도의 작은 세션들을 공개했습니다 그 세션들의 집합
- Android앱을 손쉽게 개발하도록 지원하는 android 소프트웨어 구성요소 컬렉션
- JetPack 컴포넌트로 상용구코드를 작성하지않고, 복잡한 작업을 간소화 시킵니다.

MVVM - LiveData



Data 변경 notify 방법

1. DataBinding
2. RxJava
3. Live Data

LiveData - Observable

- LiveData는 관찰가능한 (Observable) 데이터 홀더 클래스다.
- View에서의 변화를 관찰하고 View를 업데이트 한다
- ViewModel 홀로 사용한다면, View를 변경해야 할 때마다 multiple calls를 만들어줘야 하기 때문에 tedious & costly operation 을 유발한다.
- LiveData is based on Observer Pattern. 따라서 이는 ViewModel & View 의 communication을 쉽게 만들어준다
- LiveData는 Multiple Call 대신, Data변화를 자동으로 감시하고 업데이트 해준다

LiveData - Lifecycle-Aware

- LiveData는 일반적인 Observerble과는 다르게 Activity, Fragment, Service와 같은 앱 컴포넌트의 생명주기를 인식하고 잇다.
- 따라서, 현재 활성화 된 생명주기에 있는 앱 컴포넌트의 Observer만 업데이트 하도록 보장한다

LiveData VS RxJava

- RxJava와 LiveData는 LiveData가 LifeCycle을 알고 있다는 것을 제외하면 비슷하다.
- LiveData는 View가 Background에 있을 때에는 Data를 View에 업데이트 하지 않는다. 따라서 IllegaStateException 을 피할 수 있게 해준다

LiveData Benefits

- Data & UI 동기화 : 앱 생명주기에 따라 매번 UI를 갱신할 필요가 없고, 옵저버 객체 내에서 UI 갱신하는 것으로 통합할 수있다.
- 메모리 릭 방지 : Observer들은 Lifecycle 객체와 바인드 되고, 그 Lifecycle 객체가 destory 상태가 되면, 자동적으로 지워진다
- 정지된 액티비티들 때문에 Crash날 일이 없다 : 액티비티가 백스택에 있을 때와 같이 옵저버 생명주기가 비활성화 상태일 때에는 어떠한 LiveData 이벤트도 받지 않는다
- 직접 생명주기를 핸들링할 필요가 없다 : RESUME, STOP을 신경쓸 필요가 없다

LiveData Benefits

- 항상 최신 데이터를 유지 : 생명주기가 활성화가 되었을 때, UI 컴포넌트는 최신의 데이터를 받는다.
- Configuration이 변경 되었을 때 (화면 회전 등) 적절하게 대응한다.

LiveData 객체 생성

- LiveData 는 List같은 Collection 구현체를 포함하여 모든 데이터와 함께 사용될 수 있는 래퍼 클래스다
- LiveData는 일반적으로 ViewModel 인스턴스 안에 저장되고, 아래처럼 Getter 메서드를 통해 접근하도록 구현한다

```
public class NameViewModel extends ViewModel {

    // Create a LiveData with a String
    private MutableLiveData<String> mCurrentName;

    public MutableLiveData<String> getCurrentName() {
        if (mCurrentName == null) {
            mCurrentName = new MutableLiveData<String>();
        }
        return mCurrentName;
    }

    // Rest of the ViewModel...
}
```

LiveData 객체 생성

UI를 업데이트하는 LiveData 인스턴스가 Activity나 Fragment가 아닌 ViewModel에 저장되는 이유

1. Activity / Fragment 내부에 코드가 모이는 상황을 피하기 위해 - UI 컨트롤러는 데이터의 상태를 유지하는 역할을 몰라도 된다. 주어진 데이터를 잘 보여주기만 하면 된다.

2. LiveData를 특정 Activity나 Fragment 인스턴스에 종속시키지 않고, 구성변경으로부터 LiveData 인스턴스가 살아있도록 하기 위해

Update View

- Activity에 Observer를 등록할 때, onChanged() 메서드를 override할 필요가 있다
- onChanged()는 LiveData가 변경될 때 마다 trigger 된다
- 따라서 onChanged() 에서 우리는 변경된 LiveData를 View에 Upate할 수 있다
- LiveData는 View가 Background에 있을 때에는 Data를 View에 업데이트 하지 않는다. 따라서 IllegaStateException 을 피할 수 있게 해준다

LiveData는 단지 Observer에 변화를 Notify 해주는 Data Type 이다.
LiveData is like a data chagned notifier.

LiveData 객체 관찰

대부분의 경우, 앱 컴포넌트의 onCreate() 메서드가 LiveData의 관찰을 시작하기 적절한 위치이다

1. onResume() 메서드와 같이 중복으로 호출되지 않음을 보장한다

2. Activity나 Fragment 가 활성화 되자마자 화면을 표시할 데이터가 있는지를 확인할 수 있다.

LiveData 객체 갱신

- MutableLiveData 클래스는 공개적으로 setValue(T), postValue(T) 메서드를 노출하고, LiveData안에 저장된 데이터를 수정할 필요가 있다면, 반드시 해당 메서드를 통해 수정해야한다.
- 수정 가능한 MutableLiveData는 ViewModel 내부에서 사용되고, ViewModel은 수정 불가능한 LiveData로 관찰자에게 노출한다.

LiveData 객체 갱신

```
mButton.setOnClickListener(new OnClickListener() {
    @Override
    public void onClick(View v) {
        String anotherName = "John Doe";
        mModel.getCurrentName().setValue(anotherName);
    }
});
```

- 이 예제에서 setValue(T) 메서드를 호출하면 관찰자들은 값이 바뀐 John Doe로 onCahnge() 메서드를 호출할 것이다.
- 이 예제에서는 버튼 클릭을 보여줬지만, setValue(T) / postValue(T) 메서드는 네트워크 요청, 데이터베이스 조회 등 다양한 상황에서 호출 가능하다

Note : 메인스레드에서 LiveData를 갱신하려면 반드시 setValue(T) 메서드를 호출한다.
만약 작업스레드에서 갱신하려면 postValue(T) 를 호출해야 한다

Update Activity

- LiveData notifyes the observer using setValue() and postValue()
- setValue() : runs on the main thread.
- postValue() : runs on the background thread.
- getValue() : LiveData type 인스턴스에서 현재 data를 호출한다

MutableLiveData

- MutableLiveData는 LiveData Type class를 extends하는 Class
- MutableLiveData는 postValue()와 setVlaue() moethod를 제공하기 때문에 일반적으로 사용된다.

Thanks!

Any questions ?

You can find me at

- jeonyt89@gmail.com