CS194A

Android Programming Workshop

Lecture 5: May 6, 2020

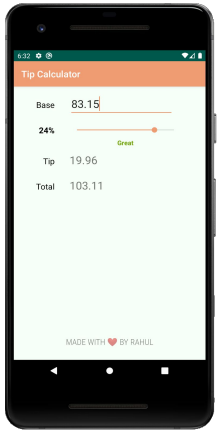
Rahul Pandey

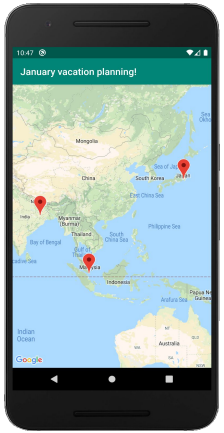
Outline

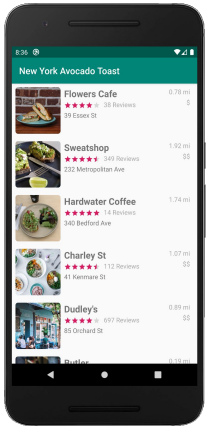
- Logistics
- Intents review
- Activity lifecycle
- Permissions

Outline

- **Logistics**
- Intents review
- Activity lifecycle
- Permissions







Assn 1: Tip Calculator

Assn 2: Google Maps

Assn 3: Yelp Clone

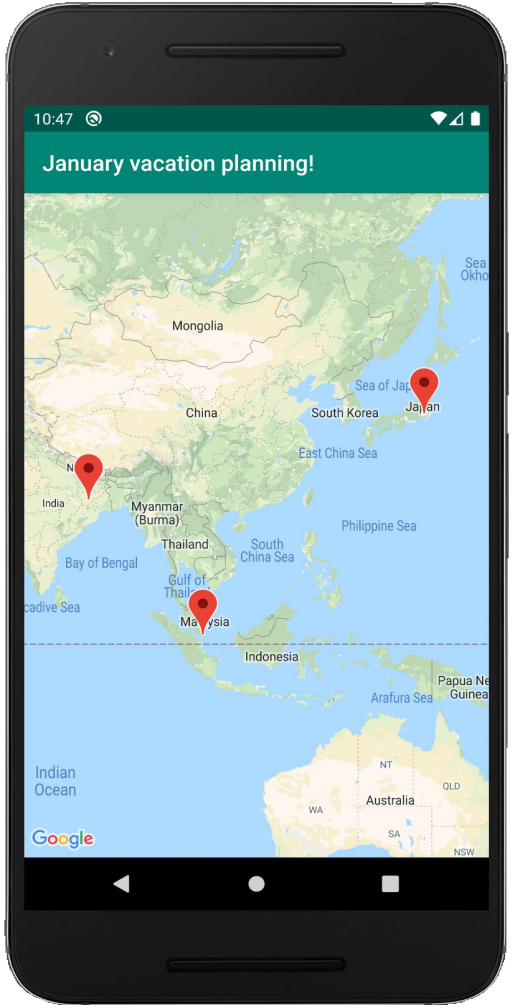
Industry panel discussion

1 2 3 4 5 6 7 8 9

Week #

Assignment 2- My Maps

- RecyclerView
- Google Maps integration
- Activities and intents



My Maps app

- Project due Sunday, **May 10, 11:59pm**
- Partner feedback due Wednesday, **May 13, 4:30pm**
- Submission through Canvas!

Mid-quarter feedback (anonymous)

