入职实务设计说明

时间：2016年3月08日星期二

作者：方名

## 界面设计：

因为是一个控制台程序。程序启动的时候会在终端列出我们我需要实现的功能菜单

----------------------------------------------------------------------------------------------------------------------| 请选择需要执行的功能 |

| 1、安装应用 |

| 2、卸载应用 |

| 3、上传文件 |

| 4、下载文件 |

| 5、备份文件 |

| 6、恢复文件 |

| 7、 退出 |

----------------------------------------------------------------------------------------------------------------------

由用户选择需要执行的功能

## 功能设计

### 1.安装应用

安装应用对应的adb命令是 adb install <package> 。指定我们所需要安装的package的路径（.apk文件路径）

### 2.卸载应用

卸载应用对应的adb命令是 adb uninstall <package> 。指定我们所需要删除的package名字。也可以加入 –k 参数 ：adb shell uninstall-k <package>

可以通过adb shell pm list packages来查询机器中已安装的package名字。或者通过grep过滤我们需要查找的package

### 3.上传文件

上传文件对应的adb命令是 adb push <local> <remote>

Local 指的是本机文件的路径

Remote 指的是远程设备的路径

### 4.下载文件

下载文件对应的adb命令是 adb pull <remote> <local>

Local 指的是本机路径

Remote 指的是远程设备文件的路径

### 5.备份文件

备份文件对应的adb命令格式如下：

adb backup [-f <file>] [-apk|-noapk] [-shared|-noshared] [-all] [-system|nosystem] [<packages...>]  
对各个参数的解释：

-f <file>  
用这个来选择备份文件存储在哪里，例如-f /backup/mybackup.ab将会使文件存储在根磁盘（Windows的C盘等等）下一个名为backup的文件夹里，并且备份文件名为mybackup.ab

-apk|-noapk  
这个决定是否在备份里包含apk或者仅仅只备份应用数据，推荐使用-apk

-shared|-noshared  
这个参数用于决定是否备份设备共享的SD card内容，默认是-noshare，主要包括内部存储中的音乐、图片和视频，推荐使用-share

-all  
这个参数是一种简单地表达“所有应用”的说法，package参数可以选择备份单独的应用，如果你不是备份某个应用，使用-all备份整个系统

-system|-nosystem  
这个参数决定-all标签是否包含系统应用，默认的是-system，根据情况可选择是否用-nosystem

<packages...>  
如果你知道应用安装包的名称（例如com.google.android.apps.plus），就可以使用该参数备份特定应用

### 6.恢复文件

恢复文件对应的adb命令是：

adb restore <your backup file dir> 后面是你备份文件的路径

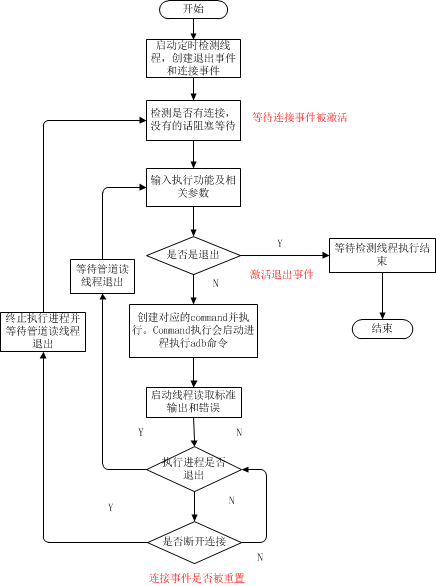
### 7.监测连接

在adb中我们可以通过adb devices命令来查看已连接的设备。若当前没有设备连接着在命令的输出中 “List of devices attached” 后面不再有设备列表

