Как сделать Navigation Drawer в Android

В этом уроке:

* Как создать макет для Navigation Drawer
* Как инициализировать Navigation Drawer
* Как обработать событие выбора пункта в списке меню Navigation Drawer
* Как установить слушатель (listener) на открытие и закрытие Navigation Drawer
* Открытие и закрытие Navigation Drawer по нажатию значка приложения в тулбаре
* Скачать образец приложения с Navigation Drawer
* Скачать Android Design Icons
* Как разместить Navigation Drawer под Toolbar’ом

**Navigation Drawer** (навигационная секция) — боковая панель, которая выводит на экран основные навигационные опции приложения в левой части экрана. Она скрыта большую часть времени, и отображается по свайпу от левого края экрана или нажатию значка приложения в **ActionBar**.

Этот урок описывает, как реализовать **Navigation Drawer**, используя API-интерфейсы, доступные в библиотеке поддержки **android.support.v4.**

Полный код приложения с **Navigation Drawer**можно скачать по ссылке внизу страницы.

Navigation Drawer Design

Прежде чем вы решите использовать Navigation Drawer в вашем приложении, вы должны ознакомиться с примерами использования и принципами проектирования, определенными в [Navigation Drawer: руководство по проектированию](https://developer.android.com/design/patterns/navigation-drawer.html" \t "_blank).

Смотрите видео, как использовать готовый шаблон [Navigation Drawer Activity в Android Studio](http://www.fandroid.info/navigation-drawer-activity-v-android-studio-struktura-shablona-primer-raboty-dizajn-android-prilozhenij/" \t "_blank) — структура шаблона, пример работы в приложении.

Создать макет Navigation Drawer

Чтобы добавить **Navigation Drawer**, объявите пользовательский интерфейс с **DrawerLayout** объектом в качестве корневого **View** вашего макета. Внутри , добавьте **View**, который содержит основной контент для экрана (основной макет, когда панель навигации скрыта) и еще один **View**, который содержит содержимое **Navigation Drawer**.

Ниже приведен код макета, который использует [DrawerLayout](https://developer.android.com/reference/android/support/v4/widget/DrawerLayout.html) с двумя дочерними **View**: [FrameLayout](https://developer.android.com/reference/android/widget/FrameLayout.html) для основного содержания ( [Фрагмент](https://developer.android.com/reference/android/app/Fragment.html) , создаваемый динамически во время работы программы), и [ListView](https://developer.android.com/reference/android/widget/ListView.html) для боковой панели навигации.

|  |
| --- |
| <android.support.v4.widget.DrawerLayout      xmlns:android="http://schemas.android.com/apk/res/android"      android:id="@+id/drawer\_layout"      android:layout\_width="match\_parent"      android:layout\_height="match\_parent">      <!-- The main content view -->      <FrameLayout          android:id="@+id/content\_frame"          android:layout\_width="match\_parent"          android:layout\_height="match\_parent" />      <!-- The navigation drawer -->      <ListView android:id="@+id/left\_drawer"          android:layout\_width="240dp"          android:layout\_height="match\_parent"          android:layout\_gravity="start"          android:choiceMode="singleChoice"          android:divider="@android:color/transparent"          android:dividerHeight="0dp"          android:background="#111"/>  </android.support.v4.widget.DrawerLayout> |

Этот макет имеет некоторые важные особенности:

Основной **View** с контентом (в **FrameLayout** выше) должен быть первым дочерним элементом в **DrawerLayout**, для того, чтобы панель навигации была над контентом.

Основному **View** устанавливается значение **match\_parent** для  ширины и высоты, поскольку он представляет весь интерфейс, когда скрыт **NavigationDrawer**.

Для**ListView** необходимо указывать параметр горизонтального выравнивания с помощью атрибута**android:layout\_gravity**. Поддержка RTL-языков (справа налево) решается указанием значения "start" вместо "left" (в таком случае **Navigation Drawer** будет появляться справа на RTL макете).

Ширину **ListView**указываемв **dp**, а высоту ставим **match\_parent**. Ширина боковой панели навигации должна быть не более 320dp, чтобы пользователь мог всегда видеть часть основного контента.

