Trường ĐH Khoa Học Tự Nhiên Tp. Hồ Chí Minh
TRUNG TÂM TIN HỌC

Lập trình Android

Bài 17: *Lập trình đa tiểu trình (Multithreading)*

Phòng LT & Mạng
http://csc.edu.vn/lap-trinh-va-csdl

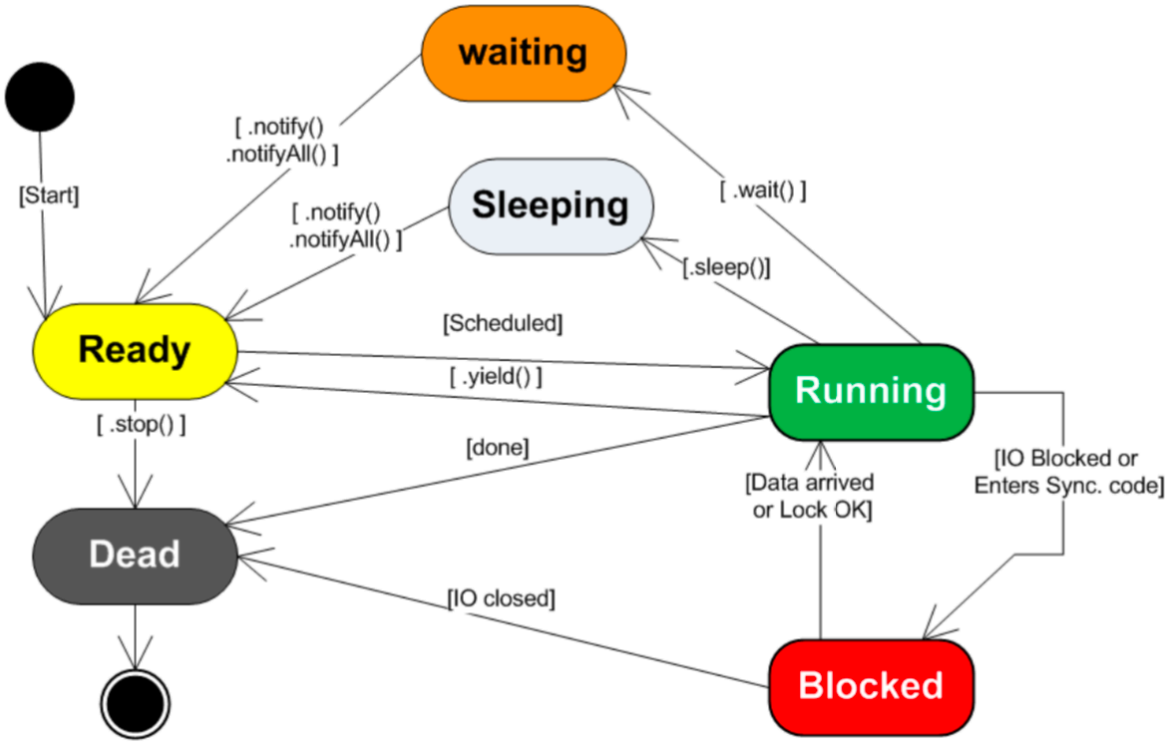
2019

Nội dung

1. Điều khiển đồng thời
2. Thread
3. Runnable
4. AsyncTask

Điều khiển đồng thời

- Trong một ứng dụng, có thể có xảy ra trường hợp cần làm nhiều việc trong cùng một thời điểm. Giải pháp là chạy các công việc này trên những **thread** độc lập.
- Máy ảo Java (JVM) cung cấp kiến trúc multi-threading cho phép thực hiện các công việc đồng thời.
- Chu trình của một Java thread như sau, cho biết các trạng thái mà một Java thread có thể chạm tới:



Tạo và thực thi Thread

- Có 2 cách để tạo và thực thi 1 Java Thread:
- **Cách 1**: Tạo 1 đối tượng thread và truyền đối tượng Runnable cho nó.

```
class RunnableClass implements Runnable {
    public void run() {
    }
}
```

```
RunnableClass runnableObj = new RunnableClass();
Thread thread = new Thread(runnableObj);
thread.start();
```

- **Cách 2**: Tạo một sub-class extends từ class Thread và override phương thức run()

```
class ThreadClass extends Thread {
    public void run() {
    }
}
```

```
ThreadClass thread = new ThreadClass();
thread.start();
```

