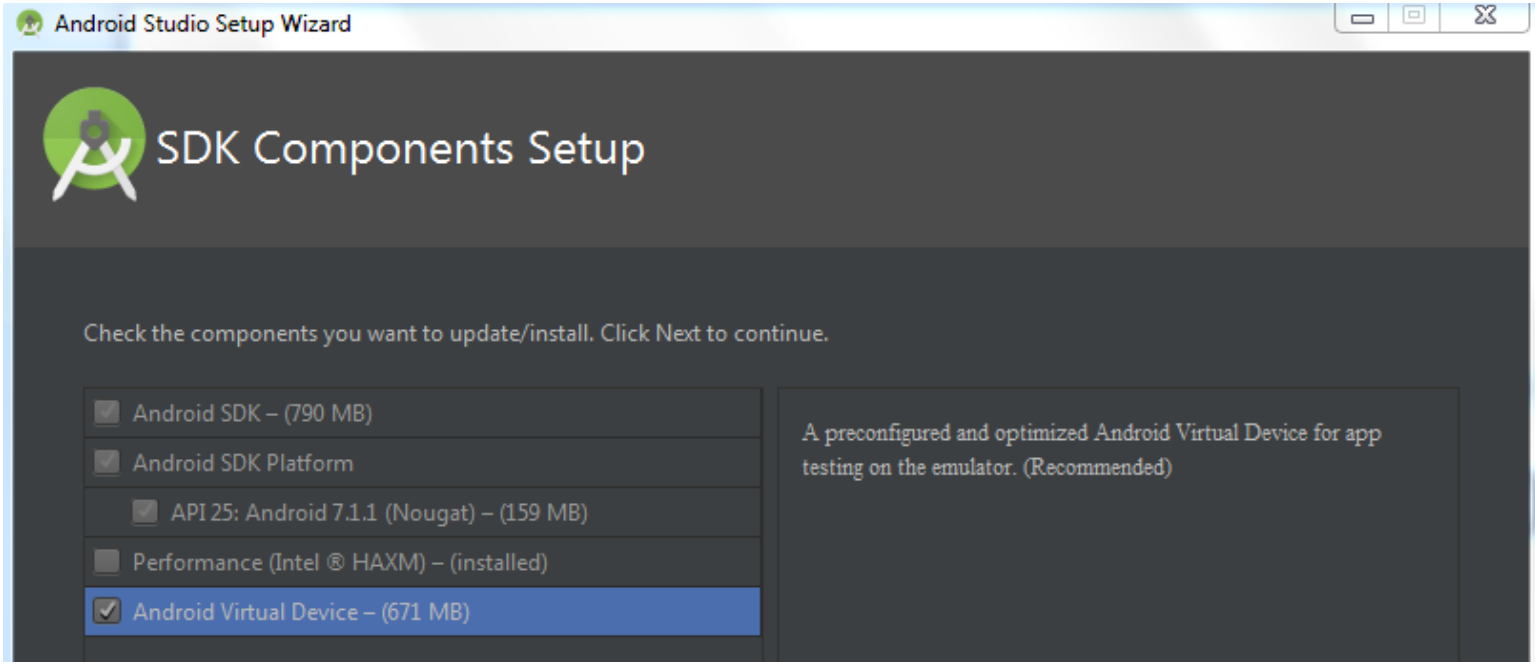
Lab Preparation

Install Android studio

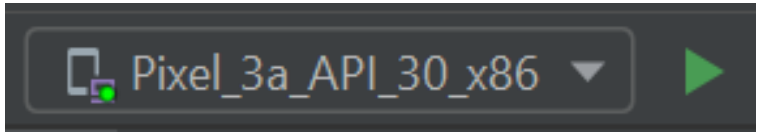
- https://developer.android.com/studio
- Download Android studio, launch .exe
- Select Android Virtual Device



