<https://blog.csdn.net/bravezhe/article/details/7396151>

工程师A修改了a.txt的第一行，提交了。  
工程师B也修改了a.txt的第一行，然后执行svn up，这时SVN提示了：（以下，你开始扮演工程师B的角色了）  
$ svn up  
在 “a.txt” 中发现冲突。  
选择: (p) 推迟，(df) 显示全部差异，(e) 编辑,  
(mc) 我的版本, (tc) 他人的版本,  
(s) 显示全部选项:  
  
我一般选择p（推迟），即引入冲突到本地，不过不会影响到SVN服务器端，可以放心。  
  
OK，开始解决冲突了。  
  
这时，会生成几个文件：  
  
a.txt  a.txt.mine  a.txt.r6328  a.txt.r6336  
  
其中a.txt中包含了工程师A、B的所有修改，以<<<<<<<、=======、>>>>>>>分隔。  
  
a.txt.mine是工程师B的修改，也就是未update前的a.txt。  
  
a.txt.r6328 是工程师A提交前的版本，即未导致冲突的版本。  
  
a.txt.r6336是工程师A提交后的版本，即导致冲突的版本。  
  
一般，查看a.txt就可以看到冲突的详情了：  
  
[yicheng@chengyi svntest]$ cat a.txt  
  
<<<<<<< .mine  
  
i also modify ,agndagnagasdg;  
  
=======  
  
i modify this line;  
  
>>>>>>> .r6336  
  
以上，<<<<<<< .mine和=======之间是工程师B（当前的“你”）修改的内容，=======与>>>>>>> .r6336之间是工程师A修改的内容。这时，最好的办法是，叫上工程师A，你们一起确定这些修改是否都需要，是否相互兼容，然后留下需要的部分，删 除<<<<<<< .mine、=======和>>>>>>> .r6336。  
  
然后，测试，测试！确定没问题之后，就可以告诉SVN，你解决冲突了：  
  
svn resolve –accept working a.txt (该命令会删除a.txt.mine  a.txt.r6328  a.txt.r6336）  
  
svn ci -m ’some comment’ a.txt  
  
这里需要注意的是，a.txt.mine  a.txt.r6328  a.txt.r6336这几个文件的存在代表着有冲突产生。如果不解决冲突，就手工删除它们，SVN服务器也会很傻的认为你解决了冲突，允许你继续之后 的工作。但是，冲突依旧存在，你的a.txt中不但有别人的修改，还有那些讨厌的<=>符号。  
  
在冲突未解决前，试图提交代码是肯定会失败的：  
  
$ svn ci -m ”  
  
svn: 提交失败(细节如下):  
  
svn: 提交终止: “/path/to/svntest/a.txt” 处于冲突状态

在使用svn update 的时候，会出现如下一些信息：  
$ svn update  
U  INSTALL  
G  README  
C  bar.c  
Updated to revision 46.  
  
那么，U 开头的信息提示你，这个文件在你本地没有修改过，文件已经根据版本库的新版本更新了。G 开头的信息提示你，这个文件在你本地已经修改过，但是和版本库中对应的版本并没有冲突的地方，svn已经合并更新了。而C 开头的信息提示你，这个文件有点麻烦，你在本地的修改和版本库中的版本修改的地方重叠了，也就是说，你修改了某一行，你的同事也修改了同一行。这个就需要你自己手工去解决了。当冲突发生时，要注意到有三件事情可以帮助你解决问题。