FULL STACK WEB DEVELOPER PART V

Introduction to Programming

how computer programs work

Panos M.

www.techcareerbooster.com

Full Stack Web Developer Part V: Introduction to Programming

Panos Matsinopoulos

This book is for sale at
http://leanpub.com/full-stack-web-developer-part-v-introduction-to-programming

This version was published on 2019-08-12

Contents

The Bundle and the TCB Subscription

Part of Bundle

This book is not sold alone. It is part of the bundle Full Stack Web Developer.

Goes with a TCB Subscription

When you purchase the bundle, then you have full access to the contents of the TCB Courses.

Each TCB Subscription Goes with the Bundle

Moreover, this goes vice-versa. If you purchase the subscription to the TCB Courses, then you are automatically eligible for the Full Stack Web Developer bundle.

Credits To Photo on Cover Page

We have designed the cover page, but the photo in the middle is the creation of our friend Telis Marin. He is an amateur photographer. He doesn't have an official Web site where you could find more of his amazing photos. Hence, if you want to see more of his work, the only way you can now do it is by making him an FB Friend here.

Telis Marin is an author, a publisher Edizioni Edilingua and a teacher trainer. He has written more than 20 books for learning Italian, which are used by schools and universities in over 80 countries.

About and Copyright Notice

Full Stack Web Developer - Part V - Introduction to Programming 1st Edition, August 2019

by Panos M. for Tech Career Booster (https://www.techcareerbooster.com)

For more information: https://www.techcareerbooster.com

Introduction To Programming

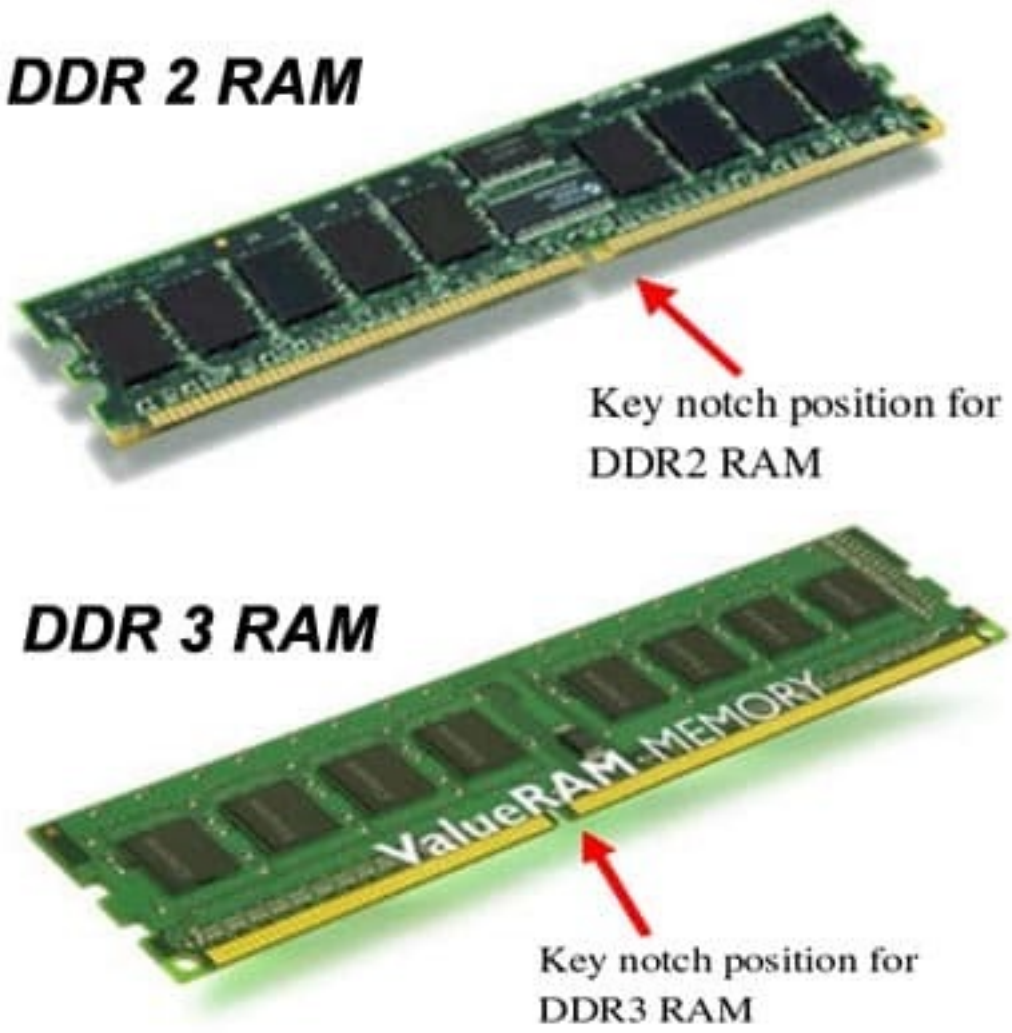
This section is the milestone section that introduces you to programming. HTML and CSS are not programming languages. Here, you will have an introduction on what a computer program is so that in the next sections we will be able to start learning a real programming language. You will also have an introduction to Linux terminal commands, necessary for you to be able to navigate in the Operating System filesystem using command line.