Create and run project

- File > New > New Project
- Empty Activity > Language Java

- Select Device > Run

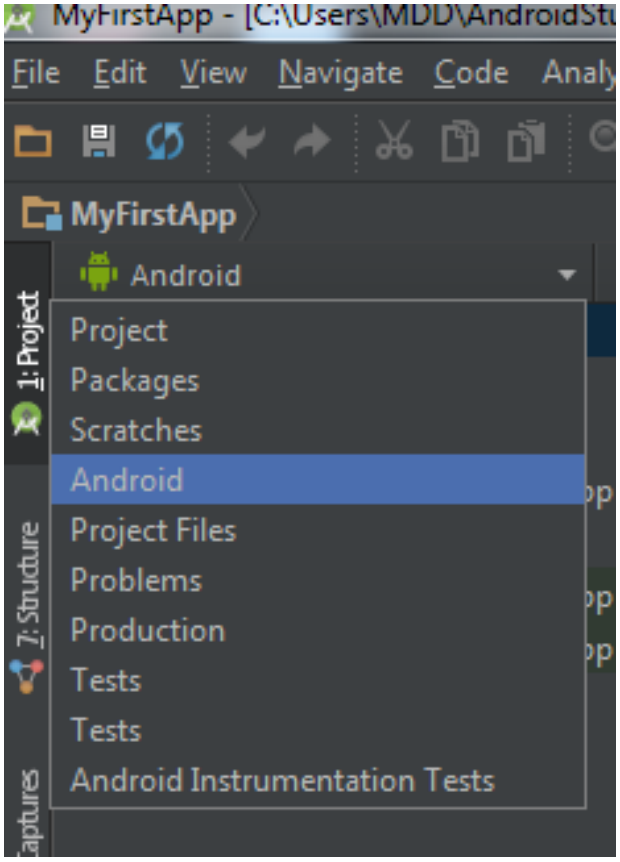


- (install Haxm if needed)

Create and run project

Project Components

- View > Tool Windows > Project > Andriod



Project Components

- app > java > com.example.myfirstapp > MainActivity.java – This is the main activity. It's the entry point for your app. When you build and run your app, the system launches an instance of this Activity and loads its layout.
- app > res > layout > activity_main.xml – This XML file defines the layout for the activity's user interface (UI). Composed of *View* objects. Can be specified for portrait and landscape mode.
- app > manifests > AndroidManifest.xml – The manifest file describes the fundamental characteristics of the app and defines each of its components: Activities/Services/Permissions/Libraries
- res
  - Drawables (like .png images)
  - Values (like strings: res/values/strings.xml)

Extensible Markup Language (XML)

- Preferred way of creating UI
  - Separates the description of the layout from any actual code that controls it
  - Can easily take a UI from one platform to another
  - Both human and machine readable

Lab 1.a Basic UI

Lab 1.a Basic UI

- The app has a interface that contains a button and a text input field.
- Once the button is pushed, generate an intent that invokes a second activity
- The second activity will display the input string

Step 1: create a UI

- app > res > layout > activity_main.xml
- Select the code tab


```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout
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:orientation="horizontal">
</LinearLayout>
```

LinearLayout is a view group that aligns all children in a single direction, vertically or horizontally. You can specify the layout direction with the android:orientation attribute.

Step 1: create a UI

- Add a Text Field Element

```
<LinearLayout
    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:orientation="horizontal">
    <EditText android:id="@+id/edit_message"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:hint="@string/edit_message" />
</LinearLayout>
```

The EditText is the standard text entry widget in Android apps. If the user needs to enter text into an app, this is the primary way for them to do that.

Step 1: create a UI

- res > values > strings.xml
- Define the String field

```
<?xml version="1.0" encoding="utf-8"?>
<resources>
    <string name="app_name">My First App</string>
    <string name="edit_message">Enter a message</string>
    <string name="button_send">Send</string>
</resources>
```

