diff

1.作用

diff 命令用于两个文档之间的比较，并指出两者的不同，他的使用权限是任何用户。

2.格式

diff [options] 源文档 目标文档

3.[options]主要参数

-a：将任何文档当作文本文档来处理。

-b：忽略空格造成的不同。

-B：忽略空行造成的不同。

-c：使用纲要输出格式。

-H：利用试探法加速对大文档的搜索。

-I：忽略大小写的变化。

-n --rcs：输出 RCS 格式。

- 指定要显示多少行的文本。此参数必须和-c 或-u 参数一并使用。

-a 或--text diff 预设只会逐行比较文本文件。

-b 或--ignore-space-change 不检查空格字符的不同。

-B 或--ignore-blank-lines 不检查空白行。

-c 显示全部内文，并标出不同之处。

-C 或--context 和执行"-c-"指令相同。

-d 或--minimal 使用不同的演算法，以较小的单位来做比较。

-D 或 ifdef 此参数的输出格式可用于前置处理器巨集。

-e 或--ed 此参数的输出格式可用于 ed 的 script 文件。

-f 或-forward-ed 输出的格式类似 ed 的 script 文件，但按照原来文件的顺序来显示不同处。

-H 或--speed-large-files 比较大文件时，可加快速度。

-l 或--ignore-matching-lines 若两个文件在某几行有所不同，而这几行同时都包含了选项中指定的字符或字符串，则不显示这两个文件的差异。

-i 或--ignore-case 不检查大小写的不同。

-l 或--paginate 将结果交由 pr 程式来分页。

-n 或--rcs 将比较结果以 RCS 的格式来显示。

-N 或--new-file 在比较目录时，若文件 A 仅出目前某个目录中，预设会显示：
Only in 目录：文件 A 若使用-N 参数，则 diff 会将文件 A 和一个空白的文件比较。

-p 若比较的文件为 C 语言的程式码文件时，显示差异所在的函数名称。

-P 或--unidirectional-new-file 和-N 类似，但只有当第二个目录包含了一个第一个目录所没有的文件时，才会将这个文件和空白的文件做比较。

-q 或--brief 仅显示有无差异，不显示周详的信息。

-r 或--recursive 比较子目录中的文件。

-s 或--report-identical-files 若没有发现所有差异，仍然显示信息。

-S 或--starting-file 在比较目录时，从指定的文件开始比较。

-t 或--expand-tabs 在输出时，将 tab 字符展开。

-T 或--initial-tab 在每行前面加上 tab 字符以便对齐。

-u,-U 或--unified= 以合并的方式来显示文件内容的不同。

-v 或--version 显示版本信息。

-w 或--ignore-all-space 忽略全部的空格字符。

-W 或--width　在使用-y 参数时，指定栏宽。

-x 或--exclude　不比较选项中所指定的文件或目录。

-X 或--exclude-from　你能将文件或目录类型存成文本文件，然后在=中指定此文本文件。

-y 或--side-by-side　以并列的方式显示文件的异同之处。

--help　显示帮助。

--left-column　在使用-y 参数时，若两个文件某一行内容相同，则仅在左侧的栏位显示该行内容。

--suppress-common-lines　在使用-y 参数时，仅显示不同之处。

-a

所有的文件都视为文本文件来逐行比较，甚至他们似乎不是文本文件.

-b

忽略空格引起的变化.

-B

忽略插入删除空行引起的变化.

--brief

仅报告文件是否相异,在乎差别的细节.

-c

使用上下文输出格式.

-C *行数（一个整数）*

--context[=*lines***]**

使用上下文输出格式，显示以指定 *行数* (一个整数), 或者是三行（当 *行数* 没有给出时. 对于正确的操作, 上下文至少要有两行.

--changed-group-format=*format*

使用 *format* 输出一组包含两个文件的不同处的行，其格式是 if-then-else .

-d

改变算法也许发现变化的一个更小的集合.这会使 *diff* 变慢 (有时更慢).

-D *name*

合并 if-then-else 格式输出, 预处理宏（由 name 参数提供）条件.

-e

--ed

输出为一个有效的 *ed* 脚本.

--exclude=*pattern*

比较目录的时候，忽略和目录中与 *pattern*（样式） 相配的.

--exclude-from=*file*

比较目录的时候，忽略和目录中与任何包含在 *file*（文件） 的样式相配的文件和目录.

--expand-tabs

在输出时扩展 tab 为空格，保护输入文件的 tab 对齐方式

-f

产生一个很象 *ed* 脚本的输出，但是但是在他们在文件出现的顺序有改变

-F *regexp*

在上下文和统一格式中，对于每一大块的不同，显示出匹配 *regexp*. 的一些前面的行.

