Tabs android material design example

The layout of the tab is visible below the toolbar with the View pager used to create swipeable views. The tabs are designed to work with fragments. Use them to swipe fragments in the form of a pager. In this article we're going to show you how to implement material design tabs in your Android app. After creating a new project, open the build.gradle level application and add a design support library because Tablayout is part of the Android Design Support Library:Add Tab layout and pager view in you layout activity_main.xmlapp-bar.xml for toolbarstyle.xmlCreate tab adpter which extends to FragmentStatePagerAdapter. Create two lists for a list of snippets and a list of headlines. Create a method of conveying the snippet and title you want to add. TabAdapter.javaCreate Fragments to view pager. We'll add snippets to the activity adapter. Tab1Fragment.javaLayout for Tab1Fragment fragment_one.xmlSimilarly create more snippets you want to add, Tab2Fragment, Tab3Fragment and so on. with their layouts. Identify the tab layout and view the pager from the layout, identify the adapter, add the header snippets, install an adapter to view the pager, and set up the tabula with a viewpager pager using tablayout. MainActivity.javaRun this code and you'll find the output below: Some options to customize the look of tabs:app:tabGravity fill in to navigate the gravity tab. Another center for centerapp navigation tabs: tabIndicatorColor@color/white for the tab indicator. Here you can see the white indicator.app:tabIndicatorHeight4dp for the height indicator.app:tabMode corrected for tab mode. Other scrolls for many other tabs.app:tabTextColor @color/semi_yellow for unelected color.app text tabs:tabSelectedTextColor @color/yellow for selected text color tabs. If you want to add icons to the tab, you have to do, this is a setIcon call method tab. Create an array of icons and assign each one to each tab like this: If you want to put only icons on the tab, change the getPageTitle method in the adapter class as below: Exit: Here you can see that there is no more customization after that. To get additional custom tabs, you should create a custom tabview for navigation tabs manually. Custom Navigation Tab Layout: Create a custom layout for tab custom_tab.xmlAdd below method in TabAdapter: Remove the code below from your activity: Add the code below to your activity after calling setAdapter () method:Now, if you want to show great text on your chosen tab or change the color of the text, then you can do so by creating a separate method in your adapter, like getSelectedTabView (as below: Code below: Exit: Now, just like TextView, add ImageView to the layout of the tab: Change in your DemoActivity.java code: Changes in you Taber Class: Here's your go-out for the custom navigation tab. You can change the style of text tabs using the font, bold, italic, color, text size, etc., as well as for the tab icon. Other examples here are to try your own face. What you're going to createThe createThe The design team at Google simply defines the functionality of Tabs in Android as follows: Tabs make it easy to study and switch between different views. In this post, you'll learn how to display tabs using TabLayout and ViewPager API. In this practical tutorial, we'll cover the following components: TabLayout and ViewPager. Different tab modes: scrolling and fixed. How to display icons instead of text for tab headers. To get a bonus, you'll also learn how to use the Android Studio template feature to quickly download your project with an interface tab. An example of the project for this tutorial can be found on our GitHub repo, so you can easily follow along. Prerequisites To be able to follow this tutorial, you need: Android Studio 3.0 or above Kotlin plugin 1.1.51 or above you can also learn all all all and out of kotlin's language in my Kotlin from Scratch series. Introducing the TabLayout Component with official Android documentation on TabLayout, he says: TabLayout provides a horizontal layout for displaying tabs. The TabLayout component is one of the components presented as part of the material design artifacts. In addition, it is also included in the design support library. In TabLayout, when the tab is selected or listened to, the user sees another page (or snippet). TabLayout can have a display tab function in one of two ways: fixed and scrolling. If the tabs are fixed, all tabs will appear on the screen at the same time. The screenshot below is the latest official WhatsApp Android app (to date) that uses TabLayout with a fixed mode configuration. In scrolling tabs, if the number of tabs becomes too wide for the screen, the user can swipe left or right to view more tabs. Here's an example of TabLayout with scrolling tab mode-demonstrated in the latest version of Google's News and Weather Android app. In addition, the information displayed on the tab can be a text, icon, or a combination of text and icon. For example, the latest Twitter app for Android uses icons instead of text on each tab. In the following sections, we'll dive into coding a simple app that TabLayout uses with ViewPager. Let's roll! Design is not just what it looks and feels like. Design is how it works. Steve Jobs 1. Create Android Studio ProjectFire up To Android Studio 3 and create a new project (you can call it TabLayoutDemo) with an empty activity called MainActivity. 