CS 370 Lab 2: Intents and Extras

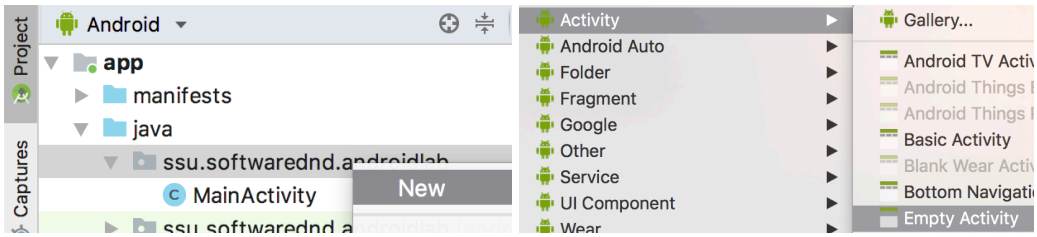
Obtain Code

1. Ensure that the lab workstation is booted into OS X
2. Create a new folder on your desktop called Repositories
3. Open a terminal session (Applications -> Utilities -> Terminal)
    - This is just a terminal window on your local machine! ***Do not log in to blue!***
4. Ensure there are no other default accounts in the OSX keychain (if on public/lab computer):
    a. Keychain management instructions: https://kb.wisc.edu/helpdesk/page.php?id=2197
    b. Search for any *github.com* entries and remove them
5. Using the command line, change directory to the Repositories folder you just created
6. Clone the repository: **git clone https://github.com/ssu-cs370-s19/AndroidLab2.git**
7. Change directory to the AndroidLab2 folder you just cloned
8. **Create a new branch** in the git repository called **LastnameFirstname**
9. Open Android Studio.
10. Open the Android project that you just cloned.

Basic Intents

1. Using Android Studio, create a new empty Activity and call it *OtherActivity*.

    file → new → Activity → Empty Activity



Android Studio will create both the layout resource file and the Java source code for you!

It may ask if you'd like to add these files to git – do it.

We now have a basic Activity that we can switch to from *MainActivity*.

2. Open *strings.xml* and add these string resources:

```
<string name="other_welcome_text">Welcome to OtherActivity!"</string>
<string name="navigate_button_text">Go to OtherActivity</string>
```

3. Open *activity_other.xml*. Add a **TextView** to the layout.
    a. Give it an android:text attribute of "@string/other_welcome_text"

    This is what we will see when entering the *OtherActivity*.

4. Open *activity_main.xml* and add a **Button** element.
    a. Give it an android:id of "@+id/navigate_button"
    b. Give it an android:text of "@string/navigate_button_text"

    We'll use this button to make the switch from Main to Other activity.

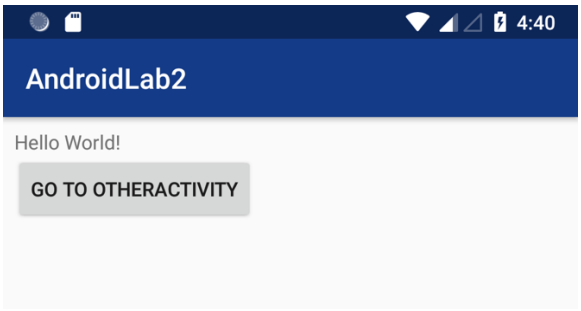
5. Open *MainActivity.java*. Add a private **Button** variable called navigateButton
6. Inside the *onCreate* method, do the following:
    a. use findViewById to get the *R.id.navigate_button* resource and assign it to *navigateButton*
    b. add an OnClickListener to *navigateButton*:

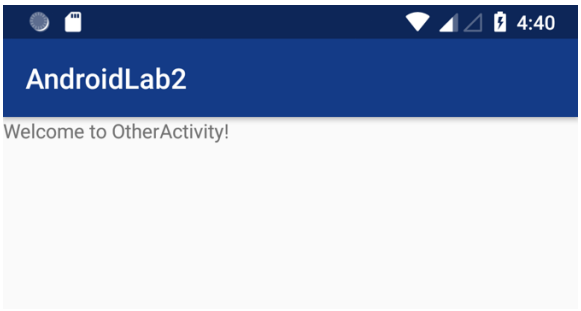
```
navigateButton.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View view) {
        Intent intent = new Intent(MainActivity.this, OtherActivity.class);
        startActivity(intent);
    }
});
```

    The first line in onClick creates an Intent, with the current Context and the Activity to start. An Intent represents an action that you want to happen. Creating one does not make the action happen right away.

    The second line actually starts the Activity defined in the Intent (makes the action happen).

7. If your app is running correctly, when you press the button you should see a different screen come up, and it should have text that says **Welcome to OtherActivity!**. If not, revisit the above steps and make sure you've followed the instructions.





8. When your app runs properly, execute the following commands to commit your changes to your branch:
    a. git add .
    b. git commit -m "part 1 complete"

Your code branch is now saved and committed to the ***local*** git repository. It has not yet been pushed up to GitHub.

Intents and Passing Information

We can pass information between Activities using the Intent!

Intents can store **Extra**s, and carry them from one Activity to the next. Each **Extra** you want add requires two things: a **key** and a **value**. The **key** is used to insert and retrieve **value**s from the Intent's collection of **Extra**s.

