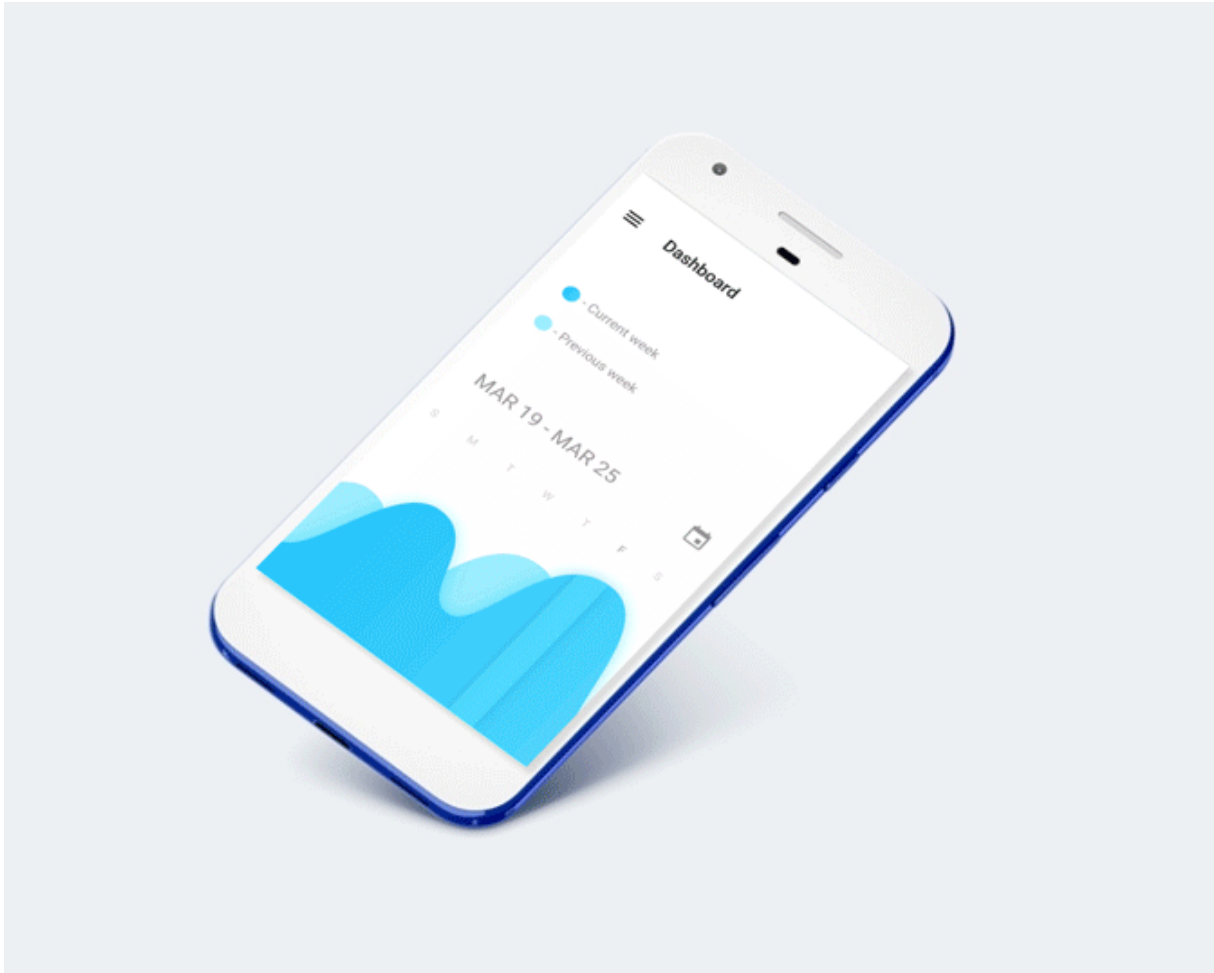


---

## SlidingRootNav

The library is a DrawerLayout-like ViewGroup, where a “drawer” is hidden under the content view, which can be shifted to make the drawer visible. It doesn’t provide you with a drawer builder.



