

Advanced Linux Usage

2018-09-11

Martin Dahlö martin.dahlo@scilifelab.uu.se

Enabler for Life Science











Same program, many files

```
$ ls -l
total 0
-rw-rw-r-- 1 dahlo dahlo 0 Sep   1 16:42 sample_1.bam
-rw-rw-r-- 1 dahlo dahlo 0 Sep   1 16:42 sample_2.bam
-rw-rw-r-- 1 dahlo dahlo 0 Sep   1 16:42 sample_3.bam
-rw-rw-r-- 1 dahlo dahlo 0 Sep   1 16:42 sample_4.bam
-rw-rw-r-- 1 dahlo dahlo 0 Sep   1 16:42 sample_5.bam
-rw-rw-r-- 1 dahlo dahlo 0 Sep   1 16:42 sample_6.bam
-rw-rw-r-- 1 dahlo dahlo 0 Sep   1 16:42 sample_7.bam
-rw-rw-r-- 1 dahlo dahlo 0 Sep   1 16:42 sample_8.bam
-rw-rw-r-- 1 dahlo dahlo 0 Sep   1 16:42 sample_9.bam
$ my_prog sample_1.bam
```



Same program, many files

```
$ ls -l
total 0
-rw-rw-r-- 1 dahlo dahlo 0 Sep    1 16:42 sample_1.bam
-rw-rw-r-- 1 dahlo dahlo 0 Sep    1 16:42 sample_2.bam
-rw-rw-r-- 1 dahlo dahlo 0 Sep    1 16:42 sample_3.bam
-rw-rw-r-- 1 dahlo dahlo 0 Sep    1 16:42 sample_4.bam
-rw-rw-r-- 1 dahlo dahlo 0 Sep    1 16:42 sample_5.bam
-rw-rw-r-- 1 dahlo dahlo 0 Sep    1 16:42 sample_6.bam
-rw-rw-r-- 1 dahlo dahlo 0 Sep    1 16:42 sample_7.bam
-rw-rw-r-- 1 dahlo dahlo 0 Sep    1 16:42 sample_8.bam
-rw-rw-r-- 1 dahlo dahlo 0 Sep    1 16:42 sample_9.bam
$ my_prog sample_1.bam
$ my_prog sample_2.bam
```



Same program, many files

```
s ls -l
total 0
-rw-rw-r-- 1 dahlo dahlo 0 Sep 1 16:42 sample 1.bam
-rw-rw-r-- 1 dahlo dahlo 0 Sep 1 16:42 sample 2.bam
-rw-rw-r-- 1 dahlo dahlo 0 Sep 1 16:42 sample 3.bam
-rw-rw-r-- 1 dahlo dahlo 0 Sep 1 16:42 sample 4.bam
-rw-rw-r-- 1 dahlo dahlo 0 Sep 1 16:42 sample 5.bam
-rw-rw-r-- 1 dahlo dahlo 0 Sep 1 16:42 sample 6.bam
-rw-rw-r-- 1 dahlo dahlo 0 Sep 1 16:42 sample 7.bam
-rw-rw-r-- 1 dahlo dahlo 0 Sep 1 16:42 sample 8.bam
-rw-rw-r-- 1 dahlo dahlo 0 Sep 1 16:42 sample 9.bam
$ my proq sample 1.bam
$ my prog sample 2.bam
$ my prog sample 3.bam
$ my prog sample 4.bam
$ my prog sample 5.bam
$ my prog sample 6.bam
$ my prog sample 7.bam
$ my prog sample 8.bam
$ my prog sample 9.bam
```



- Same program, many files
 - 10 files? Ok
 - 1000 files? Not ok



- Same program, many files
 - 10 files? Ok
 - 1000 files? Not ok
- Reproducibility
 - Self and others



- Same program, many files
 - 10 files? Ok
 - 1000 files? Not ok
- Reproducibility
 - Self and others

