Functions

- What is a function
- Function declaration and invocation
- Function arguments
- Function control

TRAINING

CENTER

Function declaration

Functions exist in most of the programming languages, and Bash is no exception. They're used mostly for grouping several commands and then executing them multiple times. They're called by their name like usual commands.

Functions are declared using this syntax (from man):

[function] *name* () { *command-list*; }

This defines a function called *name*. If a function called *name* already exists it will be overwritten.

It is important to remember that **Bash is an interpreted language** that is executed line by line. That means that you should declare the function **before you use it** in the script's text.

Function invocation

As was said earlier, you call a function just like any other command, but the functions power is that they can receive positional arguments and return values!

```
function example() {
FIRST=$1
SECOND=$2
echo “$FIRST - $SECOND”
}
```

```
example foo bar
… will result in …
“foo - bar”
```

You already know these special variables, but we’ve used them on script level only. All of them work almost the same way in functions too! If you forgot them – refer to topic **6 – Special Variables** and **15 – Positional Arguments**.

Function example

```
bash-3.2$ cat ./quiz.sh
#!/bin/bash

function play {
  RND=$(($RANDOM % $2))
  [ $1 -gt 2 ] || [ $1 -lt 0 ] && echo "NO! You have to choose between 0 and $(($2 - 1))" && exit 123
  [ $RND -ne $1 ] && echo "Wrong! The number is $RND" && exit 1
}

echo "ROUND 1: Guess the number between 0 and 1"
read NUMBER
play $NUMBER 2
echo "You've guessed right!"

echo "ROUND 2: Guess the number between 0 and 2"
read NUMBER
play $NUMBER 3
echo "You've guessed right!"

echo "ROUND 3: Guess the number between 0 and 3"
read NUMBER
play $NUMBER 4
echo "You've guessed right! You won the game!"
```

example code

```
bash-3.2$ ./quiz.sh
ROUND 1: Guess the number between 0 and 1
1
You've guessed right!
ROUND 2: Guess the number between 0 and 2
2
Wrong! The number is 0
bash-3.2$ ./quiz.sh
ROUND 1: Guess the number between 0 and 1
1
You've guessed right!
ROUND 2: Guess the number between 0 and 2
1
You've guessed right!
ROUND 3: Guess the number between 0 and 3
2
Wrong! The number is 0
```

example result

Function control

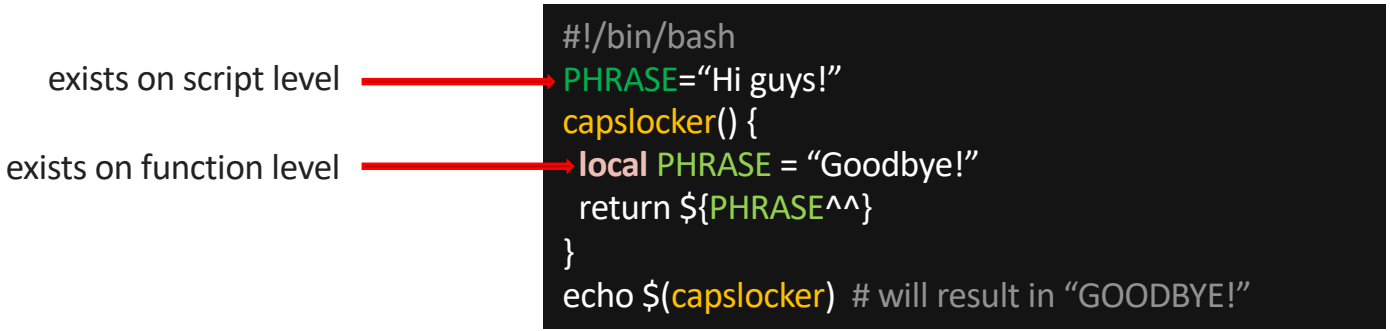
Functions can also return values with the same named command - **return**:

return – exits function and returns some value that is passed to it. If nothing is passed – returns exit code 0.

You may return any value, be it exit code or some string, depending on the use case.

Function variables scope

Every variable in Bash has global scope. That means that you can use any variable, defined outside the function, inside the function, and define or redefine variables inside the function, and they will persist through the script. However, you can make the variable exist only in function scope using the **local** keyword.



Function control example

```
#!/bin/bash

function retry () {
  local FUNCTION="$@"
  local COUNT=0
  local MAX=5
  local SLEEP=2

  while true; do
    $FUNCTION && break
    if [[ $COUNT -lt $MAX ]]; then
      ((COUNT++))
      echo "Command failed, retrying after $SLEEP seconds... ($COUNT/$MAX)"
      sleep $SLEEP
      continue
    fi
    echo "Command failed, out of retries"
    exit 1
  done
}

function healthcheck () {
  local WEBSITE_URL=$1

  curl -sIL $WEBSITE_URL | grep " 200 " > /dev/null 2>&1
  [ $? -ne 0 ] && echo "URL haven't responded 200" && return 1
  echo "URL responded 200"
}

[ $# -ne 1 ] && echo "1 argument required, got $#" && exit 1
retry healthcheck $1
```

example code

```
bash-3.2$ ./example.sh google.com
URL responded 200
bash-3.2$ ./example.sh this_website_doesnt_exist.com
URL haven't responded 200
Command failed, retrying after 2 seconds... (1/5)
URL haven't responded 200
Command failed, retrying after 2 seconds... (2/5)
URL haven't responded 200
Command failed, retrying after 2 seconds... (3/5)
URL haven't responded 200
Command failed, retrying after 2 seconds... (4/5)
URL haven't responded 200
Command failed, retrying after 2 seconds... (5/5)
URL haven't responded 200
Command failed, out of retries
bash-3.2$
```

example result

Thanks for watching!