Multiple threading with callbacks

Example 1

AbstractClasses.java

package example1;  
  
public class AbstractClasses {  
 public abstract class InternetAbstract {  
 public abstract void checkInternet(boolean status);  
 }  
}

MyThread.java

package example1;  
  
import java.io.IOException;  
import java.net.HttpURLConnection;  
import java.net.URL;  
  
public class MyThread extends Thread {  
  
 AbstractClasses.InternetAbstract internetAbs;  
  
 public MyThread(AbstractClasses.InternetAbstract inernetAbs) {  
 this.internetAbs = inernetAbs;  
 }  
  
 @Override  
 public void run() {  
 super.run();  
 //do time consuming task here  
 boolean success = false;  
 try {  
 URL url = new URL("https://google.com");  
 HttpURLConnection connection = (HttpURLConnection) url.openConnection();  
 connection.setConnectTimeout(10000);  
 connection.connect();  
 success = connection.getResponseCode() == 200;  
 } catch (IOException e) {  
 e.printStackTrace();  
 } catch (Exception ex) {  
 ex.printStackTrace();  
 }  
 internetAbs.checkInternet(success);  
  
  
 }  
}

MainClass.java

package example1;  
  
public class MainClass {  
 public static void main(String[] args) {  
  
 AbstractClasses.InternetAbstract internetabs = new AbstractClasses().new InternetAbstract() {  
 @Override  
 public void checkInternet(boolean status) {  
 if (status) {  
 System.*out*.println("Yes!!! there is intenet acccess");  
 } else {  
 System.*out*.println("Sorry!!! no internet access");  
 }  
 }  
 };  
 new MyThread(internetabs).start();  
 }  
}

* Run the mainclass.java and you’ll see status of internet bring printed. Above program is actually replacement of AsyncTask in android. For callback I’ve used abstract class.

Example2

AbstractClasses.java

package example2;  
  
public class AbstractClasses {  
 public abstract class InternetAbstract {  
 public abstract void checkInternetStatus(boolean status);  
 }  
}

WorkHorse.java

package example2;  
  
import java.io.IOException;  
import java.net.HttpURLConnection;  
import java.net.URL;  
  
  
public class WorkHorse {  
  
 String parameters;  
 AbstractClasses.InternetAbstract absinternet;  
  
  
 public WorkHorse(AbstractClasses.InternetAbstract absinternet) {  
 this.absinternet = absinternet;  
 }  
  
 public WorkHorse() {  
 }  
  
 Runnable work1 = new Runnable() {  
 @Override  
 public void run() {  
 //check intenet status  
 //do time consuming task here  
 boolean success = false;  
 try {  
 //System.out.print(parameters);//this will be null  
 URL url = new URL("https://google.com");  
 HttpURLConnection connection = (HttpURLConnection) url.openConnection();  
 connection.setConnectTimeout(10000);  
 connection.connect();  
 success = connection.getResponseCode() == 200;  
 absinternet.checkInternetStatus(success);//callback  
 } catch (IOException e) {  
 e.printStackTrace();  
 } catch (Exception ex) {  
 ex.printStackTrace();  
 }  
 }  
 };  
  
 Runnable work2 = new Runnable() {  
 @Override  
 public void run() {  
 //check wifi status or something  
 for (int i = 0; i < 5; i++) {  
 System.*out*.print(i \* 5);  
 System.*out*.print(", ");  
 }  
 }  
 };  
  
 Runnable work3 = new Runnable() {  
 @Override  
 public void run() {  
 for (int i = 0; i < 4; i++) {  
 System.*out*.println("My name is:");  
 }  
 }  
 };  
  
 //have as many independent runnables here  
  
 Thread thr1 = new Thread(work1);  
 Thread thr2 = new Thread(work2);  
 Thread thr3 = new Thread(work3);  
  
}

MainClass.java

package example2;  
public class MainClass {  
 public static void main(String[] args) {  
 AbstractClasses.InternetAbstract absinternet = new AbstractClasses().new InternetAbstract() {  
 @Override  
 public void checkInternetStatus(boolean status) {  
 if (status)  
 System.*out*.println("Yess!!! there is internet");  
 else  
 System.*out*.println("No!!! no internet today");  
 }  
 };  
 new WorkHorse(absinternet).thr1.start();  
 new WorkHorse().thr2.start();  
 new WorkHorse().thr3.start();  
 }  
  
}

* This example shows that multiple tasks being executed at same time with multiple runnables and threads. Things you should remember while doing such tasks is that each task should be independent of each other ie result of one task should not effect result of another task. In above example each task runs on different thread and any task can finish first. Among three, one task has callback feature.