### Gradle

Add this into your dependencies block.

```
1 compile 'com.yarolegovich:sliding-root-nav:1.1.1'
```

### Sample

Please see the sample app for a library usage example.

---

## Wiki

### Usage:

1. Create your content\_view.xml (example) or construct a [View](#) programatically.
2. Set the content view (for example, using [setContentView](#) in your activity).
3. Create your menu.xml (example) or construct a [View](#) programatically.
4. Now you need to inject the menu in your [onCreate](#). You can specify transformations of a content view or use the default ones.

```
1 new SlidingRootNavBuilder(this)
2   .withMenuLayout(R.layout.menu_left_drawer)
3   .inject();
```

## API

**Transformations** You can specify root transformations using [SlidingRootNavBuilder](#).

```
1 new SlidingRootNavBuilder(this)
2   .withDragDistance(140) //Horizontal translation of a view. Default ==
   180dp
3   .withRootViewScale(0.7f) //Content view's scale will be interpolated
   between 1f and 0.7f. Default == 0.65f;
4   .withRootViewElevation(10) //Content view's elevation will be
   interpolated between 0 and 10dp. Default == 8.
5   .withRootViewYTranslation(4) //Content view's translationY will be
   interpolated between 0 and 4. Default == 0
6   .addRootTransformation(customTransformation)
7   .inject();
```

[customTransformation](#) in the above example is a user-created class that implements [RootTransformation](#) interface. For an example, refer to the default transformations.

### Menu behavior

```
1 new SlidingRootNavBuilder(this)
2   .withMenuOpened(true) //Initial menu opened/closed state. Default ==
   false
3   .withMenuLocked(false) //If true, a user can't open or close the menu
   . Default == false.
4   .withGravity(SlideGravity.LEFT) //If LEFT you can swipe a menu from
   left to right, if RIGHT - the direction is opposite.
5   .withSavedState(savedInstanceState) //If you call the method, layout
   will restore its opened/closed state
6   .withContentClickableWhenMenuOpened(isClickable) //Pretty self-
   descriptive. Builder Default == true
```

---

**Controlling the layout** A call to `inject()` returns you an interface for controlling the layout.

```
1 public interface SlidingRootNav {
2     boolean isMenuClosed();
3     boolean isMenuOpened();
4     boolean isMenuLocked();
5     void closeMenu();
6     void closeMenu(boolean animated);
7     void openMenu();
8     void openMenu(boolean animated);
9     void setMenuLocked(boolean locked);
10    SlidingRootNavLayout getLayout(); //If for some reason you need to
        work directly with layout - you can
11 }
```

## Callbacks

- Drag progress:

```
1 builder.addDragListener(listener);
2
3 public interface DragListener {
4     void onDrag(float progress); //Float between 0 and 1, where 1 is a
        fully visible menu
5 }
```

- Drag state changes:

```
1 builder.addDragStateListener(listener);
2
3 public interface DragStateListener {
4     void onDragStart();
5     void onDragEnd(boolean isMenuOpened);
6 }
```

- Compatibility with `DrawerLayout.DrawerListener`:

```
1 DrawerListenerAdapter adapter = new DrawerListenerAdapter(
        yourDrawerListener, viewToPassAsDrawer);
2 builder.addDragListener(listenerAdapter).addDragStateListener(
        listenerAdapter);
```

## Special thanks

Thanks to Tayisiya Yurkiv for a beautiful GIF.

---

## License

```
1 Copyright 2017 Yaroslav Shevchuk
2
3 Licensed under the Apache License, Version 2.0 (the "License");
4 you may not use this file except in compliance with the License.
5 You may obtain a copy of the License at
6
7 http://www.apache.org/licenses/LICENSE-2.0
8
9 Unless required by applicable law or agreed to in writing, software
10 distributed under the License is distributed on an "AS IS" BASIS,
11 WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied
12
13 See the License for the specific language governing permissions and
14 limitations under the License.
```