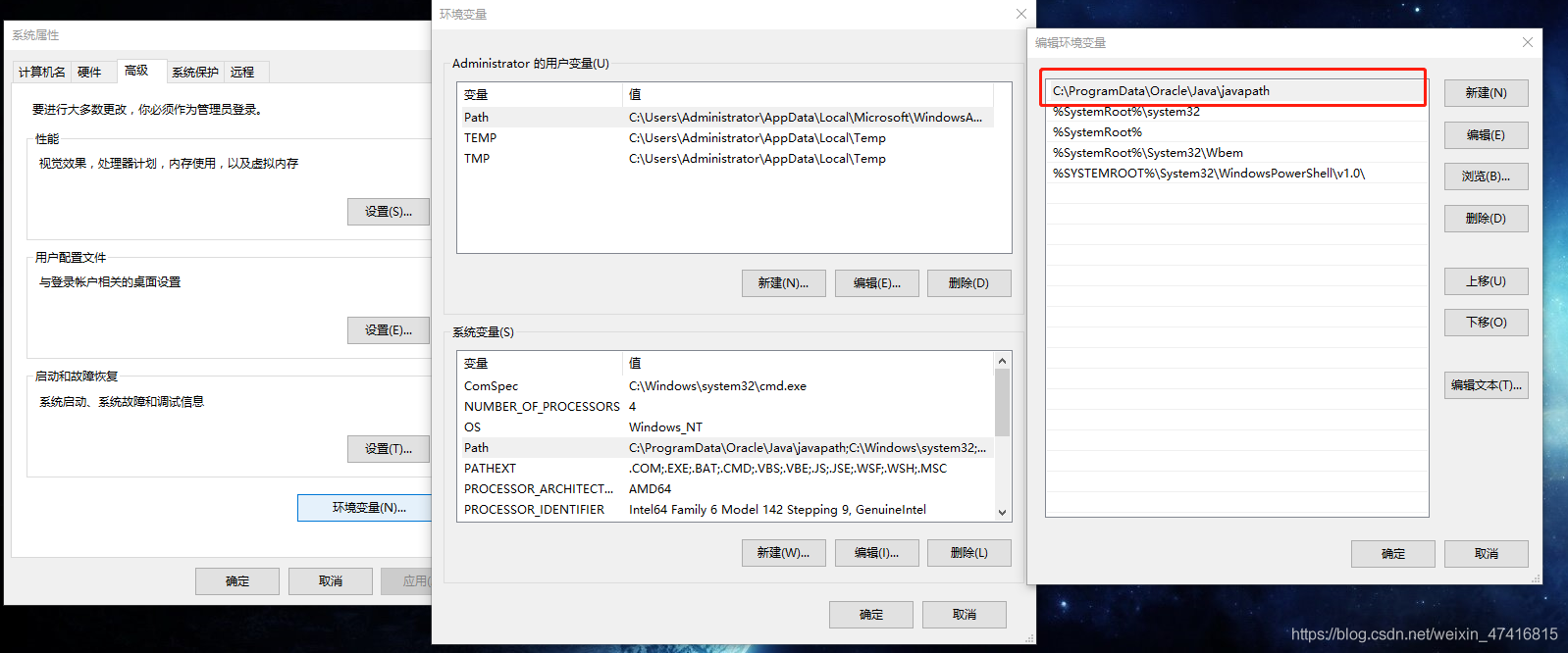
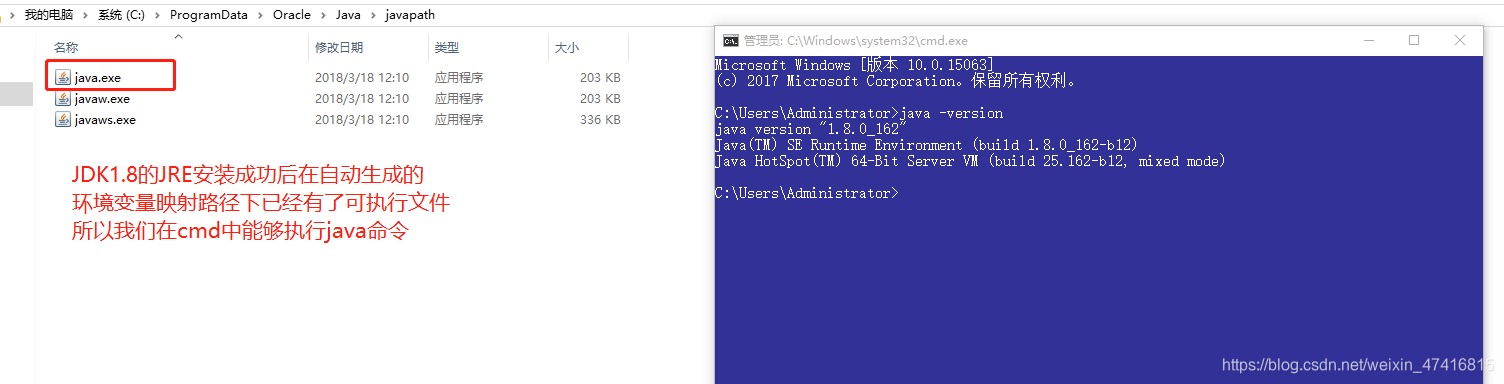
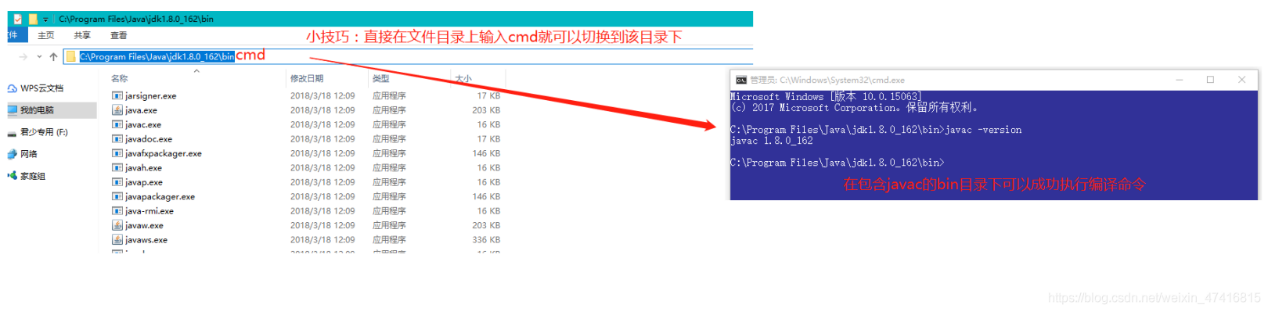
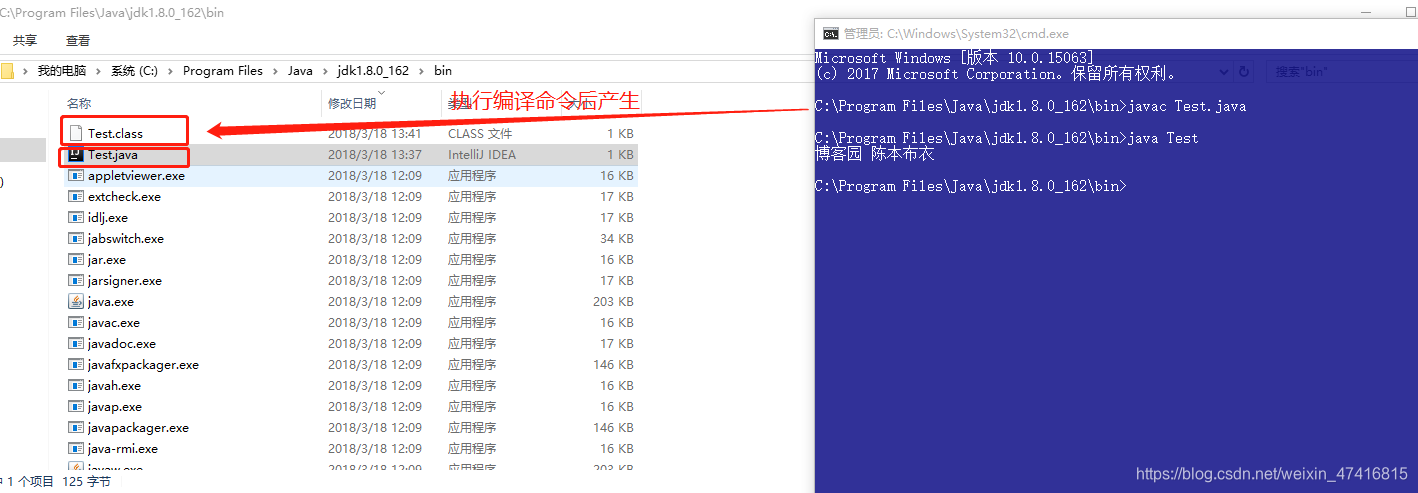
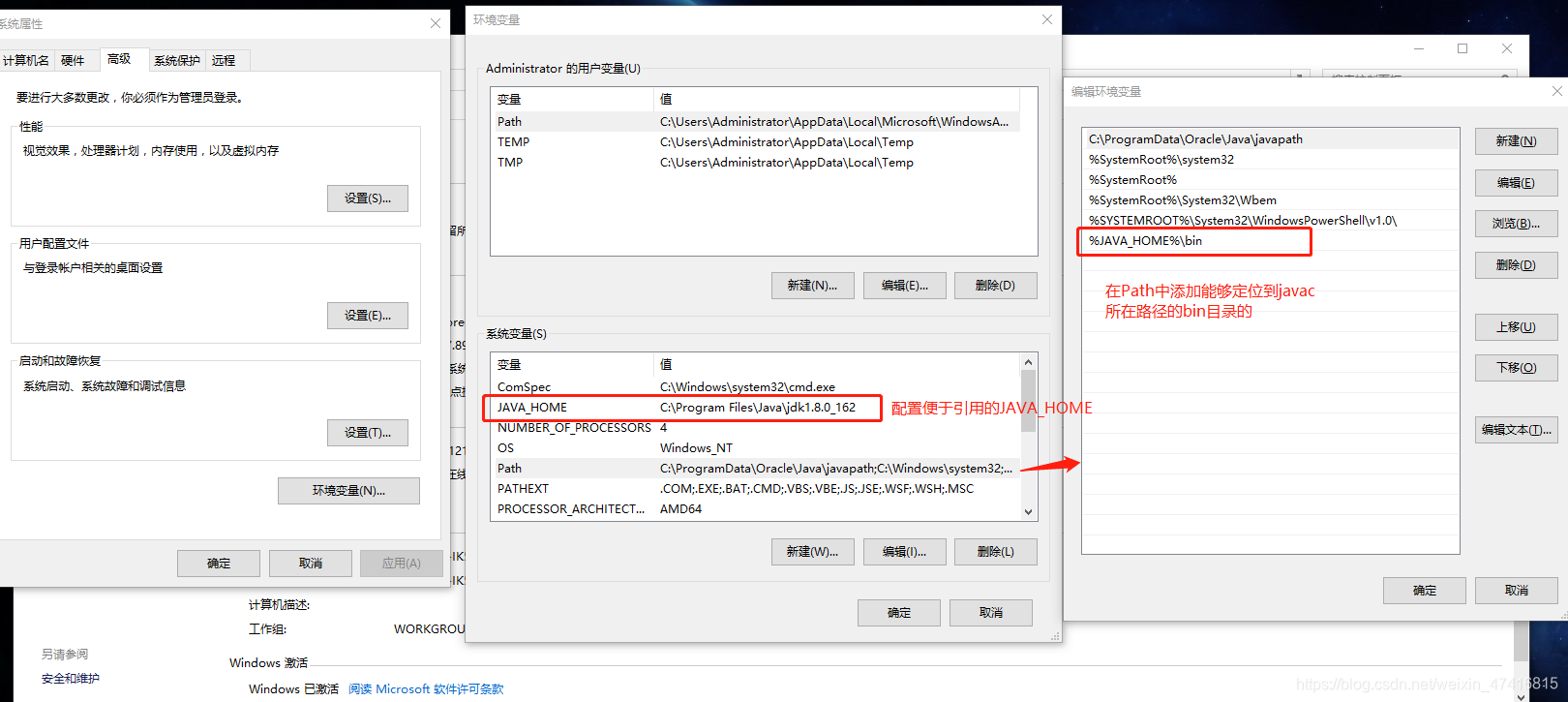
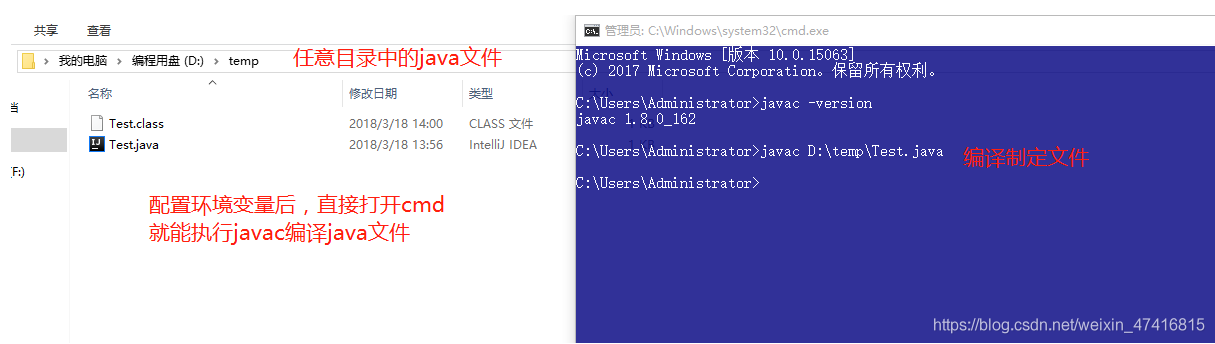
安装JDK也很简单，先是JDK，然后是JRE,如果要自定义路径，注意两者不要安装在同一个包下面。然后，作为Java开发者，就会按基本套路开始配置起环境变量。如果仅仅是作为一个代码搬运工，百度有一堆环境变量的配置方式，傻瓜也会配置，但要想成为一个高阶一些的码农，在进行一个操作之前，总该知道你干这件事的缘由——为什么要配置环境变量，如果不配置又会怎样？平时看起来只是开发入门的基础操作，深究起来其实并不简单。有人会呵呵一笑很倾城——不配环境变量，我们怎么开发？欸，这样的问题，我只能说，编程这玩意儿，很多时候不动脑子是可以的，但是不动手却不行。