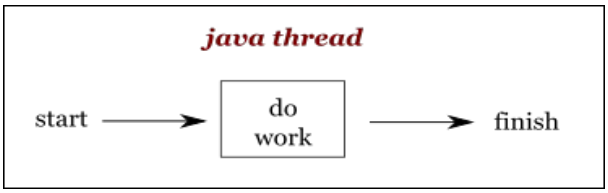
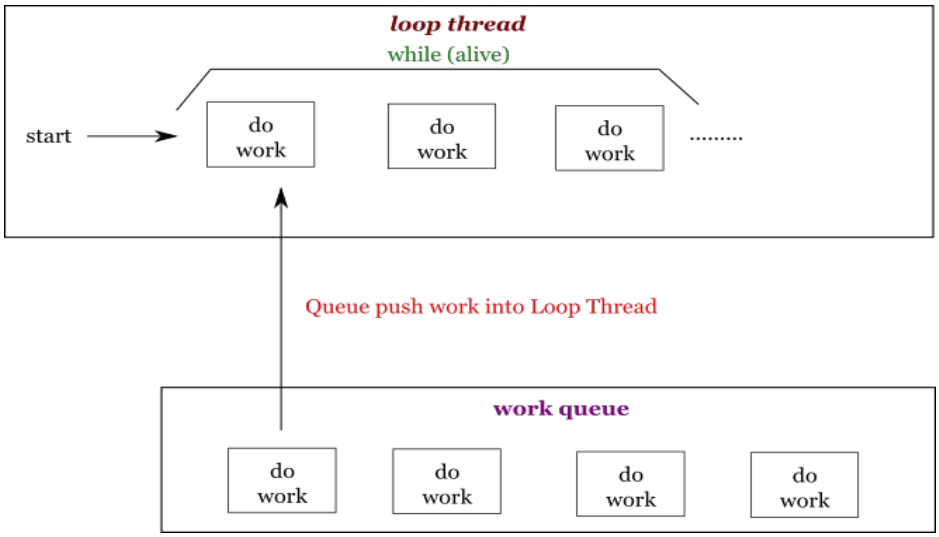
LOOP THREAD

What is Loop Thread?

Java thread executes a work and terminates after it finishes *run()* method.



Loop thread keeps alive after *run()* method.
Each loop thread associates with an work queue. Thread get work from queue to execute.
If we want to perform a work on the loop thread, we can push this work to the queue.

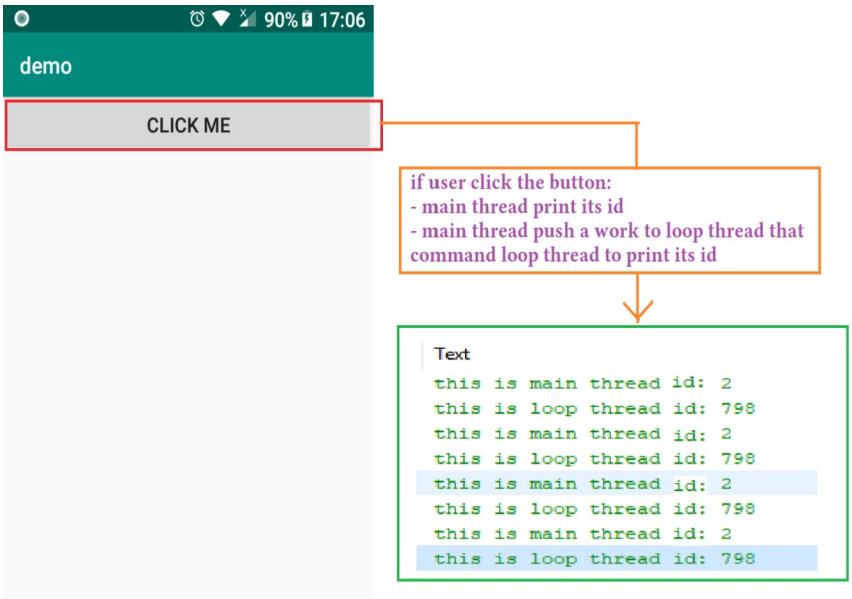


Loop Thread Sample

We will create an application that have a button.

If user click the button:

- Main thread print its id
- Main thread push into loop thread a work that requests loop thread to print its id.



