Introduction to Android Service framework

CS 436 Software Development on Mobile

Dr.Paween Khoenkaw

Department of Computer Sciece

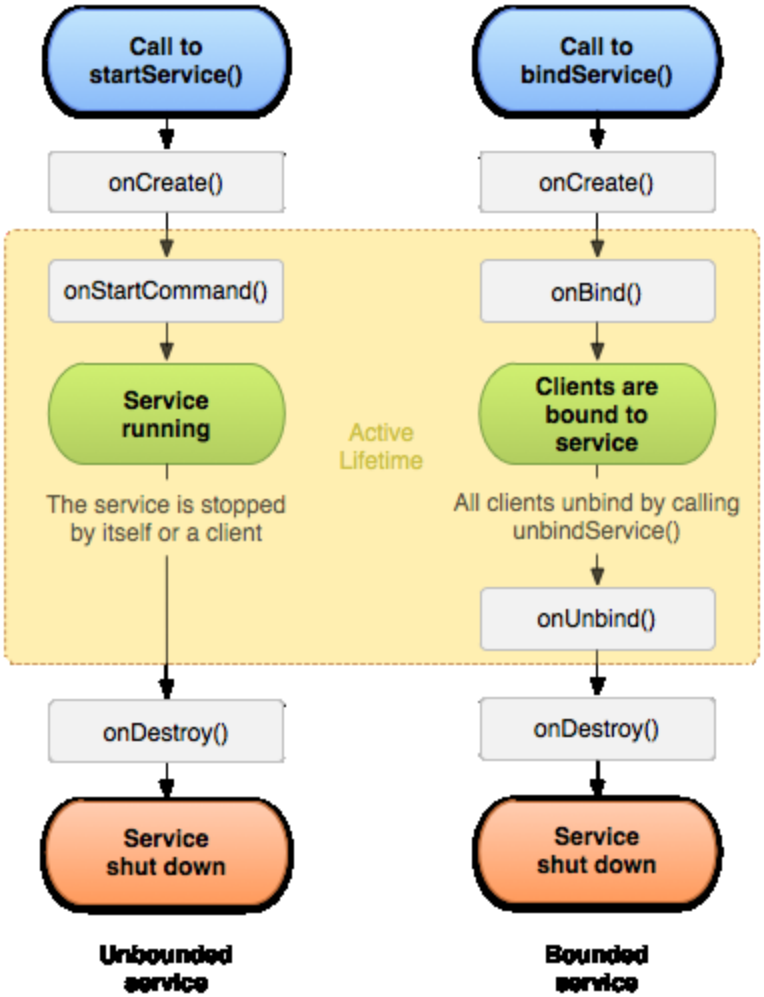
Maejo University

Android Service

Android service

-Long-running operations in the background
-No user interface
-Continue to run even if the user switches to another application
-Service can be bound or unbound

Service Lifecycle



onCreate()
onStartCommand()
onBind()
onUnbind()
onRebind()
onDestroy()

http://developer.android.com/guide/components/services.html

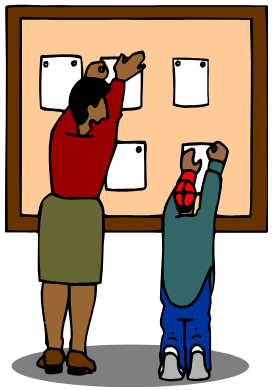
Service Lifecycle

Unbound service remains running until it stops itself or another component stops it by calling stopService().
Example: FM radio,Mp3 player, Alarm clock

Bound service runs only as long as the component is bound to it. Once the service is unbound from all clients, the system destroys it.
Example: WIFI service, GPS service

How to communicate with service

Asynchronous method using Intent (unbound service)



Synchronous method using Remote procedure call (RPC) (bound service)

Example 3.1:Kitchen timer (Unbound service)

Service

Activity

Set alarm, Start Service

Broadcast Alarm

Example 3.1:Kitchen timer (Unbound service)