Step 1: create a UI

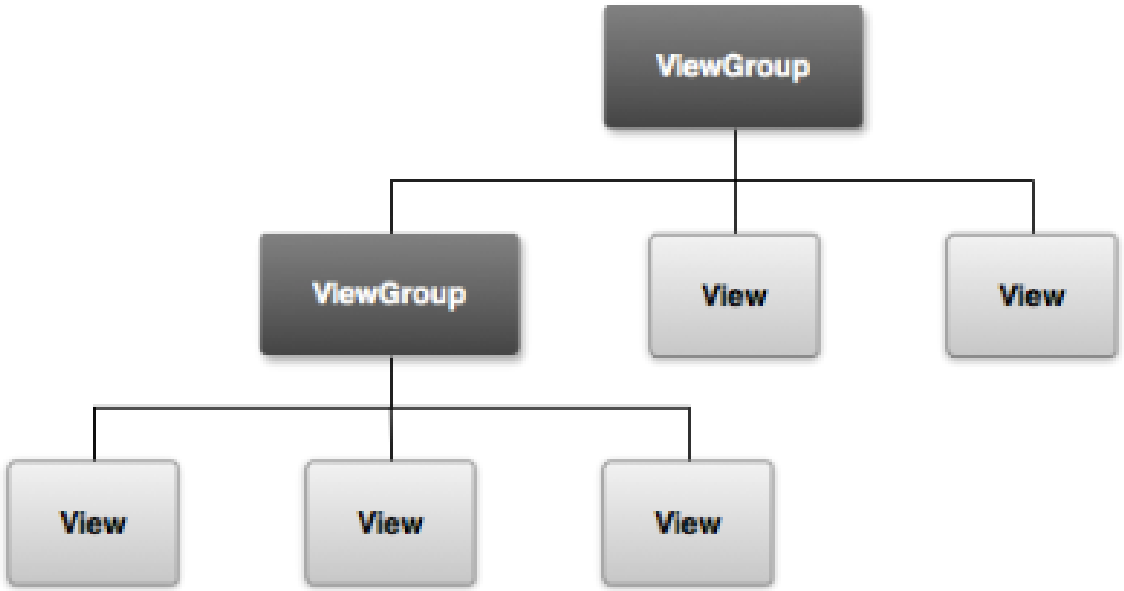
- res > layout > activity_main.xml
- Add a Button

```
<LinearLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:orientation="horizontal"
    android:layout_width="match_parent"
    android:layout_height="match_parent">
        <EditText android:id="@+id/edit_message"
          android:layout_width="wrap_content"
          android:layout_height="wrap_content"
          android:hint="@string/edit_message" />
        <Button
          android:layout_width="wrap_content"
          android:layout_height="wrap_content"
          android:text="@string/button_send" />
</LinearLayout>
```

Button: a user interface element the user can tap or click to perform an action.

Understand Basic UI Elements

- The UI for an Android app is built as a hierarchy of layouts and widgets.
- Layouts are ViewGroup objects, containers that control how their child views are positioned on the screen. e.g. LinearLayout
- Widgets are View objects, UI components e.g. Text field, Button



Run the app


Android Emulator - Nexus_5X_API_25_x86:5554
2:56
My First App
Enter a message
SEND

Play around UI

- Change the string field

```
<?xml version="1.0" encoding="utf-8"?>
<resources>
    <string name="app_name">My First App</string>
    <string name="edit_message">Enter a message</string>
    <string name="button_send">Send</string>
</resources>
```

- Change width of text field

```
<EditText android:id="@+id/edit_message"
    android:layout_weight="1"
    android:layout_width="0dp"
    android:layout_height="wrap_content"
    android:hint="@string/edit_message" />
```

Step 2: control the UI

- res > layout > activity_main.xml
- Add a field to the Button View

```
<Button
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="@string/button_send"
    android:onClick="sendMessage" />
```

Now when the button is tapped, the system calls the sendMessage() method.

Step 2: control the UI

- java > com.example.myfirstapp >MainActivity.java
- Create a Intent (an object that deliver message in run time between separate components, such as two activities)

```
public final static String EXTRA_MESSAGE =
"com.example.myfirstapp.MESSAGE";
public void sendMessage(View view) {
        Intent intent = new Intent(this, DisplayMessageActivity.class);
        EditText editText = (EditText) findViewById(R.id.edit_message);
        String message = editText.getText().toString();
        intent.putExtra(EXTRA_MESSAGE, message);
        startActivity(intent);
    }
```

Some notes

Here's what's going on in `sendMessage()`:

- The `Intent` constructor takes two parameters, a `Context` and a `Class`.

  The `Context` parameter is used first because the `Activity` class is a subclass of `Context`.

  The `Class` parameter of the app component, to which the system delivers the `Intent`, is, in this case, the activity to start.

