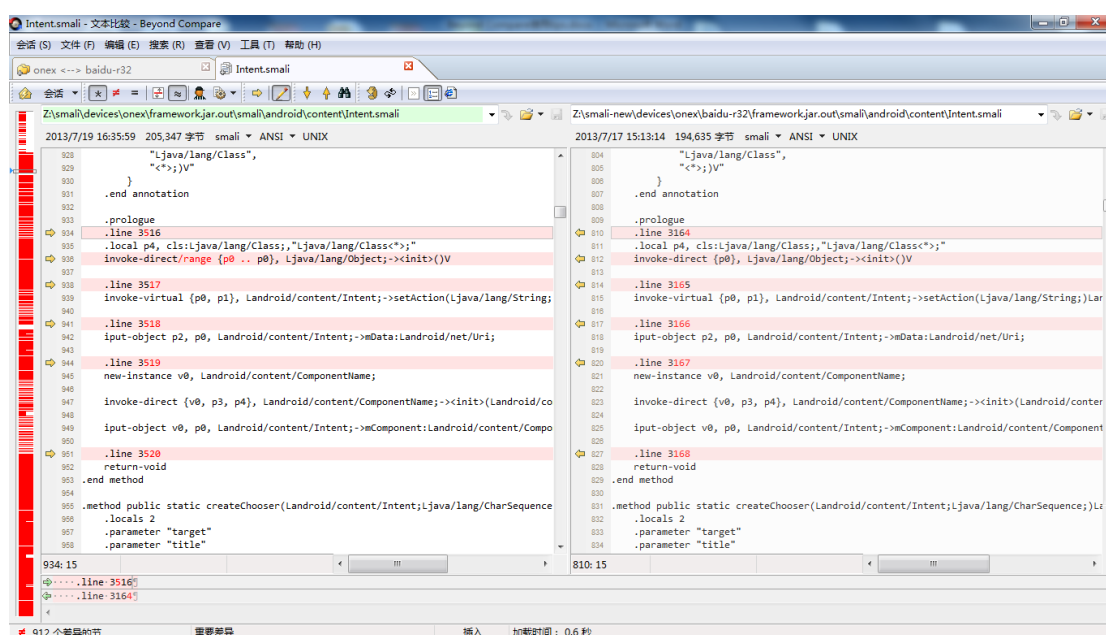



# Beyond Compare 使用技巧

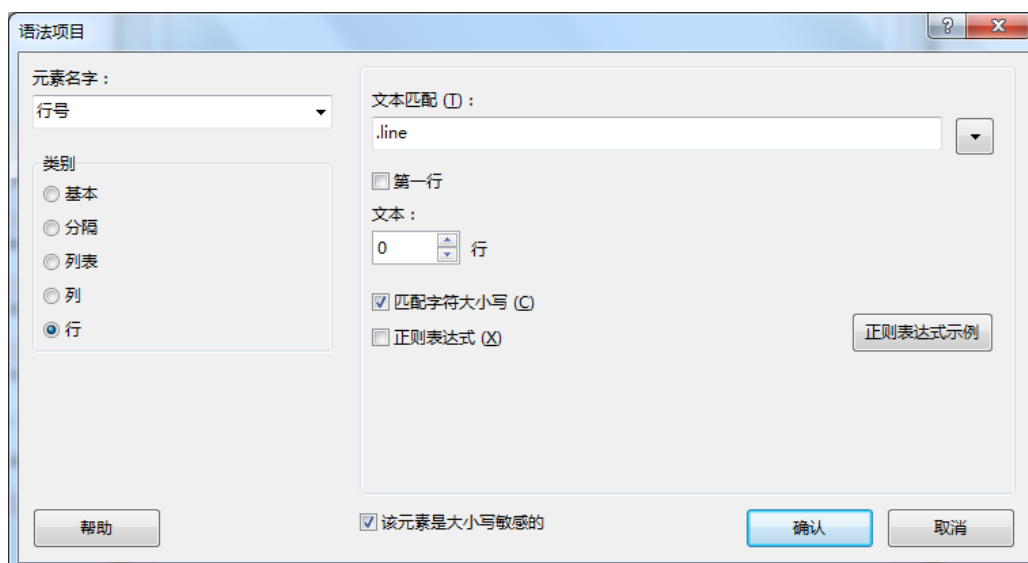
在移植 Baidu ROM 过程中,我们经常需要对比分析源包代码和工程代码两套 smali 代码,并把源包代码中的 feature 代码或 bug fix 代码合并到工程代码中,使用一个专业、高效的文件对比工具将会让我们 patch 事半功倍,这里推荐 Beyond Compare 这款优秀的文件对比软件。Beyond Compare 同时有 Windows 和 Linux 版本可使用,这里就不多介绍,具体可自行上网搜索,这里介绍利用这款软件在合并代码过程的一些使用技巧。

