

# UNIX / LINUX - SHELL FILE TEST OPERATORS EXAMPLE

<http://www.tutorialspoint.com/unix/unix-file-operators.htm>

Copyright © tutorialspoint.com

## Advertisements

We have a few operators that can be used to test various properties associated with a Unix file.

Assume a variable **file** holds an existing file name "test" the size of which is 100 bytes and has **read**, **write** and **execute** permission on –

Operator	Description	Example
<b>-b file</b>	Checks if file is a block special file; if yes, then the condition becomes true.	[ -b \$file ] is false.
<b>-c file</b>	Checks if file is a character special file; if yes, then the condition becomes true.	[ -c \$file ] is false.
<b>-d file</b>	Checks if file is a directory; if yes, then the condition becomes true.	[ -d \$file ] is not true.
<b>-f file</b>	Checks if file is an ordinary file as opposed to a directory or special file; if yes, then the condition becomes true.	[ -f \$file ] is true.
<b>-g file</b>	Checks if file has its set group ID (SGID) bit set; if yes, then the condition becomes true.	[ -g \$file ] is false.
<b>-k file</b>	Checks if file has its sticky bit set; if yes, then the condition becomes true.	[ -k \$file ] is false.
<b>-p file</b>	Checks if file is a named pipe; if yes, then the condition becomes true.	[ -p \$file ] is false.
<b>-t file</b>	Checks if file descriptor is open and associated with a terminal; if yes, then the condition becomes true.	[ -t \$file ] is false.
<b>-u file</b>	Checks if file has its Set User ID (SUID) bit set; if yes, then the condition becomes true.	[ -u \$file ] is false.
<b>-r file</b>	Checks if file is readable; if yes, then the condition becomes true.	[ -r \$file ] is true.
<b>-w file</b>	Checks if file is writable; if yes, then the condition becomes true.	[ -w \$file ] is true.
<b>-x file</b>	Checks if file is executable; if yes, then the condition becomes true.	[ -x \$file ] is true.
<b>-s file</b>	Checks if file has size greater than 0; if yes, then condition becomes true.	[ -s \$file ] is true.
<b>-e file</b>	Checks if file exists; is true even if file is a directory but exists.	[ -e \$file ] is true.

## Example

The following example uses all the **file test** operators –

Assume a variable **file** holds an existing file name **"/var/www/tutorialspoint/unix/test.sh"** the size of which is 100 bytes and has **read**, **write** and **execute** permission –

## Live Demo

```
#!/bin/sh

file="/var/www/tutorialspoint/unix/test.sh"

if [ -r $file ]
then
    echo "File has read access"
else
    echo "File does not have read access"
fi

if [ -w $file ]
then
    echo "File has write permission"
else
    echo "File does not have write permission"
fi

if [ -x $file ]
then
    echo "File has execute permission"
else
    echo "File does not have execute permission"
fi

if [ -f $file ]
then
    echo "File is an ordinary file"
else
    echo "This is sepcial file"
fi

if [ -d $file ]
then
    echo "File is a directory"
else
    echo "This is not a directory"
fi

if [ -s $file ]
then
    echo "File size is zero"
else
    echo "File size is not zero"
fi

if [ -e $file ]
then
    echo "File exists"
else
    echo "File does not exist"
fi
```

The above script will produce the following result –

```
File does not have write permission
File does not have execute permission
This is sepcial file
This is not a directory
File size is not zero
File does not exist
```

The following points need to be considered while using file test operators –

- There must be spaces between the operators and the expressions. For example,  $2+2$  is not correct; it should be written as  $2 + 2$ .
- **if...then...else...fi** statement is a decision-making statement which has been explained in the next chapter.