Service-side

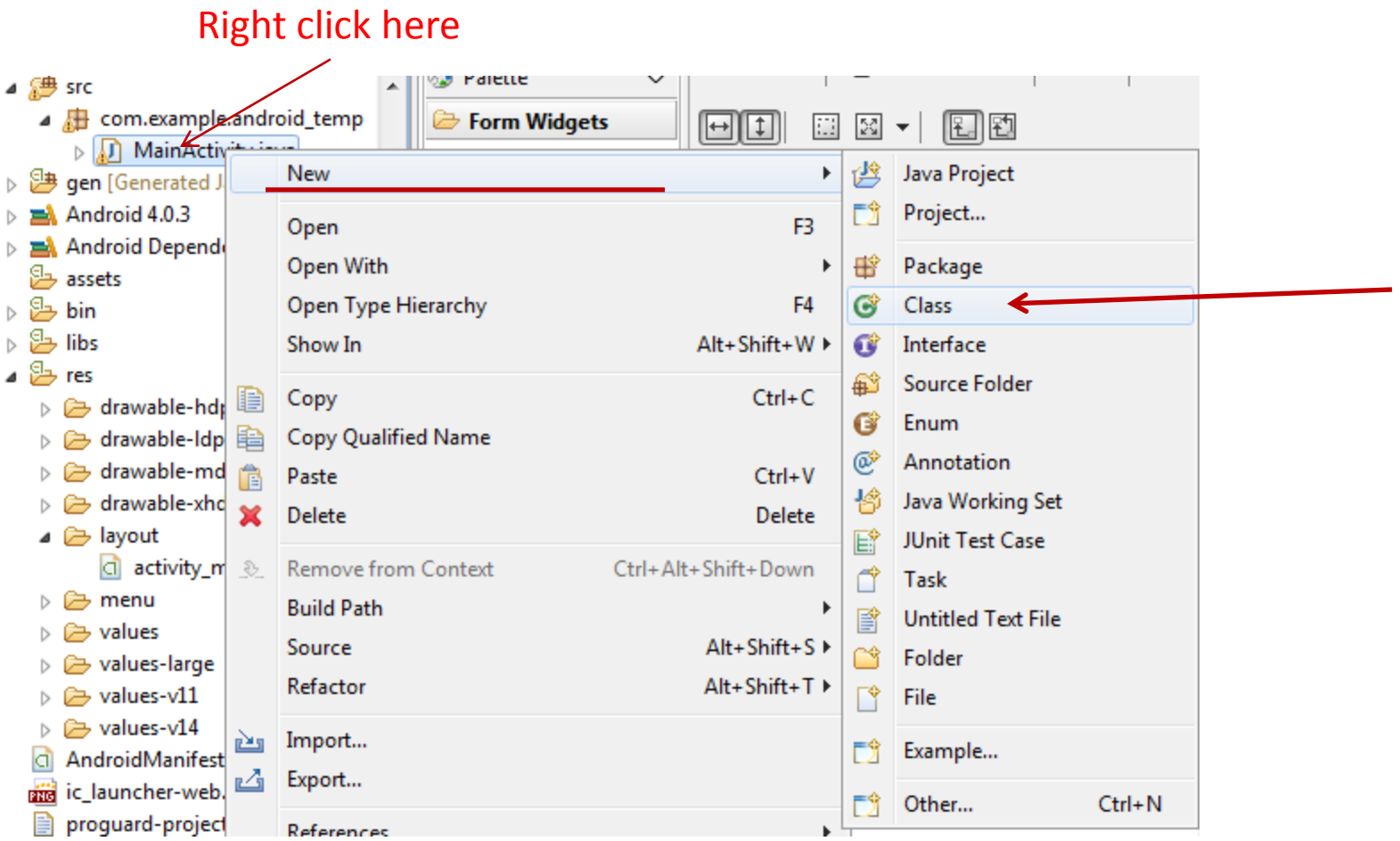
1) Create alarm clock service by extends service class
2) Implement OnCreate,OnStart,OnDestroy
3) Edit AndroidManifest to enable service

Activity-side

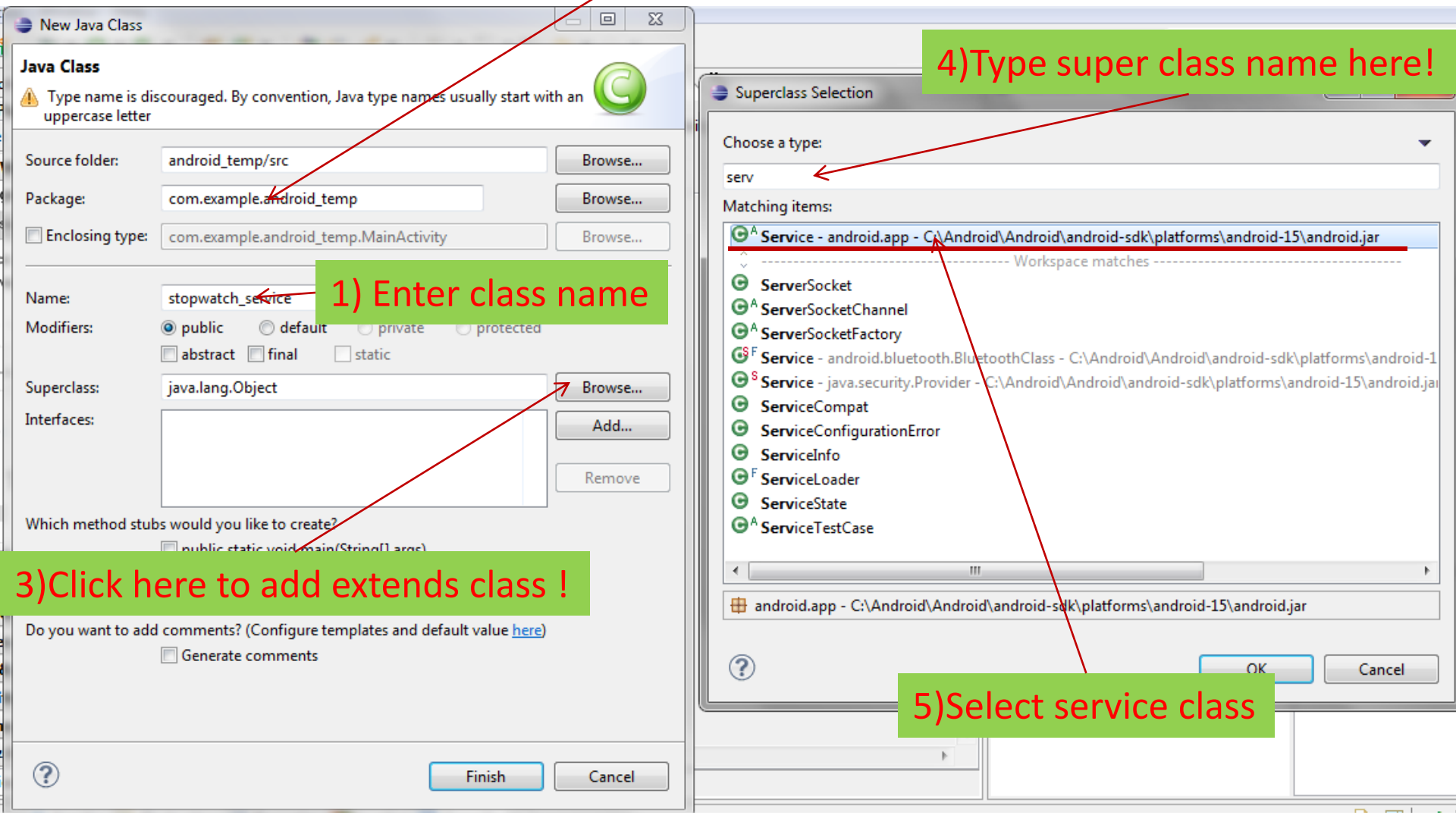
1) Instantiation service as intent
2) Send data to service using Bundle
3) Start service
4) Stop service

