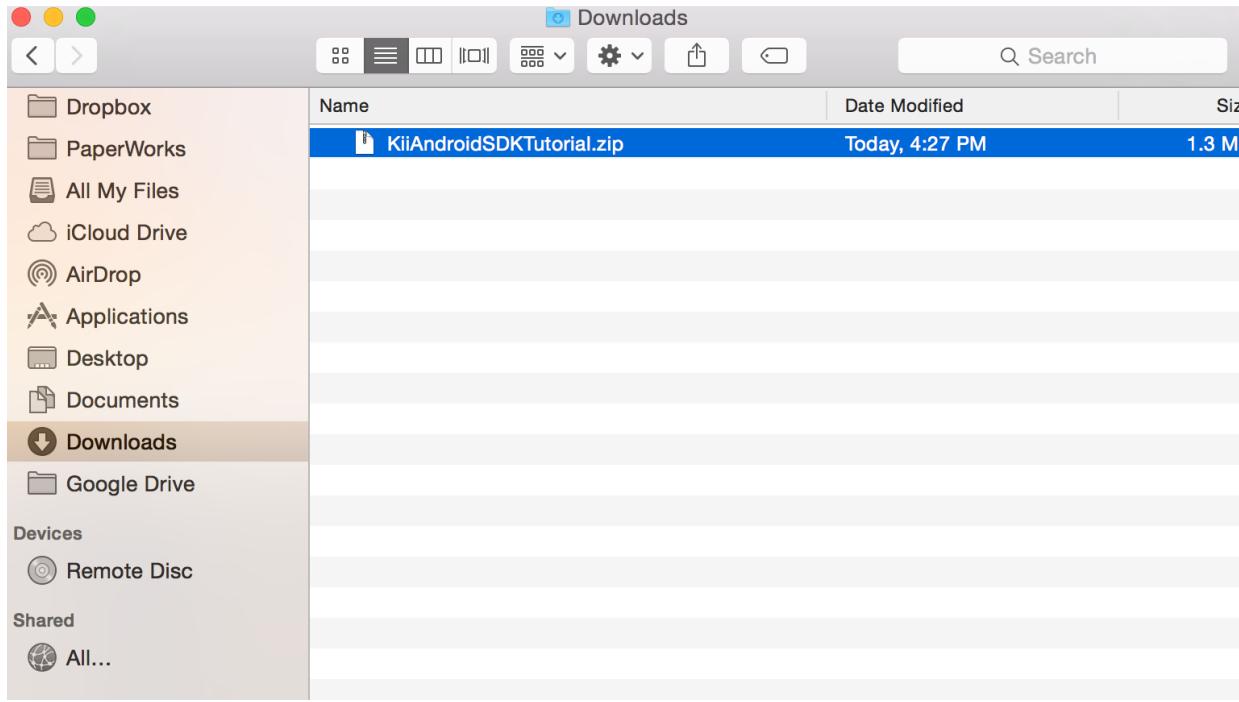


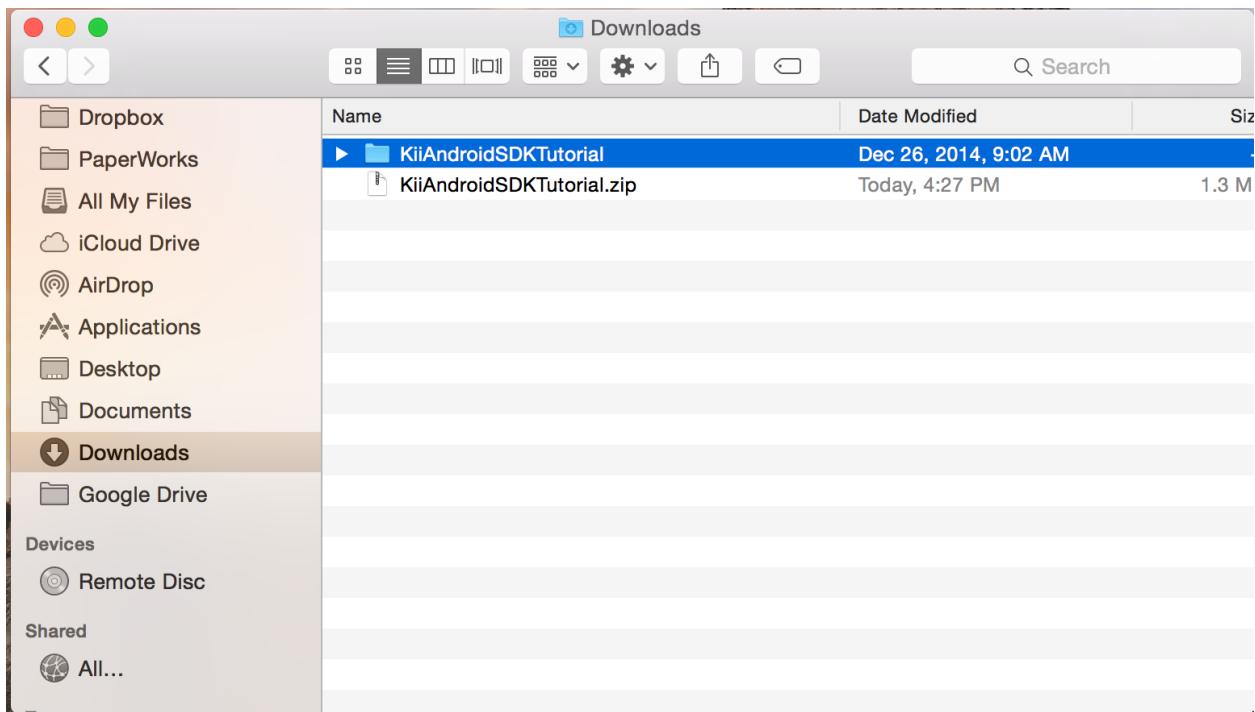
Android Tutorial Guide Page - All Steps