1 - What is a Computer Program

Summary

In this chapter you will get a first glimpse on what a computer program might be. Starting from the very basics, like the CPU and RAM:

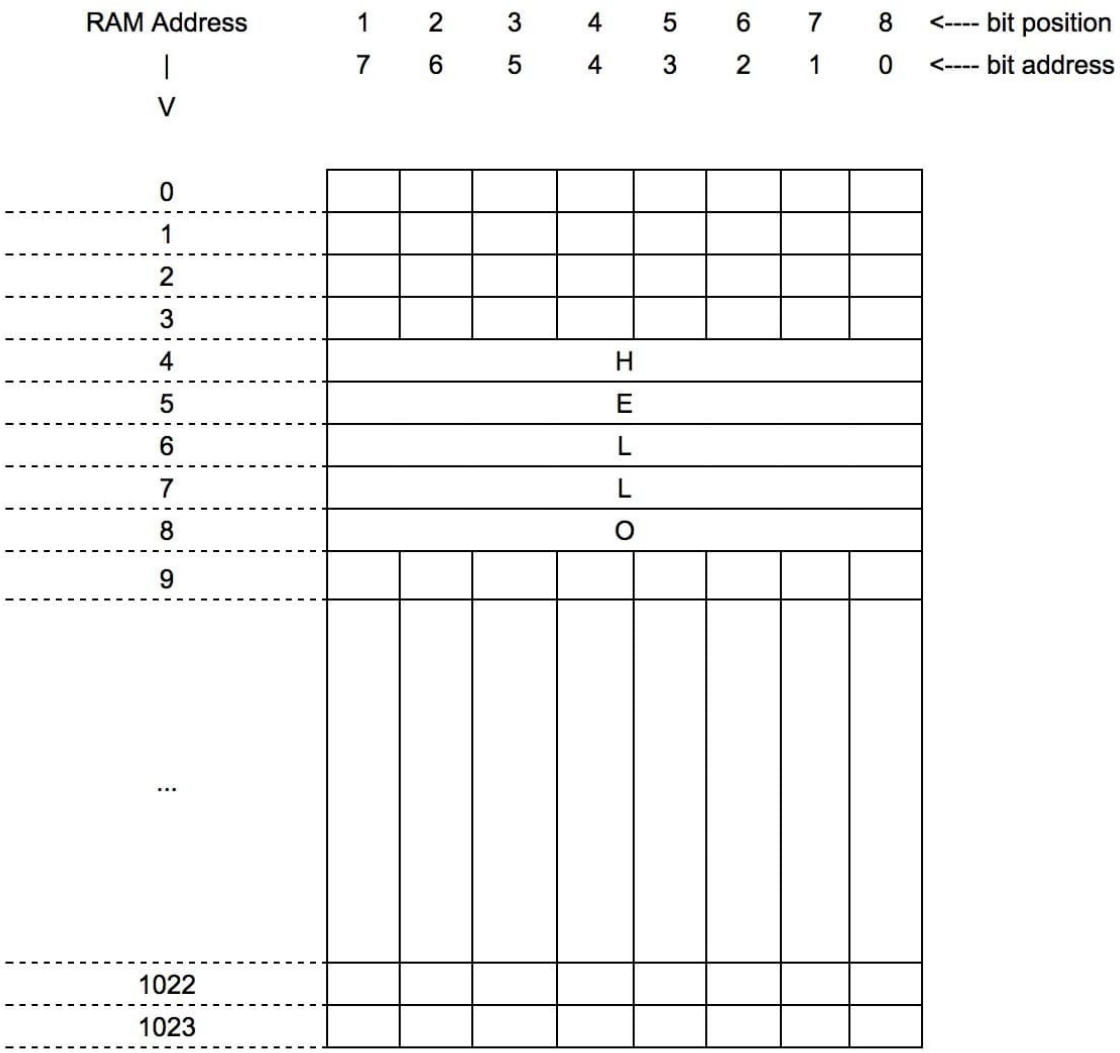
A CPU



RAM Banks

and talking about registers.

A first encounter with the RAM structure and how content is stored inside it.



RAM holds the word Hello

Then we will have a very basic introduction on Numbering Systems.

We will learn about the Assembly language and compare it to the High Level Programming Languages.

```
def compare(number)
  return if number < 97 || number > 122
  number - 32
end
```

```
; Example of IBM PC assembly language
; Accepts a number in register AX;
; subtracts 32 if it is in the range 97-122;
; otherwise leaves it unchanged.

SUB32  PROC         ; procedure begins here
       CMP  AX,97   ; compare AX to 97
       JL   DONE    ; if less, jump to DONE
       CMP  AX,122  ; compare AX to 122
       JG   DONE    ; if greater, jump to DONE
       SUB  AX,32   ; subtract 32 from AX
DONE:  RET          ; return to main program
SUB32  ENDP         ; procedure ends here
```

Compare High Level Programming Language to Assembly Equivalent

Finally, we will see the differences between Statically Typed Compiled Languages and Dynamically Typed Script Languages.

Learning Goals

1. Learn about the computer and CPU.
2. Learn about the motherboard.
3. Learn about the Program.
4. Learn about the RAM.
5. Learn about the RAM addresses.
6. Learn about the RAM content.
7. Learn about the Numbering systems.
8. Learn about the Assembly.
9. Learn about the High Level Programming Languages.
10. Learn about Statically Types Compiled Programming Languages.
11. Learn about Dynamically Typed Script Programming Languages.