Example 3.1

1) Create new class that extends service class



Example 3.1


2)This class must has identical package name with main activity
New Java Class
Java Class
Type name is discouraged. By convention, Java type names usually start with an uppercase letter
Source folder:
android_temp/src
Browse...
Package:
com.example.android_temp
Browse...
Enclosing type:
com.example.android_temp.MainActivity
Browse...
Name:
stopwatch_service
1) Enter class name
Modifiers:
public
default
private
protected
abstract
final
static
Superclass:
java.lang.Object
Browse...
Interfaces:
Add...
Remove
Which method stubs would you like to create?
3)Click here to add extends class !
Do you want to add comments? (Configure templates and default value here)
Generate comments
Finish
Cancel
4)Type super class name here!
Superclass Selection
Choose a type:
serv
Matching items:
Service - android.app - C:\Android\Android\android-sdk\platforms\android-15\android.jar
Workspace matches
ServerSocket
ServerSocketChannel
ServerSocketFactory
Service - android.bluetooth.BluetoothClass - C:\Android\Android\android-sdk\platforms\android-1
Service - java.security.Provider - C:\Android\Android\android-sdk\platforms\android-15\android.jar
ServiceCompat
ServiceConfigurationError
ServiceInfo
ServiceLoader
ServiceState
ServiceTestCase
android.app - C:\Android\Android\android-sdk\platforms\android-15\android.jar
OK
Cancel
5)Select service class