Start



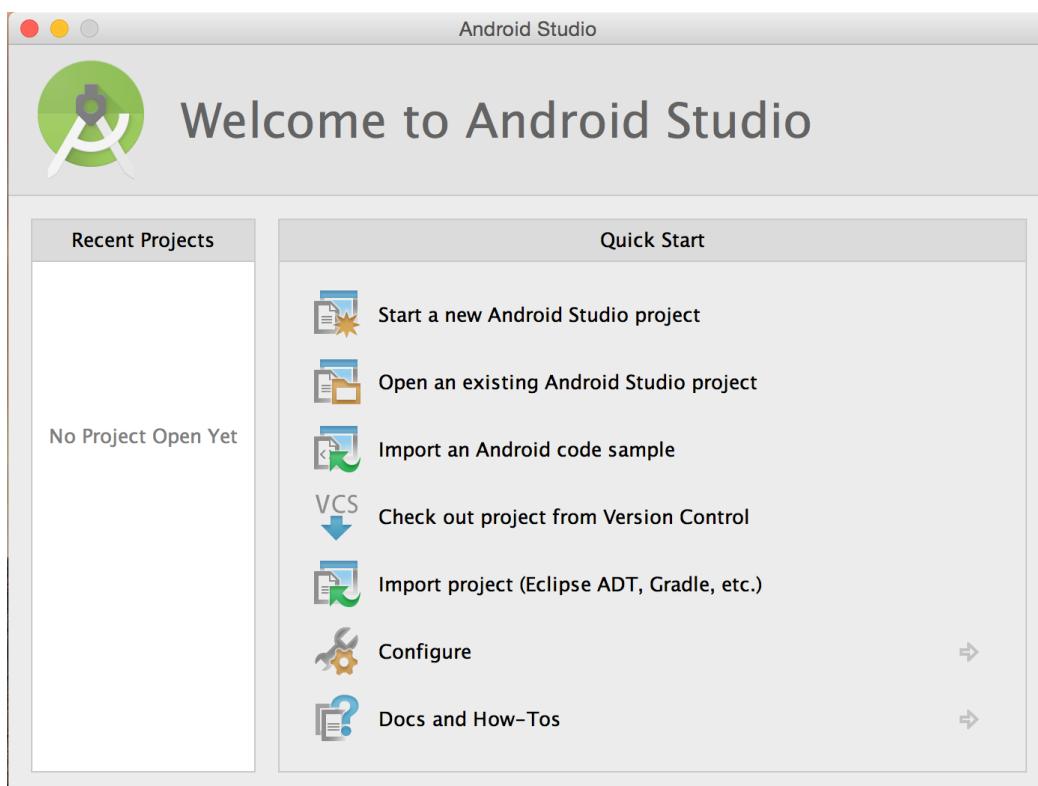
- **Download Tutorial**
 - (Android) - KiiAndroidSDKTutorial.zip