Introduction

In this chapter we are going to learn what is a computer program. Let's start.

Computer and CPU

Most of the people today they know very well what a computer is.

A Computer

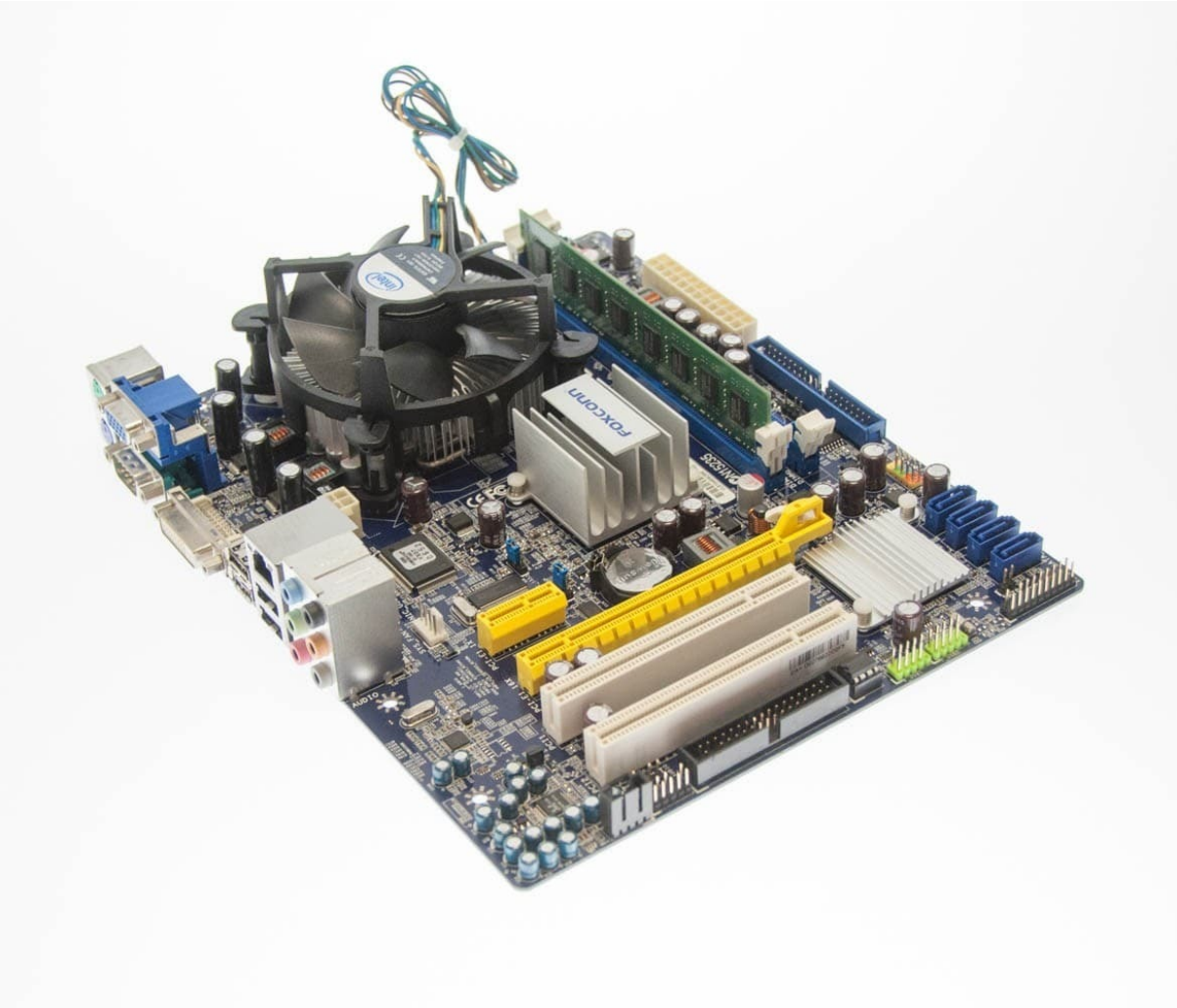
The computer is a machine that is given detailed instructions and executes them with amazing speed. Something that the humans cannot do. That's why, today, many of the tasks that used to be done by humans, are now being carried out by computers. Computers are much faster than humans and, if programmed correctly, they are less prone to error.

But, if brain is the human body part that is responsible for humans to think and act, what is the equivalent component in a computer, that allows it to execute instructions and think so fast?

It is the CPU, the Central Processing Unit.

