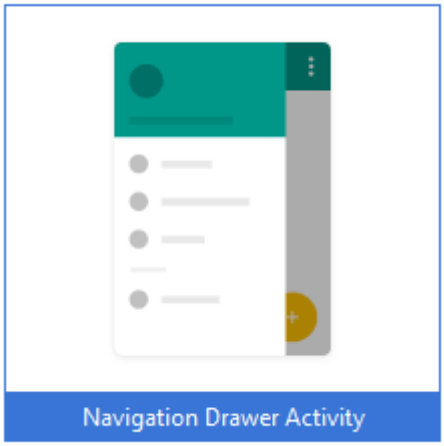
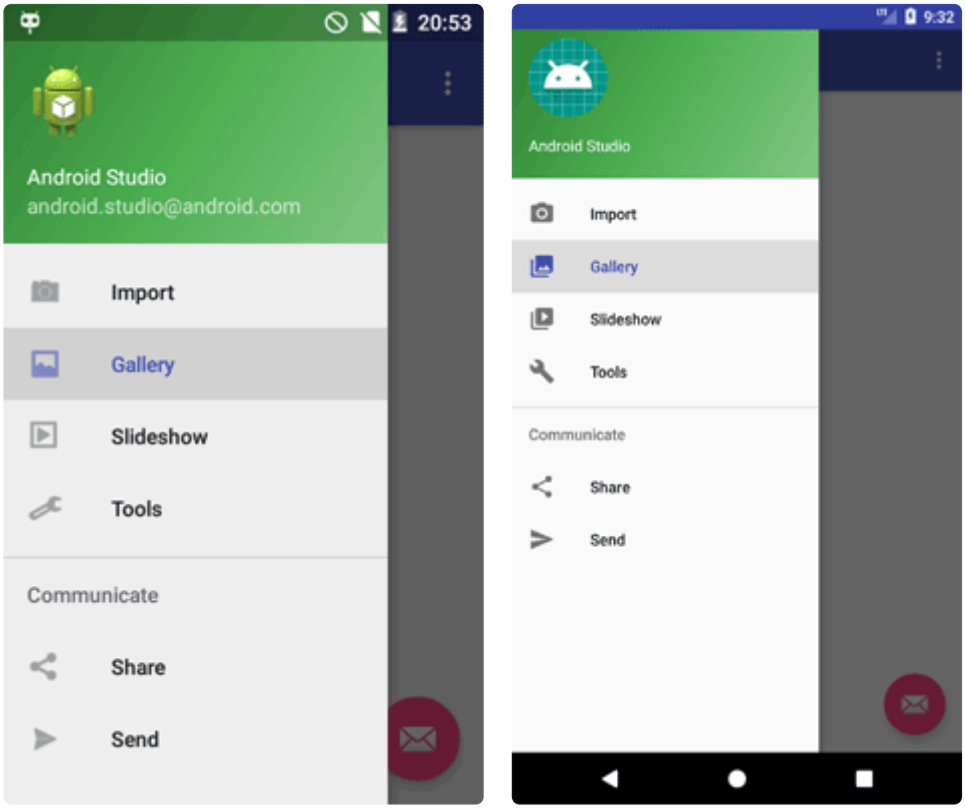
Шаблон Navigation Drawer Activity

Метки: NavigationView, DrawerLayout, Material Design

Рассмотрим шаблон **Navigation Drawer Activity**. Создадим новый проект и выберем нужный шаблон.