Open *OtherActivity* and add this line to the class (above *onCreate*):

```
public static final String NAME_KEY = "USER_NAME";
```

This will give us a **key** we can use when putting Extras into an Intent for this Activity. The **value** will be whatever we want to send when we are running the app.

Now let's make some changes to MainActivity so we have some info to pass:

1. Open *activity_main.xml* and add a TextView above the Button
    a. Give it a text of "Enter your name: "
2. Below the TextView and above the Button, add an EditText:
    a. Give it an *android:id* of "@+id/name_edit_text"
    b. Give it an *android:inputType* of "text"

    An EditText is a View that allows the user to type something in on the screen. It handles all the keyboard stuff for us. The *inputType* attribute affects what keyboard comes up, and how the EditText looks (*phone* for phone numbers, *password* shows dots as you type, etc.)

3. Open *MainActivity* and add a private EditText variable called *nameEditText*
4. In *onCreate*, use findViewById to assign the EditText in layout to the *nameEditText* variable
5. Change the navigateButton's click listener:

```
@Override
public void onClick(View v) {
    String name = nameEditText.getText().toString();
    Intent intent = new Intent(MainActivity.this, OtherActivity.class);

    intent.putExtra(OtherActivity.NAME_KEY, name);
    startActivity(intent);
}
```

    The first line will retrieve whatever is in the EditText field and save it to *name*.

    The third line puts the *name* **value** into the Intent, using *NAME_KEY* as the **key**.

This added some information that MainActivity has into the Intent.

The Intent will keep this data safe while MainActivity stops and OtherActivity begins.

These next changes will let us access the information from inside OtherActivity:

6. Open *activity_other.xml* and change the TextView there:
    a. Give it an *android:id* of "@+id/welcome_name_text"
7. Open *OtherActivity.java*. In *onCreate*, add these lines:

```
Intent newIntent = getIntent();
String name = newIntent.getStringExtra(NAME_KEY);
```

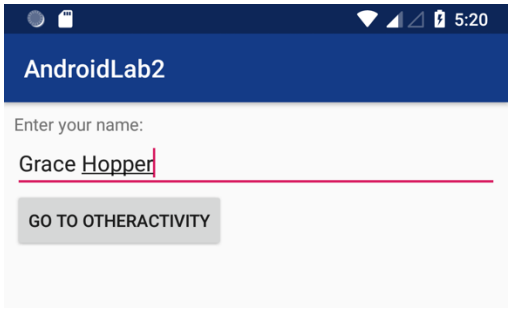
    These lines get the Intent that was used to begin this Activity and retrieve the **value** stored in the **Extra**s, using the same **key** we used to insert it in *MainActivity*.

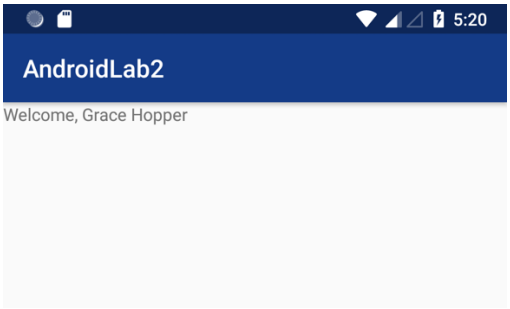
8. Finally, put this at the end of *onCreate* to show the information we've passed:

```
welcomeNameText.setText("Welcome, " + name);
```

You'll also need to declare the *welcomeText* variable and initialize it using *findViewById*.

You should be able to run your app now! Enter your name and see what happens.





9. What happens if you click the button for OtherActivity without entering text?
   Do it, and follow the stack trace to see what happened (and where).
10. To protect ourselves if the user clicked the button without entering anything, or if the Intent was somehow missing the Extra we expected, add:

```
if(name == null || name.isEmpty()) {
    name = "Anonymous";
}
```

11. Add a Button that, when clicked, starts MainActivity. You do not need to pass any Extras. The Button's text should say "Return to MainActivity".
    a. Create the Button in your layout, and give it an id and text
    b. Declare a Button variable in your Activity
    c. Initialize the Button variable using findViewById
    d. Add an OnClickListener to the button that starts MainActivity (with an Intent)

12. Switch to the Design editor view. You may see that the Button and the TextView are overlapping in the top left corner. This is because the default layout for new activities is a ConstraintLayout. Do one of the following to fix this:
    a. Use the Design editor to add constraints so the Button and TextView don't overlap
    b. Change the root layout to a LinearLayout instead (remember to add an *orientation*!)

13. If your app is running correctly, you should see a text field where you can enter text. Put your name in, and press the button. A new screen should come up that welcomes you by name. If not, revisit the above steps and make sure you've followed the instructions.

14. When your app runs properly, execute the following commands to upload your changes to your branch on GitHub:
    a. git add .
    b. git commit -m "lab 2: working code complete"
    c. git push origin *yourbranchname*

    Your code branch is now saved and committed to the git repository.

This completes Lab 2.

Food for Thought:

Android has a Back button already!
What happens when you use Back to exit OtherActivity, instead of the ReturnToMain button?
Is there a difference in behaviour?
Which is better to use?