Example3

AbstractClass.java

package example3;  
  
public class AbstractClasses {  
 public abstract class InternetAbstract {  
 public abstract void internetStatus(boolean status);  
 }  
}

MainClass.java

package example3;  
  
import java.io.IOException;  
import java.net.HttpURLConnection;  
import java.net.URL;  
  
public class MainClass {  
  
 public static void main(String[] args) {  
 Runnable work1 = new Runnable() {  
 @Override  
 public void run() {  
 //check internet status  
 //do time consuming task here  
 boolean success = false;  
 try {  
 //System.out.print(parameters);//this will be null  
 URL url = new URL("https://google.com");  
 HttpURLConnection connection = (HttpURLConnection) url.openConnection();  
 connection.setConnectTimeout(10000);  
 connection.connect();  
 success = connection.getResponseCode() == 200;  
 System.*out*.println("the status of internet is : " + success);  
 } catch (IOException e) {  
 e.printStackTrace();  
 } catch (Exception ex) {  
 ex.printStackTrace();  
 }  
 }  
 };  
  
 Runnable work2 = new Runnable() {  
 @Override  
 public void run() {  
 //check wifi status or something  
 for (int i = 0; i < 5; i++) {  
 System.*out*.print(i \* 5);  
 System.*out*.print(", ");  
 }  
 }  
 };  
 Thread thr1 = new Thread(work1);  
 Thread thr2 = new Thread(work2);  
 thr1.start();  
 thr2.start();  
  
 }  
  
  
}

If example 3 has to be do in android, it will be like following program.

Activity\_main.xml

<?xml version="1.0" encoding="utf-8"?>  
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"  
 xmlns:tools="http://schemas.android.com/tools"  
 android:layout\_width="match\_parent"  
 android:layout\_height="match\_parent"  
 android:paddingBottom="@dimen/activity\_vertical\_margin"  
 android:paddingLeft="@dimen/activity\_horizontal\_margin"  
 android:paddingRight="@dimen/activity\_horizontal\_margin"  
 android:paddingTop="@dimen/activity\_vertical\_margin"  
 tools:context="mypowerfulbag.com.threadinternetcheck.MainActivity">  
  
 <TextView  
 android:id="@+id/tvInternet"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:text="Hello World!" />  
  
 <Button  
 android:id="@+id/btnCheckStatus"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:text="Check Internet" />  
</LinearLayout>

MainActivity.java

package mypowerfulbag.com.threadinternetcheck;  
  
import android.support.v7.app.AppCompatActivity;  
import android.os.Bundle;  
import android.util.Log;  
import android.view.View;  
import android.widget.Button;  
import android.widget.TextView;  
  
import java.io.IOException;  
import java.net.HttpURLConnection;  
import java.net.URL;  
  
public class MainActivity extends AppCompatActivity {  
 TextView tvInternet;  
 Button btnCheckStatus;  
 AbstractClasses.InternetAbstract absinternet;  
 Thread thr;  
  
 @Override  
 protected void onCreate(Bundle savedInstanceState) {  
 super.onCreate(savedInstanceState);  
 setContentView(R.layout.*activity\_main*);  
 tvInternet = (TextView) findViewById(R.id.*tvInternet*);  
 btnCheckStatus = (Button) findViewById(R.id.*btnCheckStatus*);  
  
 absinternet = new AbstractClasses().new InternetAbstract() {  
 @Override  
 public void internetStatus(final boolean status) {  
 Log.*d*("crossover", "Internet status is: " + status);  
  
 runOnUiThread(new Runnable() {  
 @Override  
 public void run() {  
 //updating UI stuff must be done inside UIThread  
 tvInternet.setText("Internet: " + status);  
 }  
 });  
  
 }  
 };  
  
  
 btnCheckStatus.setOnClickListener(new View.OnClickListener() {  
 @Override  
 public void onClick(View v) {  
 System.*out*.println("checking status");  
 thr.start();  
 }  
 });  
  
 Runnable work1 = new Runnable() {  
 @Override  
 public void run() {  
 boolean success = false;  
 try {  
 //System.out.print(parameters);//this will be null  
 URL url = new URL("https://google.com");  
 HttpURLConnection connection = (HttpURLConnection) url.openConnection();  
 connection.setConnectTimeout(10000);  
 connection.connect();  
 success = connection.getResponseCode() == 200;  
 absinternet.internetStatus(success);//callback  
 } catch (IOException e) {  
 e.printStackTrace();  
 } catch (Exception ex) {  
 ex.printStackTrace();  
 }  
  
  
 }  
 };  
 thr = new Thread(work1);  
 }  
}

* Add internet permission in manifest.

<uses-permission android:name="android.permission.INTERNET" />

* Above program also has a callback feature and updating UI stuff should be handled inside runOnUiThread.