Инициализируем Drawer List

В классе **activity** в первую очередь необходимо инициализировать список элементов для **Navigation Drawer**. Как вы это сделаете зависит от контента вашего приложения. Панель навигации  включает в себя **listview**, поэтому список должен быть заполнен с помощью адаптера (например **ArrayAdapter** или **SimpleCursorAdapter**).

Например, вот как можно инициализировать **Navigation Drawer** со строкой массива:

|  |
| --- |
| public class MainActivity extends Activity {      private String[] mPlanetTitles;      private DrawerLayout mDrawerLayout;      private ListView mDrawerList;      ...        @Override      public void onCreate(Bundle savedInstanceState) {          super.onCreate(savedInstanceState);          setContentView(R.layout.activity\_main);            mPlanetTitles = getResources().getStringArray(R.array.planets\_array);          mDrawerLayout = (DrawerLayout) findViewById(R.id.drawer\_layout);          mDrawerList = (ListView) findViewById(R.id.left\_drawer);            // Set the adapter for the list view          mDrawerList.setAdapter(new ArrayAdapter<String>(this,                  R.layout.drawer\_list\_item, mPlanetTitles));          // Set the list's click listener          mDrawerList.setOnItemClickListener(new DrawerItemClickListener());            ...      }  } |

Этот код также вызывает метод **setOnItemClickListener (),** чтобы обрабатывать нажатия пунктов в списке. В следующем разделе показано, как реализовать этот интерфейс и изменить вид экрана, когда пользователь выбирает элемент.

Обработка события нажатия пункта в **Navigation Drawer**

Когда пользователь выбирает элемент в списке, система вызывает метод **onItemClick()** объекта **DrawerItemClickListener**, реализующего интерфейс **OnItemClickListener** и присвоенного списку через **setOnItemClickListener()**.

Действия в методе **onItemClick()** зависят от задач приложения. В следующем примере, при выборе каждого элемента списка в основное **View**(в**FrameLayout,** определенный по ID R.id.content\_frame) вставляется новый создаваемый фрагмент:

|  |
| --- |
| private class DrawerItemClickListener implements ListView.OnItemClickListener {      @Override      public void onItemClick(AdapterView parent, View view, int position, long id) {          selectItem(position);      }  }    /\*\* Смена фрагментов в основном окне программы \*/  private void selectItem(int position) {      // Создание нового фрагмента и вставка изображения для показа, в зависимости от выбранной позиции      Fragment fragment = new PlanetFragment();      Bundle args = new Bundle();      args.putInt(PlanetFragment.ARG\_PLANET\_NUMBER, position);      fragment.setArguments(args);        // Вставка нового фрагмента взамен существующего      FragmentManager fragmentManager = getFragmentManager();      fragmentManager.beginTransaction()                     .replace(R.id.content\_frame, fragment)                     .commit();        // Выделение выбранного элемента списка, обновление заголовка окна и закрытие бокового меню      mDrawerList.setItemChecked(position, true);      setTitle(mPlanetTitles[position]);      mDrawerLayout.closeDrawer(mDrawerList);  }    @Override  public void setTitle(CharSequence title) {      mTitle = title;      getActionBar().setTitle(mTitle);  } |

Слушатель для открытия и закрытия Navigation Drawer

Для прослушивания событий открытия и закрытия бокового меню , вызываем [setDrawerListener()](https://developer.android.com/reference/android/support/v4/widget/DrawerLayout.html" \l "setDrawerListener(android.support.v4.widget.DrawerLayout.DrawerListener)) в вашем [DrawerLayout](https://developer.android.com/reference/android/support/v4/widget/DrawerLayout.html) и передаем его интерфейсу [DrawerLayout.DrawerListener](https://developer.android.com/reference/android/support/v4/widget/DrawerLayout.DrawerListener.html). Этот интерфейс предоставляет обратные вызовы для таких событий, как [onDrawerOpened()](https://developer.android.com/reference/android/support/v4/widget/DrawerLayout.DrawerListener.html" \l "onDrawerOpened(android.view.View)) и [onDrawerClosed()](https://developer.android.com/reference/android/support/v4/widget/DrawerLayout.DrawerListener.html" \l "onDrawerClosed(android.view.View)).