布衣博主以前的固有思想也是觉得，配置环境变量是Java开发的固定姿势，可是今天为了深究环境变量这个问题，下载JDK1.8来折腾一番后发现，仅仅是安装好JDK和JRE，根本用不着配置什么环境变量就可以开发和跑项目了，这是怎么回事？到环境变量配置页查看，会发现在path的值中有个C:\ProgramData\Oracle\Java\javapath路径值生成，如以下图中展示（注：图例一为Win10系统的环境变量配置界面，和Win7有很大不同）——  
  
  
而在映射的路径文件夹中已经存在了可执行的java.exe,这似乎是说明新版JDK（1.8）在安装的时候就已经自动为我们添加好了环境配置。当博主删除该path映射的路径值C:\ProgramData\Oracle\Java\javapath的时候，集成开发环境中的eclipse就无法正常打开了，但是IDEA依然能正常使用，添加之后eclipse恢复正常。接着将JDK1.8卸载，保留运行环境JRE，这个时候eclipse依然可以正常使用，因为Eclipse比较例外的是它自己内嵌了Java编译器，对javac没有任何依赖关系,所以eclipse其实只需要依赖JRE就可以做开发并编译运行Java文件的，有了JDK只是为了方便查询源码，而这个时候IDEA少了JDK的支持就无法正常开发使用了。