

Bash `loop` statements

for loop

For loops allow repeated execution of a command sequence based on an iteration variable. Bash supports two kinds of for loop, a “list of values” and a “traditional” c-like method.

```
for varname in list
do
    commands
done
```

Note that



- Bash `for`, `in`, `do` and `done` are keywords
- list contains a list of items, which can be in the statement or fetched from a variable that contains several words separated by spaces.
- If list is missing from the for statement, then bash uses positional parameters that were passed into the shell.

Example 1:

main.sh

bash-day-for.sh

```
#!/bin/bash
i=1
for day in Mon Tue Wed Thu Fri
do
echo "$(( i++ )) : $day"
done
```



Note the counter increment command `$((i++))` which increases `i` by `1`,

each time called.





Bash c-style **for-loop**:

main.sh

bash-c-style-for.sh

```
#!/bin/bash

for (( i=1; i <= 3; i++ ))
do
    echo "Your random number $i: $RANDOM"
done
```



Note that the built-in variable **\$RANDOM** generates a random number each time called.

Bash **while** loop

Bash **while** allows for repetitive execution of a list of commands, as long as the command controlling the while loop executes successfully (exit status of zero). The syntax is:

{title="Bash while-loop Syntax", lang=bash}

```
while expression
do
    commands
done
```

Note that

- **while**, **do**, **done** are keywords
- Expression is any expression which returns a scalar value
- Commands between do and done are executed while the provided conditional expression is true.

Example:

main.sh

```
#!/bin/bash

# This script prints 4 sequential numbers 0 1 2 3.
```

bash-whileloop.sh

```
x="0"
```

```
while [ $x -lt 4 ]
do
    x=$((x+1))
    echo $x
done
```



An infinite loop:

```
#!/bin/bash

while :
do
    echo "Do something; hit [CTRL+C] to stop!"
done
```



A more (on-purpose) complicated example:

```
#!/bin/bash

select=0

echo "1. Apple"
echo "2. Orange"
echo "3. Lime"

echo -n "Please select [1,2 or 3] : "

# Loop while the variable 'select' is equal 0
# bash while loop

while [ $select -eq 0 ]; do

# read user input
read select

# bash nested if/else
if [ $select -eq 1 ] ; then

    echo "You have selected: Apple"

else

    if [ $select -eq 2 ] ; then
        echo "You have selected: Orange"
    else

        if [ $select -eq 3 ] ; then
            echo "You have selected: Lime"
        fi
    fi
fi
done
```



```

        else
            echo "Please select between 1-3 !"
            echo "1. Apple"
            echo "2. Orange"
            echo "3. Lime"
            echo -n "Please select [1,2 or 3] : "
            choice=0
        fi
    fi
fi
done

```

until loop

Bash **until loop** is very similar to the **while loop**, except that the loop executes until the condition executes successfully.

main.sh

[bash-until-loop.sh](#)

```
#!/bin/bash
```

```

i=0
# bash until loop
until [ $i -gt 3 ]; do
    echo "i :" $i
    i=$((i+1))
done

```