Однако, вместо реализации [DrawerLayout.DrawerListener](https://developer.android.com/reference/android/support/v4/widget/DrawerLayout.DrawerListener.html), если ваше activity включает [action bar](https://developer.android.com/guide/topics/ui/actionbar.html), вы можете расширить класс [ActionBarDrawerToggle](https://developer.android.com/reference/android/support/v4/app/ActionBarDrawerToggle.html). Реализуя в нем интерфейс [DrawerLayout.DrawerListener](https://developer.android.com/reference/android/support/v4/widget/DrawerLayout.DrawerListener.html),  можно переопределить обратные вызовы, что упрощает взаимодействие между значком строки меню и **Navigation Drawer** (подробнее об этом в следующем разделе).

Как обсуждалось в [гайдлайне по Navigation Drawer](https://developer.android.com/design/patterns/navigation-drawer.html" \t "_blank), когда боковое меню выезжает, нужно менять содержимое панели действий [action bar](https://developer.android.com/guide/topics/ui/actionbar.html), например, изменить заголовок и удалить элементы действий, которые являются контекстно-зависимыми к основному содержимому. Следующий код показывает, как вы можете сделать это путем переопределения метода обратного вызова [DrawerLayout.DrawerListener](https://developer.android.com/reference/android/support/v4/widget/DrawerLayout.DrawerListener.html) класса [ActionBarDrawerToggle](https://developer.android.com/reference/android/support/v4/app/ActionBarDrawerToggle.html):

|  |
| --- |
| public class MainActivity extends Activity {      private DrawerLayout mDrawerLayout;      private ActionBarDrawerToggle mDrawerToggle;      private CharSequence mDrawerTitle;      private CharSequence mTitle;      ...        @Override      public void onCreate(Bundle savedInstanceState) {          super.onCreate(savedInstanceState);          setContentView(R.layout.activity\_main);          ...            mTitle = mDrawerTitle = getTitle();          mDrawerLayout = (DrawerLayout) findViewById(R.id.drawer\_layout);          mDrawerToggle = new ActionBarDrawerToggle(this, mDrawerLayout,                  R.drawable.ic\_drawer, R.string.drawer\_open, R.string.drawer\_close) {                /\*\* Этот код вызывается, когда боковое меню переходит в полностью закрытое состояние. \*/              public void onDrawerClosed(View view) {                  super.onDrawerClosed(view);                  getActionBar().setTitle(mTitle);                  invalidateOptionsMenu(); // creates call to onPrepareOptionsMenu()              }                /\*\* Этот код вызывается, когда боковое меню полностью открывается. \*/              public void onDrawerOpened(View drawerView) {                  super.onDrawerOpened(drawerView);                  getActionBar().setTitle(mDrawerTitle);                  invalidateOptionsMenu(); // creates call to onPrepareOptionsMenu()              }          };            // Set the drawer toggle as the DrawerListener          mDrawerLayout.setDrawerListener(mDrawerToggle);      }        /\* Этот код вызывается, когда мы вызываем invalidateOptionsMenu() \*/      @Override      public boolean onPrepareOptionsMenu(Menu menu) {          // If the nav drawer is open, hide action items related to the content view          boolean drawerOpen = mDrawerLayout.isDrawerOpen(mDrawerList);          menu.findItem(R.id.action\_websearch).setVisible(!drawerOpen);          return super.onPrepareOptionsMenu(menu);      }  } |

В следующей секции описываются аргументы конструктора [ActionBarDrawerToggle](https://developer.android.com/reference/android/support/v4/app/ActionBarDrawerToggle.html) и другие шаги, необходимые для установки и настройки взаимодействия со значком **action bar icon**.

