**Activity launchMode**

**Assign a launchMode (Khai báo lauchMode)**

Cách cơ bản nhất là sử dụng thẻ tag <activity> và khai báo nó trong file AndroidManifest.xml:

<**activity**

**android:name=".MainActivity"**

**android:label="@string/app\_name"**

**android:launchMode="standard"**>

Sau đây là 4 loại launchMode được cung cấp

**1. Standard**

Đây là lauchMode mặc định. Một Activity mới sẽ luôn được khởi tạo mới cho tác vụ riêng với mỗi

Intent gửi đi. Tức Activity có thể được khởi tạo nhiều lần, mỗi Intent có thể thuộc về các task khác nhau, và một task có thể có nhiều Intent.

Ví dụ, nếu có 10 Intent được gửi để soạn thảo mail, thì sẽ có 10 activity được khởi tạo với mỗi intent riêng biệt. Như vậy có thể sẽ không giới hạn số activity được khởi tạo trên một thiết bị.

**1.1. Trước version Lollipop**

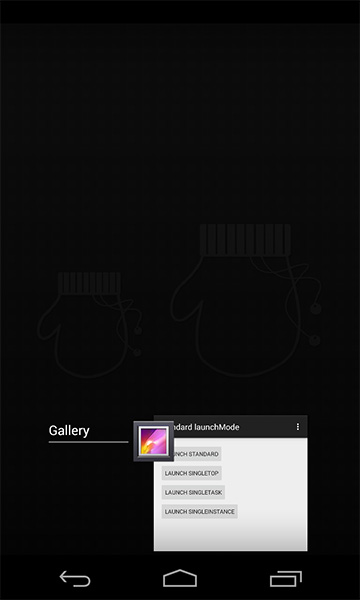
Activity sẽ được khởi tạo và đặt ngay ở trên top của stack trong cùng một task



Hình dưới sẽ mô tả cụ thể hoạt động của standard Activity, nó sẽ được đặt trong cùng một task mặc dù đó là những activity từ các ứng dụng khác nhau.



và khi nhìn bằng trình quản lý Task Manager



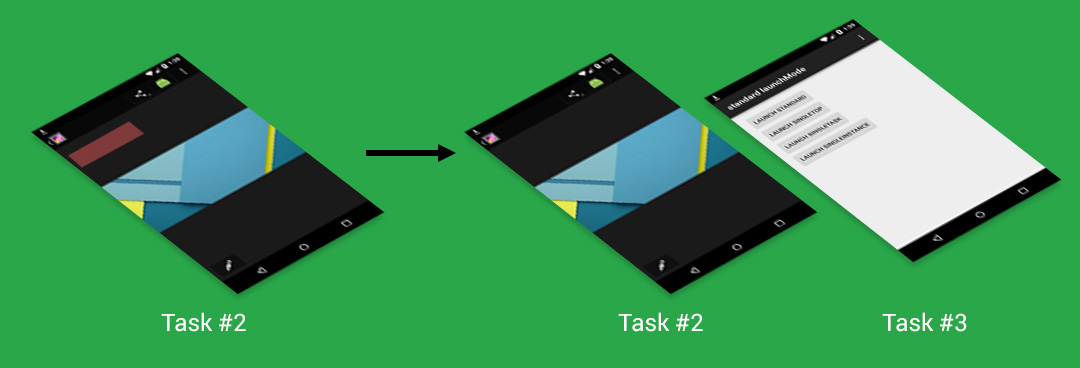
Nếu ta chuyển sang một ứng dụng khác và quay lại Gallery, ta vẫn thấy nó ở top của Gallery task. Như vậy nếu ta cần làm bất cứ điều gì với Gallery thì ta phải finish Activity ở top của task.

**1.2. Trong version Lollipop**

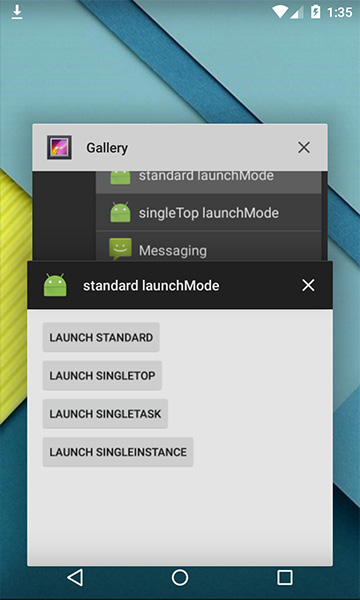
Nếu các activity đều từ cùng một ứng dụng, thì nó vẫn hoạt động như những version trước đó, activity vẫn được stack vào top của task.



