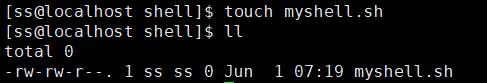
在初步了解了Shell的基本概念和特点之后，我们接下来就开始正式的学习Shell语言。

这里我们创建第一个Shell脚本文件，命名为myshell.sh。



打开文件，在文件中写入如下内容，然后保存并退出。

**[plain]** [view plain](https://blog.csdn.net/Cecilia3333/article/details/80543553) [copy](https://blog.csdn.net/Cecilia3333/article/details/80543553)

1. #！/bin/bash
2. echo "hello world"

这段代码的效果相当于我们在学习C语言时写的第一个hello  world的代码。

#！是一个约定的标记，它告诉系统这个脚本需要什么解释器来执行，即使用哪一种Shell。

通常，#!的名称叫做“Shebang”或者“She-bang”。

**在Shell脚本中用#表示注释，相当于C语言中的//。**

**echo命令用于向窗口输出文本。**

运行Shell脚本有两种方式：

**1、作为可执行程序**

https://img-blog.csdn.net/20180601223225113

chmod  +x  ./myshell.sh 使脚本具有执行权限；

./myshell.sh 执行脚本。

**2、作为解释器参数**

https://img-blog.csdn.net/20180601223645557

这种运行方式是直接运行解释器，其参数就是Shell脚本的文件名。这种方式运行的脚本，不需要在第一行指定解释器信息，写了也没有用。

为了简便书写，我们也可以写成如下这种方式：

https://img-blog.csdn.net/20180601224014678

还有一种更简便的书写方式：

https://img-blog.csdn.net/20180601224158128

在一般情况下，人们并不区分Bourne  Shell和Bourne  Again  Shell。所以，像#!/bin/sh，它同样可以改为#!/bin/bash。因此，我们执行bash  FileName与sh  FileName都可以得出结果。所以我们比较推荐最后这种写法，毕竟最简单。

* **执行方式的本质原理**

Shell会fork一个子进程并调用exec执行./myshell.sh这个程序，exec系统调用应该把子进程的代码段替换成./myshell.sh程序的代码段，并从它的\_start开始执行。然而myshell.sh是个文本文件，根本没有代码段和\_start函数，那怎么办呢？其实exec还有另外一种机制，如果要执行的是一个文本文件，并且第一行用Shebang指定了解释器，则用解释器程序的代码段替换当前进程，并且从解释器的\_start开始执行，而这个文本文件被当做命令行参数传给解释器。