Step 1



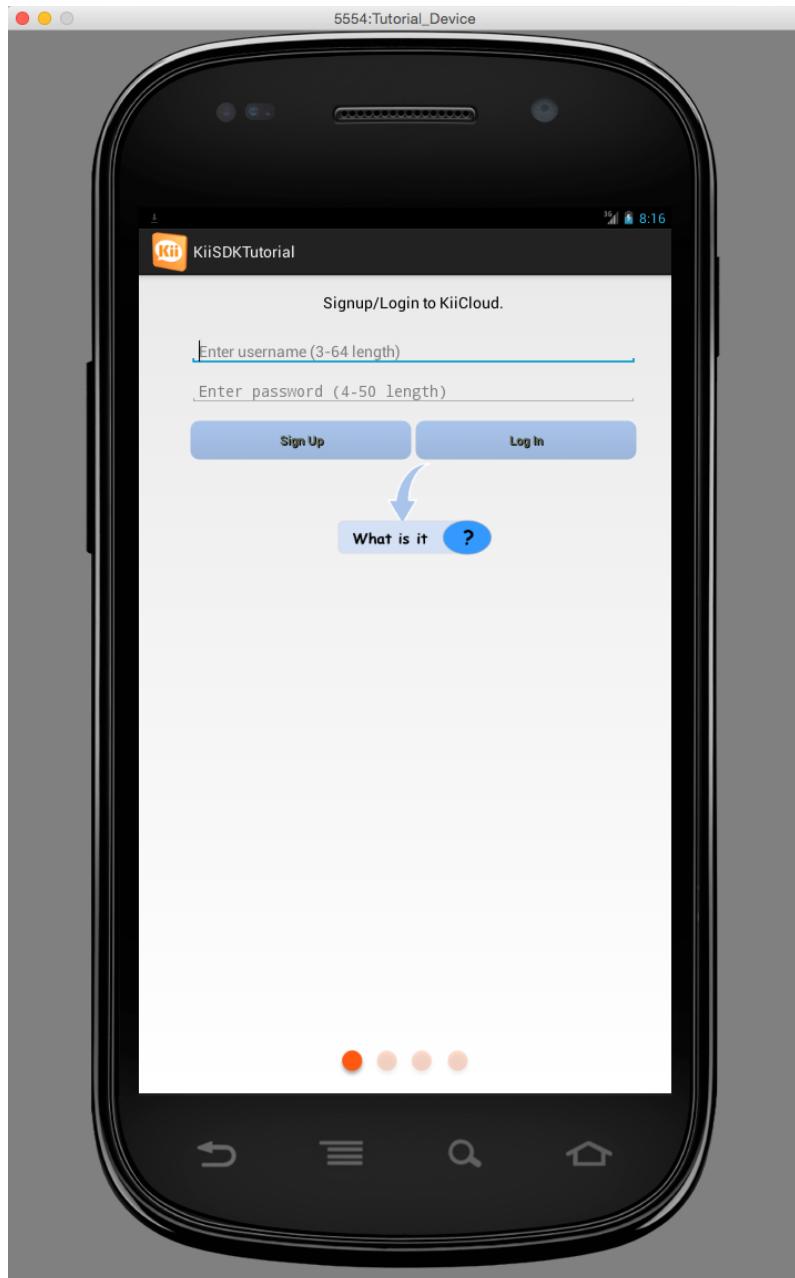
- **Unzip file**

Step 2



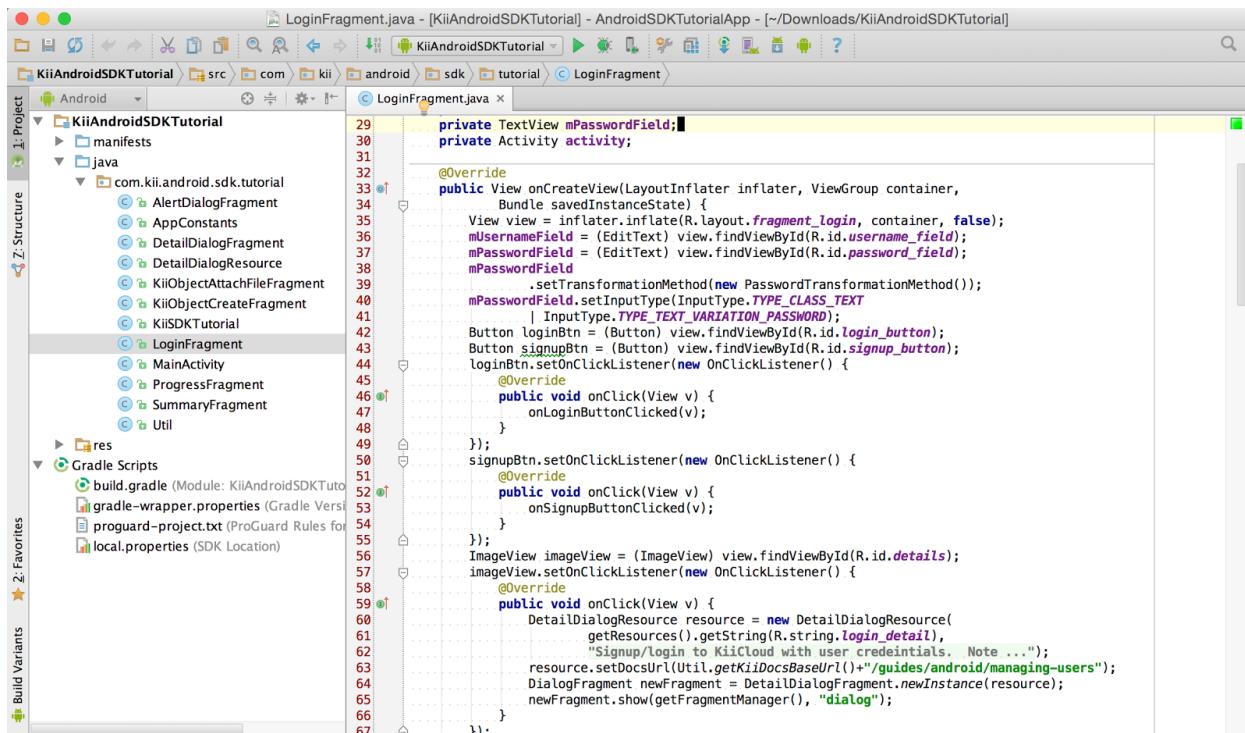
- **Open Android Studio**
 - If you don't already have Android Studio, go to download page of Android Studio. (<http://developer.android.com/sdk/index.html>)
- **Open the unzipped folder with Android Studio**
 - Click the button "Open an existing Android Studio project"
 - Choose the unzipped "KiiAndroidSDKTutorial" folder

Step 3



- **Run the project on the Android emulator or your Android device**
 - Input a username & password for either Sign Up or Log In and hit either button to start.
- **Follow instructions on the sample app:** to create users, objects, and upload file

Step 4



The screenshot shows the Android Studio interface with the project 'KiiAndroidSDKTutorial' open. The 'LoginFragment.java' file is selected in the editor. The code implements a fragment for user login, handling view creation, button clicks, and resource loading.

