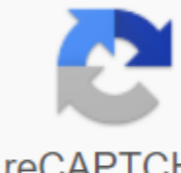


## Livedata example in android

I'm not robot



reCAPTCHA

**Continue**

In the name of the Great God, the most merciful of late, Google IO 17 has announced a new component of architecture and in this article we will talk about two of them, namely ViewModel and LiveData. First of all, suppose we have an app like this when you press the Add One button will add 1 to the value in textView, and the layout code will be as follows. The activity is that the public class MainActivity {  
 xmlns:android="http://schemas.android.com/apk/res-auto"  
 xmlns:tools="http://schemas.android.com/tools"  
 android.support.constraint.ConstraintLayout xmlns:android="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="me.a3zcs.mvp.architecurcurecomponent.MainActivity" android:layout\_width="wrap\_content" android:layout\_height="wrap\_content" app:layout\_constraintbottom\_toBottomOf="parent" app:layout\_constraintleft\_toLeftOf="parent" app:layout\_constraintright\_toRightOf="parent" app:layout\_constrainttop\_toTopOf="parent" android:id="@+id/number" android:layout\_width="wrap\_content" android:layout\_height="wrap\_content" android:text="button" app:layout\_constraintleft\_toLeftOf="parent" app:layout\_constraintright\_toRightOf="parent" android:layout\_marginTop="0dp" app:layout\_constrainttop\_toBottomOf="number/button" android.support.constraint.ConstraintLayout" expands AppCompatActivity (TextView number); AddOne button; int i No 0; @Override protected void onCreate (Bundle savedInstanceState) - super.onCreate (savedInstanceState); setContentView (R.layout.activity\_main); number = findViewById (R.id.number); addOne = findViewById (R.id.button); setNumber (i); addOne.setOnClickListener (new View.OnClickListener() - @Override public void onClick (View) - setNumber (i); The code is very simple, and works correctly, but when you change the screen mode to the landscape instead of the portrait the value of textView will return to 0, because the activity got its destruction and then create again, previously such problems that are related to configChanges solved by using onSaveInstanceState and restored onInstanceState, very nicely, we will talk about one of the new solutions to the problem, which is ViewModel. ViewModel: This is a class designed to work with view data, and the added benefit is that it stores data in a state of configChanges, very nice to start refactoring for the previous code creation model, which extends to the viewModel public class Expands ViewModel private int count; public emptinesscount this.count count; public int getCount Countdown; Very nice now we will use this class in MainActivity as an object of composition, and then create it in onCreate to become a form of activity as follows: the public class MainActivity expands AppCompatActivity (TextView number; addOne button; CounterViewModel counterViewModel; @Override protected void onCreate (Bundle savedInstanceState) - super.onCreate (savedInstanceState); setContentView (R.layout.activity\_main); ViewModel - ViewModelProviders.of (number - findViewById (R.id.number); addOne - findViewById (R.id.button); setNumber (counterViewModel.getCount()); addOne.setOnClickListener (new View.OnClickListener() - @Override public void onClick (View) - counterViewModel.setCount (counterViewModel.getCount()); setNumber (counterViewModel.getCount()); Please note that we use The ViewModelProviders, and it will store and store ViewModel for this activity, i.e., if the status of configChanges occurs - as a change of orientation - for the activity class ViewModelProviders will store ViewModel data and therefore we give it this information during viewModelProviders.of (get) now if we do rotations impact data that has been LiveData: LiveData allows lifecycleOwner, whether it's activity or snippet, to receive data from ViewModel by monitoring -observation - any modification to view Model Class, Very nice if instead of updating the user interface every time using setters and getters, we'll look at it in the following example first to make a refactor for your viewModel built, to become the next public class CounterViewModel expands to the publicViewModel (setValue We have changed the type of data we are dealing with. The button is zlt;CounterViewModel counterViewModel; The government @Override @Nullable Integer"Integer (integer) @Override; R.layout.activity\_main: R.layout.activity\_main view model - ViewModelProviders.of (it).get (CounterViewModel.class); counterViewModel.getCount.observe number - findViewById (R.id.number); addOne - findViewById (R.id.button); setNumber (counterModel.getView); @Override is also a key element in the development of the new approach. I made a comment before each line was deleted, to make it easier for you to see the edit, starting we made the Observer has only one feature that is on the change, called, if there is any data change, then we have observed on this line qlt;Object"gt; in onCreate counterViewModel.getCount.observe ), as our notes observer will call qlt;LifecycleOwner"gt; Provided that the state for lifecycleOwner in STARTED or RESUMEED, LifecycleOwner is in a STATE OF START on two occasions after calling the onStart function, secondly, before calling the onPause function, and LifecycleOwner is in a resumed state after