Nhưng đối với các intent được gửi từ những ứng dụng khác, một task mới sẽ được khởi tạo và activity mới được tạo sẽ được đặt làm root Activity.



Nhìn vào Task Manager



Như ta thấy, Task Manager trong Lollipop được sửa đổi để tốt hơn và trực quan hơn. Trong Lollipop, ta chỉ có thể chuyển về Gallery khi mà nó được đặt ở task khác.

Nếu bạn nghĩ về một Activity có thể thực hiện một công việc riêng biệt để phục vụ một intent riêng biệt, hãy nghĩ đến mode Standard.

**2. SingleTop**

SingleTop hoạt động gần giống với standard lauchMode. Một activity có thể được khwoir tạo nhiều lần, mỗi intent có thể thuộc về nhiều task khác nhau, và một task có thể chứa nhiều intent (nhưng chỉ khi activity ở top của back stack không phải là một thể hiện đang tồn tại của activity này)

Điểm khác biệt là nếu một thể hiện của activity đã tồn tại ở top của task hiện tại, sẽ không có một activity mới nào được tạo ra, thay vì đó Intent sẽ được gửi tới thể hiện của activity hiện tại thông qua phương thức onNewIntent().



Trong singleTop mode, ta phải quản lý các Intent đến trong cả onCreate() và onNewIntent() để làm cho có có thể làm việc trong tất cả các trường hợp.

Ví dụ: Nếu ta luân khởi tạo mới một SearchActivity để phục vụ những yêu cầu tìm kiếm mới, thì 10 Activity mới sẽ được tạo cho 10 searching. Như vậy để quay lại root Activity ta phải back lại 10 lần tương ứng với 10 searching. Thay vào đó, nếu SearchActivity ở top của task, ta có thể send một intent đến thể hiện của activity hiện tại và update kết quả tìm kiếm. Giờ ta vẫn chỉ có một SearchActivity ở top của stack và chỉ cần ấn một lần nút back để quay về activity trước đó. Điều này tạo nên trải nghiệm tốt hơn.

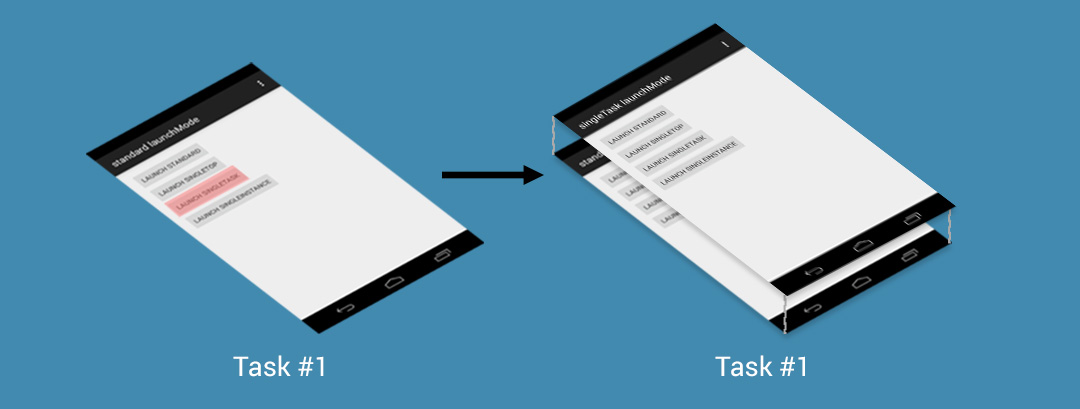
SingleTop hoạt động trên cùng một task. Nếu một Intent được gửi đến một Activity (singleTop) đang tồn tại ở top của bất kỳ một task khác thì một Activity sẽ được tạo mới giống như standard lauchMode (trước Lollipop: activity được đặt ở top của task gọi; trong Lollipop: một task mới sẽ được tạo ra).