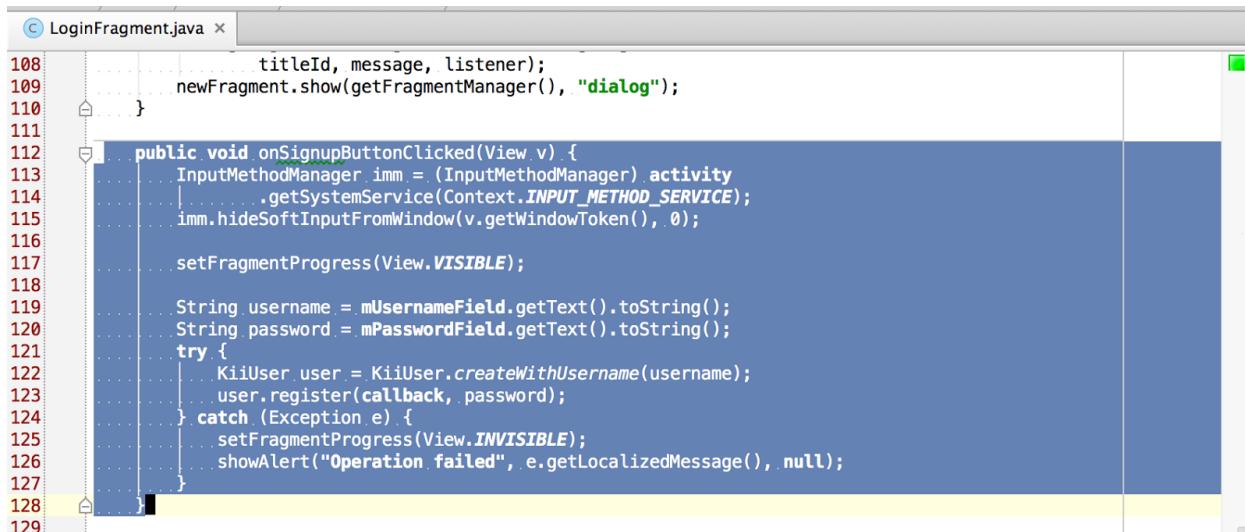
```
private TextView mPasswordField;
private Activity activity;

@Override
public View onCreateView(LayoutInflater inflater, ViewGroup container,
                         Bundle savedInstanceState) {
    View view = inflater.inflate(R.layout.fragment_login, container, false);
    mUsernameField = (EditText) view.findViewById(R.id.username_field);
    mPasswordField = (EditText) view.findViewById(R.id.password_field);
    mPasswordField.setTransformationMethod(new PasswordTransformationMethod());
    mPasswordField.setInputType(InputType.TYPE_CLASS_TEXT
        | InputType.TYPE_TEXT_VARIATION_PASSWORD);
    Button loginBtn = (Button) view.findViewById(R.id.login_button);
    Button signupBtn = (Button) view.findViewById(R.id.signup_button);
    loginBtn.setOnClickListener(new OnClickListener() {
        @Override
        public void onClick(View v) {
            onLoginButtonClicked(v);
        }
    });
    signupBtn.setOnClickListener(new OnClickListener() {
        @Override
        public void onClick(View v) {
            onSignupButtonClicked(v);
        }
    });
    ImageView imageView = (ImageView) view.findViewById(R.id.details);
    imageView.setOnClickListener(new OnClickListener() {
        @Override
        public void onClick(View v) {
            DetailDialogResource resource = new DetailDialogResource(
                getResources().getString(R.string.login_detail),
                "Signup/Login to KiiCloud with user credentials. Note ...");
            resource.setDocUrl(Util.getKiiDocsBaseUrl() + "/guides/android/managing-users");
            DialogFragment newFragment = DetailDialogFragment.newInstance(resource);
            newFragment.show(getFragmentManager(), "dialog");
        }
    });
}

private void onLoginButtonClicked(View v) {
    String username = mUsernameField.getText().toString();
    String password = mPasswordField.getText().toString();
    try {
        KiiUser user = KiiUser.createWithUsername(username);
        user.register(callback, password);
    } catch (Exception e) {
        setFragmentProgress(View.INVISIBLE);
        showAlert("Operation failed", e.getLocalizedMessage(), null);
    }
}
```

- Look at file **LoginFragment.java** for the code to create a user

Step 5



The screenshot shows the continuation of the `onSignupButtonClicked()` method from the previous code snippet. It handles user input, hides the soft keyboard, sets a progress bar, and attempts to register the user with the Kii User API.