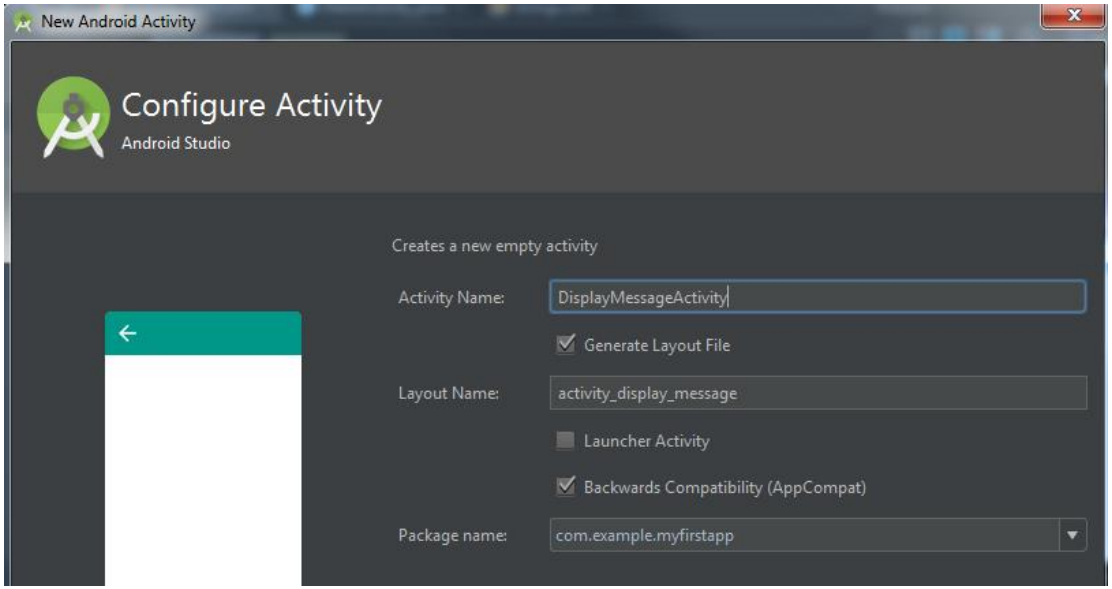
- The `putExtra()` method adds the value of `EditText` to the intent. An `Intent` can carry data types as key-value pairs called *extras*.

  Your key is a public constant `EXTRA_MESSAGE` because the next activity uses the key to retrieve the text value. It's a good practice to define keys for intent extras with your app's package name as a prefix. This ensures that the keys are unique, in case your app interacts with other apps.

- The `startActivity()` method starts an instance of the `DisplayMessageActivity` that's specified by the `Intent`. Next, you need to create that class.

Step 2: control the UI

- Create a new Activity
- Java > com.example.myapplication
- New > activity > empty activity
- Activity name: DisplayMessageActivity



Step 2: control the UI

DisplayMessageActivity.java

```
public class DisplayMessageActivity extends AppCompatActivity {

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

        Intent intent = getIntent();
        String message = intent.getStringExtra(MainActivity.EXTRA_MESSAGE);
        TextView textView = new TextView(this);
        textView.setTextSize(40);
        textView.setText(message);

        ViewGroup layout = (ViewGroup) findViewById(R.id.activity_display_message);
        layout.addView(textView);
    }
}
```