A solution - write a script!



```
total 0
-rw-rw-r-- 1 dahlo dahlo 0 Sep 1 17:18 sample_1.bam
-rw-rw-r-- 1 dahlo dahlo 0 Sep 1 17:18 sample_2.bam
-rw-rw-r-- 1 dahlo dahlo 0 Sep 1 17:18 sample_3.bam
-rw-rw-r-- 1 dahlo dahlo 0 Sep 1 17:18 sample_4.bam
-rw-rw-r-- 1 dahlo dahlo 0 Sep 1 17:18 sample_5.bam
-rw-rw-r-- 1 dahlo dahlo 0 Sep 1 17:18 sample_6.bam
-rw-rw-r-- 1 dahlo dahlo 0 Sep 1 17:18 sample_7.bam
-rw-rw-r-- 1 dahlo dahlo 0 Sep 1 17:18 sample_8.bam
-rw-rw-r-- 1 dahlo dahlo 0 Sep 1 17:18 sample_9.bam
s nano analysis.sh
```



GNU nano 2.0.9 File: analysis.sh Modified

^G Get Help ^X Exit













GNU nano 2.0.9

File: analysis.sh

Modified

my_prog sample_1.bam











GNU nano 2.0.9

File: analysis.sh

Modified

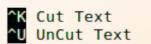
my_prog sample_1.bam
my_prog sample_2.bam

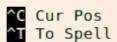














GNU nano 2.0.9 File: analysis.sh Modified

my_prog sample_1.bam my_prog sample_2.bam my_prog sample_3.bam my_prog sample_4.bam my_prog sample_5.bam my_prog sample_6.bam my_prog sample_7.bam my_prog sample_8.bam my_prog sample_9.bam















```
s l
total 4,0K
-rw-rw-r-- 1 dahlo dahlo 267 Sep 7 09:34 analysis.sh
-rw-rw-r-- 1 dahlo dahlo
                           0 Sep 1 17:18 sample 1.bam
-rw-rw-r-- 1 dahlo dahlo
                           0 Sep 1 17:18 sample 2.bam
-rw-rw-r-- 1 dahlo dahlo
                           0 Sep 1 17:18 sample 3.bam
-rw-rw-r-- 1 dahlo dahlo
                           0 Sep 1 17:18 sample 4.bam
-rw-rw-r-- 1 dahlo dahlo
                           0 Sep 1 17:18 sample 5.bam
-rw-rw-r-- 1 dahlo dahlo
                           0 Sep 1 17:18 sample 6.bam
-rw-rw-r-- 1 dahlo dahlo
                           0 Sep 1 17:18 sample 7.bam
-rw-rw-r-- 1 dahlo dahlo
                           0 Sep 1 17:18 sample 8.bam
-rw-rw-r-- 1 dahlo dahlo
                           0 Sep 1 17:18 sample 9.bam
$
```



```
s l
total 4,0K
-rw-rw-r-- 1 dahlo dahlo 267 Sep 7 09:34 analysis.sh
-rw-rw-r-- 1 dahlo dahlo
                           0 Sep 1 17:18 sample 1.bam
-rw-rw-r-- 1 dahlo dahlo
                           0 Sep 1 17:18 sample 2.bam
-rw-rw-r-- 1 dahlo dahlo
                           0 Sep 1 17:18 sample 3.bam
-rw-rw-r-- 1 dahlo dahlo
                           0 Sep 1 17:18 sample 4.bam
-rw-rw-r-- 1 dahlo dahlo
                           0 Sep 1 17:18 sample 5.bam
-rw-rw-r-- 1 dahlo dahlo
                           0 Sep 1 17:18 sample 6.bam
-rw-rw-r-- 1 dahlo dahlo
                           0 Sep 1 17:18 sample 7.bam
-rw-rw-r-- 1 dahlo dahlo
                           0 Sep 1 17:18 sample 8.bam
-rw-rw-r-- 1 dahlo dahlo
                           0 Sep 1 17:18 sample 9.bam
$ bash analysis.sh
```



```
s l
total 4,0K
-rw-rw-r-- 1 dahlo dahlo 267 Sep 7 09:34 analysis.sh
-rw-rw-r-- 1 dahlo dahlo
                           0 Sep 1 17:18 sample 1.bam
-rw-rw-r-- 1 dahlo dahlo
                           0 Sep 1 17:18 sample 2.bam
-rw-rw-r-- 1 dahlo dahlo
                           0 Sep 1 17:18 sample 3.bam
-rw-rw-r-- 1 dahlo dahlo
                           0 Sep 1 17:18 sample 4.bam
-rw-rw-r-- 1 dahlo dahlo
                           0 Sep 1 17:18 sample 5.bam
-rw-rw-r-- 1 dahlo dahlo
                           0 Sep 1 17:18 sample 6.bam
-rw-rw-r-- 1 dahlo dahlo
                           0 Sep 1 17:18 sample 7.bam
-rw-rw-r-- 1 dahlo dahlo
                           0 Sep 1 17:18 sample 8.bam
-rw-rw-r-- 1 dahlo dahlo
                           0 Sep 1 17:18 sample 9.bam
$ bash analysis.sh
```