**3. SingleTask**

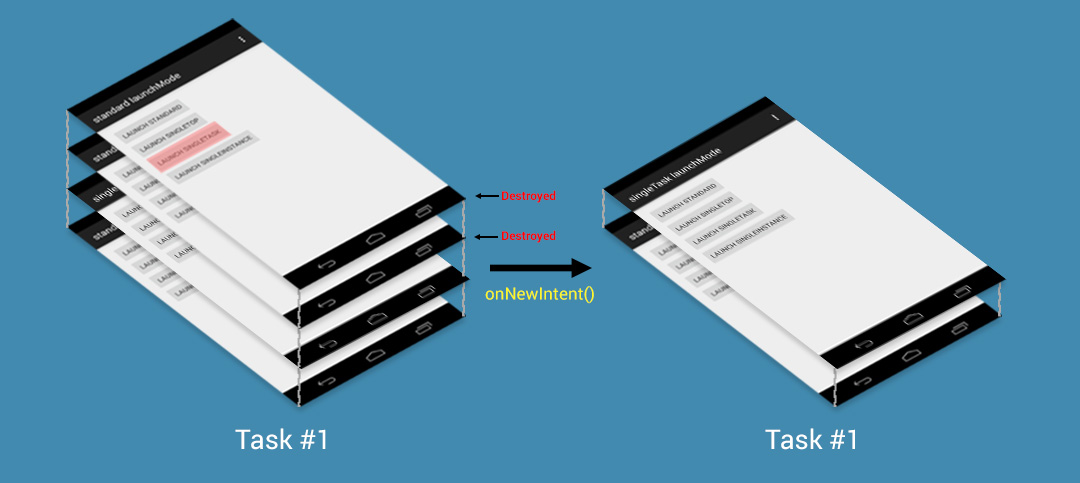
Một singleTask Activity được phép chỉ có duy nhất một thể hiện trên hệ thống (a.k.a.Singleton). Nếu một thể hiện của Activity đã tồn tại trên hệ thống, tất cả Task chứa thể hiện đó sẽ được di chuyển lên top, đồng thời Intent sẽ được thực hiện thông qua phương thức onNewIntent(). Activity sẽ được tạo và đặt ở vị trí thích hợp.

**Làm việc trên cùng một ứng dụng**

Nếu nó không phải là một thể hiện của singleTask Activity đã có trên hệ thống, nó sẽ được tạo mới và đặt vào top của stack trong cùng một Task.



Nhưng nếu một singleTask Activity đã tồn tại, tất cả Activity được đặt ở trên activity này đều được tự động destroy một cách phù hợp (lifecycle trigged). Trong khi đó, một Intent sẽ được gửi đến singleTask Activity thông qua phương thức onNewIntent().



Nhưng từ thực nghiệm, nó không hoạt động giống như được mô tả. Một singleTask Activity vẫn stack up ở top của Task'Activity stack, ta có thể sử dụng command: dumpsys activity để thấy điều đó

Task id #239

  TaskRecord{428efe30 #239 A=com.thecheesefactory.lab.launchmode U=0 sz=2}

  Intent { act=android.intent.action.MAIN cat=[android.intent.category.LAUNCHER] flg=0x10000000 cmp=com.thecheesefactory.lab.launchmode/.StandardActivity }

    Hist #1: ActivityRecord{429a88d0 u0 com.thecheesefactory.lab.launchmode/.SingleTaskActivity t239}

      Intent { cmp=com.thecheesefactory.lab.launchmode/.SingleTaskActivity }

      ProcessRecord{42243130 18965:com.thecheesefactory.lab.launchmode/u0a123}

    Hist #0: ActivityRecord{425fec98 u0 com.thecheesefactory.lab.launchmode/.StandardActivity t239}

      Intent { act=android.intent.action.MAIN cat=[android.intent.category.LAUNCHER] flg=0x10000000 cmp=com.thecheesefactory.lab.launchmode/.StandardActivity }

      ProcessRecord{42243130 18965:com.thecheesefactory.lab.launchmode/u0a123}

Để một singleTask Activity hoạt động như được mô tả: tạo mới Task và đặt Activity vào root, ta cần phải khai báo thuộc tính taskAffinity