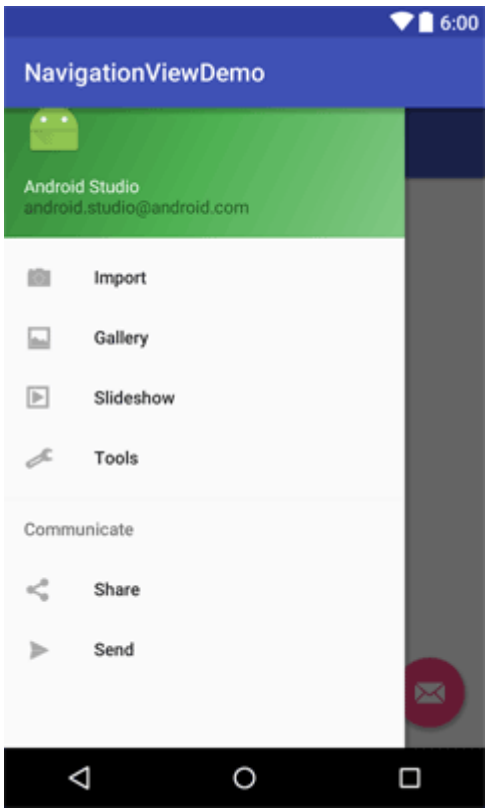
Нажмите на значок в виде трех горизонтальных полосок в заголовке. Значок в документации называется "гамбургером" (Hamburger menu). Это официальная позиция Гугла. Но в реальности значок символизирует полосатых котов (никому не рассказывайте). При нажатии слева вылезет навигационная шторка. Шторка работает как обычная шторка в ванной. По высоте она занимает весь экран, включая системную область. Можете подвигать шторку вперёд-назад, чтобы увидеть, что верхняя кромка шторки в системной области полупрозрачна и не закрывает системные значки. Подобное поведение доступно на устройствах под Android 5 и выше. На старых устройствах шторка находится под системной панелью. Недавно стал проверять работу под Android 8.0 и увидел, что шторка теперь не закрывает системную панель. Ниже для сравнения я привёл два варианта.



Шторка закрывает системную панель Шторка не закрывает системную панель

Сама шторка состоит из двух основных частей - в верхней части находится картинка и текст, а в нижней - меню со значками. Меню в свою очередь разделено на две группы. В верхней части значки можно выбрать и выбранный пункт останется выделенным. В нижней части меню пункты не выделяются. Уберите шторку обратно и вызовите теперь её не нажатием на значок гамбургера, а движением пальца от края экран в центр. Получилось? Отлично, а теперь выдвигайте шторку медленно и наблюдайте за значком гамбургера. Вы увидите, что во время движения значок трансформируется. К сожалению, шторка закрывает значок и непонятно, во что превращаются три полоски. А превращаются они в ~~три кота ж!~~ стрелку. Позже я покажу, как увидеть её. А может не покажу, я ещё не решил.

Возвращаемся в студию и начинаем изучать код проекта.



Как видите, небольшие расхождения имеются, но в целом совпадает с реальным приложением.

Посмотрим на его содержание.

```
<android.support.v4.widget.DrawerLayout
xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:id="@+id/drawer_layout"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:fitsSystemWindows="true"
    tools:openDrawer="start">

    <include
        layout="@layout/app_bar_main"
        android:layout_width="match_parent"
        android:layout_height="match_parent" />

    <android.support.design.widget.NavigationView
        android:id="@+id/nav_view"
        android:layout_width="wrap_content"
        android:layout_height="match_parent"
        android:layout_gravity="start"
        android:fitsSystemWindows="true"
        app:headerLayout="@layout/nav_header_main"
        app:menu="@menu/activity_main_drawer" />

</android.support.v4.widget.DrawerLayout>
```

Сейчас важно запомнить, что за выдвигающую шторку отвечает элемент **NavigationView**, который входит последним в контейнере **DrawerLayout** и представляет собой навигационное меню. А перед меню находится вставка **include**, указывающая на разметку **app_bar_main.xml**.

Атрибут **tools:openDrawer** позволяет указать студии, что навигационное меню нужно отобразить в раскрытом виде в режиме просмотра разметки.

NavigationView

В 2014 году Google показал новый дизайн и различные новые примеры по навигации. Но вначале они использовали подручные средства, которые были под рукой - фрагменты.

Спустя год компания разработала на основе предка **FrameLayout** новый компонент **NavigationView**, который стал частью библиотеки Android Design Support Library.

Новый подход оказался неожиданным, но логичным. Раз выдвижная шторка содержит навигационное меню, то и класс был спроектирован как меню. Вы можете создать элементы меню в ресурсах **res/menu** стандартным способом и получить готовую навигацию.

Перейдём к деталям.

Теперь в рекомендациях не указаны точные размеры шторки, хотя раньше нужно было самостоятельно указать ширину шторки. Видимо, новый класс сам обеспечивает необходимую ширину. Интерес представляют два последних атрибута.