Ví dụ

```
public class RunnableClass implements Runnable {

    @Override
    public void run() {
        for (int i = 10; i < 15; i++) {
            try {
                Thread.sleep(1000);
            } catch (InterruptedException e) {
                e.printStackTrace();
            }
            Log.i("CSC", "Runnable " + i);
        }
    }
}
```

```
public class ThreadClass extends Thread {
    @Override
    public void run() {
        super.run();
        for (int i = 0; i < 5; i++) {
            try {
                Thread.sleep(1000);
            } catch (InterruptedException e) {
                e.printStackTrace();
            }
            Log.i("CSC", "Thread " + i);
        }
    }
}
```

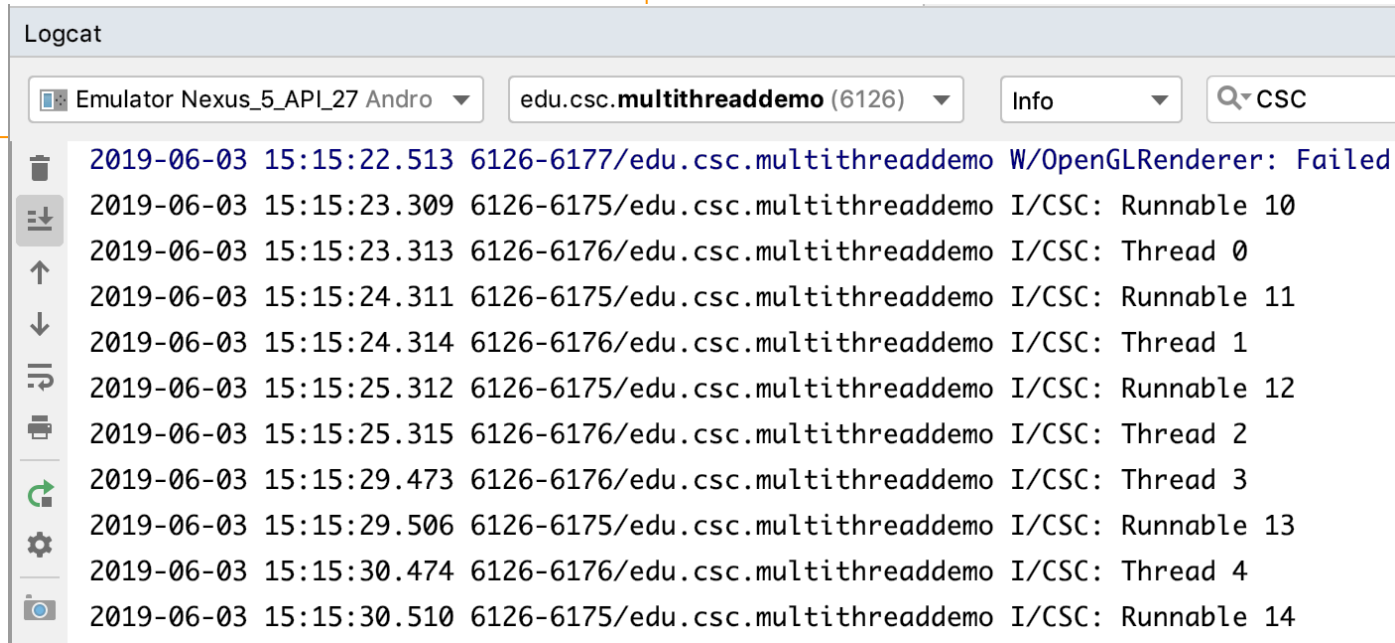
Ví dụ (tiếp)

```
public class MainActivity extends AppCompatActivity {

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

        Runnable runnable = new RunnableClass();
        Thread thread = new Thread(runnable);
        thread.start();

        ThreadClass myThread = new ThreadClass();
        myThread.start();
    }
}
```