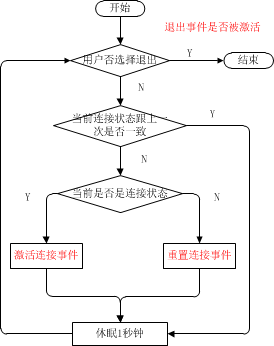
## 编码设计

### 流程图

#### 主线程执行逻辑

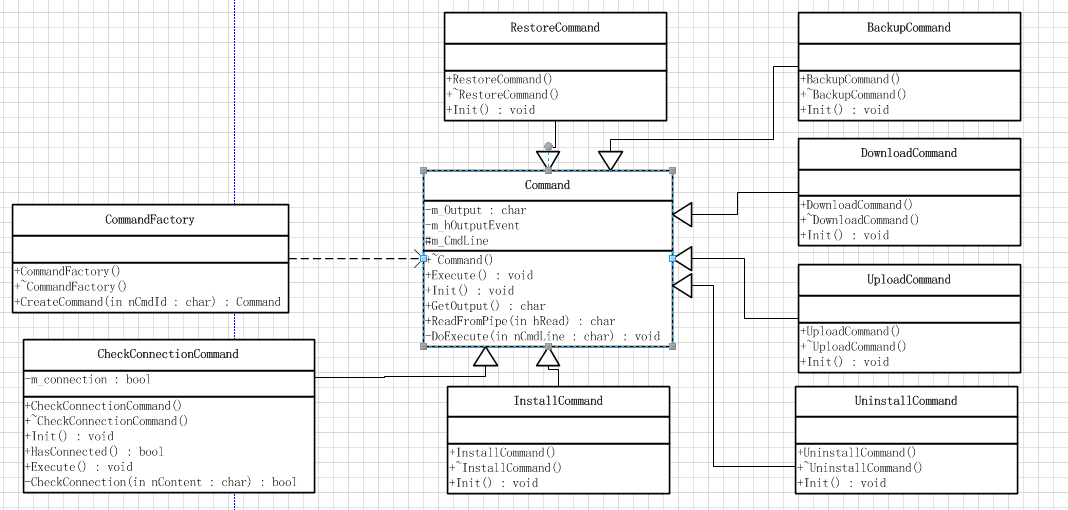


#### 检测线程执行逻辑



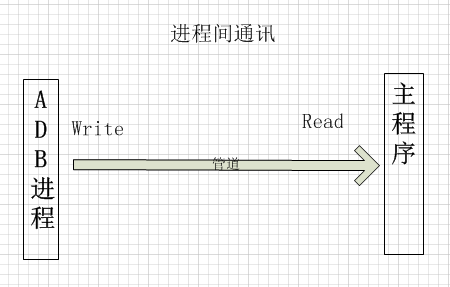
在程序启动的时候启动一个检测线程，检测线程是在定时检测当前是否有设备连接（或者设备连接被中断）。线程间的同步采用事件Event的形式。在启动开始我们会创建两个Event事件：ConnectEvent 和 ExitEvent 。当前检测线程检测到连接中断时则重置ConnectEvent，如果当前有正在执行adb命令的进程，则关闭该进程，主线程则等待新的连接。当检测到有连接时则激活该ConnectEvent，再进入我们的功能选择页面。当ExitEvent事件未被激活的时候，检测线程则会每隔一秒进行定时检测，当用户选择退出时，ExitEvent事件被激活，主线程退出主while循环，并等待检测线程退出。

### 2.类的设计图



我们把所需要执行的功能封装成一个Command。具体的功能对于一个具体的Command,继承自基类Command。通过一个CommandFactory工厂类来创建我们需要的Command。每个Command的Init的方法都是用来准备adb命令以及用户输入的相关参数，生成完整的adb命令行保存到内部的m\_CmdLine。Command的Execute首先调用Init方法，再调用基类的DoExecute来启动进程执行adb命令并通过管道机制来读取adb命令的输出。并将读取的进程的标准输出和标准错误结果保存在内部的m\_output变量中。GetOutput用来获取进程的输出信息

### 3.进程间通讯:



主进程与ADB进程间通过管道来进行通讯。创建匿名管道，将ADB进程的标准输出和标准错误重定向到管道的读端。也就相当于ADB进程往管道里面写数据。主进程从管道里面读取数据