Special variables

- Bash reserved variables
- Most commonly used variables

TRAINING
CENTER

Special variables

Bash has a number of special variables that either defined on the Bash start, or change depending on the context.

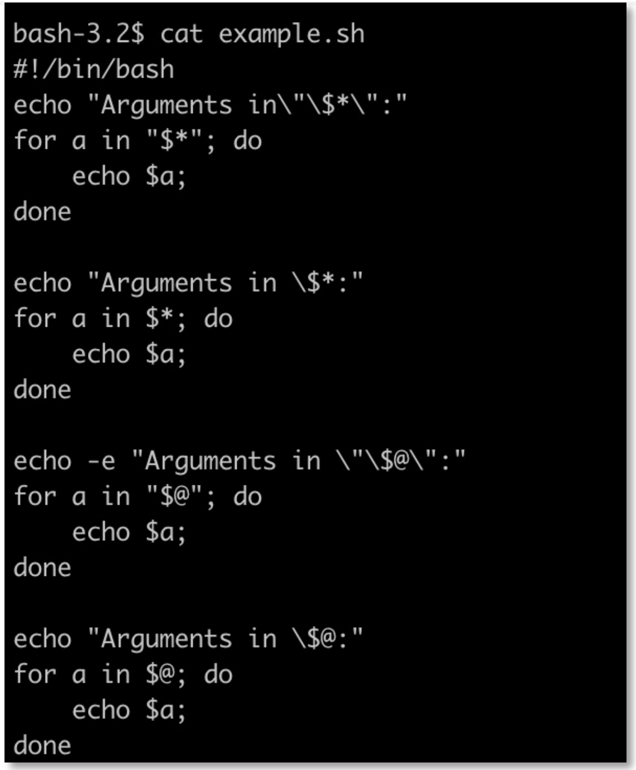
They’re useful when you’re trying to use runtime variables.

$# $_ $- $$ $! $0 $n $* $@

Arguments variables

Variable	Value
$*	All arguments in command line in a form of a single string variable, broken down by a delimiter ($IFS)
“$*”	All arguments in command line in a form of a single string variable
$@	All arguments in command line in a form of an array
“$@”	All arguments in command line in a form of quoted strings

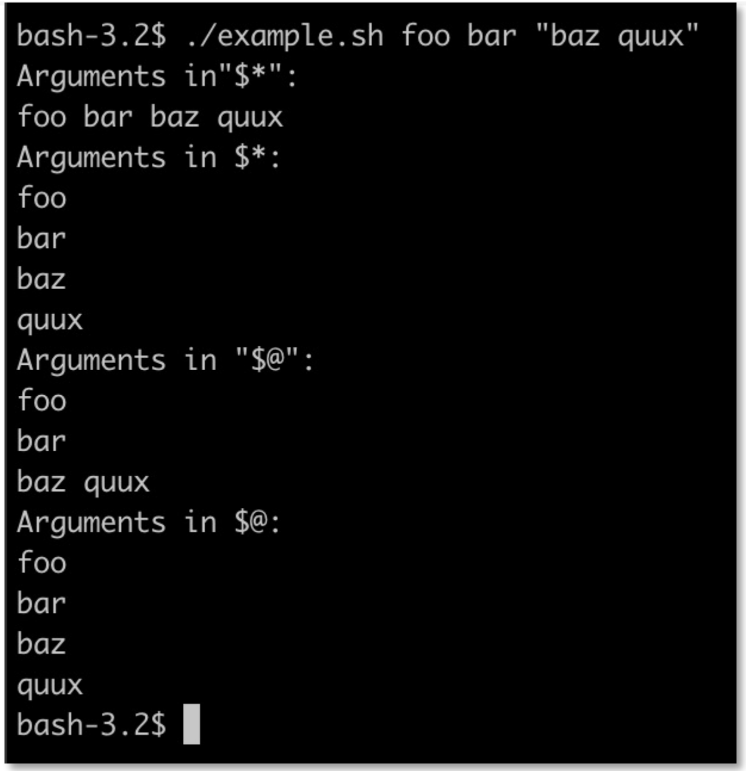
Arguments variables usage



```
bash-3.2$ cat example.sh
#!/bin/bash
echo "Arguments in\"\$*\":"
for a in "$*"; do
    echo $a;
done

echo "Arguments in \$*:"
for a in $*; do
    echo $a;
done

echo -e "Arguments in \"\$@\":"
for a in "$@"; do
    echo $a;
done

echo "Arguments in \$@:"
for a in $@; do
    echo $a;
done
```