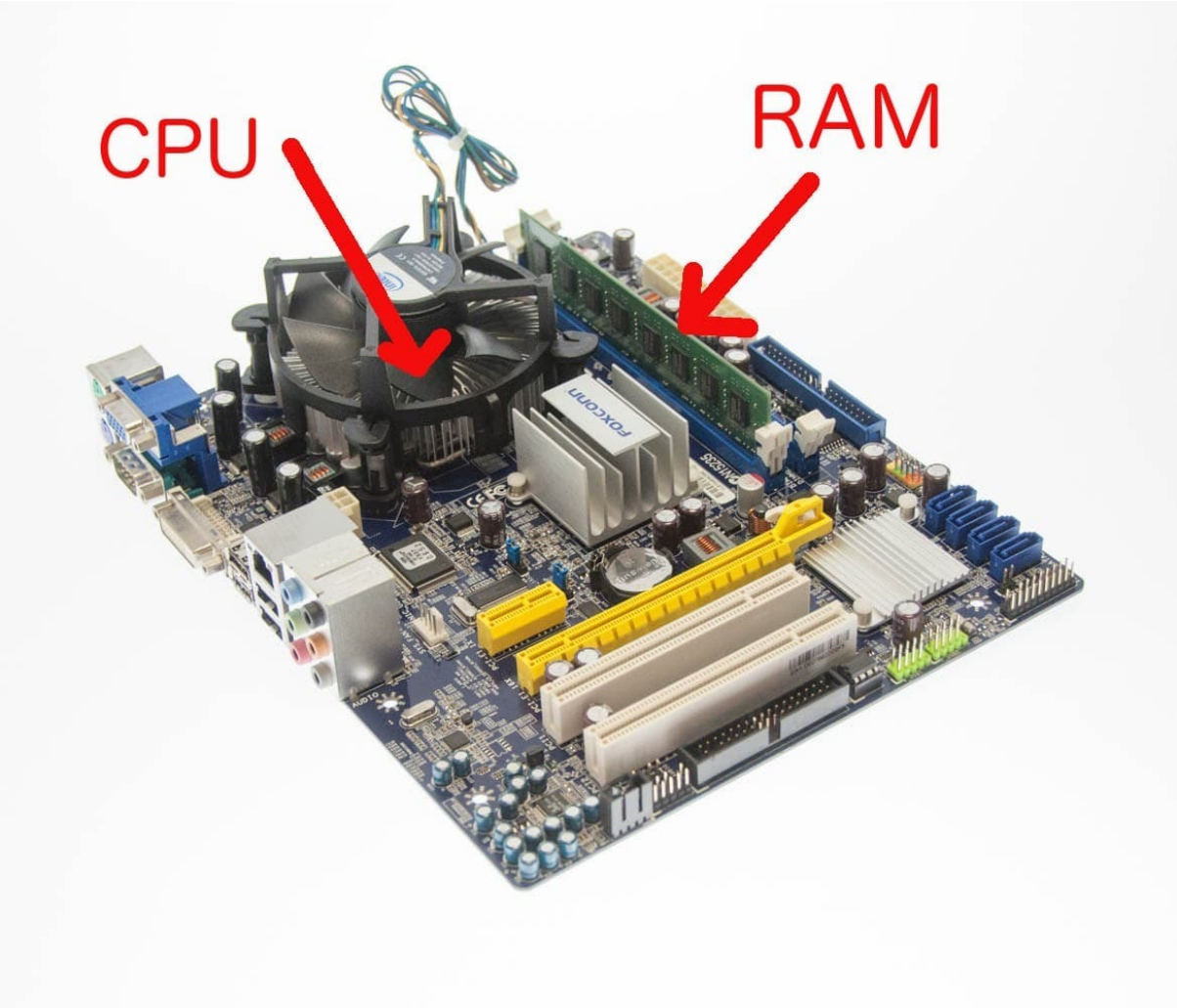
A CPU

The CPU is an electronic chip and it is usually located inside the computer chassis, plugged on the computer motherboard.



A Motherboard

The computer motherboard is a big square electronic board that integrates various electronic chips, one of which is the CPU. The other important stuff that integrates is the RAM (Random Access Memory). More about this in a while.



CPU and RAM on Motherboard

As you can see in the previous picture, the CPU is actually covered with a big fan that is always trying to keep the CPU chip cool. The RAM is another set of chips that is usually vertically positioned on the motherboard. We will talk about RAM later on.

The CPU, is the chip that executes all the instructions. Inside it, among others, there are a series of registers. The registers are part of the CPU local data storage.

Hence, we have

- the CPU
- the RAM and
- the Registers inside CPU.