2. Creating snippets (pages) We're going to create a TabLayout with just three tabs. When you select each tab, you see a different snippet or Android page. So let's now create three Android fragments for each of the Let's start with the first class of snippets and you should follow a similar process for the remaining two snippets of snippets и FragmentThree.kt. Вот мой FragmentOne.kt:import android.os.Bundle импорт android.support.v4.app.Fragment импорт android.view.LayoutInflater импорт android.view.view импорт android.view.ViewGroup класса FragmentOne : Фрагмент!!. .) надуть (R.layout.fragment_one, контейнер, ложный) объект-компаньон (весело newInstance (): FragmentOne и FragmentOne () » Вот также мой R.layout.fragment_one: <LinearLayout xmlns:android= xmlns:tools= android:layout_width=match_parent android:layout_height=match_parent android:orientation=vertical> <TextView android:layout_width=match_parent android:layout_height=match_parent android:text=FragmentOne android:gravity=center_vertical|center_horizontal></TextView> </LinearLayout> 3. Добавление TabLayout и ViewPagerTo начать использовать TabLayout и ViewPager в вашем проекте, убедитесь, что вы импортируете поддержку дизайна, а также артефакт поддержки Android, так что добавьте их в ваш модуль build.gradle файл для их импорта. зависимостей - реализация 'com.android.support:design:26.1.0' реализация 'com.android.support:support-v4:26.1.0' Также, посетите ваш res/layout/activlty_main.xml файл , чтобы включить как виджет TabLayout, так и виджет ViewPager.<?xml version=1.0 encoding=utf-8?>Здесь мы создали простой TabLayout с id tab_layout. <android.support.design.widget.CoordinatorLayout xmlns:android= xmlns:app= android:id=@+id/main_content android:layout_width=match_parent android:layout_height=match_parent android:fitssystemwindows=true> <android.support.design.widget.AppBarLayout android:id=@+id/appbar android:layout_width=match_parent android:layout_height=wrap_content android:theme=@style/AppTheme.AppBarOverlay> <android.support.v7.widget.Toolbar android:id=@+id/toolbar android:layout_width=match_parent android:layout_height=?attr/actionBarSize android:background=?attr/colorPrimary app:contentinsetstartwithnavigation=0dp app:layout_scrollflags=scroll|enterAlways app:popuptheme=@style/AppTheme.PopupOverlay></android.support.v7.widget.Toolbar> <android.support.design.widget.TabLayout android:id=@+id/tab_layout style=@style/CustomTabLayout android:layout_width=match_parent android:layout_height=?attr/actionBarSize android:layout_gravity=left android:background=@color/colorPrimary app:tabgravity=fill app :tabmode=fixed></android.support.design.widget.TabLayout> </android.support.design.widget.AppBarLayout> <android.support.v4.view.ViewPager android:id=@+id/view_pager android:layout_width=match_parent android:layout_height=match_parent app:layout_behavior=@string/appbar_scrolling_view_behavior></android.support.v4.view.ViewPager> </android.support.design.widget.CoordinatorLayout> B TabLayout XML widget, you can see that we have included some attributes, such as the app:tabMode, which must be fixed, as well as the app:tabGravity to fill. App:tabGravity property is used to set up tab items to take up the available space. We set this to fill, which will evenly distribute items across the entire width of TabLayout. Note that this will be more noticeable in wider displays such as tablets. I've also included a custom style attribute (@style/CustomTabLayout) in our TabLayout widget. CustomTabLayout parent'Widget.Design.TabLay @android out item name tabIndicatorHeight/gt;3dp'lt;item name? tabBackground?attr/selectableItemBackground)lt;/item/item name q tabTextAppearance/gt;@style/CustomTabTextAppearance @android:color/white-lt;item/gt; zlt;)gt;we begin to set up our TabLayout by setting up the attributes that will be applied on TabLayout. Here are the details for some of the attributes used: tabIndicatorColor: sets the color of the tab indicator for the currently selected tab. This can also be installed by subpoenaing setSelectedTabIndicatorColor on a TabLayout copy. tabIndicatorHeight: sets the tab indicator height for the current tab. This can also be