Code

→



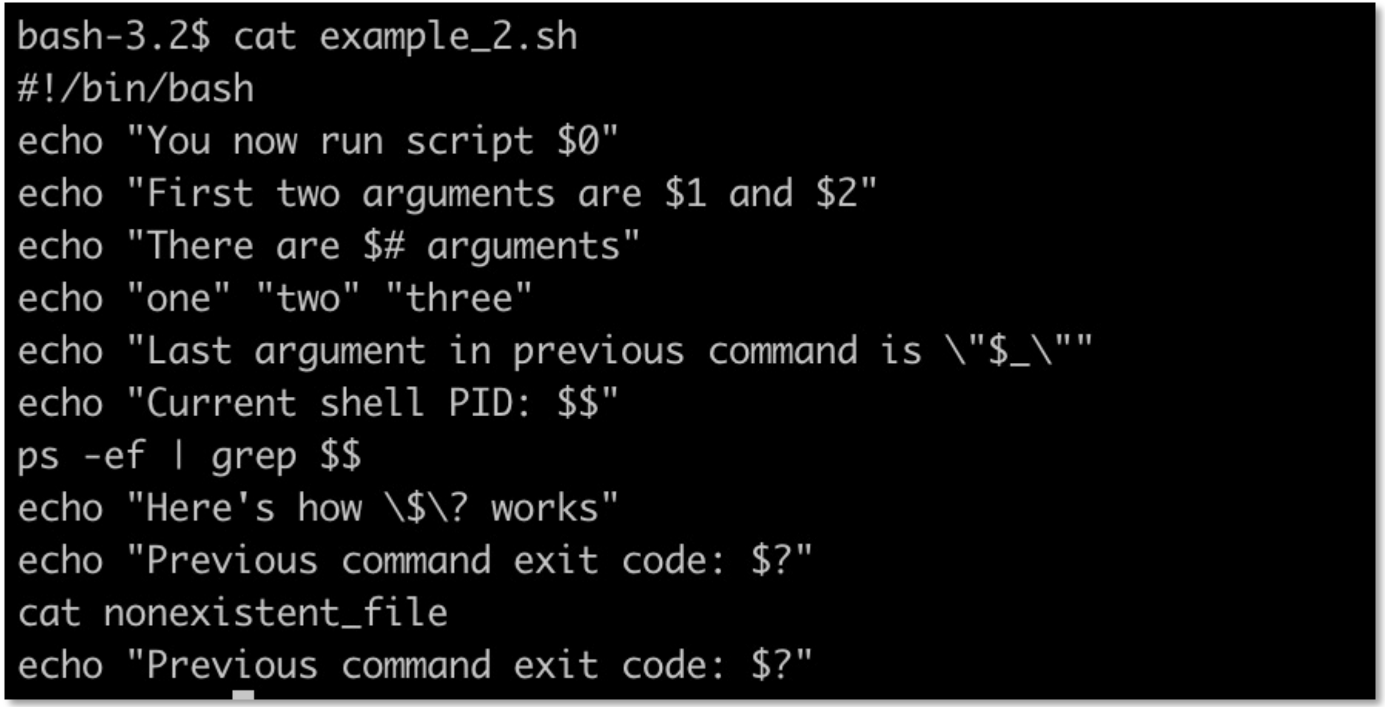
```
bash-3.2$ ./example.sh foo bar "baz quux"
Arguments in"$*":
foo bar baz quux
Arguments in $*:
foo
bar
baz
quux
Arguments in "$@":
foo
bar
baz quux
Arguments in $@:
foo
bar
baz
quux
bash-3.2$ 
```