Program

Given that we have a CPU, how can we ask it to execute operations? We need to

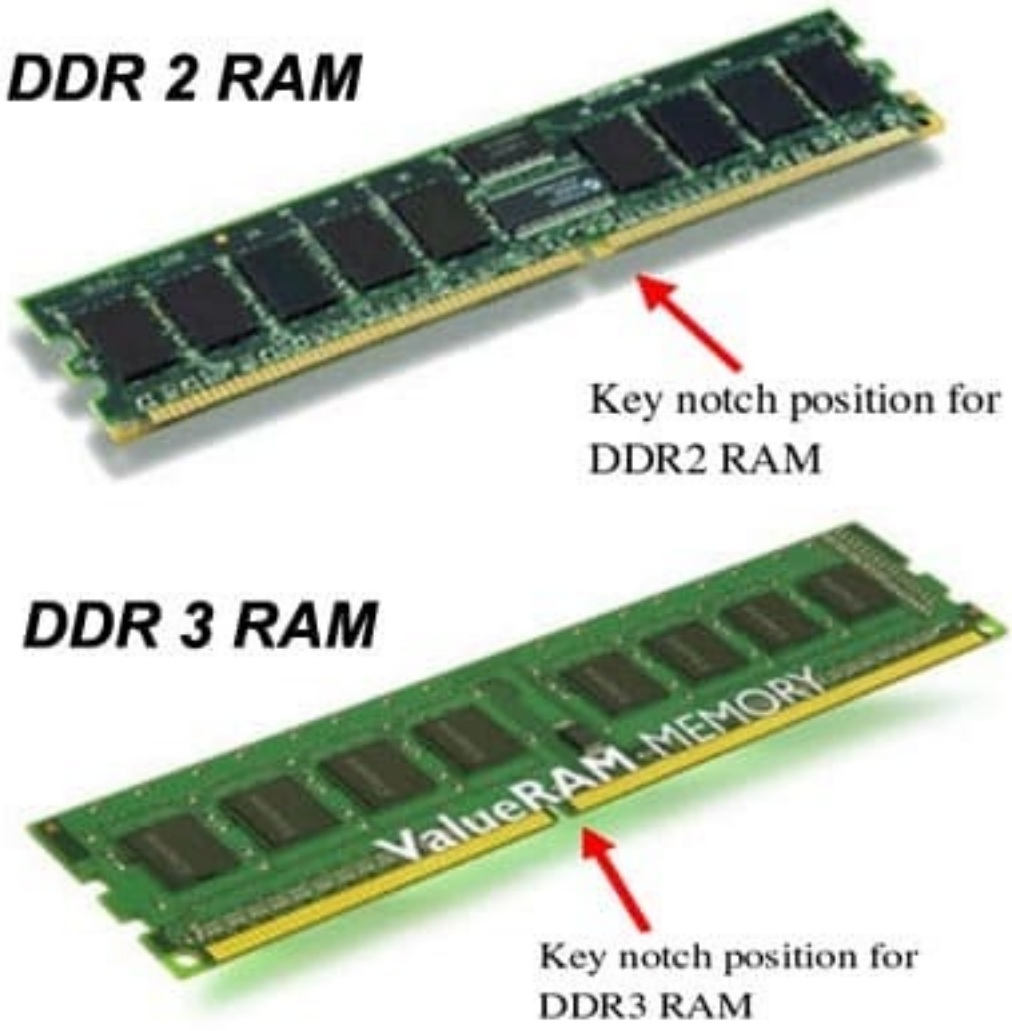
1. Prepare our data

2. Send data to CPU and
3. Ask CPU to execute operations on these data

Formally, this is a program. The program does exactly that. Prepares data and then asks CPU to execute operations on these data. In other words, a program has data and instructions.

RAM

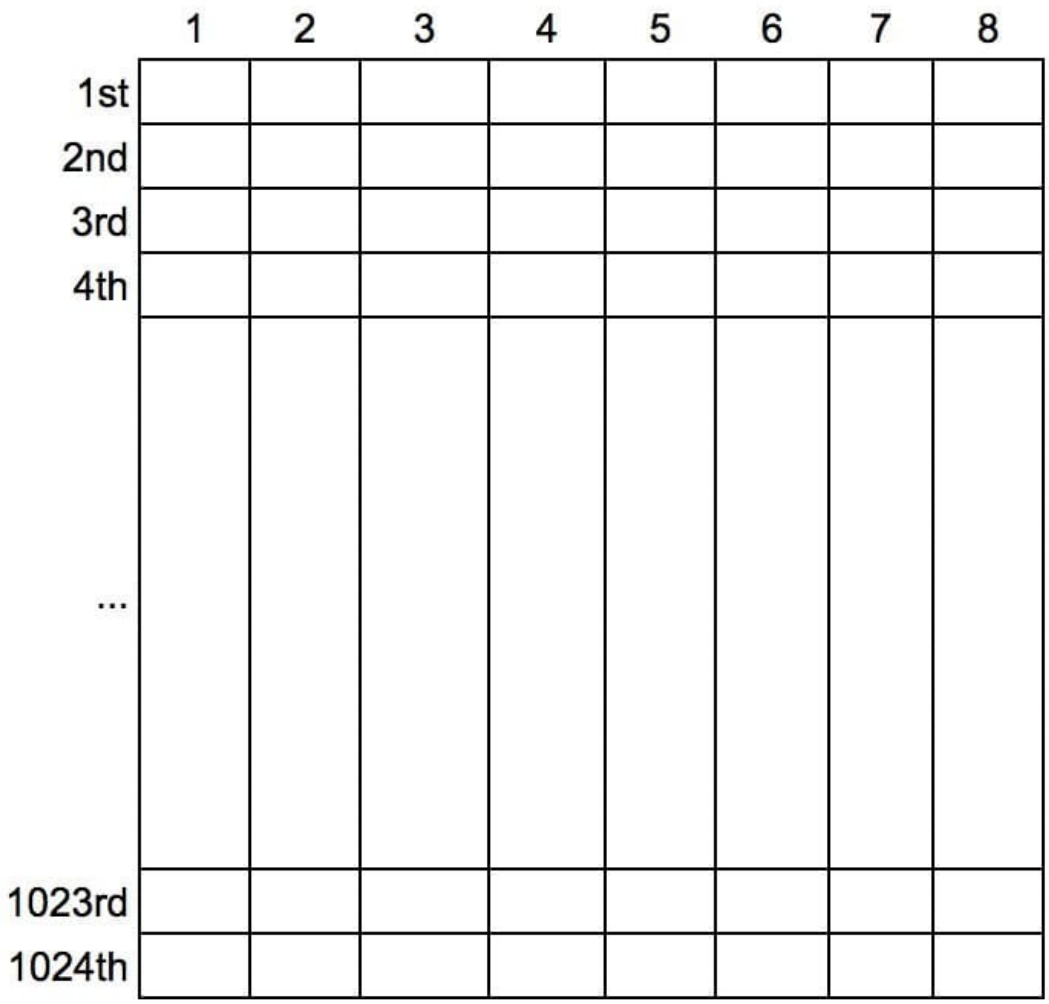
RAM, for Random Access Memory, is the electronic bank of chips that holds our program data.