## 1、忽略行号差异

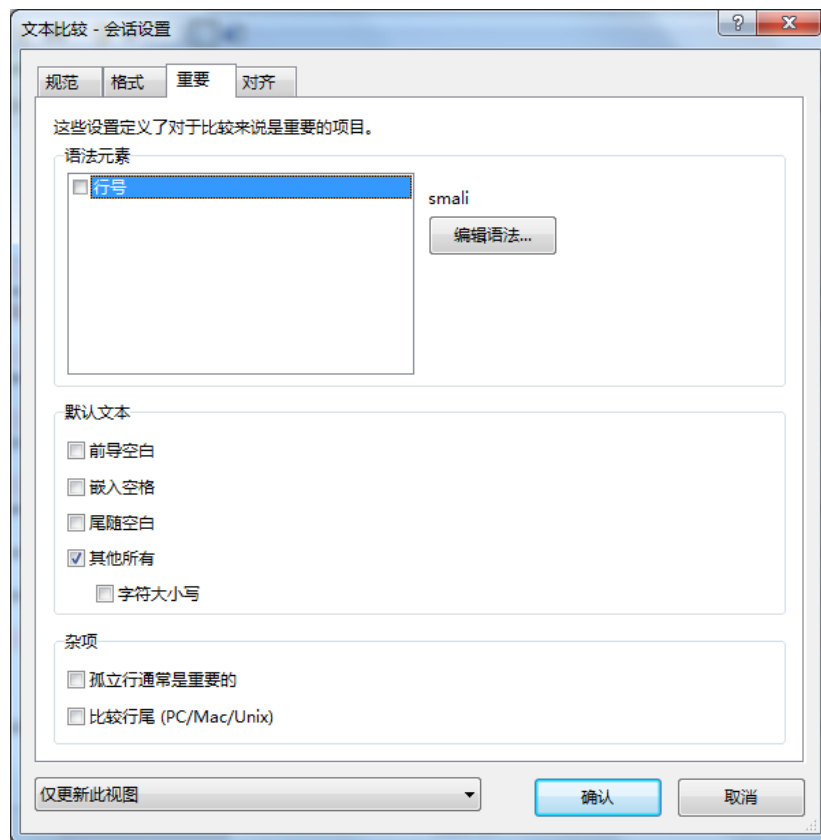
在对比 smali 代码过程,会发现很多文件的差异缩略图(下图左侧)几乎全部标红,文件差异似乎很大。其实很多只是行号的差异,我们并不关心,但这些差异会让所关心的差异难以定位,因此可以设置一个对比规则忽略掉行号的差异,设置方法如下:




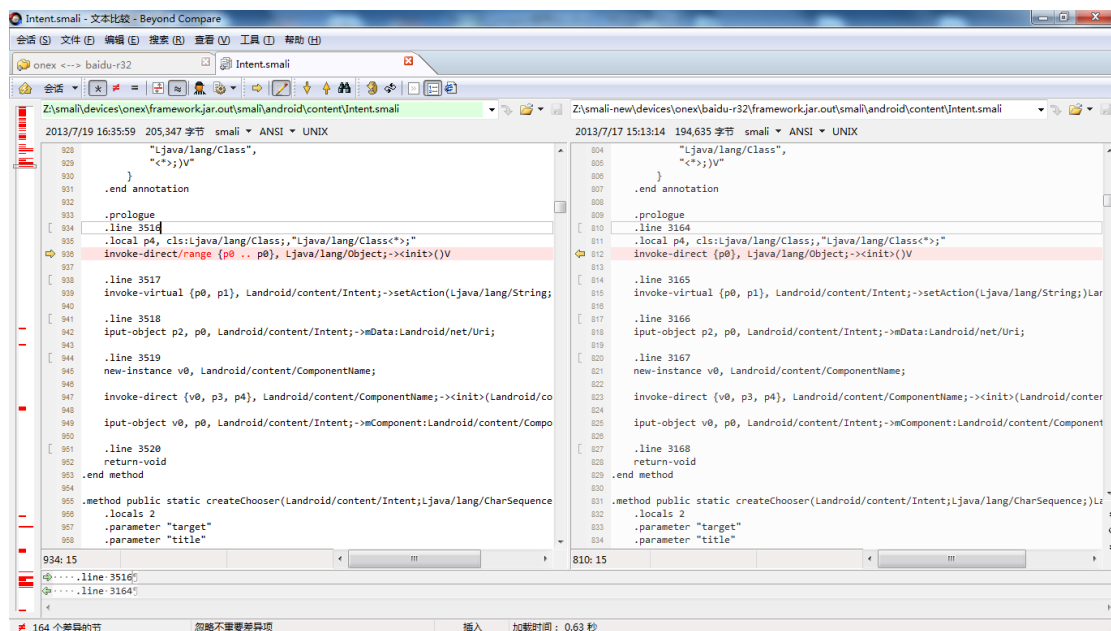
- 点击“规则”选项按钮,弹出对话框“重要”选项对话框,点击编辑语法,新建一个语法,按照下图所示配置语法项目。