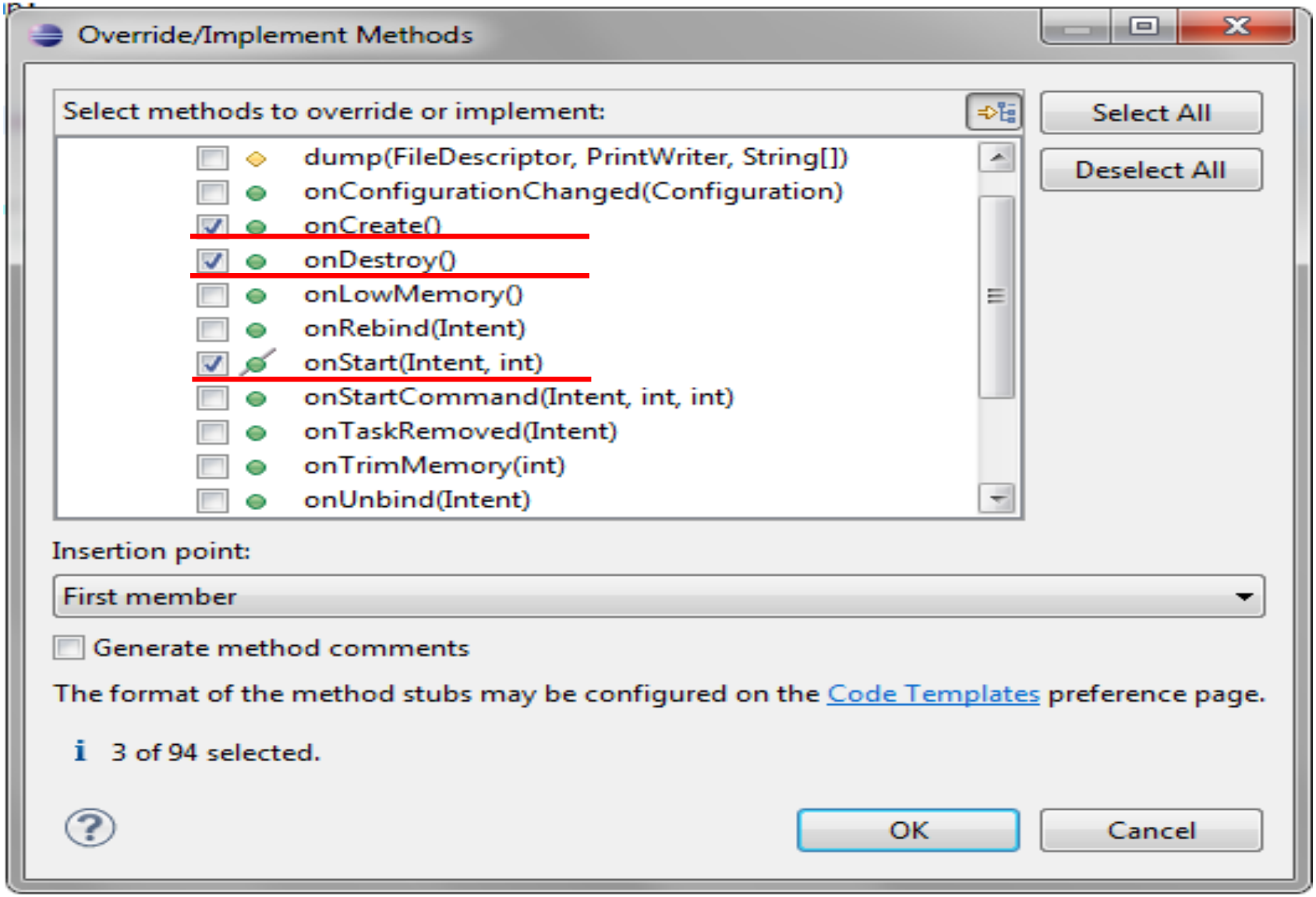
Example 3.1

The new service class was created

```
package com.example.android_temp;

import android.app.Service;

public class stopwatch_service extends Service {

    @Override
    public IBinder onBind(Intent arg0) {
        // TODO Auto-generated method stub
        return null;
    }

}
```