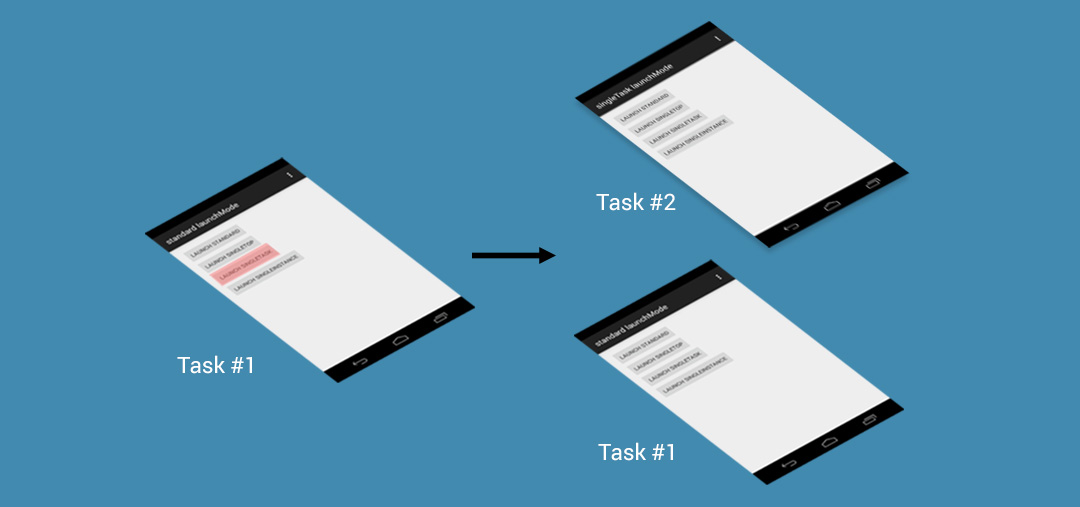
<**activity**

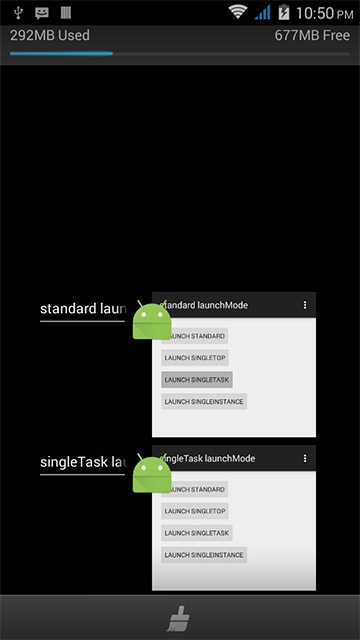
**android:name=".MainActivity"**

**android:label="@string/app\_name"**

**android:launchMode="singleTask"**

**android:taskAffinity=""**>

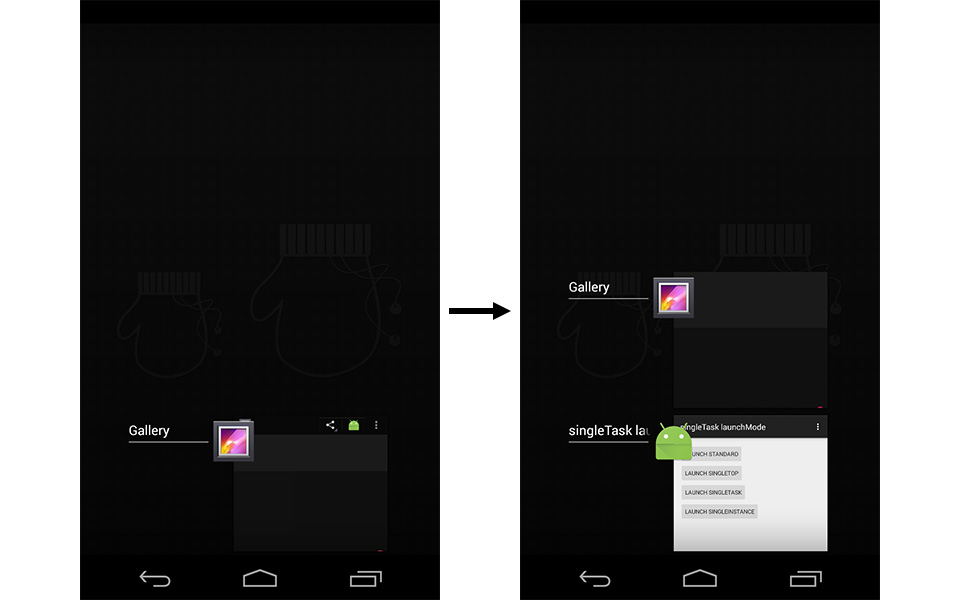




**Liên kết với các ứng dụng khác**

Lần đầu một Intent được gửi từ một ứng dụng khác và không có bất kỳ thể hiện của Activity được tạo trên hệ thống, một new Task sẽ được tạo với Activity được tạo mới nhất đặt vào root Activity.