- b. 确定后便在“重要”选项卡中生成一个“行号”勾选项，取消该选项的勾选，确定后返回。



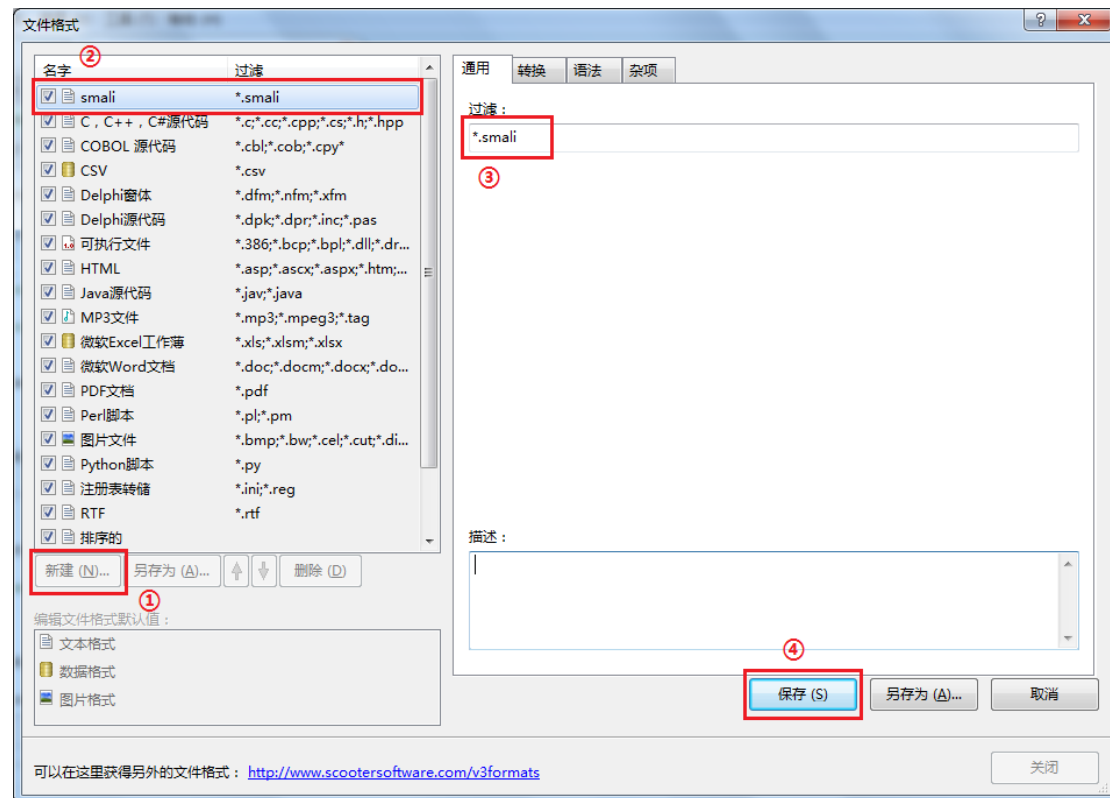
- c. 按下“忽略不重要差异项”选项按钮 ，就能忽略行号的差异，左侧缩略图便能显示出实重要代码差异的位置。



