ADVANCED PROJECT.
MATERIAL DESIGN.

Georgiy Shur

ANDROID DEVELOPER @ ACKEE

#CODECAMP

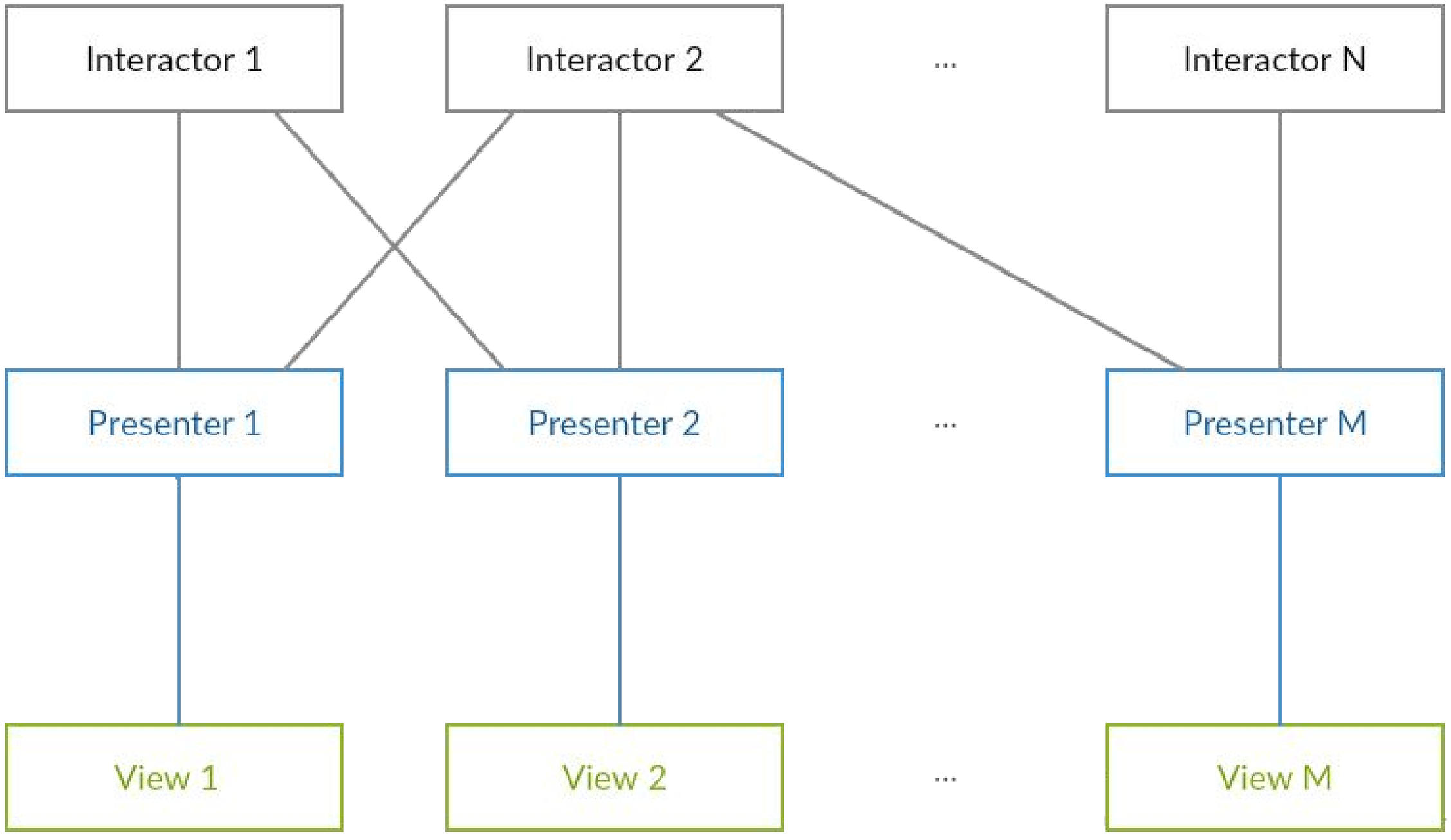
A COUPLE MORE WORDS ABOUT MVP

- Or similar patterns: MVC, MVVM,...
- Separation between data, logic and UI

A COUPLE MORE WORDS ABOUT MVP

- Model - data layer (REST API, Database, Shared preferences, Files etc.)
- View - displays data, accepts user input (Activity, Fragment, View etc.)
- Presenter - serves as “middle-man” between View and Model, contains presentation logic, caches data, is independent from view lifecycle but is aware of it

A COUPLE MORE WORDS ABOUT MVP



A COUPLE MORE WORDS ABOUT MVP

- Interactor - in most cases is singleton
- View - has multiple instances (Android specific)
- Presenter - has multiple instances. Each presenter has at most one view attached.