installed by calling the SetSelectedTabIndicatorHeight on a TabLayout instance. tabSelectedTextColor: Sets text colors for different states (normal, selected) used for tabs. The equivalent of this attribute in Java is settabTextColors. Immediately after creating our TabLayout widget in XML, the next view was ViewPager. The official documentation says the following about ViewPager:Layout Manager, which allows the user to flip left and right through the data pages... 4. Creating PagerAdapter We have to create a subclass in SampleAdapter.kt, which expands FragmentPagerAdapter. This class is responsible for managing the various fragments that will appear on the tabs. import android.support.v4.app.Fragment import android.support.v4.app.FragmentManager import android.support.v4.app.FragmentPagerAdapter class SampleAdapter (fm: FragmentManapter) : FragmentPagerAdapter (fm) - redefine fun getItem (position: Int): Fragment? - When (position) - 0 -'gt; FragmentOne.newInstance () 1 - 'gt; FragmentTwo.newInstance() 2 - FragmentThree.newInstance () still - null - override amusing getPageTitle (position: Int): CharSequence - when (position)0 -'gt; Tab 1 -'gt; Tab 2 Item 2 -'gt; Tab 3 Point still -'gt; redefine fun getCount (: Int 3 Here we redefine three of the method class: getItem, getCount, and getPageTitle. Here's an explanation of the methods: getItem: returns a snippet for a specific position in ViewPager. getCount: shows how many pages of pages be in ViewPager. getPageTitle: This method is called ViewPager to get a title line to describe the specified tab. For example, if the tab you choose is the first tab with the Tab 1 Item title, the FragmentOne page will be immediately shown to the user. 5. Initialization of Next components, we're going to initialize copies of our TabLayout, ViewPager, and SampleAdapter. Initialization will take place inside onCreate () in MainActivity.kt.import android.os.Bundle import android.support.design.widget.TabLayout import android.v4.viewPager import android.support.v7.app.AppCompatActivity import android.support.v7. widget.Toolbar-class MainActivity : AppCompatActivity - super.onCreate (savedRectostate) setContentView (R.layout.activity_main) initToolbar () val tabLayout: TabLayout - findViewById (R.id.tab_layout) val viewPager: ViewPager - findViewById (R.id.view_pager) val adapter - SampleAdapter (supportFragmentManager) viewPager.adapter - adapter tabLayout.setWithViewPager (viewPager) tabLayout.addOnTabSe lectedListener (object : TabLayout.OnTabSelectedListener - override pleasure onTabSelected (tab: TabLayout.Tab) - override the fun onTabUnselected (tab: TabLayout.Tab) - redefine the fun onTabReselected (tab: TabLayout.Tab) - private fun initToolbar () - toolbar val: Toolbar - findViewById (R.id.toolbar) setSupportActionBar (tool panel) supportActionBar!!. titled - TabLayout Demo - We got links to our TabLayout and ViewPager from R.layout.activity_main and initiated them. We also created an example of our SampleAdapter passing a fragmentManager copy as an argument. We have to provide submissions for our ViewPager, so we called setAdapter and switched to our created adapter to it. Finally, we called set UpWithViewPager on a tabLayout copy to do some work: creating the necessary tab for each page to customize the required listeners When the user clicks on the tab, he changes the page to ViewPager and shows the desired page (or snippet). Also, by updating the selected tab between the pages. In other words, this method helps us take care of changing the state of scrolling and clicking on tabs. OnTabSelectedListener is used to include the listener, which will be called when you change your tab selection. We're override: onTabSelected is triggered when the tab enters its chosen state. onTabUnselected(onTabReselected(: Is called when a tab that has already been selected is re-selected by the user. Please note that we can also set up the tab mode software, rather than through the XML layout with setTabMode on a TabLayout instance. We're going to jump. (fixed or scrolling) to this method as arguments. For example, we may TabLayout.MODE_FIXED for a fixed to scroll through mode.tabLayout.tabMode No TabLayout.MODE_FIXED tabLayout.tabMode - TabLayout.MODE_SCROLLABLENote that if you want to explicitly create tabs instead of using the setUpWithViewPager assistant method, you can use newTab instead on a TabLayout instance. Val tabLayout: TabLayout - findViewById (R.id.tab_layout) tabLayout.addTab (tabLayout.newTab ().setText (Songs)) tabLayout.addTab (tabLayout.newTab ().setText (Albums) tabLayout.addTab (tabLayout.newTab). 