## 2、smali 语法高亮

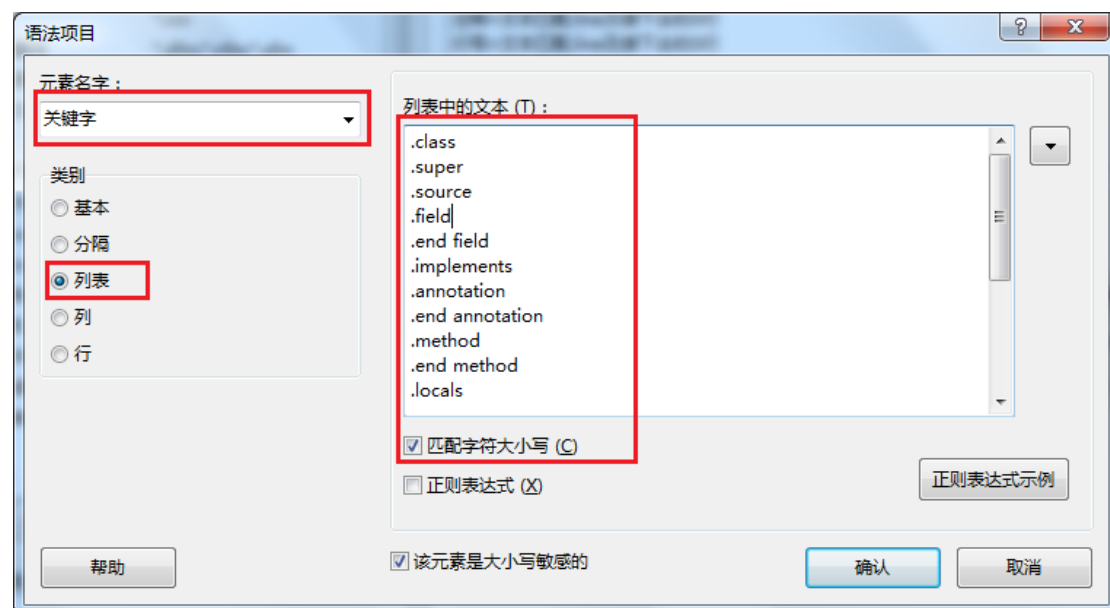
smali 代码虽然不是一种官方标准语言，但也遵循一定的语法规则，如有语法高亮辅助功能将能大大增强 smali 代码的可读性。Beyond Compare 允许自定义文件格式，对自定义的语法高亮显示，下面介绍针对 smali 代码进行语法高亮设置的方法：

- a. 在选项菜单“工具”中打开“文件格式”对话框，新建一个文本格式类型的选项并命名为“smali”，文件过滤条件为“\*.smali”，完成后先点击保存。如下图所示：



- b. 点击“语法”选项卡，这里会创建“关键字”、“字符串”和“注释”三种语法，点击“新建”弹出语法项目对话框，按照下图配置：

关键字列表添加了部分匹配字段，可根据需要自行删减。



关键字列表使用了：

```
.class
.super
.source
.field
.end field
.implements
.annotation
.end annotation
.method
.end method
.locals
.parameter
.prologue
public
private
protected
constructor
static
final
```