A COUPLE MORE WORDS ABOUT MVP

- Create interfaces for components is good practice, but depends on your project
- Polymorphism, encapsulation
- In practice makes sense for Views and Interactors

AND A LITTLE BIT ABOUT RXJAVA

- map
- doOnNext
- flatMap

AND A LITTLE BIT ABOUT RXJAVA

```
Observable.just(42)
        .map(integer -> integer + " is magic number")
    .subscribe(string -> // Do smth with our text);
```

AND A LITTLE BIT ABOUT RXJAVA

```
Observable.just(42)
        .map(integer -> integer + " is magic number")
        .subscribe(string -> // Do smth with our text);
```

AND A LITTLE BIT ABOUT RXJAVA

```
Observable.just(42)
        .map(integer -> integer + " is magic number")
.doOnNext(string -> Log.d(TAG, string))
.subscribe(string -> // Do smth with our text);
```

AND A LITTLE BIT ABOUT RXJAVA

```
Observable.just(42)
        .map(integer -> integer + " is magic number")
.doOnNext(string -> Log.d(TAG, string))
.subscribe(string -> // Do smth with our text);
```

AND A LITTLE BIT ABOUT RXJAVA

```
Observable.from(getUsers())
        .doOnNext(user -> user.setAddress(getAddress(user.getId())))
        .toList()
        .subscribe(users -> // Do smth with users);
```

AND A LITTLE BIT ABOUT RXJAVA

```
Observable.from(getUsers())
        .map(user -> getAddress(user.getId()))
        .toList()
        .subscribe(addresses -> // Do smth with addresses);
```

AND A LITTLE BIT ABOUT RXJAVA

```
Observable.just(42)
        .map(integer -> integer + " is magic number")
        .subscribe(string -> // Do smth with our text);
```

```
Observable.just(42)
        .flatMap(integer ->
                Observable.just(integer + " is magic number"))
        .subscribe(string -> // Do smth with our text);
```

AND A LITTLE BIT ABOUT RXJAVA

```
Observable.just(42)
        .map(integer -> integer + " is magic number")
        .subscribe(string -> // Do smth with our text);
```

```
Observable.just(42)
        .flatMap(integer ->
                Observable.just(integer + " is magic number"))
        .subscribe(string -> // Do smth with our text);
```

AND A LITTLE BIT ABOUT RXJAVA

```
Observable.from(getUsers())
        .flatMap(user -> getAddress(user.getId())
                        .doOnNext(address -> user.setAddress(adress))
                        .map(address -> user))
.toList()
.subscribe(users -> // Do smth with users);
```

CODE!

MATERIAL DESIGN

- Not just a bunch of views
- Visual language
- Consistent user experience, logic, branding

MATERIAL DESIGN

- Material - 3D world with paper, light and shadows
- Motion provides meaning

MATERIAL DESIGN

- Designers vs developers
- Pre-L materialization
- Support library

MATERIAL DESIGN

- Styles
- Components
- Patterns

STYLES - COLOR

- colorPrimary, colorPrimaryDark, colorAccent, textColorPrimary, colorControlNormal,...
- http://www.materialpalette.com/
- Components

STYLES - ELEVATION

- Post-L property
- Drawable imitation in older versions
- Partial support in Support Library (CardView, FloatingActionButton)

STYLES - TOUCH FEEDBACK

- Ripple effect post-L
- Separate resource files or ? attr/selectableItemBackground

COMPONENTS - NAVIGATION DRAWER

- DrawerLayout as parent element
- NavigationView vs custom layout as content

COMPONENTS - TOOLBARS AND APP BAR

- Toolbars - versatile bars containing tools :-)
- App bar - special case of toolbar (formerly action bar)
- App bar contains navigation button, title, action menu
- TabLayout + ViewPager for navigation

COMPONENTS - TOOLBARS AND APP BAR

- App bar is connected to navigation drawer through navigation button (ActionBarDrawerToggle)
- AppBarLayout helps to implement Material Design behavior like scrolling
- Additional scrolling features with CollapsingToolbarLayout

And some more… Maybe… But we shouldn’t get to this slide really

DĚKUJI ZA POZORNOST!

OTÁZKY?