AsyncTask

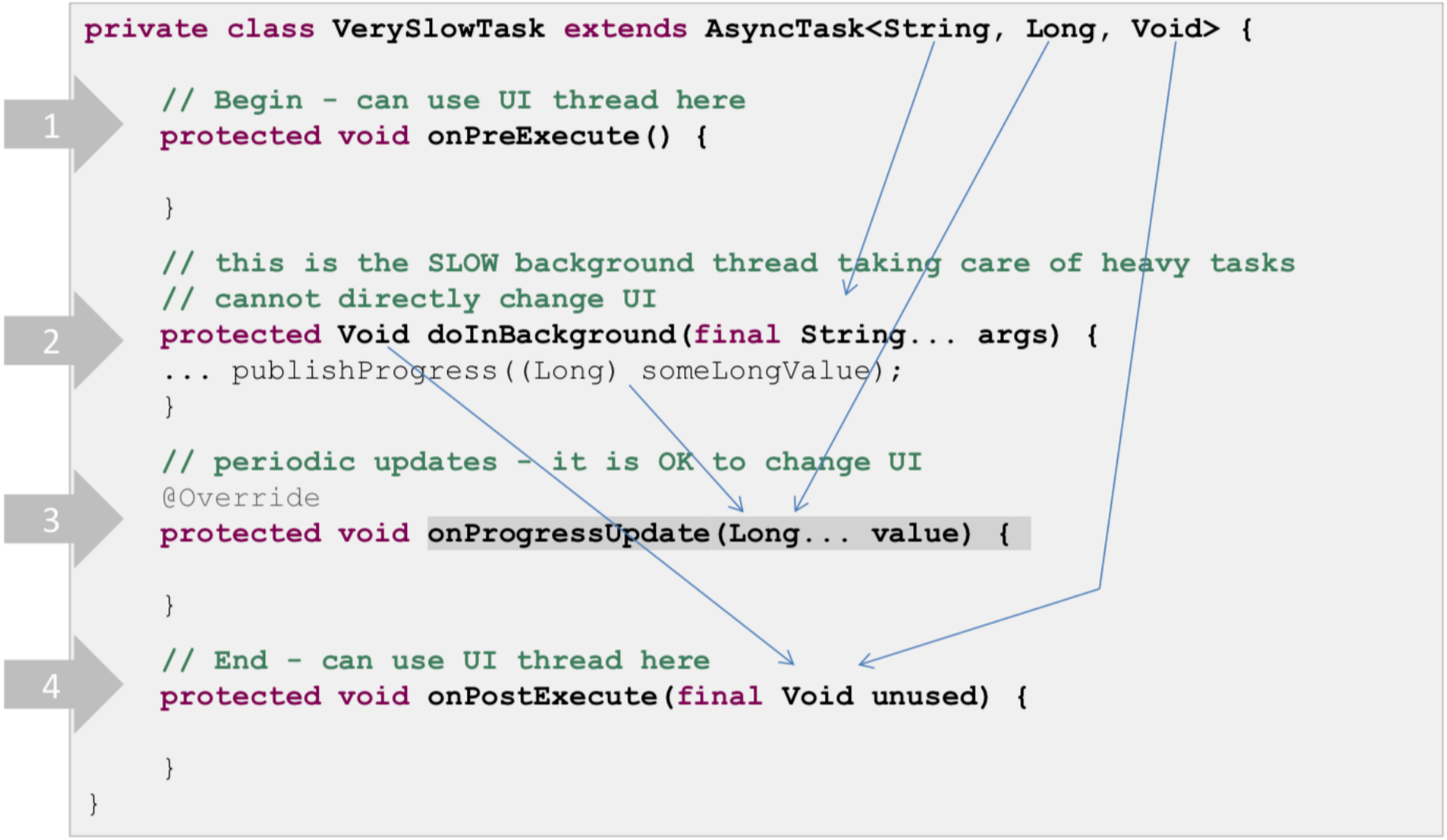
- AsyncTask cho phép thực thi 1 công việc ở background thread và trả kết quả về UI thread (main thread)
- Một AsyncTask được định nghĩa bởi các thành phần sau:

Types	States	Method
Params, Progress, Result	onPreExecute, doInBackground, onProgressUpdate, onPostExecute	publishProgress

Sử dụng AsyncTask - Types

- **AsyncTask <Params, Progress, Result>**
  - **Params**: Kiểu dữ liệu của tham số truyền vào task để thực thi
  - **Progress**: Kiểu dữ liệu của progress được gửi ra trong quá trình tính toán của background.
  - **Result**: Kiểu dữ liệu của kết quả trả về sau quá trình tính toán ở background.
  - Chú ý:
    - Tham số nào không sử dụng: thay thế bằng **Void**
    - **String…** tương đương **String[]**