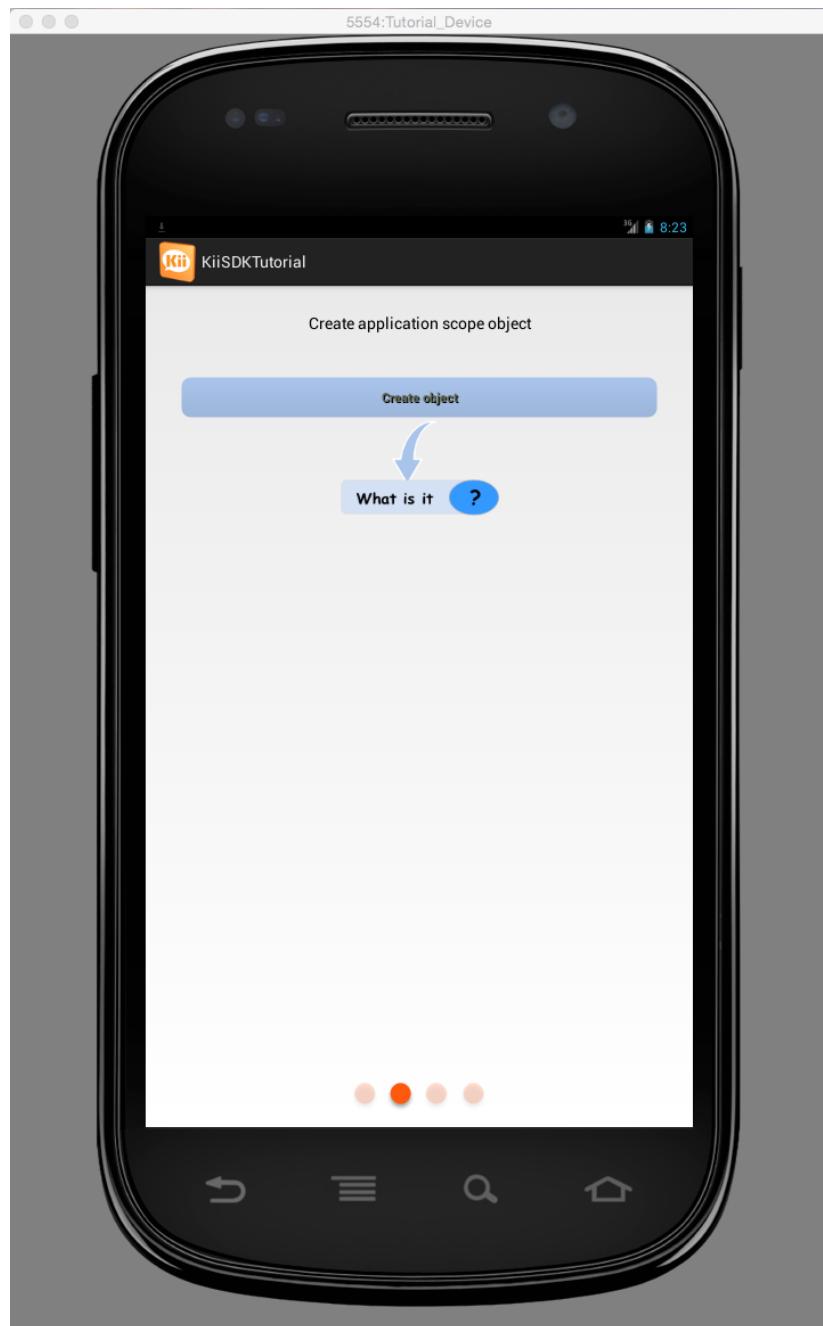
```
    ...
    imm.hideSoftInputFromWindow(v.getWindowToken(), 0);

    setFragmentProgress(View.VISIBLE);

    String username = mUsernameField.getText().toString();
    String password = mPasswordField.getText().toString();
    try {
        KiiUser user = KiiUser.createWithUsername(username);
        user.register(callback, password);
    } catch (Exception e) {
        setFragmentProgress(View.INVISIBLE);
        showAlert("Operation failed", e.getLocalizedMessage(), null);
    }
}
```

- Code to register a user

Step 6



- Hit button “Create KiiObject” in application, check dev portal(inside buckets - tutorial)