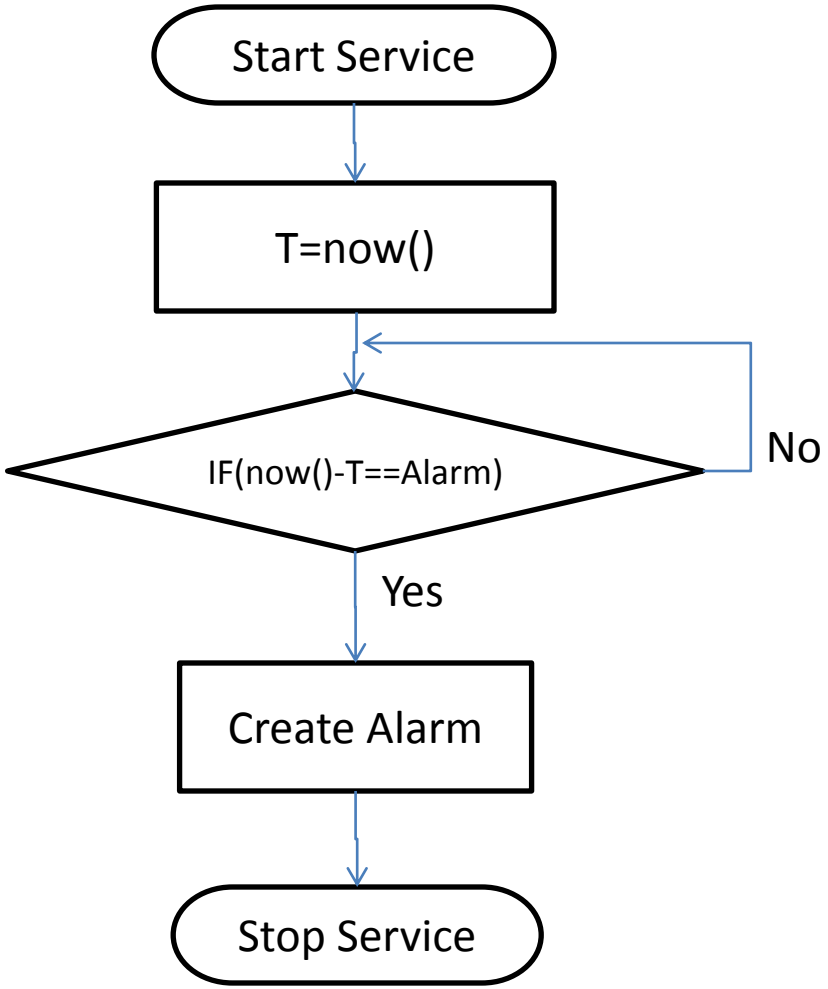
Example 3.1

2)Override these method



Example 3.1

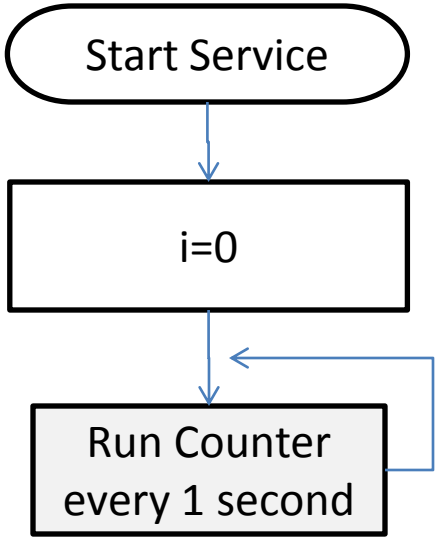
3)The algorithm

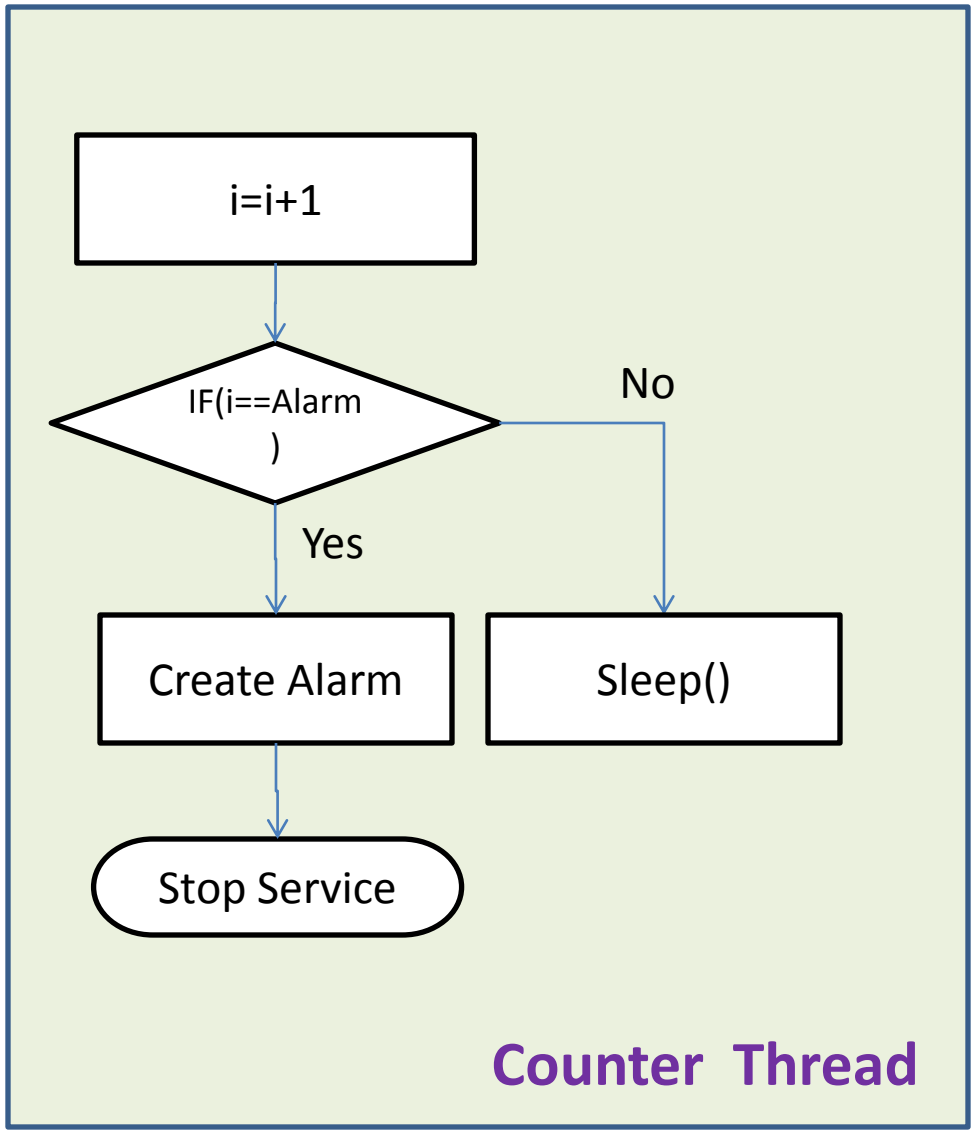


The wrong way !!!!!!

Example 3.1

3)The algorithm





The right way !!!!!!

Example 3.1

4)Create counter thread class using timertask as superclass

```
private class timertask extends TimerTask{

@Override
public boolean cancel() {
return super.cancel();
}

@Override
public void run() {
  // write your code here !!!!


}

@Override
public long scheduledExecutionTime() {
return super.scheduledExecutionTime();
}
}
```

This section will be execute every n second

Example 3.1

4) The example of complete class

```
private class timertask extends TimerTask{
@Override
public boolean cancel() {
// TODO Auto-generated method stub
return super.cancel();
}
@Override
public void run() {
c++;
if(c==alert_at)
{
Looper.prepare();
Toast.makeText(getApplicationContext(), "Ring Ring Ring...", Toast.LENGTH_LONG).show();
stopwatch_service.this.stopSelf();
Looper.loop();
}
Log.d("service", String.valueOf(alert_at-c));
}
@Override
public long scheduledExecutionTime() {
// TODO Auto-generated method stub
return super.scheduledExecutionTime();
}
}
```

