**Android 近期任务是如何杀进程的**

从近期任务划去一个卡片到系统杀死这个月应用的一系列流程。

**任务消去流程**

划去卡片后，会调用：

com.android.server.am.ActivityManagerService#removeTask

com.android.server.wm.ActivityTaskManagerService#removeTask

com.android.server.wm.ActivityStackSupervisor#removeTaskByIdLocked

// 这个方法中有各种情况判断这个进程是否需要杀死

com.android.server.wm.ActivityStackSupervisor#cleanUpRemovedTaskLocked

com.android.server.am.ActivityManagerService.LocalService#killProcessesForRemovedTask

|  |
| --- |
| Java void cleanUpRemovedTaskLocked(TaskRecord tr, boolean killProcess, boolean removeFromRecents) {  if (removeFromRecents) {  mRecentTasks.remove(tr);  }  ComponentName component = tr.getBaseIntent().getComponent();  if (component == null) {  Slog.w(TAG, "No component for base intent of task: " + tr);  return;  }   // Find any running services associated with this app and stop if needed.  final Message msg = PooledLambda.obtainMessage(ActivityManagerInternal::cleanUpServices,  mService.mAmInternal, tr.userId, component, new Intent(tr.getBaseIntent()));  mService.mH.sendMessage(msg);   if (!killProcess) {  return;  }   // Determine if the process(es) for this task should be killed.  final String pkg = component.getPackageName();  ArrayList<Object> procsToKill = new ArrayList<>();  ArrayMap<String, SparseArray<WindowProcessController>> pmap =  mService.mProcessNames.getMap();  for (int i = 0; i < pmap.size(); i++) {   SparseArray<WindowProcessController> uids = pmap.valueAt(i);  for (int j = 0; j < uids.size(); j++) {  WindowProcessController proc = uids.valueAt(j);  if (proc.mUserId != tr.userId) {  // Don't kill process for a different user.  continue;  }  if (proc == mService.mHomeProcess) {  // Don't kill the home process along with tasks from the same package.  continue;  }  if (!proc.mPkgList.contains(pkg)) {  // Don't kill process that is not associated with this task.  continue;  }   if (!proc.shouldKillProcessForRemovedTask(tr)) {  // Don't kill process(es) that has an activity in a different task that is also  // in recents, or has an activity not stopped.  return;  }   if (proc.hasForegroundServices()) {  // Don't kill process(es) with foreground service.  return;  }   // Add process to kill list.  procsToKill.add(proc);  }  }   // Kill the running processes. Post on handle since we don't want to hold the service lock  // while calling into AM.  final Message m = PooledLambda.obtainMessage(  ActivityManagerInternal::killProcessesForRemovedTask, mService.mAmInternal,  procsToKill);  mService.mH.sendMessage(m); } |

**进程的类型**