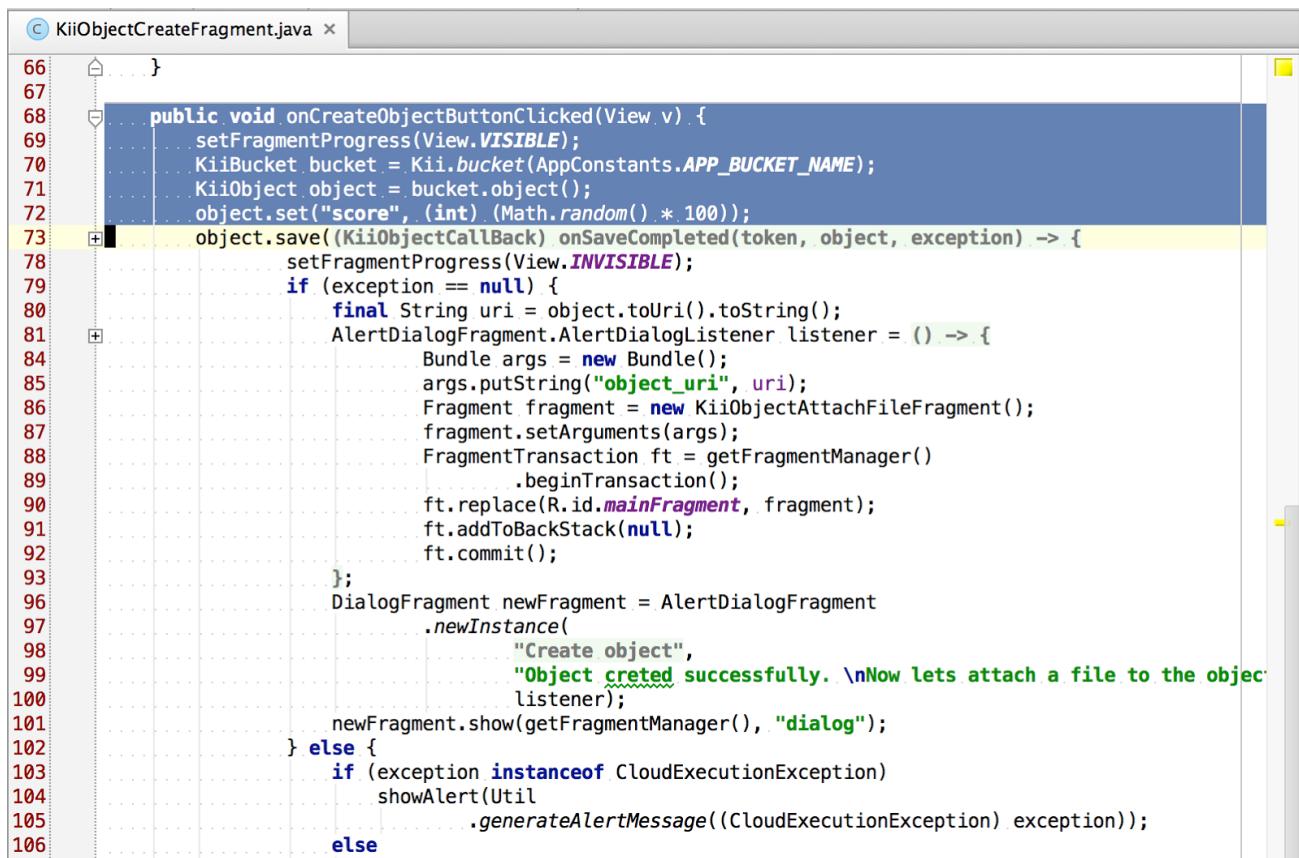
Step 7

The screenshot shows the Android Studio interface with the project 'KiiAndroidSDKTutorial' open. The code editor displays the file 'KiiObjectCreateFragment.java'. The class 'KiiObjectCreateFragment' is defined, extending Fragment. It overrides the onCreateView method to inflate a layout and set up a button click listener. The button's onClickListener calls a method named 'onCreateObjectClicked'. The code also handles progress bar visibility and other UI logic. The Java code is color-coded for syntax highlighting.

```
1 package com.kii.android.sdk.tutorial;
2
3 import ...
4
5 public class KiiObjectCreateFragment extends Fragment {
6     @Override
7     public View onCreateView(LayoutInflater inflater, ViewGroup container,
8         Bundle savedInstanceState) {
9         View view = inflater.inflate(R.layout.fragment_create_object,
10             container, false);
11         Button createObjectBtn = (Button) view
12             .findViewById(R.id.create_object_button);
13         createObjectBtn.setOnClickListener((v) -> {
14             KiiObjectCreateFragment.this.onCreateObjectClicked(
15                 v));
16         });
17         setPageImage(2);
18         ImageView imageView = (ImageView) view.findViewById(R.id.details);
19         imageView.setOnClickListener((v) -> {
20             DetailDialogResource resource = new DetailDialogResource(
21                 "KiiObject details",
22                 "Creating object means storing arbitrary key/value");
23             resource.setId(R.drawable.datastore);
24             resource.setDocsUrl(Util.getKiiDocsBaseUrl() + "/guides/android");
25             DetailDialogFragment newFragment = DetailDialogFragment.newInstance();
26             newFragment.show(getFragmentManager(), "dialog");
27         });
28         return view;
29     }
30
31     void setFragmentProgress(int v) {
32         ProgressFragment fragment = (ProgressFragment) getFragmentManager()
33             .findFragmentById(R.id.progressFragment);
34         if (fragment != null && fragment.isInLayout()) {
35             fragment.setProgressBarVisibility(v);
36         }
37     }
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59 }
```