- https://forms.gle/qYuoFKa7Pb3MQ8a18
- Should only take a few minutes

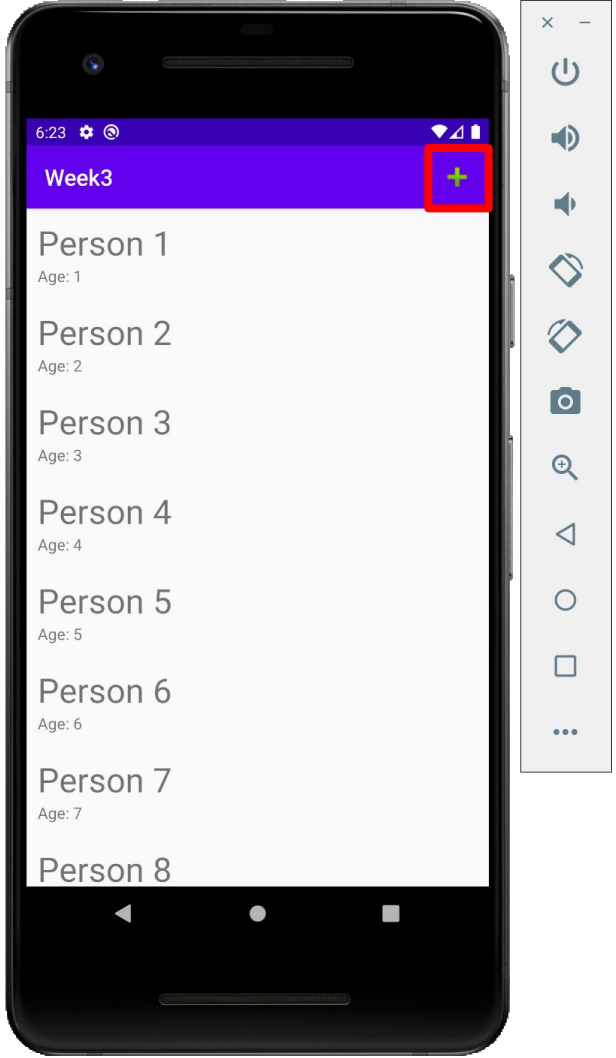
Outline

- Logistics
- **Intents review**
- Activity lifecycle
- Permissions

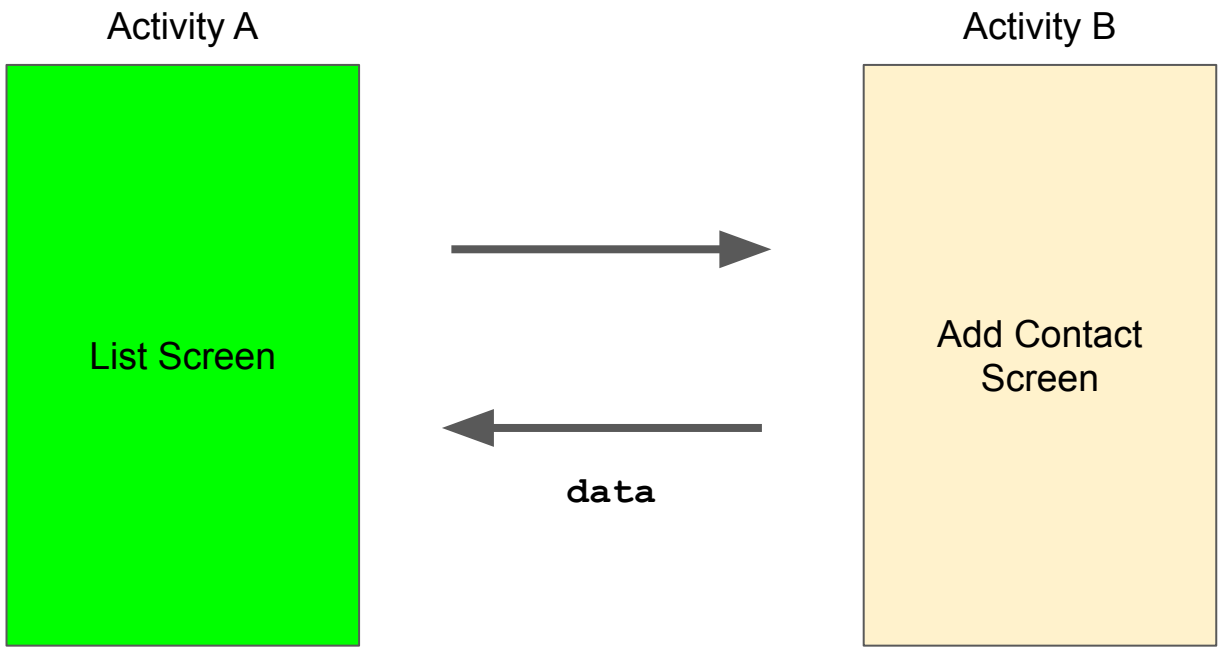
What is an intent?

Types of Intents

- Explicit intent: launch other activities in your app
  - `val myIntent = Intent(this, ActivityName::class.java)`
  - `startActivity(myIntent)`
- Implicit intent: request to perform an action based on a desired action
  - `val browserIntent = Intent(Intent.ACTION_VIEW, Uri.parse("url.com"))`
  - `startActivity(browserIntent)`
  - Common implicit intents: start a phone call, take a picture, open the browser/maps


6:23
Week3
Person 1
Age: 1
Person 2
Age: 2
Person 3
Age: 3
Person 4
Age: 4
Person 5
Age: 5
Person 6
Age: 6
Person 7
Age: 7
Person 8