RAM Banks

The RAM is divided in positions. Let's assume that each position has a size of 1 Byte. A RAM with 1024 Bytes size has 1024 different positions in order to store a piece of information. Each position is of limited size. Let's assume that it is 8 bits. 1 Byte is equal to 8 bits, and we can tell that each address position is of size 8 bits.

Look at the following picture. It shows a table of 1024 positions. The table corresponds to a RAM of size 1024Bytes. Note that each byte holds 8 bits.

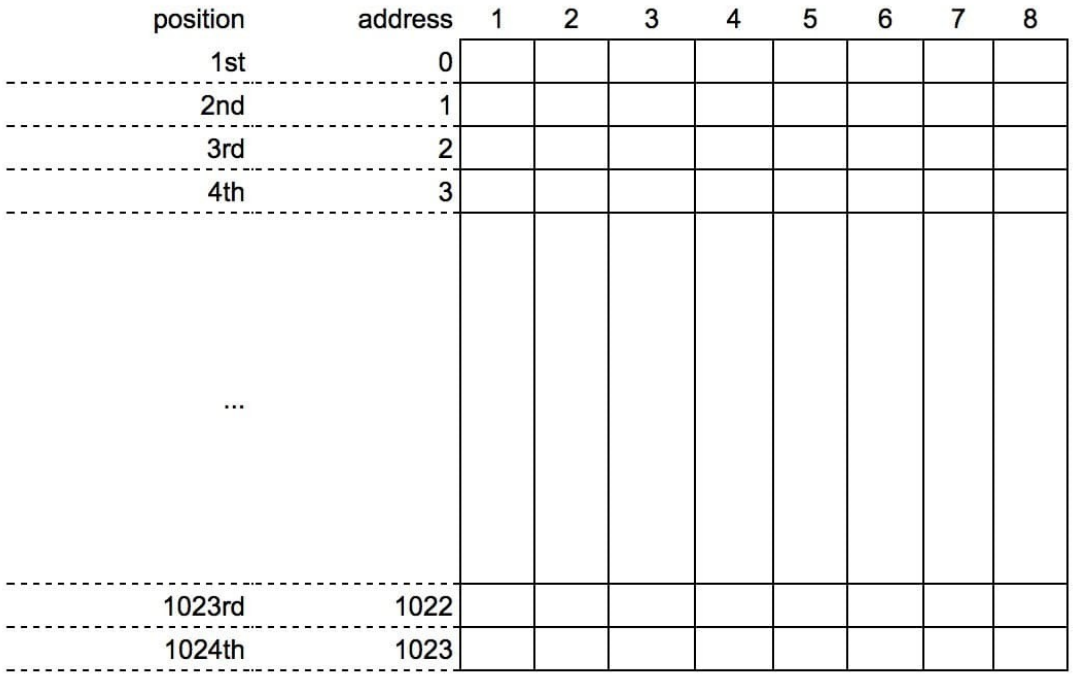


RAM of 1024 Bytes

RAM Addresses

In order for CPU to be able to carry out operations on the data, we need to have a way to tell CPU which positions of the RAM space are occupied by our data. To help out this process RAM positions are given a distinct identification number. It is the number 0 for the 1st position, the number 1 for the 2nd position, the number 2 for the third position e.t.c. Hence, the distinct identification number for the RAM 1000 position is 999, i.e. the index of the position - 1. Since these numbers are used to reference a specific position inside the available set of RAM positions, we use the term “address” to name this number.

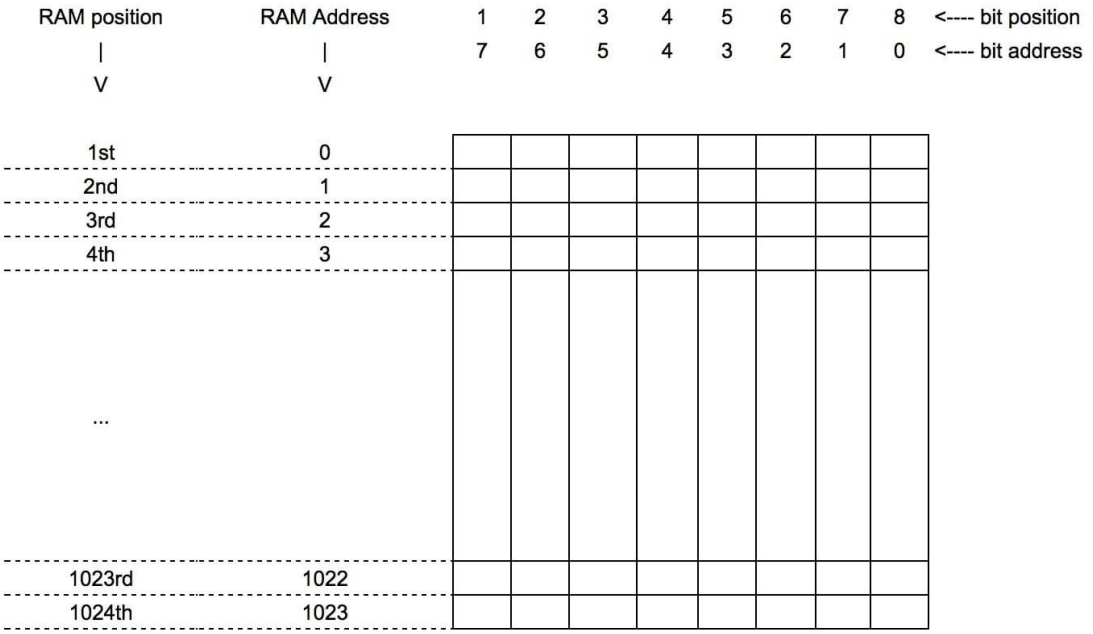
Hence, the address of the first RAM position is 0, the address of the second RAM position is 1, the address of the third RAM position is 2, e.t.c.