Still not OK for 1000 or more files!



Assigning

```
my_variable=5
my_variable="nice text"
```



Assigning

```
my_variable=5
my_variable="nice text"
```

Using

```
$my_variable
```



Assigning

```
my_variable=5
my_variable="nice text"
```

Using

\$my variable

```
$ my variable="Rackham"
```



Assigning

```
my_variable=5
my_variable="nice text"
```

Using

\$my variable

```
$ my_variable="Rackham"
$ echo "Hello, $my variable! "
```



Assigning

```
my_variable=5
my_variable="nice text"
```

Using

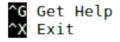
\$my variable

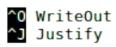
```
$ my_variable="Rackham"
$ echo "Hello, $my_variable! "
Hello, Rackham!
```

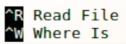


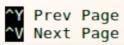
GNU nano 2.0.9 File: analysis.sh Modified

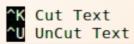
my_prog sample_1.bam my_prog sample_2.bam my_prog sample_3.bam my_prog sample_4.bam my_prog sample_5.bam my_prog sample_6.bam my_prog sample_7.bam my_prog sample_8.bam my_prog sample_9.bam

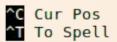














GNU nano 2.0.9 File: analysis.sh Modifie

prefix="sample"

my_prog sample_1.bam
my_prog sample_2.bam

my_prog sample_3.bam

my_prog sample_4.bam

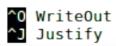
my_prog sample_5.bam

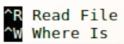
my_prog sample_6.bam
my prog sample 7.bam

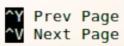
my_prog_sample_7.ban

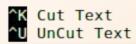
my_prog sample_8.bam
my_prog sample 9.bam

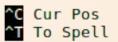
^G Get Help









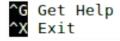


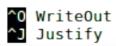


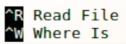
GNU nano 2.0.9 File: analysis.sh Modified

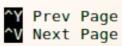
prefix="sample"

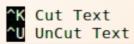
my_prog \${prefix}_1.bam
my_prog \${prefix}_2.bam
my_prog \${prefix}_3.bam
my_prog \${prefix}_4.bam
my_prog \${prefix}_5.bam
my_prog \${prefix}_6.bam
my_prog \${prefix}_7.bam
my_prog \${prefix}_8.bam
my_prog \${prefix}_8.bam
my_prog \${prefix}_9.bam

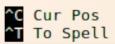










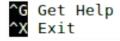


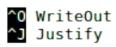


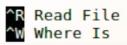
GNU nano 2.0.9 File: analysis.sh Modified

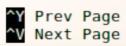
prefix="dog"