activity_display_message.xml

```
android:id="@+id/activity_display_message"
```

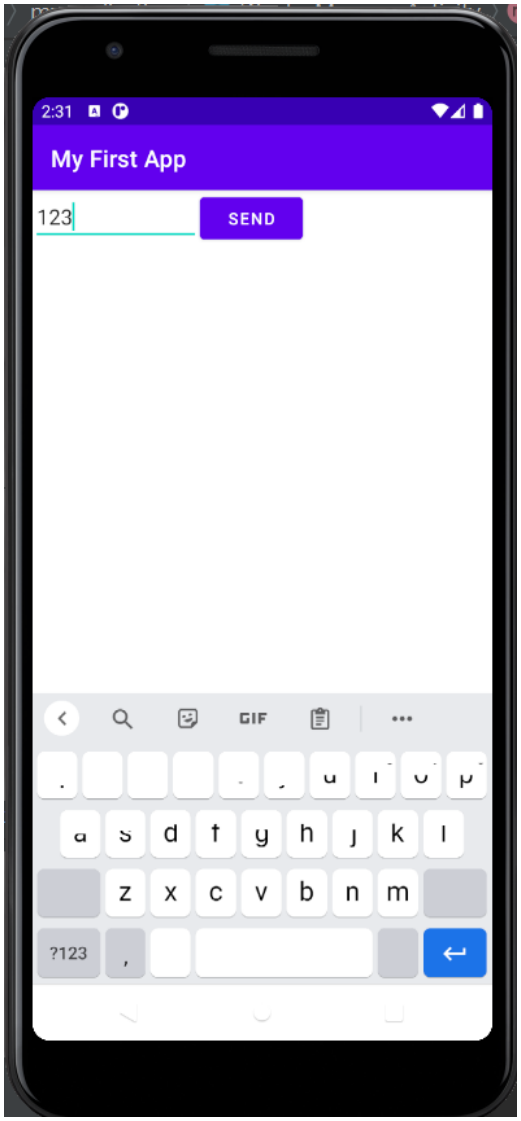
Note: The XML layout generated by previous versions of Android Studio might not include the android:id attribute. The call findViewById() will fail if the layout does not have the android:id attribute. If this is the case, open activity_display_message.xml and add the attribute android:id="@+id/activity_display_message" to the layout element.

Step 2: control the UI

- In MainActivity.java and DisplayMessageActivity.java

Use Alt + Enter to import class

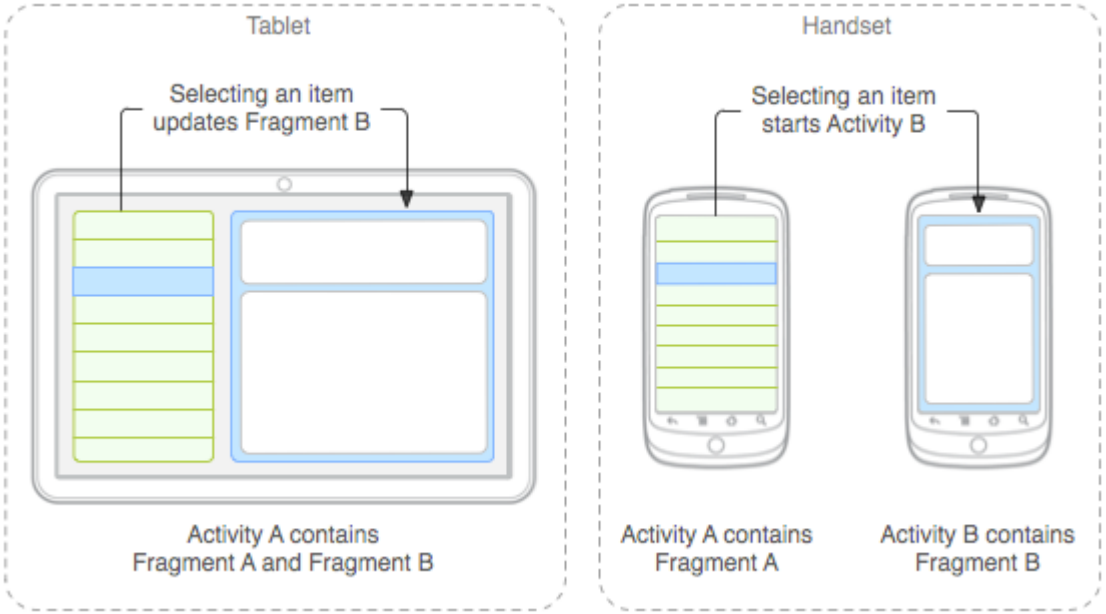
Run the App



Lab 1.b FragmentBasics

Fragments

- A Fragment represents a portion of user interface in an Activity.
- can combine multiple fragments in a single activity, or reuse a fragment in multiple activities.