Тег **NavigationView** содержит ссылку на собственную разметку в атрибуте **app:headerLayout**, который указывает на файл **nav_header_main.xml** (верхняя часть шторки), а также на меню в атрибуте **app:menu**, который ссылается на ресурс меню **menu/activity_main_drawer.xml**.

Откроем файл **nav_header_main.xml** и посмотрим на разметку шторки.

```
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    android:layout_width="match_parent"
    android:layout_height="@dimen/nav_header_height"
    android:background="@drawable/side_nav_bar"
    android:gravity="bottom"
    android:orientation="vertical"
    android:paddingBottom="@dimen/activity_vertical_margin"
    android:paddingLeft="@dimen/activity_horizontal_margin"
    android:paddingRight="@dimen/activity_horizontal_margin"
    android:paddingTop="@dimen/activity_vertical_margin"
    android:theme="@style/ThemeOverlay.AppCompat.Dark">

    <ImageView
        android:id="@+id/imageView"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:paddingTop="@dimen/nav_header_vertical_spacing"
        app:srcCompat="@mipmap/ic_launcher_round" />

    <TextView
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:paddingTop="@dimen/nav_header_vertical_spacing"
        android:text="Android Studio"
        android:textAppearance="@style/TextAppearance.AppCompat.Body1" />

    <TextView
        android:id="@+id/textView"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="android.studio@android.com" />

</LinearLayout>
```

Разметка состоит из **ImageView** и двух **TextView**, размещённых в контейнере **LinearLayout**. Фон контейнера определён в ресурсе **drawable/side_nav_bar.xml** и представляет собой градиент.

```
    <gradient
        android:angle="135"
        android:centerColor="#009688"
        android:endColor="#00695C"
        android:startColor="#4DB6AC"
        android:type="linear" />
</shape>
```

Остальные атрибуты понятны и не требуют пояснений.

Можно (но не нужно) настроить верхнюю часть шторки не через XML, а программно.

```
NavigationView navigationView = (NavigationView) findViewById(R.id.nav_view);
// Наполняем шапку элементами
View headerLayout = navigationView.inflateHeaderView(R.layout.nav_header_main);
ImageView headerImageView = headerLayout.findViewById(R.id.imageView);
```

После обновления одной из версий библиотеки **Design Support**, доступ к шапке осуществляется теперь через другой код.

```
View headerLayout = navigationView.getHeaderView(0);
```

Теперь рассмотрим ресурс навигационного меню **res/menu/activity_main_drawer.xml**.

```
<menu xmlns:android="http://schemas.android.com/apk/res/android">

    <group android:checkableBehavior="single">
        <item
            android:id="@+id/nav_camera"
            android:icon="@drawable/ic_menu_camera"
            android:title="Import" />
        <item
            android:id="@+id/nav_gallery"
            android:icon="@drawable/ic_menu_gallery"
            android:title="Gallery" />
        <item
            android:id="@+id/nav_slideshow"
            android:icon="@drawable/ic_menu_slideshow"
            android:title="Slideshow" />
        <item
            android:id="@+id/nav_manage"
            android:icon="@drawable/ic_menu_manage"
            android:title="Tools" />
    </group>

    <item android:title="Communicate">
        <menu>
            <item
                android:id="@+id/nav_share"
                android:icon="@drawable/ic_menu_share"
                android:title="Share" />
            <item
                android:id="@+id/nav_send"
                android:icon="@drawable/ic_menu_send"
                android:title="Send" />
        </menu>
    </item>

</menu>
```

Принцип создания элементов меню остался стандартным. Каждый пункт меню представляет собой тег **item** с указанием значка и текста. Для группировки используется элемент **group**. Поведение элементов меню в группе регулируется атрибутом **android:checkableBehavior**. В примере используется значение **single** - при нажатии на пункт меню, он останется выделенным (принцип переключателя **RadioButton**). Всего доступно три варианта.

- **single** – можно выбрать один элемент группы (переключатель)
- **all** - можно выбрать все элементы группы (флажок)
- **none** – элементы не выбираются

Также следует обратить внимание, что теперь проект ссылается на векторные рисунки, которые находятся в папке **drawable-21**.