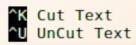
my_prog \${prefix}_1.bam
my_prog \${prefix}_2.bam
my_prog \${prefix}_3.bam
my_prog \${prefix}_4.bam
my_prog \${prefix}_5.bam
my_prog \${prefix}_6.bam
my_prog \${prefix}_7.bam
my_prog \${prefix}_8.bam
my_prog \${prefix}_8.bam
my_prog \${prefix}_9.bam

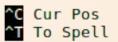














```
for var in 1 2 3;
do
    echo $var
done
```

```
$ bash loop_test.sh
1
2
3
```



```
for var in text works too;
do
    echo $var
done
```

```
$ bash loop_test.sh
text
works
too
$
```



```
for var in mix them 5;
do
    echo $var
done
```

```
$ bash loop_test.sh
mix
them
5
```

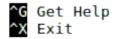


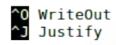
GNU nano 2.0.9

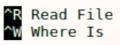
File: analysis.sh

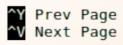
prefix="sample"

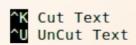
for i in 1 2 3 4 5 6 7 8 9;
 do
 my_prog \${prefix}_\$i.bam
 done

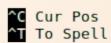














GNU nano 2.0.9

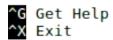
File: analysis.sh

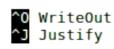
```
prefix="sample"

for i in 1 2 3 4 5 6 7 8 9;

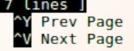
do
    echo my_prog ${prefix}_$i.bam
done
```

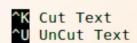
Debugging!

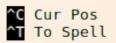














GNU nano 2.0.9

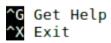
File

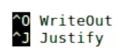
```
prefix="sample"

for i in 1 2 3 4 5 6 7 8 9;
  do
      echo my_prog ${prefix}_$i.bam
  done
```

```
Loops
```

```
$ bash analysis.sh
my_prog sample_1.bam
my_prog sample_2.bam
my_prog sample_3.bam
my_prog sample_4.bam
my_prog sample_5.bam
my_prog sample_6.bam
my_prog sample_7.bam
my_prog sample_8.bam
my_prog sample_9.bam$
```









```
$ ls *.bam
sample_1.bam sample_3.bam sample_5.bam sample_7.bam sample_9.bam
sample_2.bam sample_4.bam sample_6.bam sample_8.bam
```

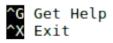
Wildcard *

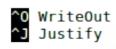


GNU nano 2.0.9

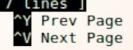
File: analysis.sh

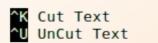
```
prefix="sample"
for file in $( ls *.bam );
do
    echo my_prog $file
done
```











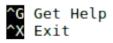


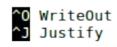


GNU nano 2.0.9

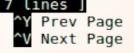
File: analysis.sh

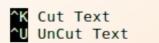
```
for file in $( ls *.bam );
do
    echo my_prog $file
done
```

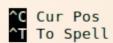














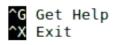
GNU nano 2.0.9

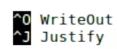
File:

```
for file in $( ls *.bam );
do
    echo my_prog $file
done
```

Loop over files

```
$ bash analysis.sh
my_prog sample_1.bam
my_prog sample_2.bam
my_prog sample_3.bam
my_prog sample_4.bam
my_prog sample_5.bam
my_prog sample_6.bam
my_prog sample_7.bam
my_prog sample_7.bam
my_prog sample_8.bam
my_prog sample_9.bam
$
```









\$ bash analysis.sh /path/to/my/bams

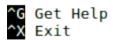


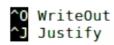
Loop over files

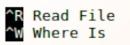
GNU nano 2.0.9 File: analysis.sh

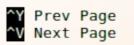
Modified

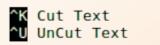
```
for file in $( ls $1/*.bam );
do
    echo my_prog $file
done
```