Pop-up message box

Stop Service

Example 3.1

5)Put them together

```
public class stopwatch_service extends Service {
int c=0,alert_at=-1;
Timer timer1;
timertask task1;
@Override
public void onCreate() {
super.onCreate();
timer1=new Timer();
task1=new timertask();
timer1.scheduleAtFixedRate(task1, 100, 1000);
Log.d("service", "Create");
}
@Override
public void onDestroy() {
super.onDestroy();
timer1.cancel();
Log.d("service", "destroy");
}
@Override
public void onStart(Intent intent, int startId) {
super.onStart(intent, startId);
Log.d("service", "start");

Bundle extras = intent.getExtras();
if (extras != null) {
alert_at=extras.getInt("alert_at");
}
c=0;
}}
```

public
Declare class and variable

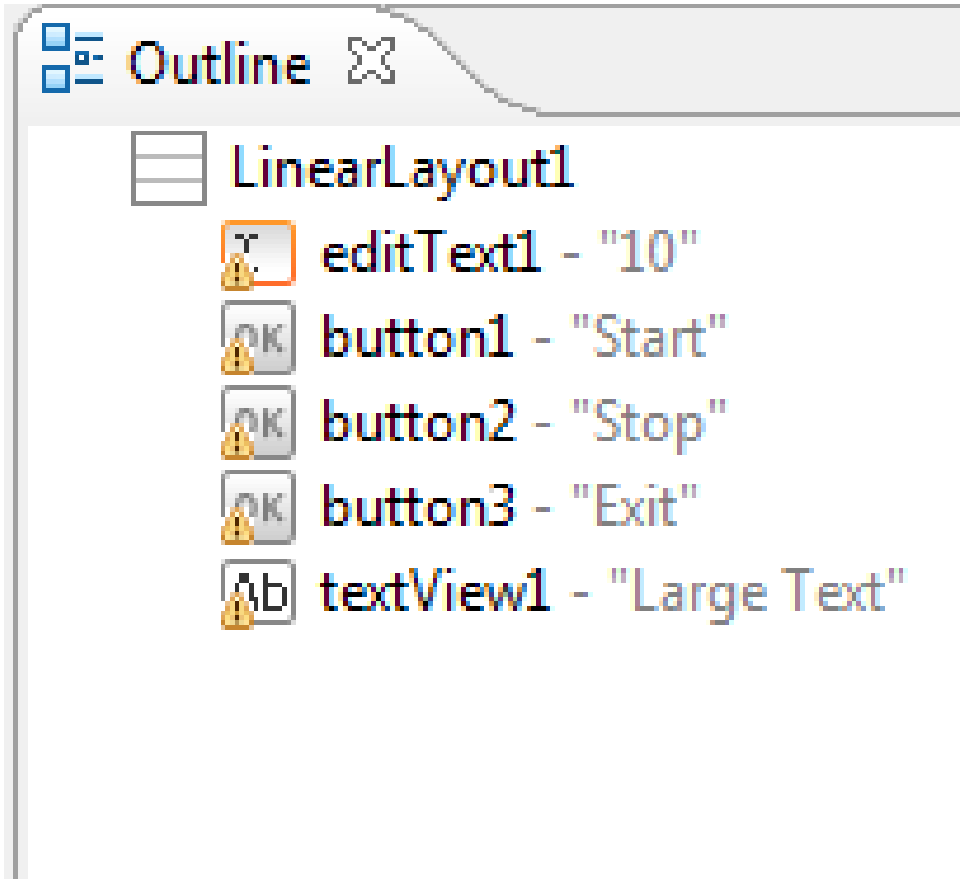
OnCreate()
Timer and Task Instantiation and schedule to run every 1000ms

OnDestroy()
Destroy scheduler

OnStart()
Get bundle value

Example 3.1

6)Design layout



Example 3.1

7)Start service in Activity

Instantiate the service intent

```
final Intent service1=new Intent(MainActivity.this
,stopwatch_service.class);
```