Осталось рассмотреть тег **include**, который ссылается на файл ресурса **res/layout/app_bar_main.xml**. Он вам будет знаком по шаблону **Blank Activity**, который мы изучали в статье Библиотека Android Support Design. Только там он находился в файле **activity_main.xml**, а здесь его перенесли в файл **app_bar_main.xml**. Всё остальное осталось без изменений. Повторяться не будем.

Теперь изучим код активности для работы со шторкой.

В классе активности реализуется интерфейс **OnNavigationItemSelectedListener** с его методом **onNavigationItemSelected()**:

```
public class MainActivity extends AppCompatActivity
        implements NavigationView.OnNavigationItemSelectedListener {

    @SuppressWarnings("StatementWithEmptyBody")
    @Override
    public boolean onNavigationItemSelected(MenuItem item) {
        // Handle navigation view item clicks here.
        int id = item.getItemId();

        if (id == R.id.nav_camara) {
            // Handle the camera action
        } else if (id == R.id.nav_gallery) {

        } else if (id == R.id.nav_slideshow) {

        } else if (id == R.id.nav_manage) {

        } else if (id == R.id.nav_share) {

        } else if (id == R.id.nav_send) {

        }

        DrawerLayout drawer = (DrawerLayout) findViewById(R.id.drawer_layout);
        drawer.closeDrawer(GravityCompat.START);
        return true;
    }
}
```

closeDrawer() для закрытия шторки.

Добавим код для первого пункта меню.

```
if (id == R.id.nav_camera) {
    // Handle the camera action
    Toast.makeText(getApplicationContext(), "Вы выбрали камеру",
Toast.LENGTH_SHORT).show();
}
```

При нажатии кнопки "Назад" проверяется состояние шторки. Если шторка открыта (**isDrawerOpen()**), то её закрываем с помощью метода **closeDrawer()**.

```
@Override
public void onBackPressed() {
    DrawerLayout drawer = (DrawerLayout) findViewById(R.id.drawer_layout);
    if (drawer.isDrawerOpen(GravityCompat.START)) {
        drawer.closeDrawer(GravityCompat.START);
    } else {
        super.onBackPressed();
    }
}
```

В методе **onCreate()** происходит инициализация шторки.

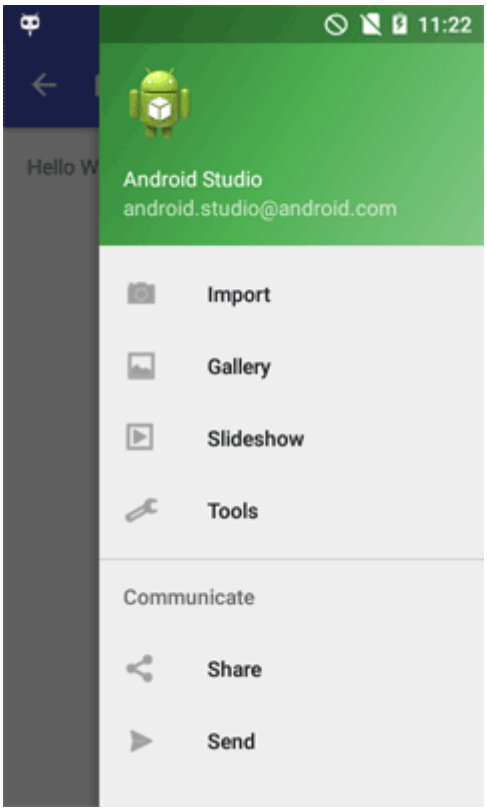
```
@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_main);
    ...

    DrawerLayout drawer = (DrawerLayout) findViewById(R.id.drawer_layout);
    ActionBarDrawerToggle toggle = new ActionBarDrawerToggle(
            this, drawer, toolbar, R.string.navigation_drawer_open,
R.string.navigation_drawer_close);
    drawer.addDrawerListener(toggle);
    toggle.syncState();

    NavigationView navigationView = (NavigationView) findViewById(R.id.nav_view);
    navigationView.setNavigationItemSelectedListener(this);
}
```

Теперь поговорим об изменениях, которые можно внести в проект.

```
<android.support.design.widget.NavigationView
    ...
    android:layout_gravity="end"/>
```



На самом деле смысла в этом не оказалось. Да, шторка выдвигается вручную. Но если нажать на значок гамбургера, то приложение валится с ошибкой. Любое нажатие в меню шторки также приводит к ошибке. Теоретически можно написать код, который исправит проблему, но он будет сложным. Забудьте об этом совете.