Now, Lets implement the application.
Design Layout

```
<?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:orientation="vertical"
    tools:context=".MainActivity">

    <Button
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:text="Click Me"
        android:onClick="onUserClick"/>

    <ImageView
        android:id="@+id/image_view"
        android:layout_width="match_parent"
        android:layout_height="match_parent" />

</LinearLayout>
```

Implement Loop Thread

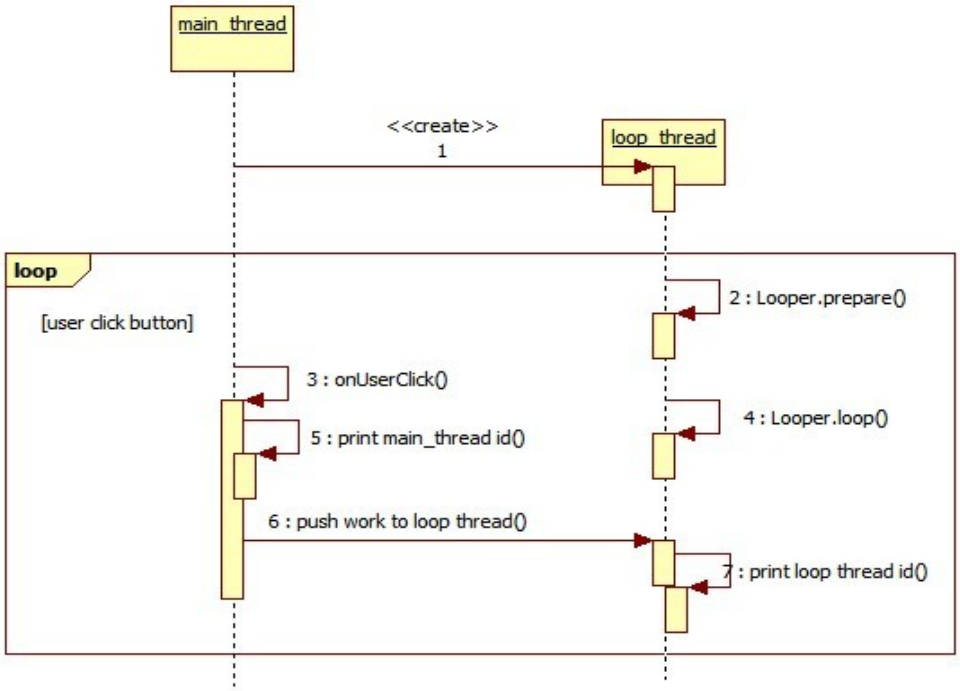
```
package invistd.demo;

import android.os.Handler;
import android.os.Looper;

public class LoopThread extends Thread{
    public Handler mHandler;

    @Override
    public void run() {
        //create work queue for thread
        Looper.prepare();

        //Handler will be used to push work to thread
        mHandler = new Handler();

        //keep thread alive
        Looper.loop();
    }
}
```

Implement Main Activity

```
package invistd.demo;

import android.os.Handler;
import android.os.Looper;
import android.os.Message;
import android.support.v7.app.AppCompatActivity;
```

```
import android.os.Bundle;
import android.util.Log;
import android.view.View;

public class MainActivity extends AppCompatActivity {

    LoopThread mLoopThread;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);

        mLoopThread = new LoopThread();
        mLoopThread.start();//start loop thread
    }

    void onUserClick(View v) {
        Log.i("maxter", "this is main thread id: " + Thread.currentThread().getId());

        //push work to loop thread
        mLoopThread.mHandler.post(new Runnable() {
            @Override
            public void run() {
                Log.i("maxter", "this is loop thread id: " +
Thread.currentThread().getId());
            }
        });
    }

    @Override
    protected void onDestroy() {
        super.onDestroy();
        mLoopThread.mHandler.getLooper().quit();
    }
}
```

Sequence Diagram



Analyze The Sample

How to create a loop thread?

In *Thread::run(),* we perform belows steps:
- Call *Looper.prepare()* to create work queue for the thread.
- Create a *Handler*, the *Handler* will be used to push work.
- Call *Looper.loop()* to keep thread alive.

How to push work to loop thread?

- Call *Handler.post(work),*
- *work* is a Runnable object.

Main Thread Is A Loop Thread

Main thread has an associated work queue and keep alive, by default.

We can push work to it by this code.

```
//create work
Runnable work = new Runable(){...};

//create main handler
Handler mainHandler = new Handler(Looper.getMainLooper());

//post work
mainHandler.post(work);
```