--forward-ed

产生象 **ed** 脚本的输出，但是它们在文件出现的顺序有改变。

-h

这选项现在已没作用，它呈现 Unix 的兼容性.

-H

使用启发规则加速操作那些有许多离散的小差异的大文件.

--horizon-lines=*lines*

比较给定行数的有共同前缀的最后行，和有共同或缀的最前行.

-i

忽略大小写.

-I *regexp*

忽略由插入，删除行（由 regexp 参数提供参考）带来的改变.

--ifdef=*name*

合并 if-then-else 格式输出，预处理宏（由 name 参数提供）条件.

--ignore-all-space

在比较行的时候忽略空白.

--ignore-blank-lines

忽略插入和删除空行

--ignore-case

忽略大小写.

--ignore-matching-lines=*regexp*

忽略插入删除行（由 regexp 参数提供参考）.

--ignore-space-change

忽略空白的数量.

--initial-tab

在文本行（无论是常规的或者格式化的前后文关系）前输出 tab 代替空格. 引起的原因是 tab 对齐方式看上去象是常规的一样.

-l

产生通过 *pr* 编码的输出.

-L *label*

--label=*label*

使用 *label* 给出的字符在文件头代替文件名输出.

--left-column

以并列方式印出两公共行的左边

--line-format=*format*

使用 *format* 输出所有的行，在 if-then-else 格式中.

--minimal

改变算法也许发现变化的一个更小的集合.这会使 *diff* 变慢 (有时更慢).

-n

输出 RC-格式 diffs; 除了每条指令指定的行数受影响外 象 **-f** 一样。

-N

--new-file

在目录比较中，如果那个文件只在其中的一个目录中找到，那么它被视为在另一个目录中是一个空文件.

--new-group-format=*format*

使用 *format* 以 if-then-else 格式输出只在第二个文件中取出的一个行组

--new-line-format=*format*

使用 *format* 以 if-then-else 格式输出只在第二个文件中取出的一行

--old-group-format=*format*

使用 *format* 以 if-then-else 格式输出只在第一个文件中取出的一个行组

--old-line-format=*format*

使用 *format* 使用 *format* 以 if-then-else 格式输出只在第一个文件中取出的一行

-p

显示带有 c 函数的改变.

-P

在目录比较中，如果那个文件只在其中的一个目录中找到，那么它被视为在另一个目录中是一个空文件.

--paginate

产生通过 *pr* 编码的输出.

-q

仅报告文件是否相异,不报告详细的差异.

-r

当比较目录时，递归比较任何找到的子目录.

--rcs

输出 RC-格式 diffs; 除了每条指令指定的行数受影响外 象 **-f** 一样。

--recursive

当比较目录时，递归比较任何找到的子目录.

--report-identical-files

-s

报告两个文件相同.

-S *file*

当比较目录时，由 *file* 开始. 这用于继续中断了的比较.

--sdiff-merge-assist

打印附加的信息去帮助 *sdiff*. *sdiff* 在运行 *diff* 时使用这些选项. 这些选项不是特意为使用者直接使用而准备的。

--show-c-function

显示带有 c 函数的改变.

--show-function-line=*regexp*

在上下文和统一的格式，对于每一大块的差别，显示出匹配 *regexp*. 的一些前面的行

--side-by-side

使用并列的输出格式.

--speed-large-files

使用启发规则加速操作那些有许多离散的小差异的大文件.

--starting-file=*file*

当比较目录时，由 *file* 开始. 这用于继续中断了的比较.

--suppress-common-lines

在并列格式中不印出公共行。

-t

在输出时扩展 tab 为空格，保护输入文件的 tab 对齐方式

-T

在文本行（无论是常规的或者格式化的前后文关系）前输出 tab 代替空格.引起的原因是 tab 对齐方式看上去象是常规的一样.

--text

所有的文件都视为文本文件来逐行比较，甚至他们似乎不是文本文件.

-u

使用统一的输出格式.

--unchanged-group-format=*format*

使用 *format* 输出两个文件的公共行组，其格式是 if-then-else.

--unchanged-line-format=*format*

使用 *format* 输出两个文件的公共行，其格式是 if-then-else.

--unidirectional-new-file

在目录比较中，如果那个文件只在其中的一个目录中找到，那么它被视为在另一个目录中是一个空文件.

-U *lines*

--unified[=*lines***]**

使用前后关系格式输出，显示以指定 *行数* (一个整数), 或者是三行（当 *行数* 没有给出时. 对于正确的操作, 上下文至少要有两行.

-v

--version

输出 *diff* 版本号.

-w

在比较行时忽略空格

-W *columns*

--width=*columns*

在并列格式输出时，使用指定的列宽.

-x *pattern*

比较目录的时候，忽略和目录中与 *pattern*（样式） 相配的.