RAM Addresses

From now on, we will be refering to RAM positions by their address number.

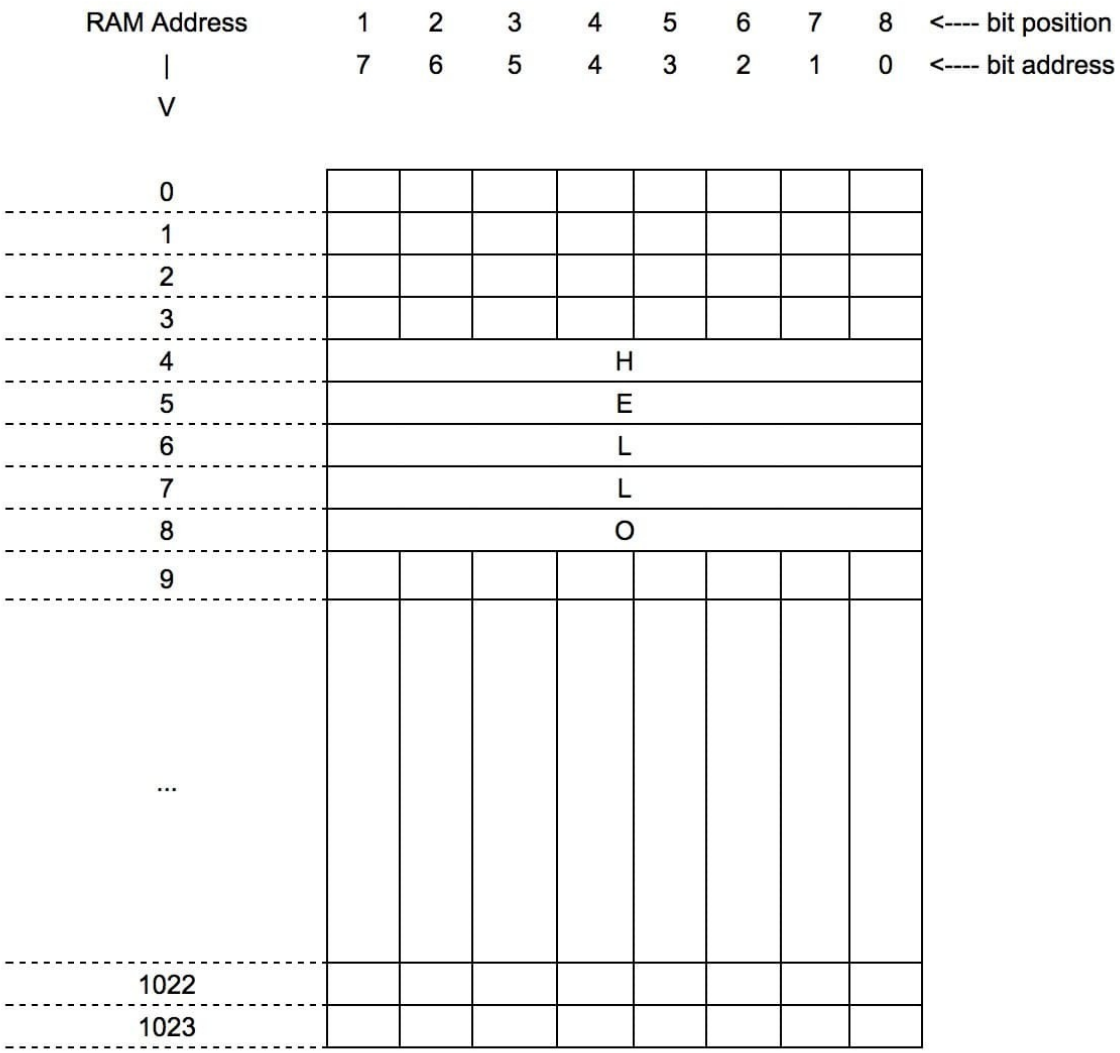
Besides numbering the RAM positions with an integer number starting from 0, we use the same scheme to number bit positions inside a specific RAM position. We address them with an integer starting from 0, up to 7, since we have 8 bits inside each position. Moreover, usually, the addressing starts from the right hand side.



RAM Addresses and Bit Addresses

RAM content

RAM is useful because it allows us to store our data inside. In the following example, you can see that we have stored the word “HELLO”. Each letter occupies 1 RAM position. And the addresses that our data occupy are 4, 5, 6, 7, 8.