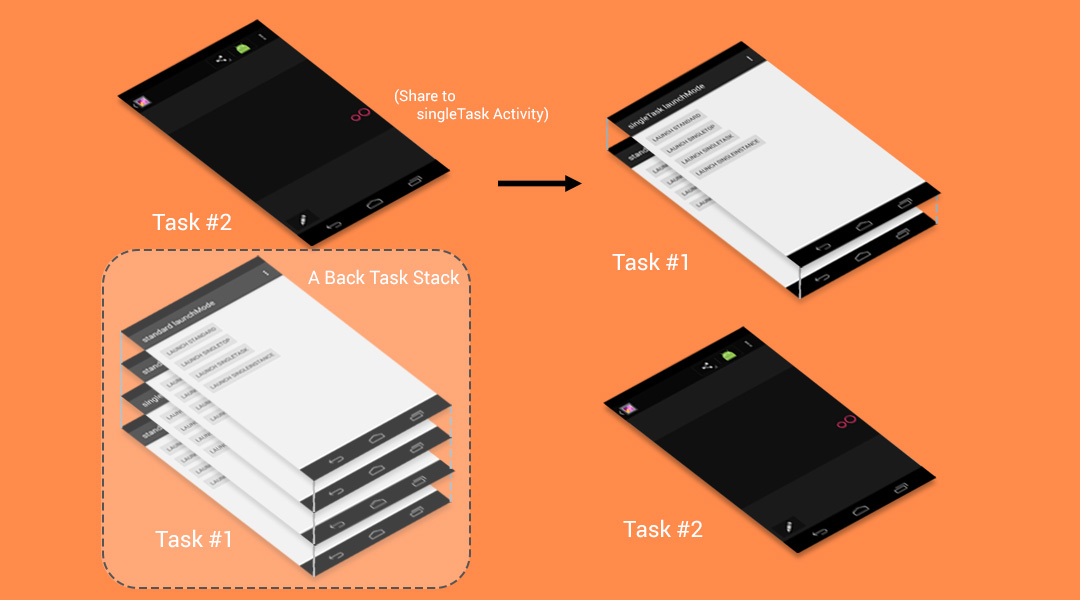




Trừ khi đó là một Task của ứng dụng gọi đến singleTask Activity có sẵn, thì Activity mới nhất được đặt ở top.



Trong trường hợp đó là thể hiện của activity có sẵn tại bất kỳ Task nào, tất cả Task sẽ được di chuyển lên top và bất kỳ single Activity được đặt ở trên singleTask Activity sẽ bị phá hủy với lifecycle. Nếu nút back được ấn, user sẽ phải đi qua tất cả activity trong stack trước khi quay lại caller Task.