-X *file*

比较目录的时候，忽略和目录中与任何包含在 *file*（文件） 的样式相配的文件和目录.

-y

使用并列格式输出

diff 命令的使用

有这样两个文件：

程序清单 1 ： hello.c

```
#include
int main(void)
{
    char msg[] = "Hello world!";

    puts(msg);
    printf("Welcome to use diff commond.\n");

    return 0;
}
```

程序清单 2：hello_diff.c

```
#include
#include
int main(void)
{
    char msg[] = "Hello world,fome hello_diff.c";

    puts(msg);
    printf("hello_diff.c says,'Here you are,using diff.'\n");

    return 0;
}
```

我们使用 diff 命令来查看这两个文件的不同之处，有一下几种方便的方法：

1、普通格式输出：

```
[root@localhost diff]# diff hello.c hello_diff.c
1a2
> #include
5c6
       char msg[] = "Hello world,fome hello_diff.c";
8c9
       printf("hello_diff.c says,'Here you are,using diff.'\n");
[root@localhost diff]#
```

上面的“1a2”表示后面的一个文件"hello_diff.c"比前面的一个文件"hello.c"多了一行

"5c6"表示第一个文件的第 5 行与第二个文件的第 6 行有区别

2、并排格式输出

```
[root@localhost diff]# diff hello.c hello_diff.c -y -W 130
#include                                                #include
                                                            > #include
int main(void)                                                 int main(void)
{                                                              {
       char msg[] = "Hello world!";                                    |          char msg[] = "Hello
world,fome hello_diff.c";
       puts(msg);                                                        puts(msg);
       printf("Welcome to use diff commond.\n");                         |          printf("hello_diff.c
```

```
says,'Here you are,using diff.'\
        return 0;                                                    return 0;
}                                                                }
[root@localhost diff]#
```

这种并排格式的对比一目了然，可以快速找到不同的地方。

-W 选择可以指定输出列的宽度，这里指定输出列宽为 130

3、上下文输出格式

```
[root@localhost diff]# diff hello.c hello_diff.c -c
*** hello.c     2007-09-25 17:54:51.000000000 +0800
--- hello_diff.c        2007-09-25 17:56:00.000000000 +0800
***************
*** 1,11 ****
  #include

  int main(void)
  {
!       char msg[] = "Hello world!";

        puts(msg);
!       printf("Welcome to use diff commond.\n");

        return 0;
  }
--- 1,12 ----
  #include
+ #include

  int main(void)
  {
!       char msg[] = "Hello world,fome hello_diff.c";

        puts(msg);
!       printf("hello_diff.c says,'Here you are,using diff.'\n");

        return 0;
  }
[root@localhost diff]#
```

这种方式在开头两行作了比较文件的说明，这里有三中特殊字符：

＋　　比较的文件的后者比前着多一行

－　　比较的文件的后者比前着少一行

！　　比较的文件两者有差别的行

4、统一输出格式

```
[root@localhost diff]# diff hello.c hello_diff.c -u
--- hello.c     2007-09-25 17:54:51.000000000 +0800
```

```
+++ hello_diff.c        2007-09-25 17:56:00.000000000 +0800
@@ -1,11 +1,12 @@
#include
+#include

int main(void)
{
-       char msg[] = "Hello world!";
+        char msg[] = "Hello world,fome hello_diff.c";

       puts(msg);
-       printf("Welcome to use diff commond.\n");
+        printf("hello_diff.c says,'Here you are,using diff.'\n");

       return 0;
}
[root@localhost diff]#
```

正如看到的那样，统一格式的输出更加紧凑，所以更易于理解，更易于修改。

5、其他

假如你想查看两个文件是否不同又不想显示差异之处的话，可以加上－q 选项：

```
[root@localhost diff]# diff hello.c hello_diff.c -q
Files hello.c and hello_diff.c differ
[root@localhost diff]#
```

另外你还可以提供一些匹配规则来忽略某中差别，可以用 -I regexp

```
[root@localhost diff]# diff hello.c hello_diff.c -c -I include
*** hello.c     2007-09-25 17:54:51.000000000 +0800
--- hello_diff.c        2007-09-25 17:56:00.000000000 +0800
***************
*** 2,11 ****

  int main(void)
  {
!       char msg[] = "Hello world!";

       puts(msg);
!       printf("Welcome to use diff commond.\n");

       return 0;
  }
--- 3,12 ----

  int main(void)
  {
!       char msg[] = "Hello world,fome hello_diff.c";
```

```
        puts(msg);
!       printf("hello_diff.c says,'Here you are,using diff.'\n");

        return 0;
  }
[root@localhost diff]#
```

这里通过“ -I include”选项来忽略带有“ include”字样的行