- Look at file **KiiObjectCreateFragment.java** for the code to create an object

Step 8

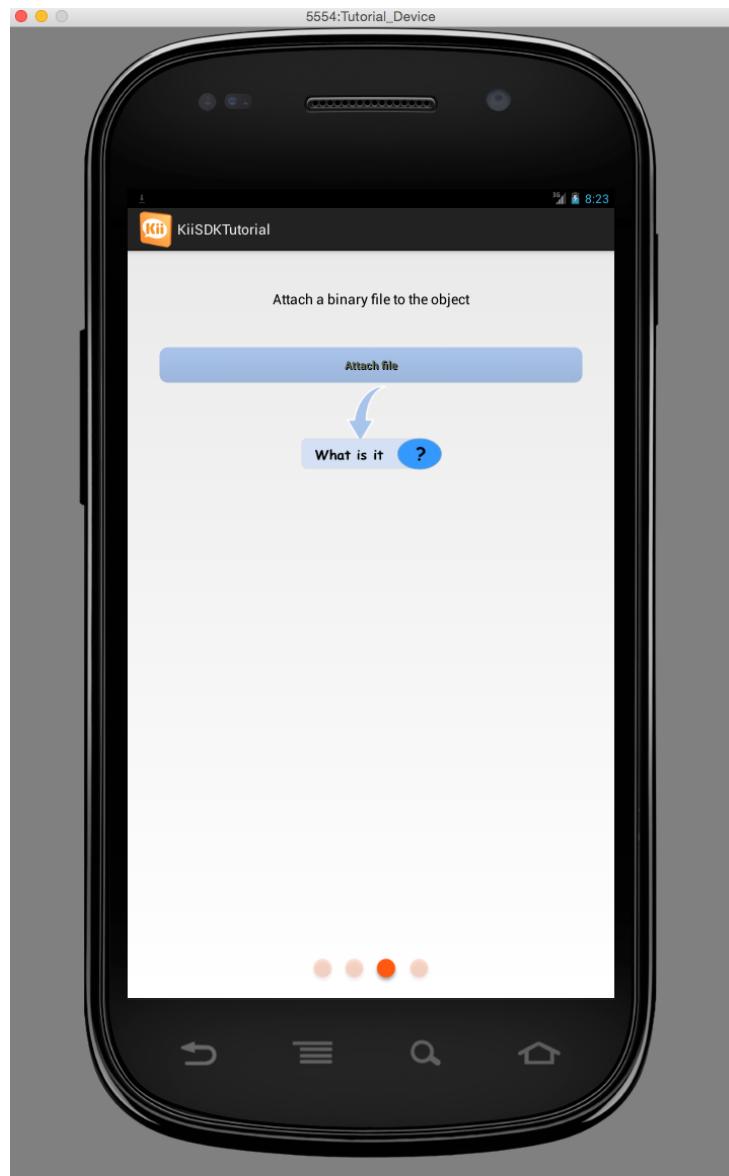


The screenshot shows the Java code for the `KiiObjectCreateFragment.java` file. The code handles the creation of a new object in a bucket and displays a success message in a dialog fragment.

```
66     }
67
68     public void onCreateObjectButtonClicked(View v) {
69         setFragmentProgress(View.VISIBLE);
70         KiiBucket bucket = Kii.bucket(AppConstants.APP_BUCKET_NAME);
71         KiiObject object = bucket.object();
72         object.set("score", (int)(Math.random() * 100));
73         object.save((KiiObjectCallBack) onSaveCompleted(token, object, exception) -> {
74             setFragmentProgress(View.INVISIBLE);
75             if (exception == null) {
76                 final String uri = object.toUri().toString();
77                 AlertDialogFragment.AlertDialogListener listener = () -> {
78                     Bundle args = new Bundle();
79                     args.putString("object_uri", uri);
80                     Fragment fragment = new KiiObjectAttachFileFragment();
81                     fragment.setArguments(args);
82                     FragmentTransaction ft = getFragmentManager()
83                         .beginTransaction();
84                     ft.replace(R.id.mainFragment, fragment);
85                     ft.addToBackStack(null);
86                     ft.commit();
87                 };
88                 DialogFragment newFragment = AlertDialogFragment
89                     .newInstance(
90                         "Create object",
91                         "Object created successfully. Now lets attach a file to the object",
92                         listener);
93                 newFragment.show(getFragmentManager(), "dialog");
94             } else {
95                 if (exception instanceof CloudExecutionException)
96                     showAlert(Util
97                         .generateAlertMessage((CloudExecutionException) exception));
98             }
99         });
100    }
101 }
```

- **Code to create objects**

Step 9



- Hit button “Attach File” in application, check dev portal (data browser)
 - you can download the uploaded image/jpeg file or open the file from browser by a generated publishing url of this data.

