Shell脚本编写建议

by zpo

目标

Shell脚本面向内部技术人员，编写目标如下：

- 脚本编写简单小巧，结构清晰，便于维护

  1. 脚本内容需要分为若干步骤，同时打印步骤序号和步骤名称
  2. 一个脚本做的事情保持在4-5步左右，脚本内容过多则进行相应拆分,避免某一个脚本过于复杂
  3. 文件部署尽量清晰，及时清理废弃脚本和废弃文件夹，被注释掉的无用代码及时删除
  4. 入口脚本应该提供**-help**功能

- 日志输出可以快速定位问题，打印格式规范统一

  1. 日志止于出错位置，并且能够明确指出错误位置
  2. 能够看到执行脚本的脚本名与执行时的具体参数
  3. 关键日志和非关键日志区分开来
  4. 不打印看起来没有意义或没有人会看的日志
  5. 调试脚本时打印的辅助日志，调试完成后请**务必**删掉

规约

命名风格统一

- 文件名、内部变量名使用 ***lowerCamelCase*** 小写驼峰形式，如：buildServerConsole.sh，resultNum
- 函数名使用 ***UpperCamelCase*** 大写驼峰形式，如：ReportFailure，AfterShell
- 常量名、路径名使用以下划线连接的大写单词形式，如: QUIET_RUN，SCRIPTS_AUTOBUILD_DIR

日志输出规范

编写了一套Shell的日志方法，目前能够实现：统一格式打印，分步执行，时间戳，出错后停止输出日志，出错后打印Shell堆栈，区分关键日志与非关键日志。编写Shell脚本时请按照以下规范调用日志方法。

1. 脚本开始时，调用 *BeforeShell* 函数，传入所有执行参数$@。该函数会打印执行脚本名称与所有参数：

   ```
   BeforeShell $@

   2018-09-05 [17:52:21]
   2018-09-05 [17:52:21]    +   testforlog.sh  BEGIN   +
   2018-09-05 [17:52:21]    +   testforlog.sh -n -r -android 90   +
   2018-09-05 [17:52:21]
   ```

2. 脚本执行步骤时，打印步骤序号和步骤名称。调用 *Step* 函数，传入两个参数，第一个为步骤序号，第二个为步骤名称。该函数会打印步骤标志：

```
Step 1 "Copy Unity Resource"

2018-09-05 [17:52:21]
2018-09-05 [17:52:21] ==================== testforlog.sh  STEP 1 : Copy Unity Resource ==
2018-09-05 [17:52:21]
```

3. 脚本结束时，调用 *AfterShell* 函数。该函数会打印脚本结束标志：

```
AfterShell

2018-09-05 [17:52:21]
2018-09-05 [17:52:21]    +   testforlog.sh  FINISH   +
2018-09-05 [17:52:21]
```

4. 需要打印关键步骤成功的标志时，调用 *ReportSuccess* 函数，可以传入关键步骤名称参数(默认为空)。该函数会打印成功结果标志：

```
ReportSuccess "Build Dynamic Framework Project"

****     Build Dynamic Framework Project SUCCESS      ****
```

5. 需要打印错误标志时，调用 *ReportFailure* 函数。该函数会打印调用者的行号、脚本名，并阻止之后的日志输出，并通过内建命令caller输出一个Shell的调用堆栈：

```
ReportFailure

****     line 10 testforlog.sh REPORTED FAILURE     ****


    CALLER LIST
 - line 10 a testforlog.sh
 - line 14 b testforlog.sh
 - line 17 main testforlog.sh
```

6. 非关键命令，如svn操作，cp命令执行前请调用 *BlankLine* 打印一个空行

7. 如果需要打印带TAG的Shell日志，可以在source logForShell.sh时传入TAG参数(默认为空)，则日志打印时均会带上该TAG输出：

```
source ./logForShell.sh [Shell]
Show "Using TAG to LOG"

2018-09-11 [10:08:26] [Shell] Using TAG to LOG
```

其他规范

1. 脚本设计编写时应考虑按功能模块拆分成独立脚本，但是执行时应该在一个Shell进程中执行，避免重复做大量初始化环境的工作

2. 不能打印只有变量没有名称的日志

> 如Show "$PLATFORM",
> 至少应该写成Show "PLATFORM : $PLATFORM"

3. 定义函数时在函数名前加上***function***:

```
function UpperCamelCase(){
        ...
}
```

4. if-else流程控制，then和条件判断请写在同一行:

```
if [ ... ];then
  ...
elif [ ... ]; then
  ...
else
  ...
fi
```

5. 区分$0与$BASH_SOURCE的区别，需要打印当前执行脚本名时使用$BASH_SOURCE

   TOBEDONE...