Lab 1.b FragmentBasics

- Create two fragments. One is a list fragment that contains all news headlines, the other is a fragment that displays the news article
- Display the two fragments inside a single activity

- File > New > New Project
- Empty Activity > Language Java
- Name: FragmentBasics

- Java > com.example.fragmentbasics > New > Java Class
- Create Ipsum.java class (That is a capital i)

```
package com.example.fragmentbasics;

public class Ipsum {
    static String[] Headlines = {
        "Article One",
        "Article Two"
    };
    static String[] Articles = {
        "Article One\n\nExcepteur pour-over occaecat squid biodiesel umami gastropub, nulla laborum salvia dreamcatcher fanny pack. Ullamco culpa retro ea, trust fund excepteur eiusmod direct trade banksy nisi lo-fi cray messenger bag. Nesciunt esse carles selvage put a bird on it gluten-free, wes anderson ut trust fund twee occupy viral. Laboris small batch scenester pork belly, leggings ut farm-to-table aliquip yr nostrud iphone viral next level. Craft beer dreamcatcher pinterest truffaut ethnic, authentic brunch. Esse single-origin coffee banksy do next level tempor. Velit synth dreamcatcher, magna shoreditch in american apparel messenger bag narwhal PBR ennui farm-to-table.",
        "Article Two\n\nVinyl williamsburg non velit, master cleanse four loko banh mi. Enim kogi keytar trust fund pop-up portland gentrify. Non ea typewriter dolore deserunt Austin. Ad magna ethical kogi mixtape next level. Aliqua pork belly thundercats, ut pop-up tattooed dreamcatcher kogi accusamus photo booth irony portland. Semiotics brunch ut locavore irure, enim etsy laborum stumptown carles gentrify post-ironic cray. Butcher 3 wolf moon blog synth, vegan carles odd future."
    };
}
```

- Create a new fragment

Java > com.example.fragmentbasics > New > Fragment > Fragment (Blank)

- Fragment name: NewsFragment

```
public class NewsFragment extends Fragment{
		final static String ARG_POSITION = "position";
		int mCurrentPosition = -1;

		@Override
		public View onCreateView(LayoutInflater inflater, ViewGroup container,
				Bundle savedInstanceState) {

				// Inflate the layout for this fragment
				return inflater.inflate(R.layout.fragment_news, container, false);
		}

		@Override
		public void onStart() {
				super.onStart();

				//During startup, check if there are arguments passed to the fragment.
				// onStart is a good place to do this because the layout has already been
				// applied to the fragment at this point so we can safely call the method
				// below that sets the article text.

				Bundle args = getArguments();
				if (args != null) {

						// Set article based on argument passed in
						updateArticleView(args.getInt(ARG_POSITION));

						} else if (mCurrentPosition != -1) {
								// Set article based on saved instance state defined during onCreateView
								updateArticleView(mCurrentPosition);
						}

				}

				public void updateArticleView(int position) {
						TextView article = (TextView) getActivity().findViewById(R.id.news);
						article.setText(Ipsum.Articles[position]);
						mCurrentPosition = position;
				}

				@Override
				public void onSaveInstanceState(Bundle outState) {
						super.onSaveInstanceState(outState);
						// Save the current article selection in case we need to recreate the fragment
						outState.putInt(ARG_POSITION, mCurrentPosition);
				}
		}
```

- Create a new fragment

Java > com.example.fragmentbasics > New > Fragment > Fragment (List)

- Fragment name: HeadlineFragment

- Java > com.example.fragmentbasics > HeadlineFragment.java

```
public class HeadlineFragment extends ListFragment {
    OnHeadlineSelectedListener mCallback;
    // Container Activity must implement this interface
    public interface OnHeadlineSelectedListener {
        public void onArticleSelected(int position);
    }
    @Override
    public void onAttach(Activity activity) {
        super.onAttach(activity);

        // This makes sure that the container activity has implemented
        // the callback interface. If not, it throws an exception
        try {
            mCallback = (OnHeadlineSelectedListener) activity;
        } catch (ClassCastException e) {
            throw new ClassCastException(activity.toString()
                    + " must implement OnHeadlineSelectedListener");
        }
    }

    @Override
    public void onListItemClick(ListView l, View v, int position, long id) {
        // Notify the parent activity of selected item
        mCallback.onArticleSelected(position);

        // Set the item as checked to be highlighted when in two-pane layout
        getListView().setItemChecked(position, true);
    }
}
```