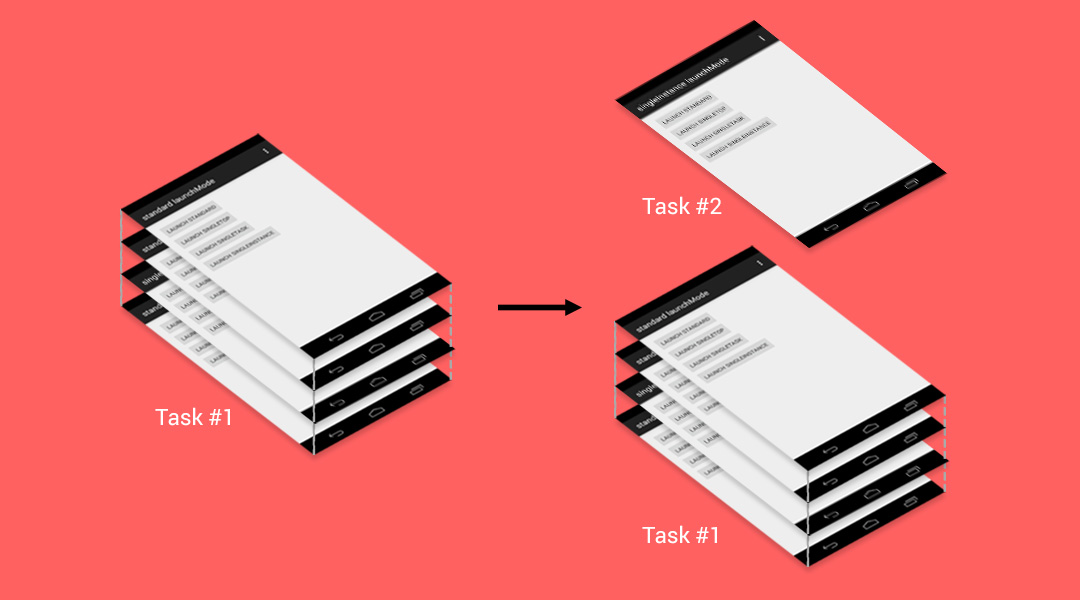


Ví dụ: bất kỳ Entry Point Activity cho Email Client's Inbox hay Social Network's Timline đều được thiết kế để không có quá một thể hiện, nên singleTask sẽ có thể thực hiện một cách rất tốt điều này. Đồng thời cũng cần sử dụng một cách khôn ngoan vì Activity có thể bị destroy mà user không công nhận.

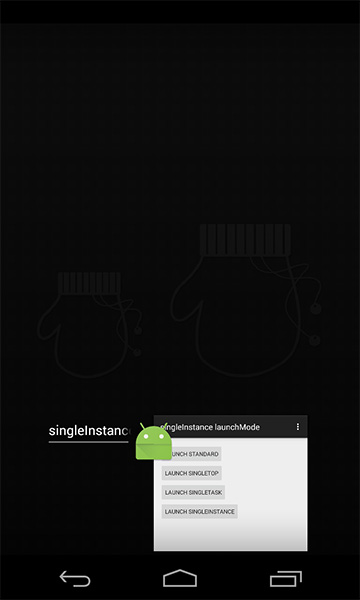
**4. SingleInstance**

singleInstance rất giống với singleTask mode. Điểm khác biệt là Task chỉ chứa một Activity, một singleInstance. Nếu một Activity khác được gọi từ Activity loại này, một Task mới sẽ được tự động khởi tạo để đặt Activity mới. Tương tự vậy, nếu một singleInstance Activity được gọi, một Task mới sẽ được khởi tạo để đặt Activity.

Theo những thông tin được cung cấp bởi dumpsys, có thể thấy 2 Task trong hệ thống, nhưng chỉ có một Task được xuất hiện trong Task Manager phụ thuộc vào cái nào được di chuyển lên top lần cuối. Vậy nên, mặc dù có Task vẫn hoạt động ở background nhưng chúng ta không thể chuyển nó lên foregound.

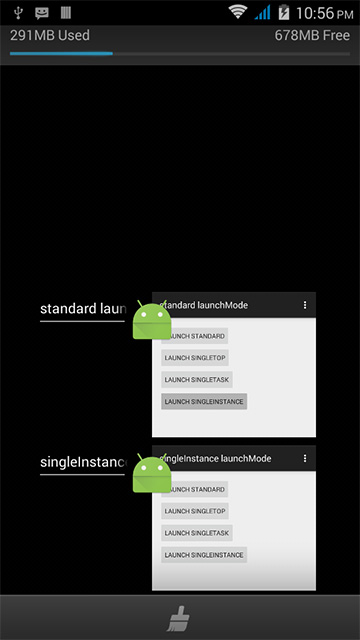


nhìn trong Task Manager



Vì Task này chỉ có thể có một Activity, ta không thể switch back lại Task#1 được nữa. Chỉ có một cách đó là chạy lại ứng dụng từ launcher nhưng singleInstance Task sẽ bị ẩn ở background.

Tuy nhiên ta có thể khai báo thuộc tính taskAffinity (như singleTask) để có thể thực hiện việc trên.



Chế độ này ít được sử dụng.Một số trường hợp được sử dụng như là một Activity cho Launcher hay nếu bạn chắc chắn 100% rằng ứng dụng chỉ có duy nhật một Activity. Lời khuyên: không nên sử dụng chế độ này, trừ khi thật cần thiết.

**5. Intent Flags**

Bên cạnh cách khai báo trực tiếp trong file AndroidManifest.xml, ta có thể chỉ định hành vi thông qua Intent Flags:

Intent intent = new Intent(StandardActivity.this, StandardActivity.class);

intent.addFlags(Intent.FLAG\_ACTIVITY\_SINGLE\_TOP);

startActivity(intent);