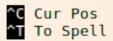














GNU nano 2.0.9

File:

```
for file in $( ls $1/*.bam );
do
    echo my_prog $file
done
```

Loop over files

```
$ bash analysis.sh /path/to/my/bams
my_prog /path/to/my/bams/sample_1.bam
my_prog /path/to/my/bams/sample_2.bam
my_prog /path/to/my/bams/sample_3.bam
my_prog /path/to/my/bams/sample_4.bam
my_prog /path/to/my/bams/sample_5.bam
my_prog /path/to/my/bams/sample_6.bam
my_prog /path/to/my/bams/sample_7.bam
my_prog /path/to/my/bams/sample_8.bam
my_prog /path/to/my/bams/sample_9.bam
```

```
^G Get Help
^X Exit
```

```
^0 WriteOut
^J Justify
```

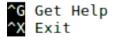
```
^R Read F
^W Where
```

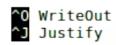


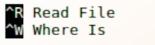
Loop over files

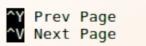
GNU nano 2.0.9 File: analysis.sh Modified

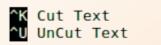
```
for file in $( ls $1/*.bam );
do
    my_prog $file
done
```

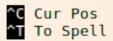












```
if condition; then
  action
fi
```



```
if true; then
  echo "This is true"
fi
```

result:

This is true

```
if false; then
  echo "This is true"
fi
```

result:



```
if [[ 5 < 9 ]]; then
  echo "This is true"
fi</pre>
```

result:

This is true

```
if [[ 5 > 9 ]]; then
  echo "This is true"
fi
```

result:

```
if [[ 5 == 9 ]]; then
  echo "This is true"
fi
```

result:



```
if [[ "Hello" == "Hello" ]]; then
  echo "This is true"
fi
```

result:

This is true



```
if [[ "Hello" == "Hi" ]]; then
  echo "This is true"
fi
```

result:



```
if [[ "Hello" == "Hel"* ]]; then
  echo "This is true"
fi
```

result:

This is true



```
for file in $( ls $1/*.bam );
do
    echo my_prog $file
done
```



```
for file in $( ls $1/*.bam );
do
    if [[ ... != "dog"* ]]; then
       echo my_prog $file
    fi
done
```



```
for file in $( ls $1/*.bam );
do
    if [[ ... != "dog"* ]]; then
       echo my_prog $file
    fi
done
```

Ex: \$file is /path/to/dog_1.bam



```
for file in $( ls $1/*.bam );
do
    if [[ ... != "dog"* ]]; then
       echo my_prog $file
    fi
done
```

Ex: \$file is /path/to/dog_1.bam

basename \$file



```
for file in $( ls $1/*.bam );
do
    if [[ ... != "dog"* ]]; then
       echo my_prog $file
    fi
done
```

Ex: \$file is /path/to/dog_1.bam

basename \$file

dog_1.bam



```
for file in $( ls $1/*.bam );
do
    if [[ $(basename $file) != "dog"* ]]; then
        echo my_prog $file
    fi
done
```

Ex: \$file is /path/to/dog_1.bam

basename \$file

dog_1.bam



```
for file in $( ls $1/*.bam );
do
    if [[ $(basename $file) != "dog"* ]]; then
        my_prog $file
    fi
done
```

Ex: \$file is /path/to/dog_1.bam

basename \$file

dog_1.bam



- Programming is programming
 - Perl, Python, Bash, and more



- Programming is programming
 - Perl, Python, Bash, and more

```
for file in $( ls $1/*.bam );
do
    if [[ $(basename $file) != "dog"* ]]; then
        my_prog $file
    fi
done
```



- Programming is programming
 - Perl, Python, Bash, and more



- Programming is programming
 - Perl, Python, Bash, and more



- Programming is programming
 - Perl, Python, Bash, and more
- Start with one, git gud, (learn another)



- Programming is programming
 - Perl, Python, Bash, and more
- Start with one, git gud, (learn another)

PYTHON



Laboratory time! (yet again)