6. Testing AppFinally, you can run the app! android.design.widget.TabLayout android:id'id/tabs android:layout_width'match_parent android:layout_height'wrap_content android.design.widget.TabItem android:id'id/tabItem android:layout_width'wrap_content android:layout_height'wrap_content android:text'songs'gt;'lt;/android.support.design.widget.TabItem'gt; android.design.widget.TabItem android:id'id/tabItem2 android:layout_width'wrap_content android:layout_height'wrap_content android:text'Albums'gt;'lt;'lt;'lt;'android.support.design.widget.TabItem'gt; android.design.widget.TabItem android:id'id/tabItem3 android:layout_width'wrap_content android:layout_height'wrap_content android:text'artists'gt;'lt;/android.support.design.widget.TabItem'gt; Try to interact with the app by swiping left or right and clicking on the tabs. 7. Scroll TabsThe official material design guidelines on tabs says the following about scrolling tabs: Scroll tabs to display a subset of tabs at any given time. They may contain longer tab labels and more tabs than fixed tabs. Scroll tabs are best used to view contexts in touch interfaces when users don't need to directly compare tab labels. Let's see how to create tabs with the scroll mode configuration. I made the title for each of the tabs longer than before. Here's the result in a fixed mode: You can see that TabLayout used several lines to display each of the tab names. In some situations, it will even truncation the title! This creates a bad user experience, so if your tab headers should be very long, you should consider using a scroll mode. Note also that if you're going to have more than four tabs, it's a good idea to do a scroll tab. Let's change the app:tabMode property from fixed to scrolling. you can also customize the tab.qlayout.design.website.!-- ... -- app: tabModescrollable/zgt;Remember, you can also set up a software tab mode, as discussed earlier. 8. Displaying Tab IconsLet is now immersed in заменить текст элемента вкладки с иконками вместо. класс MainActivity : AppCompatActivity () - переопределить удовольствие onCreate (сохраненоВостяниеГосударство: Комплект?) { // ... tabLayout.setupWithViewPager (viewPager) tabLayout.getTabAt (0)!!. setIcon (андроид. R.drawable.ic_dialog_email) tabLayout.getTabAt(1)!!. setIcon (андроид. R.drawable.ic_dialog_info) R.drawable.ic_dialog_info) // ... } // ... } Here we called getTabAt () on the example of TabLayout. Calling this method will return the tab in the index. Next, we call setIcon. Calling this method will set the icon displayed on this tab. I've also set a tab mode to be fixed. I'm still redefining getPageTitle () inside SampleAdapter. SampleAdapter class (fm: FragmentManager) : FragmentPagerAdapter (fm) to override fun getPageTitle (position: Int): CharSequence - when (position) - 0 -'gt; TAB 1 1 -'gt; TAB 2 2 -'gt; TAB 3 still - . . . . . Here's the result: Now, if you only want icons, you just don't override getPageTitle. 9. Bonus: Using Android Studio TemplatesIndeces writing as much code just to create a tab interface or activity from scratch, Android Studio 3.0 has some pre-existing code templates (available in Java and Kotlin) to help start your project. One such template can be used to create an activity tab. I'll show you how to use this handy feature in Android Studio 3. For a new project, light Android Studio 3. Enter the app's name and click Next. You can leave by default as they are in dialogue targeted Android devices. Click the next button again. In the activity app to mobile dialogue, scroll down and select Tabbed Activity. Click the next button after that. In the last conversation, scroll down to the navigation style drop-off menu and select Action Bar Tabs (with ViewPager). Finally, click the Finish button to take all configurations. Android Studio has now helped us create a project with an activity tab. Really cool! It is highly recommended to study the code generated. In an existing Android Studio project, to use this template, just go to File's Activity and Tabbed Activity. And follow similar steps that were described earlier. The templates included in Android Studio are good for simple layouts and basic apps, but if you want to start running the app even more, you might want to consider some of the app templates available on the Envato market. They are a huge time savers for experienced developers, helping them cut through the slog of creating an application from scratch and focus their talents rather than on the unique and individual parts of creating a new app. Conclusion In this tutorial, you learned how to create a tab interface in Android using TabLayout and ViewPager API Scratch. We've also explored how to easily and quickly use Android Studio templates to create an interface with tabs. I highly recommend checking the official material design guidelines for tabs to learn more about how to design and use the tabs in Android. To learn more about coding for Android, check out some of our other courses and tutorials here at Envato Tuts! Toots! Toots! android material design tabs example. android sliding tabs with material design example