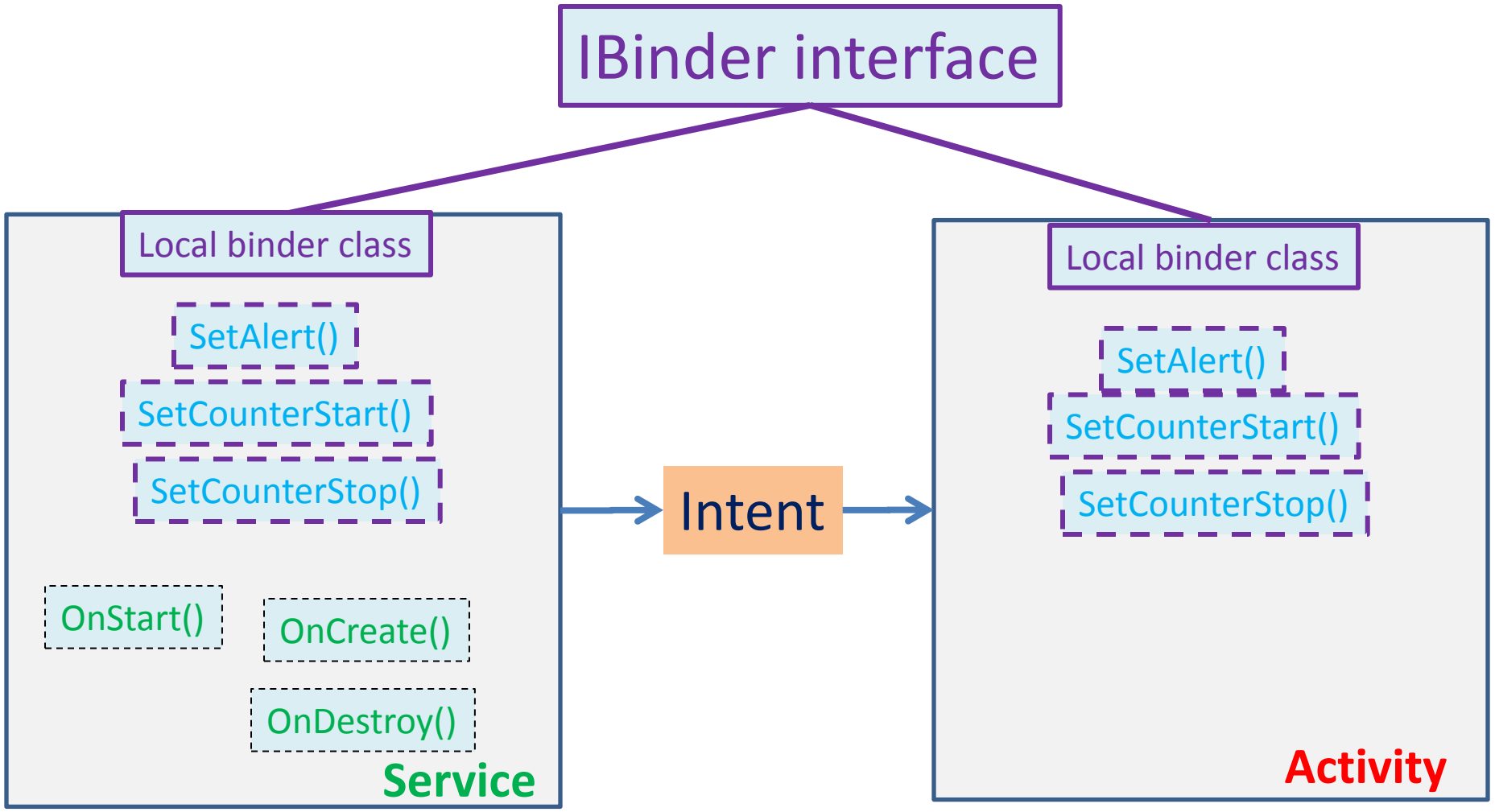
Put extra data in service intent and start service

```
service1.putExtra("alert_at“, String_Second_To_Alert);
startService(service1);
```

Manually stop service

```
stopService(service1);
```

Example 3.2:Kitchen timer (Bound service)



*An RPC interface can include only methods. All methods are executed synchronously

Example 3.2:Kitchen timer (Bound service)

Service-side

1) Create alarm clock service by extends service class
2) Implement OnCreate,OnStart,OnDestroy, onBind
3) Create local binder class extends binder class
4) Create Get and Set method for RPC call
5) Use intent to raise alarm event
6) Edit AndroidManifest to enable service

Activity-side

1) Instantiation service as intent
2) ~~Start service~~ Bind with service
3) Use RPC to get and set data
4) ~~Stop service~~ Unbind with service

Example 3.2:Kitchen timer (Bound service)

-Create bondable service

1)Create LocalBinder class

```
public class LocalBinder extends Binder {
    stopwatch_service getService() {
   return stopwatch_service.this;
    }
  }
```

2)Create LocalBinder instance in global section

```
private final IBinder mBinder = new LocalBinder();
```

3)Return LocalBinder instance back to activity at the onBind moment

```
@Override
public IBinder onBind(Intent arg0) {
return mBinder;
}
```

Example 3.2:Kitchen timer (Bound service)

-Bind Activity with service

1)Create instant of service

stopwatch_service service1;

Service class Service instance

2)Create service intent

final Intent serviceIntent1=new Intent(MainActivity.this ,stopwatch_service.class);

3)Activate binder

bindService(serviceIntent1, mConnection, Context.*BIND_AUTO_CREATE);*

Example 3.2:Kitchen timer (Bound service)

-Bind Activity with service

4) Defines callbacks for service binding, passed to bindService()

```
private ServiceConnection mConnection = new ServiceConnection() {

@Override
public void onServiceConnected(ComponentName name, IBinder service) {
LocalBinder binder = (LocalBinder) service;
        service1 = binder.getService();
        mBound = true;
}

@Override
public void onServiceDisconnected(ComponentName name) {
    mBound = false;
}
   };
```

Example 3.2:Kitchen timer (Bound service)

-Bind Activity with service

5) Unbound service when do not use

```
@Override
protected void onStop() {
// TODO Auto-generated method stub
super.onStop();
if (mBound) {
        unbindService(mConnection);
        mBound = false;
    }
}
```

Example 3.2:Kitchen timer (Bound service)

-Bind Activity with service

6) Now you can call method in service synchronously

```
service1.setSecond( TimeToCook);
service1.setStart();
textview1.setText(service1.getRemain());
..... Etc.
```

Example 3.2:Kitchen timer (Bound service)

-Broadcast custom intent from service

1) Define intent string

```
public static final String Action_Tick ="example.ex3.stopwatch_service.action.Tick";
public static final String Action_Alarm ="example.ex3.stopwatch_service.action.Alarm";
```

2) Broadcast intent

```
Intent TickIntent = new Intent();
TickIntent.setAction(Action_Tick);
sendBroadcast( TickIntent);
```