接着卸载JRE,这个时候开发工具肯定是没法使用了，关键是在环境变量的Path值C:\ProgramData\Oracle\Java\javapath也消失了，对应路径下的文件也被删除了，这就说明在安装JRE的时候，是会为我们自动配置好环境变量的。环境变量都没配置就成功的布衣博主实在不甘心，卸载了JDK1.8之后又单独安装了JDK1.7继续尝试，安装完成后Path值并没有自动添加，博主也没有主动的去配置任何环境变量，但是打开Eclipse或者Idea等IDE，依然可以正常开发跑项目。尝试至此，似乎证明了长久以来大家配置环境变量只是思维和行为固化后的自以为是，纯粹是多此一举，对日常开发调试并没有卵用。难道，这么多程序员都在信奉了一个错误的认知而且丝毫不觉得有异？

要搞解开这个疑惑，还是得搞清楚环境变量到底是干什么的，为什么要配置它。所谓环境变量，其实就是操作系统中用来指定操作系统运行环境的一些参数，比如临时文件夹位置和系统文件夹位置等。当要求系统运行一个程序而没有告诉它程序所在的完整路径时，系统除了在当前目录下面寻找此程序外，还应到环境变量的Path中指定的路径值去找。也就是说当执行可执行文件找不到位置的时候，就会去Path中配置的路径去找。因此可以设想如果直接在可执行文件所在文件夹取执行，那不就是找得到吗？为了证明，博主先打开cmd执行编译命令javac，返回不是内部或外部命令，当我们切换到JDK的bin目录C:\Program Files\Java\jdk1.8.0\_162\bin路径去执行javac命令的时候，就能够成功执行，好像配置了环境变量一样——  
  
既然可以执行javac，那么我们就在该目录下测试编译一个Java文件。先用记事本编写测试代码 public class Test { public static void main(String[] args) { System.out.println(“博客园 陈本布衣”); } } ，文件名必须和类名保持一致Test.java,然后我们执行编译命令并运行文件——  
  
可是正常情况下我的Java文件是不可能放在bin目录下的，所以我们接着配置好环境变量，使我们能够在任意目录编译Java文件——  
  
  
配置环境变量的目的似乎达到了，仅仅是为了不用切换到JDK的bin目录就能执行编译命令的方便，可是，现在都是用集成开发环境如idea或eclipse等开发并编译Java程序，如果不是使用命令行javac的方式来编译Java程序