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

    int layout = android.R.layout.simple_list_item_activated_1;
    setListAdapter(new ArrayAdapter<String>(getActivity(), layout, Ipsum.Headlines));
}

@Override
public void onStart() {
    super.onStart();

    if (getFragmentManager().findFragmentById(R.id.news_fragment) != null) {
        getListView().setChoiceMode(ListView.CHOICE_MODE_SINGLE);
    }
}
```

- Java > com.example.fragmentbasics > MainActivity.java
- implement the interface for the list fragment

```
public class MainActivity extends AppCompatActivity implements
HeadlineFragment.OnHeadlineSelectedListener{

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

    public void onArticleSelected(int position) {

        NewsFragment newsFragment = (NewsFragment)
getSupportFragmentManager().findFragmentById(R.id.news_fragment);
        newsFragment.updateArticleView(position);

    }
}
```

- res > layout > activity_main.xml
- Organize these two fragments

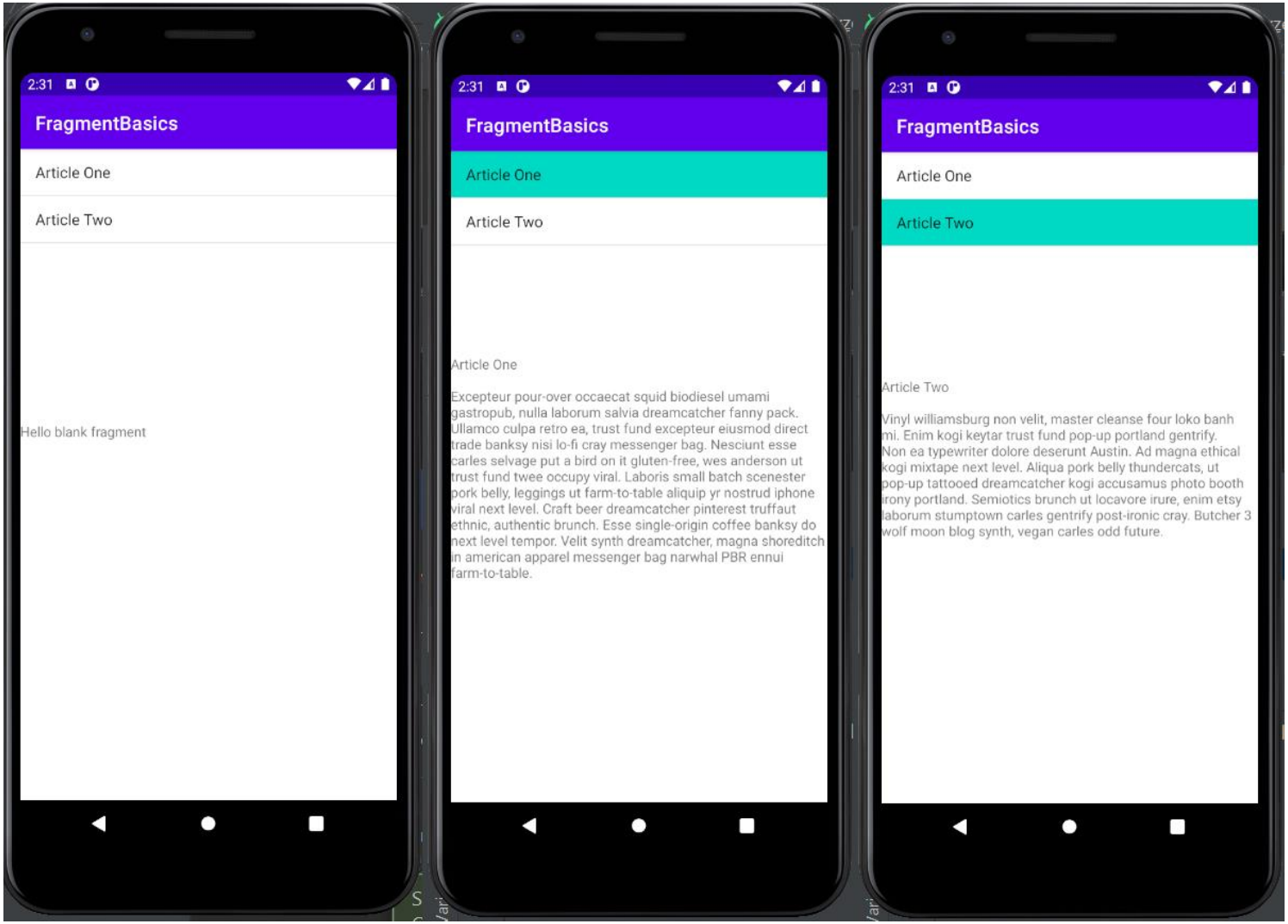
```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
android:orientation="vertical"
android:id="@+id/fragment_container"
android:layout_width="match_parent"
android:layout_height="match_parent">

