http://www.geocities.jp/geo_sunisland/variable.html

基本的なこと

■変数に値を設定する

```
VAR1=123456
VAR2=hogehoge
VAR3=" a b c d e f g "
```

■変数の値を参照する

・変数の値を表示

```
# 変数 VAR1 の値を表示
echo $VAR1

# 変数 VAR の値と「1」を表示
echo ${VAR}1

# 変数 VAR2 の値と変数 VAR3 の値を表示
echo ${VAR2}${VAR3}
```

変数内に改行文字がある場合

```
echo $test
```

とすると、改行はスペースとして出力される。

```
echo "$test"
```

とすると、改行がそのまま改行で出力される

・変数の値を他の変数へ設定する

```
# 変数 VAR に変数 VAR1 の値を設定
VAR=$VAR1

# 変数 VAR に変数 VAR の値と「1」を設定 ( 変数 VAR の値の最後に「1」を付加する )
```

VAR=${VAR}1

```
# 変数 VAR に変数 VAR2 と変数 VAR3 の値を結合した値を設定
VAR=${VAR2}${VAR3}
```

変数の値を参照するには変数名に「$」を付ける。
変数の値を参照する場合は変数名の頭に「$」を付ける。参照する変数を明確にする場合は「{}」(中括弧) で変数名を囲む。

■コマンドの実行結果を変数に設定する

```
VAR=`command`
```

コマンドの実行結果を、エラーも含めて変数に設定したい場合は次のようにする。

```
VAR=`command 2>&1`
```

バッククォートと同様の機能は「$(command)」でも実現可能。

```
VAR=$(command)
```

■変数の値を変数名として値を参照する

```
eval echo '$'$VAR
```

例

```
$ FOO="BAR"
$ BAR="SUCCESS"
$ eval echo '$'$FOO
SUCCESS
```

※↑変数 FOO に設定されている値を変数名として参照している。

文字列操作

http://open-groove.net/shell/summary-variables/

■ ${var#pattern} ${var##pattern}

var の値の先頭から pattern にマッチする部分が削除された状態で展開される。
の場合は最短一致、## は最長一致となる。pattern にはパス名の展開と同じ規則が適用される。

```
$ echo ${animal}
$ animal=”monkey:cat:dog:cow”
$ echo ${animal}
monkey:cat:dog:cow
$ echo ${animal#monkey}
:cat:dog:cow  ← monkey が削除された状態で出力
$ echo ${animal#m?}
nkey:cat:dog:cow
$ echo ${animal#m*}
onkey:cat:dog:cow
$ echo ${animal##m*}
```

■ ${var%pattern}${var%%pattern}

と逆で var の値の後ろから pattern にマッチする部分が削除された状態で展開される。% の場合は最短一致、%% は最長一致となる。pattern にはパス名の展開と同じ規則が適用される。

■ ${var/pattern/str}${var//pattern/str}${var/pattern}${var//pattern}

var の値のうち pattern にマッチする部分が str に置換された状態で展開される。
/ の場合は最初にマッチした部分のみが、// の場合はマッチする部分すべてが置き換えられる。

```
$ echo ${meal}  ←変数が未設定
$ meal=”catfood:catmilk”
$ echo $meal
catfood:catmilk
```

```
$ echo ${meal/cat/dog}
dogfood:catmilk  ←最初にマッチした”cat”のみdogに置換
$ echo ${meal//cat/dog}
dogfood:dogmilk  ←すべての”cat”がdogに置換された
```

■ ${#var}

変数 var の値の文字列に置き換えられる。

■オフセット位置から長さ分を取得

```
HOGE="abcdef"
echo ${HOGE:0:2}
$ ab

echo ${HOGE:4:2}
$ ef
```

■オフセットから最後まで出力

```
HOGE="abcdef"
echo ${HOGE:2}
$ cdef
```

■末尾からのオフセット位置を指定

マイナスを指定すると末尾からのオフセット位置になる

```
HOGE="abcdef"
echo ${HOGE:0:-2}
$ abcd
```