Тем не менее, можно реализовать забавный эффект - добавить вторую шторку на экран. Первая будет работать главной и реагировать на нажатие значка, а вторая будет дополнительной для вывода какой-то информации. Достаточно в разметку добавить второй **NavigationView** с атрибутом **android:layout_gravity="end"**

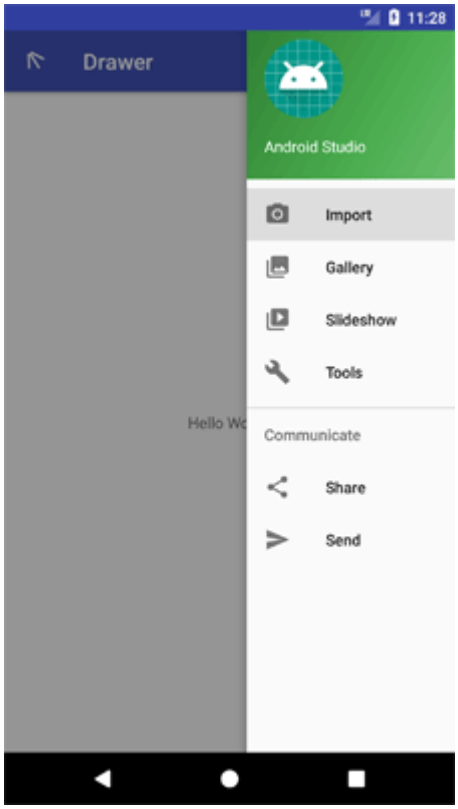
```
<android.support.v4.widget.DrawerLayout
xmlns:android="http://schemas.android.com/apk/res/android"
     ...

    <android.support.design.widget.NavigationView
        ...
        android:layout_gravity="start"
        ... />

    <android.support.design.widget.NavigationView
        android:id="@+id/nav_view2"
        android:layout_width="wrap_content"
        android:layout_height="match_parent"
        android:layout_gravity="end"
        android:fitsSystemWindows="true"
        app:headerLayout="@layout/nav_header_main"
        app:menu="@menu/activity_main_drawer" />

</android.support.v4.widget.DrawerLayout>
```

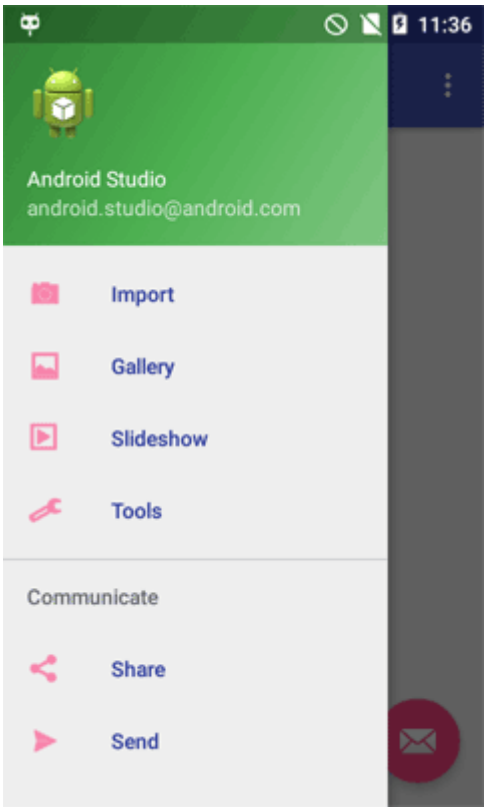
Кстати, если выдвигать правую шторку, то можно наблюдать трансформацию значка без помех. Скриншот во время частичного выдвижения шторки справа.



Для изменения цвета значков и текста в навигационном меню используйте атрибуты **app:itemIconTint** и **app:itemTextColor**.

```
    ...
    app:itemIconTint="@color/colorAccent"
    app:itemTextColor="@color/colorPrimaryDark"/>
```

Данным атрибутам соответствуют методы **setItemIconTintList()** и **setItemTextColor()** компонента **NavigationView**.