<fragment android:name="com.example.fragmentbasics.HeadlineFragment"
    android:id="@+id/headlines_fragment"
    android:layout_weight="1"
    android:layout_width="match_parent"
    android:layout_height="wrap_content" />

<fragment android:name="com.example.fragmentbasics.NewsFragment"
    android:id="@+id/news_fragment"
    android:layout_weight="2"
    android:layout_width="match_parent"
    android:layout_height="wrap_content" />

</LinearLayout>
```

- In HeadlineFragment.java and NewsFragment.java, use Alt + Enter to import class

- res > layout > fragment_news.xml

```
<TextView
    android:id="@+id/news"
    android:layout_width="match_parent"
    android:layout_height="match_parent“
    android:text="@string/hello_blank_fragment" />
```

Run the app


2:31
FragmentBasics
Article One
Article Two
Hello blank fragment
2:31
FragmentBasics
Article One
Article Two
Article One
Excepteur pour-over occaecat squid biodiesel umami gastropub, nulla laborum salvia dreamcatcher fanny pack. Ullamco culpa retro ea, trust fund excepteur eiusmod direct trade banksy nisi lo-fi cray messenger bag. Nesciunt esse carles selvage put a bird on it gluten-free, wes anderson ut trust fund twee occupy viral. Laboris small batch scenester pork belly, leggings ut farm-to-table aliquip yr nostrud iphone viral next level. Craft beer dreamcatcher pinterest truffaut ethnic, authentic brunch. Esse single-origin coffee banksy do next level tempor. Velit synth dreamcatcher, magna shoreditch in american apparel messenger bag narwhal PBR ennui farm-to-table.
2:31
FragmentBasics
Article One
Article Two
Article Two
Vinyl williamsburg non velit, master cleanse four loko banh mi. Enim kogi keytar trust fund pop-up portland gentrify. Non ea typewriter dolore deserunt Austin. Ad magna ethical kogi mixtape next level. Aliqua pork belly thundercats, ut pop-up tattooed dreamcatcher kogi accusamus photo booth irony portland. Semiotics brunch ut locavore irure, enim etsy laborum stumptown carles gentrify post-ironic cray. Butcher 3 wolf moon blog synth, vegan carles odd future.

Other activities

- Play around different layouts
  - Table Layout
  - RelativeLayout
  - FrameLayout
- Run the app in the phone

Assignment

- Finish both lab 1.a and 1.b.

Show TA the results on Jan 26/Jan 28,

or at the latest Feb 2/Feb 4

- Extra credits: change of phone orientations. Build an app with two fragments (a headline fragment and a news fragment). The app should display only a single fragment when the phone is held in portray position, and display both fragments at the same time when the phone is held in landscape position.

References

- Developer's Guide
  - http://developer.android.com/guide/index.html
- API Reference
  - http://developer.android.com/reference/packages.html

- A good webpage
  - http://users.jyu.fi/~mijoahon/android/

References

- Install Android Studio (https://developer.android.com/studio/install.html)

- Install Android Virtual Device

https://developer.android.com/studio/run/managing-avds.html

- Install drivers for Android phone

(Windows:https://developer.android.com/studio/run/oem-usb.html
*nix:https://developer.android.com/studio/run/device.html)

- Enable Android Development

https://www.kingoapp.com/root-tutorials/how-to-enable-usb-debugging-mode-on-android.htm

References

- First app:

https://developer.android.com/training/basics/firstapp/creating-project.html

- Lab 1.a

https://developer.android.com/training/basics/firstapp/creating-project.html

- Lab 1.b

https://developer.android.com/training/basics/fragments/index.html