Sử dụng AsyncTask

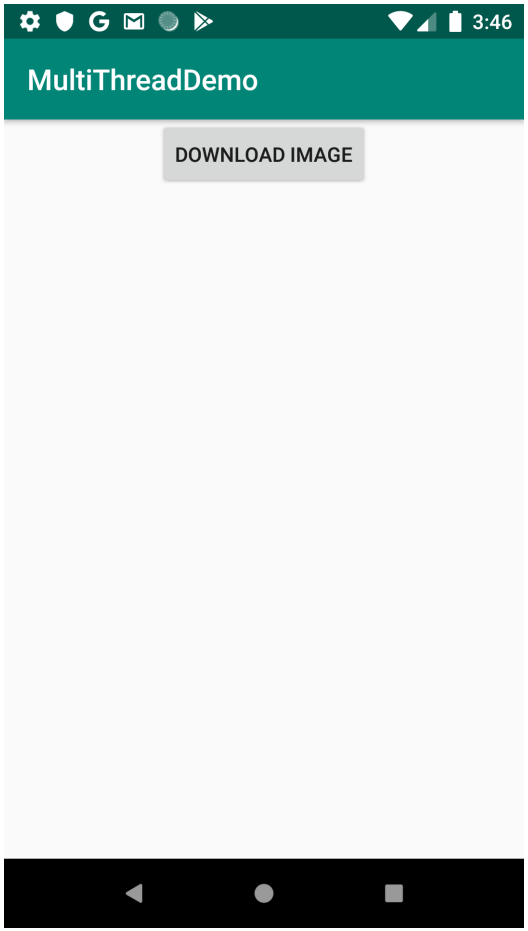


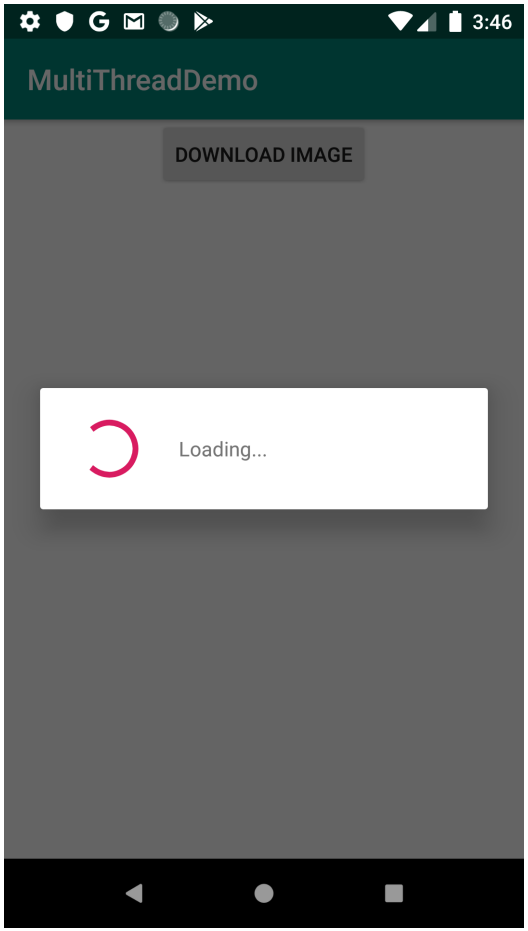
Sử dụng AsyncTask - Method

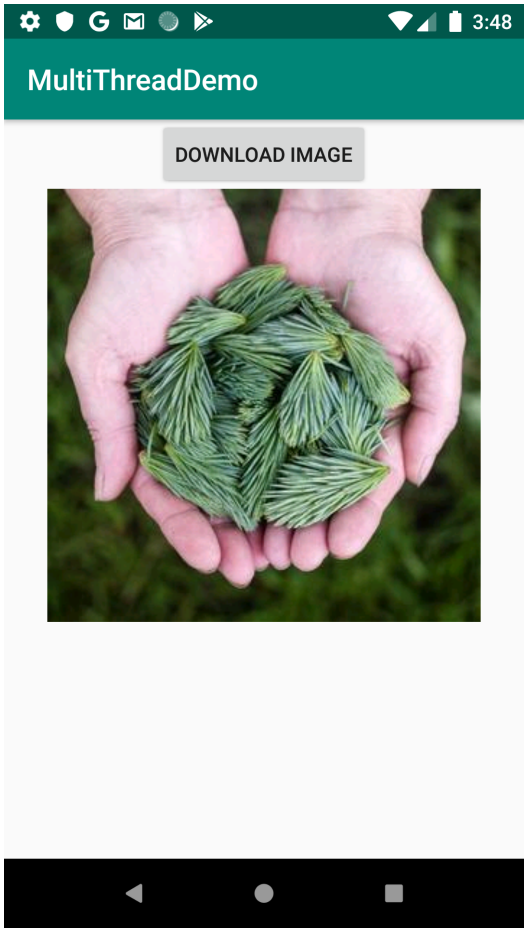
- **onPreExecute():** được gọi từ UI thread ngay sau khi thực thi task. Bước này thường dùng để setup task, vd: hiển thị một progress bar lên giao diện
- **doInBackground(Params…)**: được gọi ở background thread ngay sau *onPreExecute()* hoàn tất. Bước này thường thực hiện các tính toán, xử lý tốn nhiều thời gian. Bước này còn gọi đến *publishProgress(Progress…)* để gửi một hoặc nhiều thông tin về progress, các giá trị này được gửi đến UI thread trong phương thức *onProgressUpdate(Progress…)*
- **onProgressUpdate(Progress…)**: được gọi ở UI thread sau khi gọi *publishProgress(Progress…)*. Phương thức này dùng để thông tin đến người dùng về progress trong lúc background đang xử lý, tính toán.
- **onPostExecute(Result)**: được gọi ở UI thread sau khi quá trình xử lý ở background kết thúc. Kết quả tính toán của background được truyền tới bước này chính là tham số của phương thức. Ở bước này, UI thread thực hiện cập nhật giao diện với kết quả trả về (và ẩn progress bar đã hiển thị ở onPreExecute()).