Так как наличие шторки не совсем очевидно, можно при первом запуске показать шторку в раскрытом состоянии. Далее можно запомнить состояние шторки при первом запуске в настройках, чтобы во второй раз выводить её уже в закрытом состоянии. Можете самостоятельно реализовать эту возможность.

```
// показываем в открытом состоянии в onCreate()
drawer.openDrawer(GravityCompat.START);
```

Напоследок покажу превращение значка гамбургера в стрелку в явном виде, как и обещал в начале статьи. Напомню, что по рекомендации Material Design шторка должна закрывать всю область экрана. Но если вы хотите поместить шторку под заголовком, то следует немного поправить разметку. Откроем файл **app_bar_main.xml** и вырежем из него небольшой кусок. Затем в файле **activity_main.xml** добавим **LinearLayout** в качестве корневого контейнера и вставим скопированный ранее кусок кода.

```
<LinearLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical">

    <android.support.design.widget.AppBarLayout
        android:layout_height="wrap_content"
        android:layout_width="match_parent"
        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:popupTheme="@style/AppTheme.PopupOverlay"/>

    </android.support.design.widget.AppBarLayout>

    <android.support.v4.widget.DrawerLayout
        android:id="@+id/drawer_layout"
        xmlns:android="http://schemas.android.com/apk/res/android"

        xmlns:tools="http://schemas.android.com/tools"
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:fitsSystemWindows="true"
        tools:openDrawer="start">

        <include
            layout="@layout/app_bar_main"
            android:layout_width="match_parent"
            android:layout_height="match_parent"/>

        <android.support.design.widget.NavigationView
            android:id="@+id/nav_view"
            android:layout_width="wrap_content"
            android:layout_height="match_parent"
            android:layout_gravity="start"
            android:fitsSystemWindows="true"
            app:headerLayout="@layout/nav_header_main"
            app:menu="@menu/activity_main_drawer"/>

    </android.support.v4.widget.DrawerLayout>
</LinearLayout>
```

Сама анимация значка зависит от переменной **toggle** (объект класса **ActionBarDrawerToggle**). Если вы её уберёте, то никакого значка в заголовке приложения не будет.

Можно поменять цвет значка гамбургера. Откроем файл стилей **res/values/styles.xml** и добавим:

```
<!-- Base application theme. -->
<style name="AppTheme" parent="Theme.AppCompat.Light.DarkActionBar">
    <item name="drawerArrowStyle">@style/DrawerArrowStyle</item>
</style>

<style name="DrawerArrowStyle" parent="Widget.AppCompat.DrawerArrowToggle">
    <item name="spinBars">true</item>
    <item name="color">@android:color/white</item>
</style>
```

Элемент **spinBars** со значением **true** позволяет использовать анимацию. В противном случае значок будет статичным.

Навигация

В шаблоне присутствует метод **onNavigationItemSelected()** с аннотацией **@SuppressWarnings("StatementWithEmptyBody")** (Оператор с пустым телом). Нам нужно добавить свой код для навигации, который должен реагировать на нажатия в меню шторки. Нам понадобятся фрагменты. Для примера создадим первый фрагмент.

```
import android.os.Bundle;
import android.support.v4.app.Fragment;
import android.view.LayoutInflater;
import android.view.View;
import android.view.ViewGroup;

public class FirstFragment extends Fragment {

    @Override
    public View onCreateView(LayoutInflater inflater, ViewGroup container,
                             Bundle savedInstanceState) {
        return inflater.inflate(R.layout.fragment_first, container, false);
    }
}
```

Разметка для первой активности.

```
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
                android:layout_width="match_parent"
                android:layout_height="match_parent"
                android:background="@android:color/holo_green_dark">

    <TextView
        android:id="@+id/textView3"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_centerHorizontal="true"
        android:layout_centerVertical="true"
        android:text="First Fragment"
        android:textAppearance="?android:attr/textAppearanceLarge"/>
</RelativeLayout>
```

По такому же образу создайте второй фрагмент **SecondFragment** и т.д.