字符串语法匹配规则：

语法项目

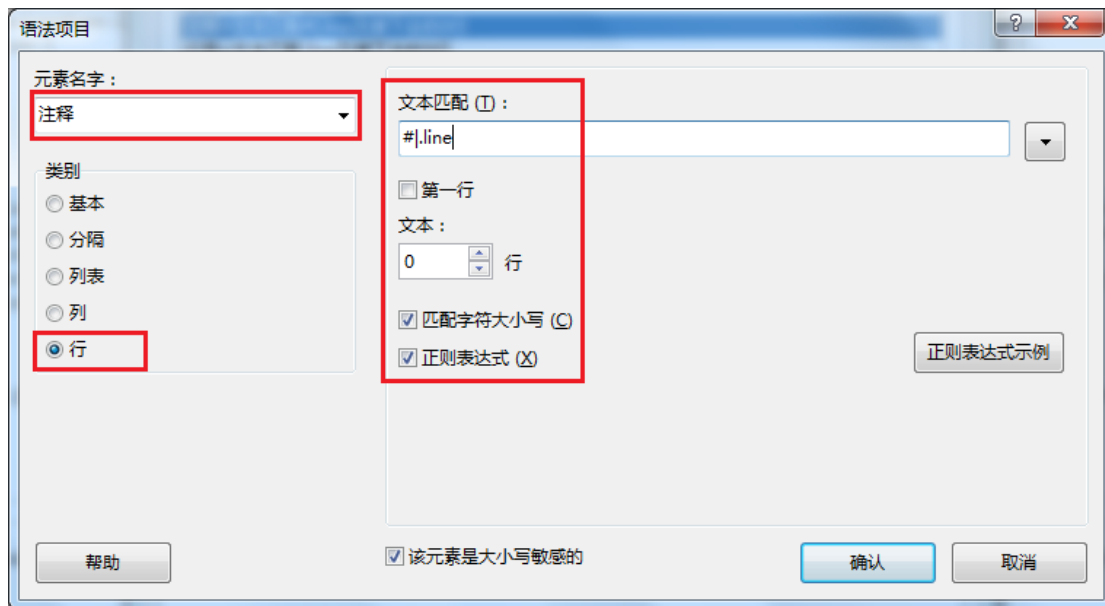
元素名字：  
字符串

类别：  
☐ 基本  
☒ 分隔  
☐ 列表  
☐ 列  
☐ 行

文本从 (T)：  
到：  
转义符：  
☒ 在行尾停止  
☒ 匹配字符大小写 (C)  
☐ 正则表达式 (X)

帮助 该元素是大小写敏感的 确认 取消

注释语法匹配使用正则表达式匹配了“#”和“.line”两个字段：



- c. 创建完成后便把“关键字”、“字符串”和“注释”三种语法添加进匹配规则列表中，保存后关闭文件格式对话框。



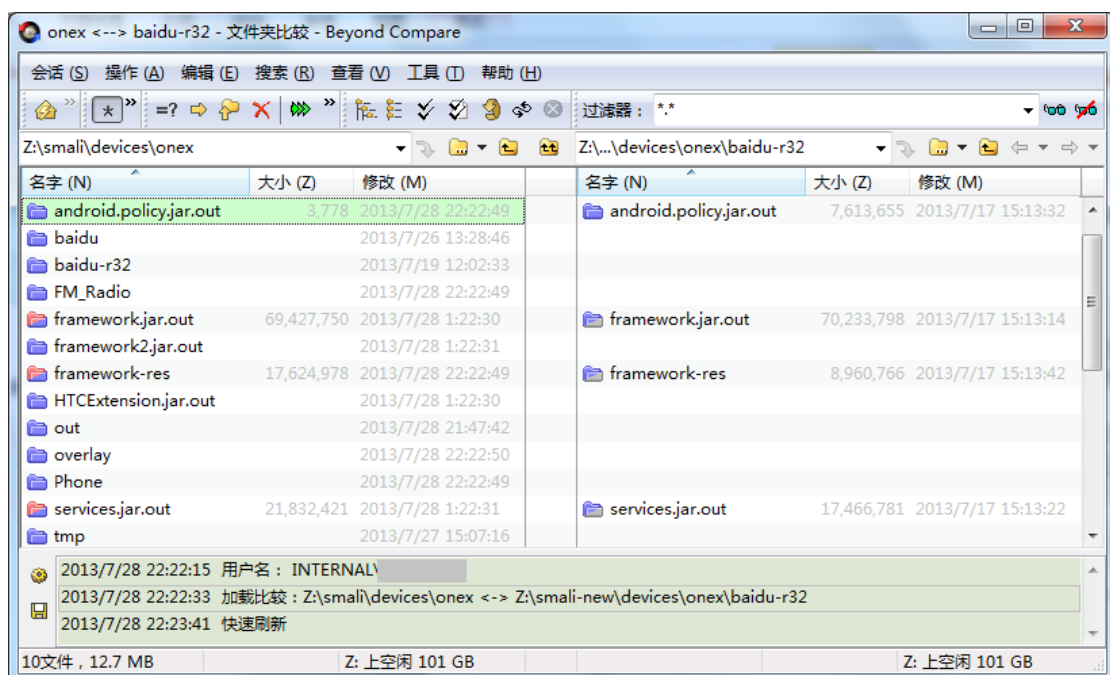
- d. 选中选项菜单“查看”中的“语法高亮”选项，Beyond Compare 即可对所有 smali 文件进行 smali 语法高亮显示。

```
984     return-object v0
985 .end method
986
987 .method public static getIntent(Ljava/lang/String;)Landroid/content/Intent;
988     .locals 1
989     .parameter "uri"
990     .annotation system Ldalvik/annotation/Throws;
991         value = {
992             Ljava/net/URISyntaxException;
993         }
994     .end annotation
995
996     .annotation runtime Ljava/lang/Deprecated;
997     .end annotation
998
999     .prologue
1000     .line 3609
1001     const/4 v0, 0x0
1002
1003     invoke-static {p0, v0}, Landroid/content/Intent;->parseUri(Ljava/lang/String;I)La
1004
1005     move-result-object v0
1006
1007     return-object v0
1008 .end method
1009
1010 .method public static getIntentOld(Ljava/lang/String;)Landroid/content/Intent;
1011     .locals 22
1012     .parameter "uri"
```