Открыть и закрыть **панель навигации** по  значку приложения

Пользователи могут открывать и закрывать **navigation drawer** жестом от левого края экрана. Если вы используете [action bar](https://developer.android.com/guide/topics/ui/actionbar.html), или пришедший ему на смену в Android 5.0 API новый виджет [Toolbar](https://developer.android.com/reference/android/widget/Toolbar.html" \t "_blank), вы также должны позволять пользователям открывать и закрывать **navigation drawer**, нажав на значок приложения. Кроме того, рядом со значком приложения должен быть специальный значок, обозначающий наличие **navigation drawer**. Вы можете реализовать все это поведение при помощи [ActionBarDrawerToggle](https://developer.android.com/reference/android/support/v4/app/ActionBarDrawerToggle.html), как показано в предыдущем разделе.

Чтобы использовать [ActionBarDrawerToggle](https://developer.android.com/reference/android/support/v4/app/ActionBarDrawerToggle.html), создайте экземпляр при помощи конструктора с такими аргументами:

* [Activity](https://developer.android.com/reference/android/app/Activity.html) , в котором размещается боковая панель навигации.
* [DrawerLayout](https://developer.android.com/reference/android/support/v4/widget/DrawerLayout.html).
* drawable ресурс, используемый в качестве индикатора панели.Стандартный навигационный значок панели доступен в [Download the Action Bar Icon Pack](http://developer.android.com/downloads/design/Android_Design_Icons_20130926.zip).
* Строковый ресурс для обозначения открытой панели  (для специальных возможностей).
* Строковый ресурс для обозначения закрытой панели (для специальных возможностей).

Теперь, в зависимости от использования класса [ActionBarDrawerToggle](https://developer.android.com/reference/android/support/v4/app/ActionBarDrawerToggle.html) в вашем списке панели навигации, вы должны вызвать [ActionBarDrawerToggle](https://developer.android.com/reference/android/support/v4/app/ActionBarDrawerToggle.html) в нескольких местах жизненного цикла **activity**:

|  |
| --- |
| public class MainActivity extends Activity {      private DrawerLayout mDrawerLayout;      private ActionBarDrawerToggle mDrawerToggle;      ...        public void onCreate(Bundle savedInstanceState) {          ...            mDrawerLayout = (DrawerLayout) findViewById(R.id.drawer\_layout);          mDrawerToggle = new ActionBarDrawerToggle(                  this,                  /\* host Activity \*/                  mDrawerLayout,         /\* DrawerLayout object \*/                  R.drawable.ic\_drawer,  /\* nav drawer icon to replace 'Up' caret \*/                  R.string.drawer\_open,  /\* "open drawer" description \*/                  R.string.drawer\_close  /\* "close drawer" description \*/                  ) {                /\*\* Called when a drawer has settled in a completely closed state. \*/              public void onDrawerClosed(View view) {                  super.onDrawerClosed(view);                  getActionBar().setTitle(mTitle);              }                /\*\* Called when a drawer has settled in a completely open state. \*/              public void onDrawerOpened(View drawerView) {                  super.onDrawerOpened(drawerView);                  getActionBar().setTitle(mDrawerTitle);              }          };            // Set the drawer toggle as the DrawerListener          mDrawerLayout.setDrawerListener(mDrawerToggle);            getActionBar().setDisplayHomeAsUpEnabled(true);          getActionBar().setHomeButtonEnabled(true);      }        @Override      protected void onPostCreate(Bundle savedInstanceState) {          super.onPostCreate(savedInstanceState);          // Sync the toggle state after onRestoreInstanceState has occurred.          mDrawerToggle.syncState();      }        @Override      public void onConfigurationChanged(Configuration newConfig) {          super.onConfigurationChanged(newConfig);          mDrawerToggle.onConfigurationChanged(newConfig);      }        @Override      public boolean onOptionsItemSelected(MenuItem item) {          // Pass the event to ActionBarDrawerToggle, if it returns          // true, then it has handled the app icon touch event          if (mDrawerToggle.onOptionsItemSelected(item)) {            return true;          }          // Handle your other action bar items...            return super.onOptionsItemSelected(item);      }        ...  } |

Перевод [источника](https://developer.android.com/intl/ru/training/implementing-navigation/nav-drawer.html). [Скачать исходный код.](http://developer.android.com/shareables/training/NavigationDrawer.zip)