**post :post causes the Runnable to be added to the message queue,**

**Runnable : Represents a command that can be executed. Often used to run code in a different Thread.**

**run () : Starts executing the active part of the class' code. This method is called when a thread is started that has been created with a class which implements Runnable.**

**getView().post(new Runnable() {**

**@Override**

**public void run() {**

**getView().startAnimation(a);**

**}**

**});**

**code : getView().startAnimation(a);**

**in your code,**

**post causes the Runnable (the code will be run a in different thread) to add the message queue.**

**So startAnimation will be fired in a new thread when it is fetched from the messageQueue**

**[EDIT 1]**

**Why do we use a new thread instead of UI thread (main thread)?**

**UI Thread :**

**When application is started, Ui Thread is created automatically**

**it is in charge of dispatching the events to the appropriate widgets and this includes the drawing events.**

**It is also the thread you interact with Android widgets with**

**For instance, if you touch the a button on screen, the UI thread dispatches the touch event to the widget which in turn sets its pressed state and posts an invalidate request to the event queue. The UI thread dequeues the request and notifies the widget to redraw itself.**

**What happens if a user press a button which will do longOperation ?**

**((Button)findViewById(R.id.Button1)).setOnClickListener(**

**new OnClickListener() {**

**@Override**

**public void onClick(View v) {**

**final Bitmap b = loadImageFromNetwork();**

**mImageView.setImageBitmap(b);**

**}**

**});**

**The UI freezes. The program may even crash.**

**public void onClick(View v) {**

**new Thread(new Runnable() {**

**public void run() {**

**final Bitmap b = loadImageFromNetwork();**

**mImageView.setImageBitmap(b);**

**}**

**}).start();**

**}**

**It breaks the android rule that never update UI directly from worker thread**

**Android offers several ways to access the UI thread from other threads.**

**Activity.runOnUiThread(Runnable)**

**View.post(Runnable)**

**View.postDelayed(Runnable, long)**

**Handler**

**Like below,**

**View.post(Runnable)**

**public void onClick(View v) {**

**new Thread(new Runnable() {**

**public void run() {**

**final Bitmap b = loadImageFromNetwork();**

**mImageView.post(new Runnable() {**

**public void run() {**

**mImageView.setImageBitmap(b);**

**}**

**});**

**}**

**}).start();**

**}**

**Handler**

**final Handler myHandler = new Handler();**

**(new Thread(new Runnable() {**

**@Override**

**public void run() {**

**final Bitmap b = loadImageFromNetwork();**

**myHandler.post(new Runnable() {**

**@Override**

**public void run() {**

**mImageView.setImageBitmap(b);**

**}**

**});**

**}**

**})).start();**

**}**

**enter image description here**

**For more info**

**http://android-developers.blogspot.com/2009/05/painless-threading.html**

**http://www.aviyehuda.com/blog/2010/12/20/android-multithreading-in-a-ui-environment/**