### 3、全工程文件夹比较

机型工程下 smali 代码目录结构繁杂、文件数量庞大，我们又经常需要同时对比分析源代码和工程代码多个文件，如何方便快捷地定位到需要对比的文件，这里介绍使用 Beyond Compare 全工程文件比较的方法。

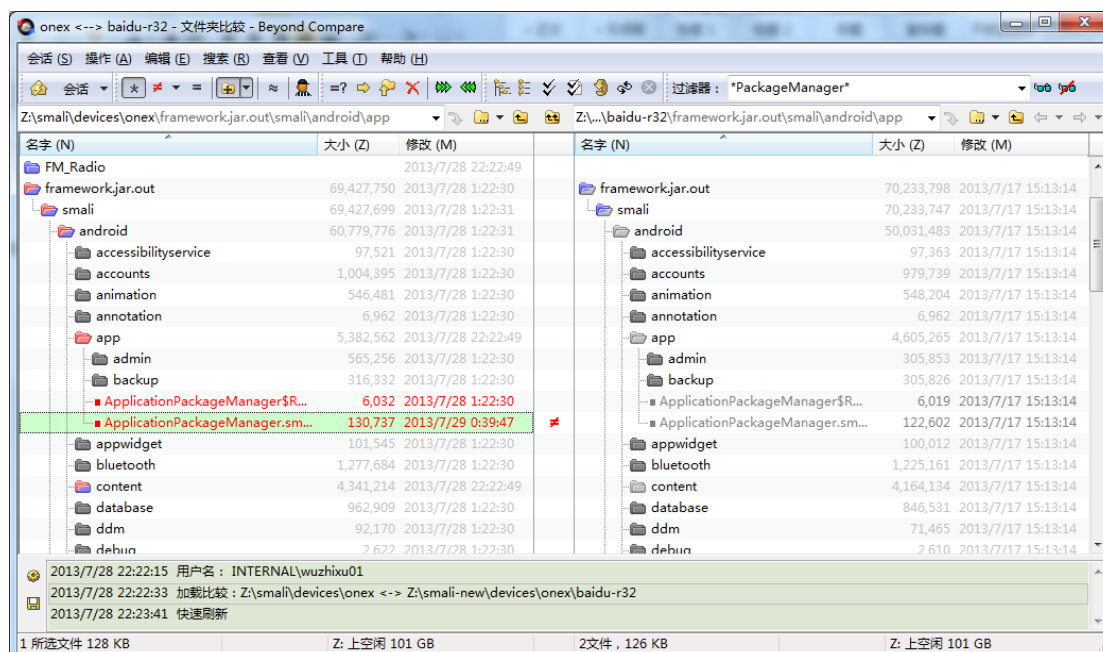
- a. 新建“文件夹比较”会话，分别添加源包代码文件夹和工程代码文件夹。



b. 在上方过滤器中输入需要查找的文件名关键词。



c. 所有文件名不包含输入关键字的文件会过滤掉而不显示，包含差异文件父目录颜色会标红，可以快速定位到查找的文件。



#### 4、把 Beyond Compare 做为 git 的可视化比较工具

准确说这应该是 git 的使用技巧,但涉及 Beyond Compare 这个工具,这里一并做下介绍。

a. 在 Linux 下,执行以下命令配置 git 的默认可视化比较工具。

```
git config --global diff.tool bc3
git config --global difftool.prompt false
```

b. 使用 Beyond Compare 比较两个 commit 的差异, git difftool 命令其他使用与 git diff 一致。

```
git difftool commit_old [commit_new] [file] &
```