Lab 3: Service

Submission Due: 2/19/2021

Objective

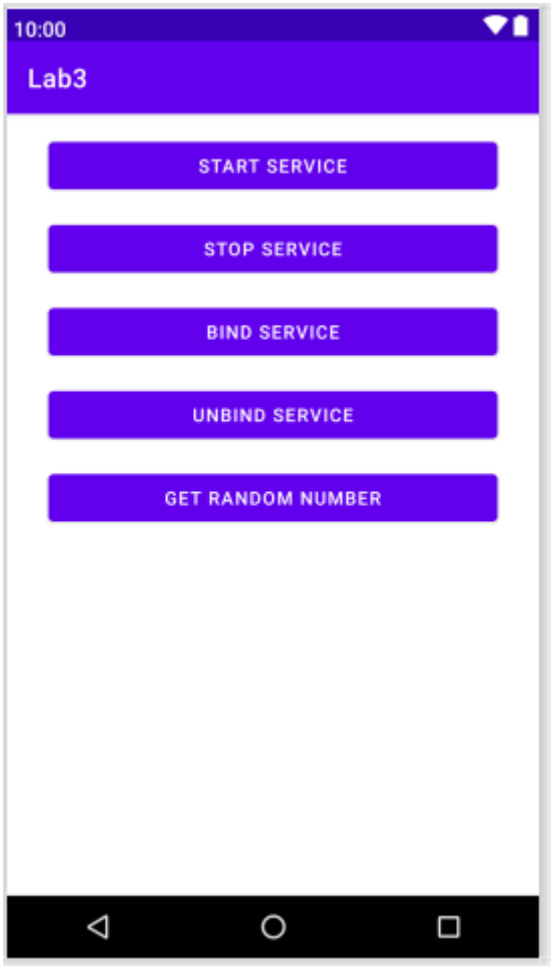
1. Learn to create a service that generates random numbers.
2. Learn how to start/stop the service and how to bind/unbind to the service

Before Lab

1. Download the lab3 starter code from Camino (Files -> Lab -> Lab3_StarterCode).
2. Unzip and open the Lab3_StarterCode project in Android Studio.

Lab3_StarterCode

Starter code contains MainActivity with code for setting layout, initializing UI elements and has Button click event listeners for the following buttons –



App Description & Tasks

In this lab, you are going to create a service, implement callback methods that get invoked by the system when the service is started, stopped, bound, and unbound. In the service, you will also implement a method to generate random numbers.

In the MainActivity, you will complete the implementation of methods to start, stop, bind and unbind to the created service.

You need to complete the following tasks:

Task 1: Create a service & implement random number generator method

1. Create a service called **MyService**
    a. Project -> New -> Service -> Service
    b. Set Class Name: MyService.
    c. Check Exported and Enabled
    d. Click on Finish
    e. Open AndroidManifest.xml and observe that service metadata is added.
2. In MyService, implement a method called **generateRandomNumber()-**
    a. generate random integer every second
    b. Add info log to print the thread id and generated random number

    Hint:

    You can use Thread.sleep() to generate every second, Random.nextInt() to generate random integer, Thread.currentThread().getId() to get thread ID.

    Do not use while(true) to generate random every second, instead use while(*variable*) so that you can start and stop the generator by setting the variable true and false respectively.

Task 2: Start & stop service

1. In the MainActivity, implement the method **startMyService()**
    a. Add info log to print the message- Starting Service.
    b. Add another info log to print the thread id of the MainActivity.
    c. Start MyService using explicit intent by calling startService().
    (Tip: Similar to startActivity() we use to navigate from one activity to another)
2. In the MyService, implement the callback method **onStartCommand** which gets invoked by the system when startService() is called.
    a. Add info log to print the message – On Start Service
    b. Add info log to print Thread ID & Start ID
    c. If the random number generator is not running then start the generator in a separate thread.
    d. Set the return value from onStartCommand() to constant - START_NOT_STICKY

(For more details, refer https://developer.android.com/guide/components/services)

Code Hint:

```
Runnable runnable = () -> {
    //ToDo: set the boolean variable to True
    generateRandomNumber();
};
Thread t= new Thread(runnable);
t.start();
```

3. In the MainActivity, implement stopMyService() to stop the service
    a. Add info log with the message "Stopping service"
    b. Stop MyService by calling stopService()
    (Note: You need to pass the same intent object which you used for startService()
4. In MyService, implement the callback onDestroy() that system invokes when stopService() is called.
    a. Add info log – "On Destroy".
    b. Set the boolean variable to false.
    c. Set thread (t) reference to null as it is no longer needed.

Task 3: Bind & unbind Service

1. In MyService, create an instance of Binder that contains a **public** method called **getService()** which **returns** this MyService **instance** to the clients as shown-

```
class MyLocalBinder extends Binder{
    public MyService getService(){
        return MyService.this;
    }
}

private Binder myBinder = new MyLocalBinder();
```

2. Implement the onBind() callback
    a. Add info log – "On Bind"
    b. Return myBinder
3. Implement the onUnbind() callback & add info log – "On Unbind"
4. In MainActivity, within bindMyService() define callbacks for service binding, which will be later passed to bindService()
    Code Hint:

```
MyService myService;
private ServiceConnection myConnection;

private void bindMyService(){
    if myConnection is null then
        log (info) - Binding service
        myConnection = new ServiceConnection() {
```

```
            @Override
      public void onServiceConnected(ComponentName componentName,
IBinder iBinder) {
                log(info) “On Service Connected”
                Set variable isBound to True
                MyService.MyLocalBinder myBinder =
(MyService.MyLocalBinder) iBinder;
                myService = myBinder.getService();
            }

            @Override
  public void onServiceDisconnected(ComponentName componentName){
                log(info) “On Service Disconnected”
                Set variable isBound to False
            }
        };
        bindService(sIntent,myConnection,BIND_AUTO_CREATE);
    }
}
```

5. In MainActivity, within unbindMyService add code to unbind the bound service
    a. Check if service is bound (isBound should be true). If true then
        i. Add info log to print “Unbinding service”
        ii. Call unbindService()

```
unbindService(myConnection);
```

        iii. Set myConnection reference to null as it is no longer needed
        iv. Set isBound to false

Task 4: Get the random number from the service

1. In MyService, you need to implement a **public** method called **getRandomNumber** that **returns** the **random number** generated by generateRandomNumber()
   (Activity binds to MyService and calls getRandomNumber() to get the generated random number from MyService)
   Tip: Declare integer variable say ‘rNum’ as instance variable. Set rNum in generateRandomNumber() and return rNum from getRandomNumber()
2. In MainActivity, implement **getNumber**() check if service is bound i.e if isBound is
    a. True then call getRandomNumber() of MyService. Set text of TextView (tvRandomNumber) to display the return value from the function call.
    b. False then set the text of TextView (tvRandomNumber) to an empty string.

Task 5: Execute & Observe

1. Run the app on an emulator or an Android device.
2. Test whether the application is working as expected.
3. Observe the logs-

a. When Start Service button is clicked, random numbers are generated every second
b. MyService->Generate Random Number will be running on a different thread and not on main thread (Different Thread IDs)
c. Click Bind Service button and then Get Random Number button, the random number displayed on the screen is same as that observed in logcat log.
d. Service is not destroyed when you click Stop Service button as it is still bound (As long as 1 connection exists, service instance won't be destroyed)
e. To destroy, Click on Unbind.
f. Now try Start and Stop Service. onDestroy will be called as there are no connections.

Deliverables

1. Final code (File format: zip)