Example 3.2:Kitchen timer (Bound service)

-Receive intent in activity

1) Create intent filter in onResume()

```
IntentFilter intentFilter = new IntentFilter("example.ex3.stopwatch_service.action.Tick");
IntentFilter intentFilter2 = new
IntentFilter("example.ex3.stopwatch_service.action.Alarm");
```

2) Override onReceived action in onResume()

```
mReceiver = new BroadcastReceiver() {

        @Override
        public void onReceive(Context context, Intent intent) {

                textview1.setText(String.valueOf(service1.getRemain()));
        }
    };
```

** If you create intent filter in onCreate() it will not work!!! **

Example 3.2:Kitchen timer (Bound service)

-Receive intent in activity

1) Register intent receiver

```
this.registerReceiver(mReceiver, intentFilter);
```

2) Don’t forget to unregister receiver

```
@Override
protected void onPause() {
super.onPause();
this.unregisterReceiver(this.mReceiver);
}
```

Example 3.2:Kitchen timer (Bound service)

-New timertask in service

```
private class timertask extends TimerTask{

@Override
public boolean cancel() {
return super.cancel();
}
@Override
public void run() {
c++;
Intent TickIntent = new Intent();
 TickIntent.setAction(Action_Tick);
 sendBroadcast( TickIntent);
if(c==alert_at)
{
 TickIntent.setAction(Action_Alarm);
      sendBroadcast( TickIntent);
task1.cancel();
}
}
@Override
public long scheduledExecutionTime() {
return super.scheduledExecutionTime();
}
}
```

Second update intent (tick)

Alarm intent

Example 3.2:Kitchen timer (Bound service)

-Additional method in service

```
@Override
public void onCreate() {
                super.onCreate();
                timer1=new Timer();
                task1=new timertask();
        }
public int getRemain()          {
                return (alert_at-c);
        }
public void setSecond(int iSecond){
                alert_at=iSecond;
        }
public void setStart(){
                task1.cancel();
                task1=new timertask();
                timer1.scheduleAtFixedRate(task1, 100, 1000);
                c=0;
        }
public void setStop(){
                task1.cancel();
        }
```

OnCreate

Get remain second

Get alarm second

Start timer

Cancel timer task

Concurrence task

Intent Service

IntentService is a base class for Services that handle asynchronous requests (expressed as Intents) on demand. Clients send requests through startService(Intent) calls

Concurrence task

Intent Service

1) Create service class by extends IntentService
2) Register service in AndroidManifest
3) Start service using intent
4) Communicate with main thread using broadcast receiver

Concurrence task

Intent Service

1) Create service class by extends IntentService

```
public class primeFinderService extends IntentService {
        public primeFinderService(String name) {
                super(name);
        }
        public primeFinderService() {
                super("primeFinderService");
        }
        @Override
        protected void onHandleIntent(Intent intent) {
                        long x=findPrime(intent.getLongExtra("data",5));
                        Intent broadcastIntent = new Intent();
                        broadcastIntent.setAction("primefinder is done the job");
                        broadcastIntent.addCategory(Intent.CATEGORY_DEFAULT);
                        broadcastIntent.putExtra("result", x);
                        sendBroadcast(broadcastIntent);
                }
                private long findPrime(long x){...}
                private boolean isPrime(long x){...}
  }
```

Project: Prime_intentservice

Concurrence task

Intent Service

1) Create service class by extends IntentService

```
private long findPrime(long x){
long i=3,c=0;
while(true)
 {
              if(isPrime(i)==true)
              {
             c++;
              if(c==x)return i;
              Intent broadcastIntent = new Intent();
broadcastIntent.setAction("primefinder update progress");
broadcastIntent.addCategory(Intent.CATEGORY_DEFAULT);
broadcastIntent.putExtra("result", c);
sendBroadcast(broadcastIntent);
              }
              i++;
}
}
```

Project: Prime_intentservice

Concurrence task

Intent Service

2) Register service in AndroidManifest

```
<service android:name="primeFinderService"></service>
```

Concurrence task
Intent Service

3) Start service using intent

```
primefinder = new Intent(this,primeFinderService.class);
primefinder.putExtra("data",Long.valueOf(edittext1.getText().toString()) );
startService(primefinder);
```

Concurrence task

Intent Service

4) Communicate with main thread using broadcast receiver

```
IntentFilter filter = new IntentFilter("primefinder is done the job");
    filter.addCategory(Intent.CATEGORY_DEFAULT);
    ResponseReceiver receiver = new ResponseReceiver();
    registerReceiver(receiver, filter);

 IntentFilter filter2 = new IntentFilter("primefinder update progress");
   filter2.addCategory(Intent.CATEGORY_DEFAULT);
   ProgressReceiver receiver2 = new ProgressReceiver();
   registerReceiver(receiver2, filter2);

public class ResponseReceiver extends BroadcastReceiver {
@Override
public void onReceive(Context arg0, Intent arg1) {
textview1.setText(String.format("%d", arg1.getLongExtra("result", 0)));
                              }}
public class ProgressReceiver extends BroadcastReceiver {\
@Override
public void onReceive(Context arg0, Intent arg1) {
textview1.setText(String.format("%d primes was found", arg1.getLongExtra("result", 0)));
                              }}
```

Thank you ☺