

# Bash – Check if variable is set

## Bash – Check if variable is set

---

To check if a variable is set in Bash Scripting, use `-v var` or `-z ${var}` as an expression with `if` command.

This checking of whether a variable is already set or not, is helpful when you have multiple script files, and the functionality of a script file depends on the variables set in the previously run scripts, etc.

In this tutorial, we shall learn the syntax and usage of the above mentioned expressions with examples.

### Syntax

---

Following is the syntax of boolean expressions which check if the variable is set:

```
[[ -v variableName ]]
```

```
[[ -z variableName ]]
```

The boolean expression returns true if the variable is set and false if the variable is not set.

### Example 1 – Check if Variable is Set using -v

---

In this example, we use `[[ -v variableName ]]` boolean expression to check if variables `a` and `b` are set with the help of [bash if else](#) statement.

### Bash Script File

---

```
#!/bin/bash

a=10

# a: variable is set
if [[ -v a ]];
then
    echo "variable named a is already set"
else
    echo "variable a is not set"
fi
```

```

# a: variable is set
if [[ -v a ]];
then
    echo "variable named a is already set"
else
    echo "variable a is not set"
fi

# b: variable is not set
if [[ -v b ]];
then
    echo "variable named b is already set"
else
    echo "variable b is not set"
fi

```

### Output

```

~/workspace/bash$ ./bash-if-variable-is-set-example
variable named a is already set
variable b is not set

```

Variable `a` is defined and assigned a value of 10 and hence is set. For variable `b`, we have not defined it by assigning a value. So, we got the result that variable `b` is not set.

## Example 2 – Check if Variable is Set using -z

In this example, we use `[[ -z ${variableName} ]]` boolean expression to check if variables `a` and `b` are set with the help of [bash if else](#) statement.

### Bash Script File

```

#!/bin/bash

a=10

# a: variable is set
if [[ -z ${a} ]];
then
    echo "variable a is not set"
else
    echo "variable named a is already set"
fi

# b: variable is not set
if [[ -z ${b} ]];
then
    echo "variable b is not set"
else
    echo "variable named b is already set"
fi

```

### Output

```

~/workspace/bash$ ./bash-if-variable-is-set-example
variable named a is already set
variable b is not set

```

## Conclusion

In this [Bash Tutorial](#), we have learnt to check if a variable is set or not using `[[ -v variableName ]]` or `[[ -z ${variableName} ]]`, with the help of example Bash scripts.

### Bash Shell Scripting

- ◆ [Bash Tutorial](#)
- ◆ [Bash Script Example](#)
- ◆ [Bash File Extension](#)
- ◆ [Bash Echo](#)
- ◆ [Bash Comments](#)
- ◆ [Bash Variable](#)
- ◆ [Bash Command Line Arguments](#)
- ◆ [Bash Read User Input](#)
- ◆ [Bash Read Password](#)
- ◆ [Bash Date Format](#)
- ◆ [Bash Sleep](#)

### Operators

- ◆ [Bash Arithmetic Operators](#)

### Conditional Statements

- ◆ [Bash If](#)
- ◆ [Bash If Else](#)
- ◆ [Bash Else If](#)
- ◆ [Bash Case](#)

### Loops

- ◆ [Bash For Loop](#)
- ◆ [Bash While Loop](#)
- ◆ [Bash Until Loop](#)

## Strings

- ◆ [Bash String Manipulation Examples](#)
- ◆ [Bash String Length](#)
- ◆ [Bash If String Equals](#)
- ◆ [Bash Split String](#)
- ◆ [Bash SubString](#)
- ◆ [Bash Concatenate String](#)
- ◆ [Bash Concatenate Variables to Strings](#)

## Functions

- ◆ [Bash Function](#)
- ◆ [Bash Override Builtin Commands](#)

## Arrays

- ◆ [Bash Array](#)

## Files

- ◆ [Bash Write to File](#)
- ◆ [Bash Read File](#)
- ◆ [Bash Read File line by line](#)
- ◆ [Bash If File Exists](#)
- ◆ [Bash If File is Directory](#)
- ◆ [Bash If File is Readable](#)

## Bash Others

- ⇒ [Bash Check if variable is set](#)