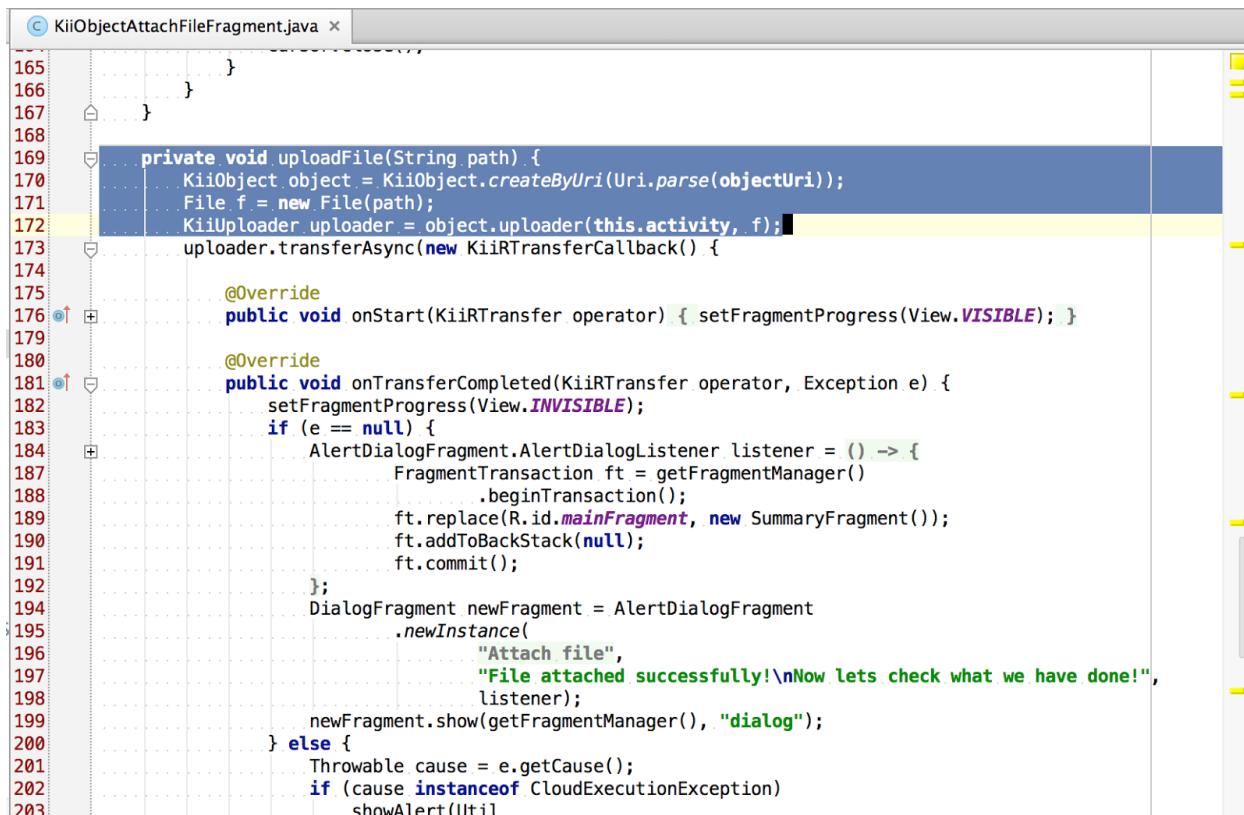
Step 10

The screenshot shows the Android Studio interface with the code editor open to the file `KiiObjectAttachFileFragment.java`. The code implements a fragment for attaching files to a Kii Object. It includes imports for `com.kii.android.sdk.tutorial`, `android.R`, and `DetailDialogResource`. The class `KiiObjectAttachFileFragment` extends `Fragment` and overrides `onCreateView` to inflate a layout and handle file attachment logic. It also handles button click events to attach files and show a dialog.

```
1 package com.kii.android.sdk.tutorial;
2
3 import ...
4
5 public class KiiObjectAttachFileFragment extends Fragment {
6     private static final String TAG = "KiiObjectAttachFileFragment";
7     String objectUri = null;
8     private static final int PICK_IMAGE = 1;
9     private Activity activity;
10
11     @Override
12     public View onCreateView(LayoutInflater inflater, ViewGroup container,
13             Bundle savedInstanceState) {
14         View view = inflater.inflate(R.layout.fragment_attach_file, container,
15             false);
16         Bundle args = getArguments();
17         objectUri = args.getString("object_uri");
18         Button attachButton = (Button) view
19             .findViewById(R.id.attach_file_button);
20         attachButton.setOnClickListener((v) -> { onAttachFileButtonClick(v); });
21         ImageView imageView = (ImageView) view.findViewById(R.id.details_image);
22         imageView.setOnClickListener((v) -> {
23             DetailDialogResource resource = new DetailDialogResource(
24                 "Attach file to KiiObject",
25                 "The file will be uploaded to kiicloud associated with your account",
26                 resource.setImageResource(R.drawable.bodyattach),
27                 resource.setDocsUrl(Util.getKiDocsBaseUrl() + "/guides/android/attachment/"));
28             DetailDialogFragment newFragment = DetailDialogFragment.newInstance(resource);
29             newFragment.show(getFragmentManager(), "dialog");
30         });
31         return view;
32     }
33
34     @Override
35     public void onActivityResult(int requestCode, int resultCode, Intent data) {
36         if (requestCode == PICK_IMAGE) {
37             if (resultCode == RESULT_OK) {
38                 Uri uri = data.getData();
39                 // Handle file upload logic here
40             }
41         }
42     }
43 }
```

- Look at file `KiiObjectAttachFileFragment.java` for the code to handle files

Step 11



```
165     }
166 }
167 }
168
169 private void uploadFile(String path) {
170     KiiObject object = KiiObject.createByUrl(Uri.parse(objectUri));
171     File f = new File(path);
172     KiiUploader uploader = object.uploader(this.activity, f);
173     uploader.transferAsync(new KiiRTransferCallback() {
174
175         @Override
176         public void onStart(KiiRTransfer operator) { setFragmentProgress(View.VISIBLE); }
177
178         @Override
179         public void onTransferCompleted(KiiRTransfer operator, Exception e) {
180             setFragmentProgress(View.INVISIBLE);
181             if (e == null) {
182                 AlertDialogFragment.AlertDialogListener listener = () -> {
183                     FragmentTransaction ft = getFragmentManager()
184                         .beginTransaction();
185                     ft.replace(R.id.mainFragment, new SummaryFragment());
186                     ft.addToBackStack(null);
187                     ft.commit();
188                 };
189                 DialogFragment newFragment = AlertDialogFragment
190                     .newInstance(
191                         "Attach file",
192                         "File attached successfully!\nNow lets check what we have done!",
193                         listener);
194                 newFragment.show(getFragmentManager(), "dialog");
195             } else {
196                 Throwable cause = e.getCause();
197                 if (cause instanceof CloudExecutionException)
198                     showAlert(Util
199
200
201
202
203 }
```

- **Code to create and upload a file**

End

Congratulations!

You have successfully completed the Android Tutorial and you are now ready to create your app.