Ví dụ

- Tải về một hình ảnh từ Internet và hiển thị lên ImageView







Ví dụ (tiếp)

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

    <Button
        android:id="@+id/btnDownload"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Download Image" />

    <ImageView
        android:id="@+id/ivDownload"
        android:layout_width="300dp"
        android:layout_height="300dp" />
</LinearLayout>
```

activity_main.xml

Ví dụ (tiếp)

```
public class MainActivity extends AppCompatActivity {
    ProgressDialog dialog; URL imageUrl = null; InputStream is = null;
    Bitmap bitmap = null; ImageView ivDownload = null;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
    }
    // Khai class ImageDownloaderAsyncTask kế thừa từ AsynTask
    private class ImageDownloaderAsyncTask extends AsyncTask<String, Void, Bitmap> {
        @Override
        protected void onPreExecute() {
            super.onPreExecute();
        }
        @Override
        protected Bitmap doInBackground(String... strings) {
            return bitmap;
        }
        @Override
        protected void onPostExecute(Bitmap bitmap) {
            super.onPostExecute(bitmap);
        }
    }
}
```

url string của image cần tải về

bitmap kết quả (hình ảnh được tải về)

MainActivity.java

Ví dụ (tiếp)

```
@Override
    protected void onPreExecute() {
        super.onPreExecute();
        dialog = new ProgressDialog(MainActivity.this);
        dialog.setMessage("Loading...");
        dialog.setIndeterminate(false);
        dialog.setCancelable(false);
        dialog.show();
    }
```

Hiển thị progress dialog trước khi thực hiện tác vụ

Lấy url string từ tham số đầu vào

```
@Override
    protected Bitmap doInBackground(String... strings) {
        try {
            imageUrl = new URL(strings[0]);
            HttpURLConnection conn = (HttpURLConnection) imageUrl.openConnection();
            conn.setDoInput(true);
            conn.connect();
            is = conn.getInputStream();
            BitmapFactory.Options options = new BitmapFactory.Options();
            options.inPreferredConfig = Bitmap.Config.RGB_565;
            bitmap = BitmapFactory.decodeStream(is, null, options);
        } catch (IOException e) {
            e.printStackTrace();
        }
        return bitmap;
    }
```

- Thực hiện việc download hình ảnh
- Trả về bitmap kết quả sau khi thực hiện xong

Ví dụ (tiếp)

```
@Override
protected void onPostExecute(Bitmap bitmap) {
        super.onPostExecute(bitmap);
        dialog.hide();
        if (ivDownload != null)
            ivDownload.setImageBitmap(bitmap);
}
```

- Ẩn progress dialog
- Hiển thị bitmap kết quả lên image view

```
public class MainActivity extends AppCompatActivity {

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        ivDownload = findViewById(R.id.ivDownload);
        Button btnDownload = findViewById(R.id.btnDownload);

        btnDownload.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                ImageDownloaderAsyncTask downloader = new ImageDownloaderAsyncTask();
                downloader.execute("https://picsum.photos/300");
            }
        });
    }
```

- Khởi tạo image downloader
- Gọi hàm execute với tham số là url string của image cần tải về

Q&A