Определим **RelativeLayout** в файле **content_main.xml** в качестве контейнера.

```
<RelativeLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:id="@+id/container"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:paddingBottom="@dimen/activity_vertical_margin"
    android:paddingLeft="@dimen/activity_horizontal_margin"
    android:paddingRight="@dimen/activity_horizontal_margin"
    android:paddingTop="@dimen/activity_vertical_margin"
    app:layout_behavior="@string/appbar_scrolling_view_behavior"
    tools:context="ru.alexanderklimov.navigationdrawerdemo.MainActivity"
    tools:showIn="@layout/app_bar_main">

    <!--<TextView-->
        <!--android:layout_width="wrap_content"-->
        <!--android:layout_height="wrap_content"-->
        <!--android:text="Hello World!"/>-->
</RelativeLayout>
```

Теперь можем написать недостающий код для навигации по фрагментам в **MainActivity**.

```
@Override
public boolean onNavigationItemSelected(MenuItem item) {

    // Создадим новый фрагмент
    Fragment fragment = null;
    Class fragmentClass = null;

    // Handle navigation view item clicks here.
    int id = item.getItemId();

    if (id == R.id.nav_camera) {
        // Handle the camera action
        fragmentClass = FirstFragment.class;

    } else if (id == R.id.nav_gallery) {
        fragmentClass = SecondFragment.class;
    } else if (id == R.id.nav_slideshow) {

    } else if (id == R.id.nav_manage) {

    } else if (id == R.id.nav_share) {

    } else if (id == R.id.nav_send) {

    }

    try {
        fragment = (Fragment) fragmentClass.newInstance();
    } catch (Exception e) {
        e.printStackTrace();
    }

    // Вставляем фрагмент, заменяя текущий фрагмент
    FragmentManager fragmentManager = getSupportFragmentManager();
    fragmentManager.beginTransaction().replace(R.id.container, fragment).commit();
    // Выделяем выбранный пункт меню в шторке
    item.setChecked(true);
    // Выводим выбранный пункт в заголовке
    setTitle(item.getTitle());

    DrawerLayout drawer = (DrawerLayout) findViewById(R.id.drawer_layout);
    drawer.closeDrawer(GravityCompat.START);
    return true;
}
```

меню шторки

В версии 23.1.0 появилась возможность добавлять дополнительные компоненты в меню шторки. Создадим новый файл разметки **nav_header_switch.xml**.

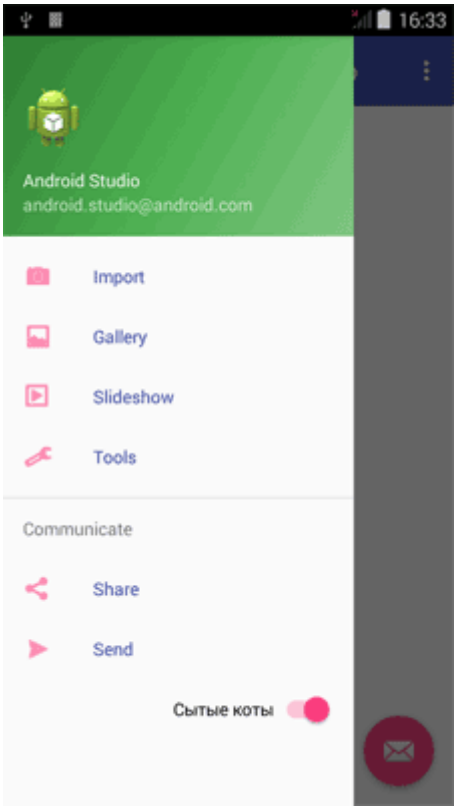
```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:orientation="horizontal">

    <android.support.v7.widget.SwitchCompat
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:text="Сытые коты"/>
</LinearLayout>
```

Связываем разметку с новым элементом меню.

```
<item
    android:id="@+id/nav_switch"
    app:actionLayout="@layout/nav_header_switch"
    android:title=""/>
```

Атрибут **android:title** я оставил пустым, так как текст уже задан в разметке. Вы можете задать свой текст здесь.