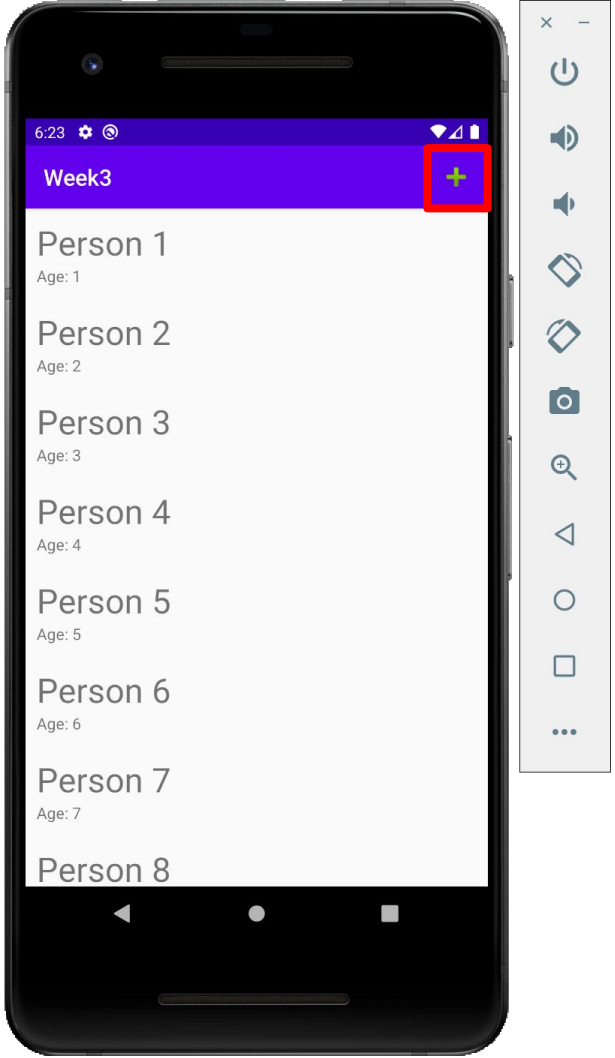
Returning data to the parent



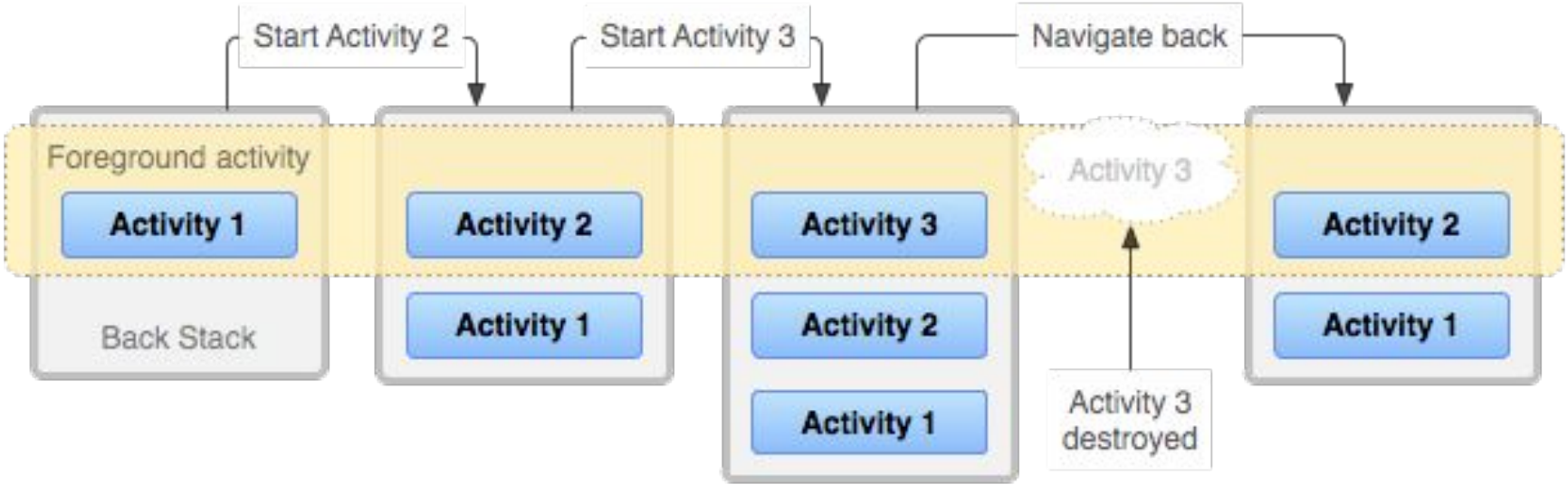
Getting a result back from a launched activity

- Sometimes you'll want to get data from the launched activity, e.g.
  - `MainActivity` launched `AddContactActivity`: adding a contact to the list
  - Intent to take a picture
- Call `startActivityForResult` rather than `startActivity`.
  - Pass a request code along with the intent
  - Returns immediately, but the Android system will call another method...
- `onActivityResult` is called when the second activity is done
  - Second activity should call `setResult` and `finish` to communicate back

Code with me!


