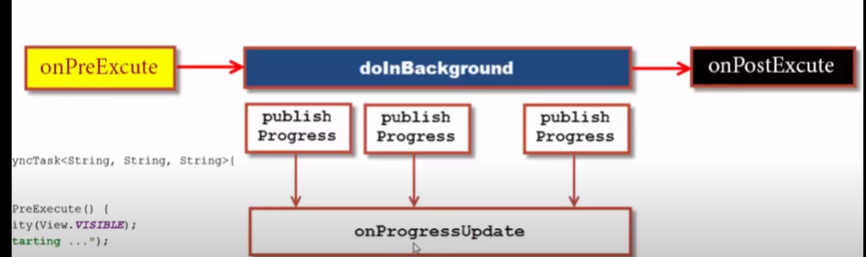
Giới thiệu :

1. AsyncTask dùng để làm gì:

* Trong Android, đôi khi chúng ta sẽ thực hiện một số operation mà thời gian từ lúc gửi request đến lúc nhận response có thể hơi lâu như network operation: download ảnh, down phim…; database query, lấy thông tin danh sách ứng dụng trong hệ điều hành, gửi file bluetooth… Người ta hay gọi chúng là **Background operations**.
* Main Thread (thread dùng để update giao diện) thì thực thi công việc theo cơ chế hàng đợi FIFO, do đó nếu ta thực hiện background operation bằng main thread sẽ dẫn tới hiện tượng đơ, treo màn hình do công việc sau muốn chạy phải đợi công việc trước được thực hiện xong.
* Vậy nên Google làm ra AsyncTask để thực hiên **background operation**. Không dùng Main Thread nữa.



Để sử dụng asynctask ta phải khai báo nó trong 1 class để thực thi

Asynctask có 4 function :

**onPreExcute**

**doInBackground**

**onPostExcute**

**onProgerssUpdate**

**doInBackground :**

là thằng sử lý chính làm việc chính trong tất cả 4 hàm

và nó bắt buộc phải được gọi

nó có thể chạy độc lập 1 mình không liên quan 3 tk còn lại

3 tk còn có thể gọi hoặc ko gọi cũng ok

Trong lúc nó sử lý công việc của mình thì nó chỉ làm việc nó chạy nó không làm bất cứ thứ gì liên quan tới việc thay đổi giao diện

Thứ tự chạy và cách chạy như sau:

Với asynctask lần đầu tiên nó vào thì nó sẽ chạy onPreExcute

Sau đó tiếp tục đến doInBackground (hàm sử lý chính công việc) sau khi doInBackground hoạt động xong(kết thúc ) thì nó sẽ trả kết quả ví dụ như a+b =? Thì nó gang kết quả nó trả về cho tk onPostExcute( onPostExcute nó chỉ chạy khi nào tk doInBackground thực hiện xog và nó nhận kết quả từ doInBackground trả về)

Và tk onProgressUpdate (một quá trình sử lý cập nhật ) cập nhập lại trạng thái của doInBackground

Muốn biết tiến trình xoá , hay download 1 file chẳng hạn bạn muốn biết nó down dc bao nhìu % thì thằng onPostExcute nó chỉ nhận khi mà nó có cái file đó rồi và nó thông báo download thành công còn bây giờ muốn biết bao nhiu phần trăm thì phải dựa vào onProgressUpdate tức là trong lúc sử lý doInBackground download dc 5% 10% nó sẽ trả ra ngoài giao diện nhưng nó không trả trực tiếp được trong doInBackground ta gọi 1 hàm là public Progress trong đây nó sẽ cập nhật quá trình thằng doInBackground nó sử lý gì gì đó nó trả về public progress và public progress nó sẽ truyền ra function bên ngoài onProgressUpdate để nó hiển thị lên màn hình của mình mình gán vào text view hay gì đó hoặc progressbar và đó là cách hoặc động và thứ tự sử lý

Params là tham số có thể được nhận từ execute(), **có thể là một mảng các tham số con**. và Params sẽ là input của doInBackground(), Process là input của onProgressUpdate(), đầu ra này nhận từ doInBackground() thông qua phương thức publishProgress(). onProgressUpdate() có thể cập nhật giao diện lúc runtime. Result là đầu ra của doInBackground() và chính là kết quả trả về ở onPostExecute(). AsyncTask chạy trên Worker Thread còn Handler chạy trên Main Thread (hay Thread sinh ra nó).

\*Note

Ứng dụng trở nên không phản hồi vì luồng chính đang chờ tải xuống hoàn tất và đã chuyển sang trạng thái treo.

Khi phản hồi (tệp PNG) từ máy chủ đang được chờ đợi, ứng dụng sẽ bắt đầu trạng thái unreponsive (không phản hồi) từ khi Main thread bắt đâu chờ đợi hoạt động tải xuống cho đến khi hoàn tất.

Để khắc phục điều này, chúng ta có thể tạo luồng mới và thực hiện các phương thức để thực hiện các hoạt động này tương tự như chúng ta vẫn thường làm trong các ứng dụng Java thông thường khác để UI vẫn có thể đáp ứng.

Nhưng việc xử lý với các luồng riêng biệt này có thể tạo ra một số vấn đề bổ sung khi chúng ta cố gắng cập nhật giao diện người dùng dựa trên kết quả của các hoạt động được thhực hiện do bộ công cụ giao diện người dùng Android không an toan.

Vi vậy Android đã cân nhắc tất cả vấn đề này và tạo ra một lớp chuyên dụng có tên AsyncTask, để xử lý các tác vụ / thao tác cần được thực hiện ở chế độ nền không đông bộ.

AysncTask là gì?

AsyncTask là một lớp Android trừu tượng giúp các ứng dụng Android xử lý luồng UI chính theo cách hiệu quả. Lớp AsyncTask cho phép chúng ta thực hiện các tác vụ / thao tác nền lâu dài và hiển thị kết quả trên luồng UI mà không ảnh hưởng đến luồng chính.

Khi nào nên sử dụng AT?

Giả sử bạn đã tạo một ứng dụng Android đơn giản tải tệp MP3 từ Internet khi khởi chạy ứng dụng. Biểu đồ trạng thái bên dưới hiển thị chuỗi các hoạt động sẽ diễn ra khi bạn khởi chạy ứng dụng bạn đã tạo.