Execution

Special variables

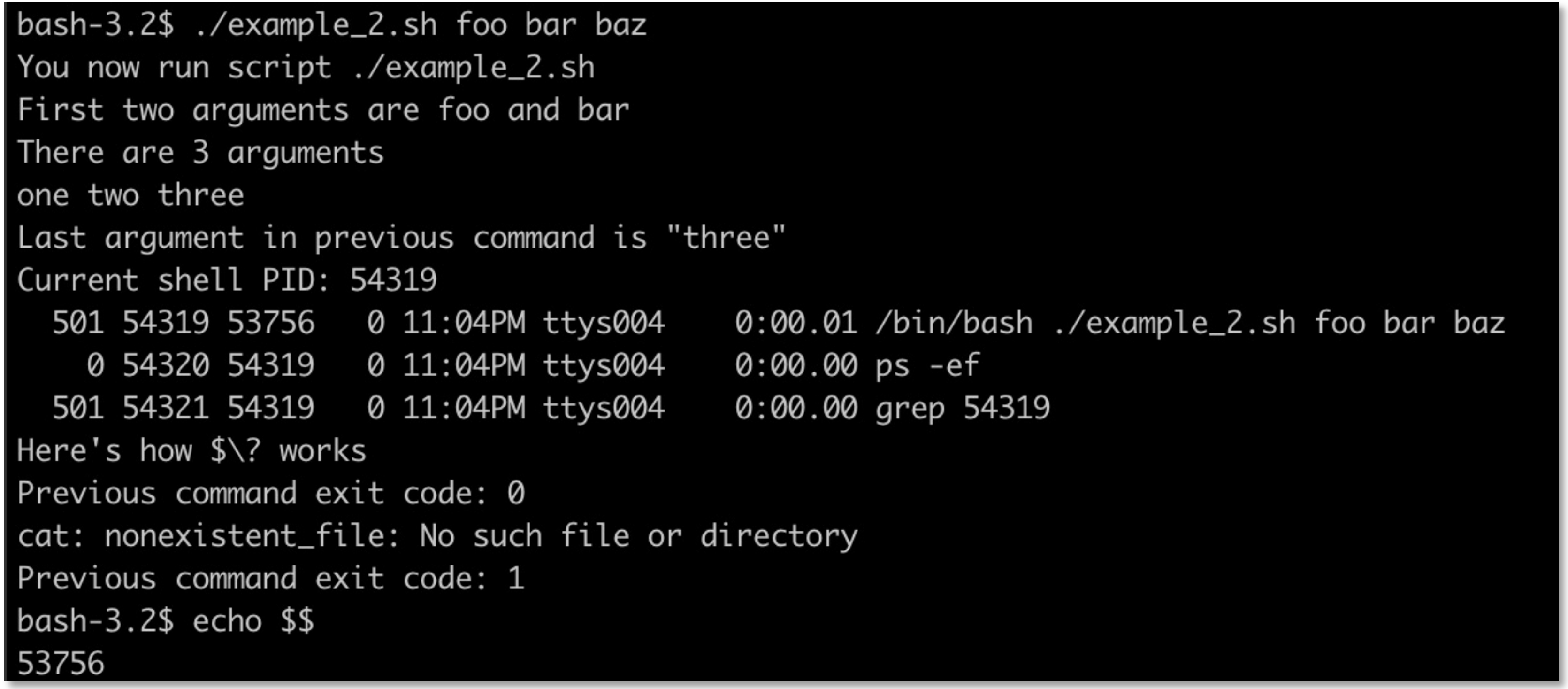
Variable	Value
$#	Number of arguments passed in a command line
$_	Last argument of the previous command
$0	Name of a script being executed
$n or ${n}	N-th (positional) argument passed in a command line
$-	Set of Bash flags enabled in current shell (man bash → SHELL BUILTIN COMMANDS)
$$	Process number of the shell or current command
$!	Process number of last command in background (jobs)
$?	Exit code of the last command

Special variables usage



```
bash-3.2$ cat example_2.sh
#!/bin/bash
echo "You now run script $0"
echo "First two arguments are $1 and $2"
echo "There are $# arguments"
echo "one" "two" "three"
echo "Last argument in previous command is \"$_\""
echo "Current shell PID: $$"
ps -ef | grep $$
echo "Here's how \$\? works"
echo "Previous command exit code: $?"
cat nonexistent_file
echo "Previous command exit code: $?"
```

Special variables usage



```
bash-3.2$ ./example_2.sh foo bar baz
You now run script ./example_2.sh
First two arguments are foo and bar
There are 3 arguments
one two three
Last argument in previous command is "three"
Current shell PID: 54319
  501 54319 53756   0 11:04PM ttys004    0:00.01 /bin/bash ./example_2.sh foo bar baz
    0 54320 54319   0 11:04PM ttys004    0:00.00 ps -ef
  501 54321 54319   0 11:04PM ttys004    0:00.00 grep 54319
Here's how $\? works
Previous command exit code: 0
cat: nonexistent_file: No such file or directory
Previous command exit code: 1
bash-3.2$ echo $$
53756
```

Running Commands

Thanks for watching!