6:23
Week3
Person 1
Age: 1
Person 2
Age: 2
Person 3
Age: 3
Person 4
Age: 4
Person 5
Age: 5
Person 6
Age: 6
Person 7
Age: 7
Person 8

Activity back stack- like a stack of plates



Nullability in Kotlin

Java

```
String name = null;
int length = name.length();     //  runtime crash
if (name != null) {
  int length = name.length();  // ok
}
```

Kotlin

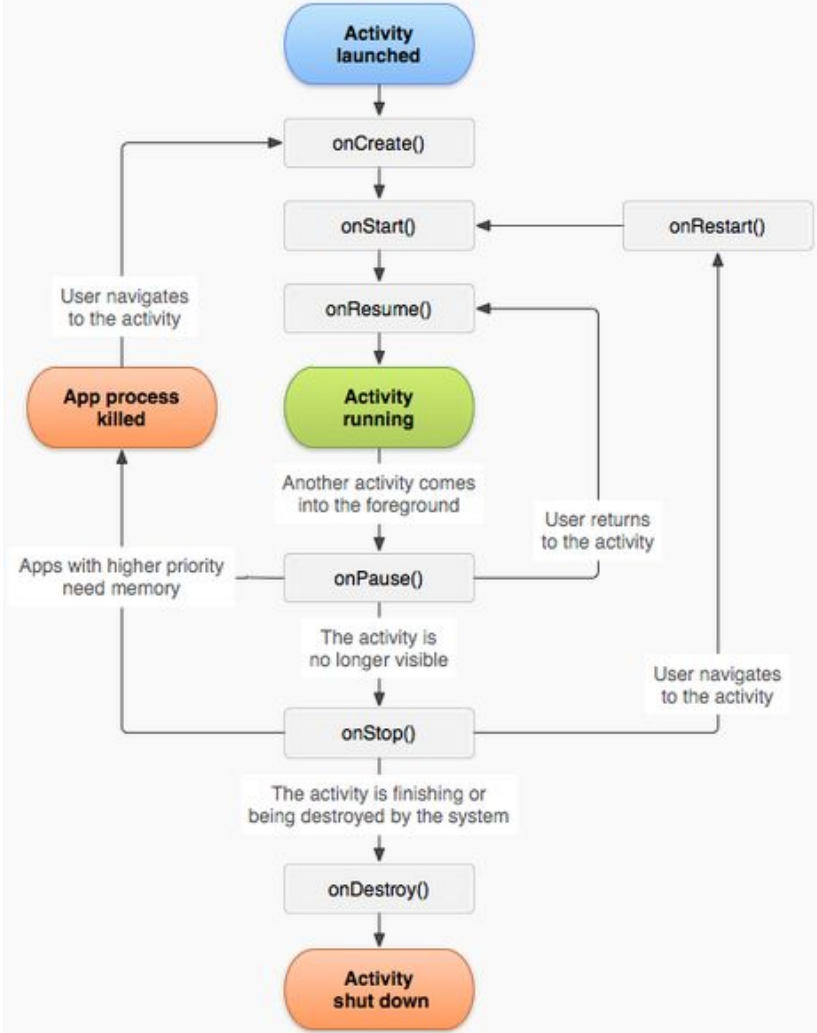
```
val bad: String = null          //  compiler error!
val name: String? = null        // ok
val lengthBad = name.length()   //  compiler error!
val length1 = name?.length() ?: 0
if (name != null) {
  val length2 = name.length()
}
```

Outline

- Logistics
- Intents review
- **Activity lifecycle**
- Permissions

Activity can be in a number of states

- States:
  - **Resumed**: activity is in foreground
  - **Paused**: activity is partially obscured by another activity. Activity cannot receive user input or execute code
  - **Stopped**: activity is hidden/in the background. Things like member variables are maintained
  - **Destroyed**: Resources of the activity are reclaimed by the Android system. E.g. back button press
- Android system will notify you when a state transition happens


Activity launched
onCreate()
onStart()
onRestart()
onResume()
User navigates to the activity
App process killed
Activity running
Another activity comes into the foreground
User returns to the activity
Apps with higher priority need memory
onPause()
The activity is no longer visible
User navigates to the activity
onStop()
The activity is finishing or being destroyed by the system
onDestroy()
Activity shut down

Prep for next week

- Finish **My Maps**

- Submit peer feedback through Canvas