Откройте файл **res/menu/activity_main_drawer.xml** и добавьте атрибут **app:actionViewClass="android.widget.TextView"** ко второму и третьему элементу меню из шаблона. Теперь эти элементы будут связаны с текстовыми метками.

```
<item
    android:id="@+id/nav_gallery"
    android:icon="@drawable/ic_menu_gallery"
    app:actionViewClass="android.widget.TextView"
    android:title="Gallery"/>
<item
    android:id="@+id/nav_slideshow"
    android:icon="@drawable/ic_menu_slideshow"
    app:actionViewClass="android.widget.TextView"
    android:title="Slideshow"/>
```

Объявим текстовые метки и инициализируем их в методе **onCreate()**. В отдельном методе будем управлять их свойствами.

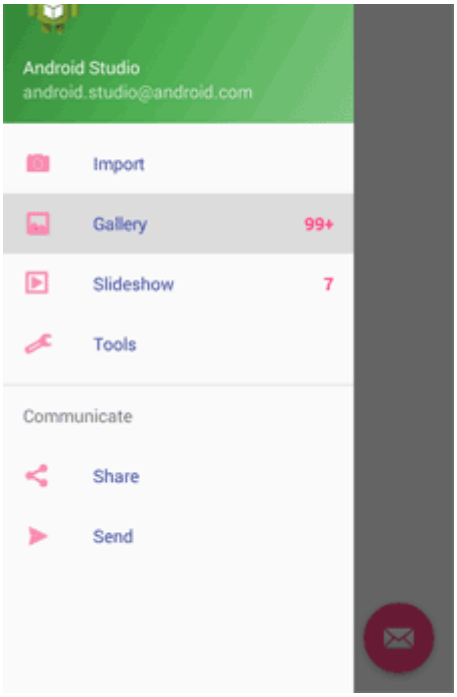
```
private TextView mGalleryTextView;

@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);

    ...

    mGalleryTextView = (TextView)
MenuItemCompat.getActionView(navigationView.getMenu().
            findItem(R.id.nav_gallery));
    mSlideshowTextView = (TextView)
MenuItemCompat.getActionView(navigationView.getMenu().
            findItem(R.id.nav_slideshow));
    // метод для счетчиков
    initializeCountDrawer();
}

private void initializeCountDrawer() {
    mGalleryTextView.setGravity(Gravity.CENTER_VERTICAL);
    mGalleryTextView.setTypeface(null, Typeface.BOLD);
    mGalleryTextView.setTextColor(getResources().getColor(R.color.colorAccent));
    mGalleryTextView.setText("99+");
    mSlideshowTextView.setGravity(Gravity.CENTER_VERTICAL);
    mSlideshowTextView.setTypeface(null, Typeface.BOLD);
    mSlideshowTextView.setTextColor(getResources().getColor(R.color.colorAccent));
    mSlideshowTextView.setText("7");
}
```



Вы можете переделать метод под себя, чтобы динамически изменять показания счётчика.

Затемнение и тень

Когда выдвигается шторка, то основная часть активности затемняется. Можем убрать затемнение или выбрать другой цвет.

```
// выбираем цвет
drawer.setScrimColor(Color.TRANSPARENT);
```

Тень от шторки убирается также просто.

```
drawer.setDrawerElevation(0f);
```

Сдвигаем содержимое экрана

При выдвижении шторки можно сдвинуть основное содержание. Потребуется небольшая модификация кода. Для начала нужно присвоить идентификатор контейнеру **ConstraintLayout** в **content_main.xml**.

```
<android.support.constraint.ConstraintLayout
xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:id="@+id/content"
    ...>
```

размер, используя второи параметр метода **slideOffset**.

```
final ConstraintLayout content = findViewById(R.id.content);

ActionBarDrawerToggle toggle = new ActionBarDrawerToggle(
        this, drawer, toolbar, R.string.navigation_drawer_open, R.string
        .navigation_drawer_close) {
    @Override
    public void onDrawerSlide(View drawerView, float slideOffset) {
        super.onDrawerSlide(drawerView, slideOffset);

        float slideX = drawerView.getWidth() * slideOffset;
        content.setTranslationX(slideX);

                // а также меняем размер
                content.setScaleX(1 - slideOffset);
        content.setScaleY(1 - slideOffset);
    }
};
drawer.addDrawerListener(toggle);
toggle.syncState();
```

Дополнительное чтение

Обсуждение статьи на форуме.

DrawerLayout

Библиотека mxn21/FlowingDrawer с прикольным эффектом.