RAM holds the word Hello

The above picture shows that, for the example, the letter `H` is stored inside the RAM position `4`. However, you need to know that the actual information stored inside the RAM position is a series of zeros (`0`) and ones (`1`). In other words, the computer knows how to convert the `H` letter into a very specific sequence of `0`s and `1`s and store that inside the RAM position.

RAM Address	1	2	3	4	5	6	7	8	<---- bit position
|	7	6	5	4	3	2	1	0	<---- bit address
V									
0									
1									
2									
3									
4	0	1	0	0	1	0	0	0	H
5	0	1	0	0	0	1	0	1	E
6	0	1	0	0	1	1	0	0	L
7	0	1	0	0	1	1	0	0	L
8	0	1	0	0	1	1	1	1	O
9									
...									
1022									
1023									

RAM holds word HELLO as 0s and 1s

We will not delve into how characters are being converted to 0s and 1s. For the interested reader, take a look at the ASCII table.

In summary,

- RAM is the area we store our data.
- Each position in RAM is referenced by its address.
- Each position holds a series of 0s and 1s that correspond to some actual real piece of information.

Note: Converting a piece of information to a series of 0s and 1s is a process which is called encoding. Turning 0s and 1s back to the useful piece of information is a process which is called decoding.

Numbering Systems

We all know how to count and how to represent quantities with numbers. On a daily basis and for any common calculations, humans, are using numbers that are composed by the digits `0, 1, 2, 3, 4, 5, 6, 7, 8, 9`. These digits are part of the numbering system that we use, and in particular of the `decimal` numbering system, or the numbering system which has as base the number `10`.

However, in Computer Science, the most popular numbering system is the `binary` numbering system. Also, computer programming, besides the `binary` numbering system, usually deals with `octal` and with `hexadecimal` numbering system.

Hence, the popular numbering systems in the world of computers and programming are:

1. `binary`, which uses 2 digits: `0` and `1`
2. `octal`, which uses 8 digits: `0`, `1`, `2`, `3`, `4`, `5`, `6`, `7`
3. `decimal`, which uses 10 digits: `0`, `1`, `2`, `3`, `4`, `5`, `6`, `7`, `8`, `9`
4. `hexadecimal`, which uses 16 digits: `0`, `1`, `2`, `3`, `4`, `5`, `6`, `7`, `8`, `9`, `A`, `B`, `C`, `E`, `F`

We will not spend more on numbering systems in this chapter. We will do it in a later chapter.

> *Information:* The decimal numbering system became so popular in the human world, because it is based on the number 10, which equals the number of fingers that humans have on their hands. Humans were always using their fingers to count small quantities and execute simple calculations.

Assembly

At the outset of the computer programming era, developers had to use a very difficult programming language, which was called `assembly`. Actually, assembly is still needed in some cases, even today.

The following picture is how an assembly program looks like.

```
; Example of IBM PC assembly language
; Accepts a number in register AX;
; subtracts 32 if it is in the range 97-122;
; otherwise leaves it unchanged.

SUB32  PROC          ; procedure begins here
       CMP  AX,97    ; compare AX to 97
       JL   DONE     ; if less, jump to DONE
       CMP  AX,122   ; compare AX to 122
       JG   DONE     ; if greater, jump to DONE
       SUB  AX,32    ; subtract 32 from AX
DONE:  RET           ; return to main program
SUB32  ENDP          ; procedure ends here
```

FIGURE 17. Assembly language

Program Written in Assembly

Actually, this is not a full program. It is a very tiny program unit, called procedure that takes some input and does an action on it.

1. Takes as input a number
2. If the number is less than 97 or greater than 122 it does nothing.
3. If the number is in the range 97 - 122, subtracts the number 32 from this input number.

> *Side Note:* The input is stored inside the register `AX`. The content of `AX` is then compared to the number 97 (`CMP AX,97`). If it is less (`JL` - `J`ump if `L`ess), control flow jumps to `DONE` point where it returns (`RET`), which means that it essentially finishes. If it is not less, the content of `AX` is then compared to the number 122 (`CMP AX,122`). If it is greater (`JG` - `J`ump if `G`reater), control flow jumps to `DONE`. If it is not greater, we subtract 32 (`SUB AX,32`) and the procedure returns having stored in `AX` the result of the subtraction.
>
> You do not have to understand assembly or being able to write assembly in order to become a Web developer. That was only an example to get an idea how assembly looks like.

As you can see, the program is difficult to read and it is also very verbose, even if does a very simple thing.

High Level Programming Languages

Nowadays, most of the developers are using high level programming languages.

In the following picture, you can see the same comparison procedure, that we saw earlier return in assembly language, now written in a high level programming language:

```
def compare(number)
  return if number < 97 || number > 122
  number - 32
end
```

```
; Example of IBM PC assembly language
; Accepts a number in register AX;
; subtracts 32 if it is in the range 97-122;
; otherwise leaves it unchanged.

SUB32  PROC         ; procedure begins here
       CMP  AX,97   ; compare AX to 97
       JL   DONE    ; if less, jump to DONE
       CMP  AX,122  ; compare AX to 122
       JG   DONE    ; if greater, jump to DONE
       SUB  AX,32   ; subtract 32 from AX
DONE:  RET          ; return to main program
SUB32  ENDP         ; procedure ends here
```

Compare High Level Programming Language to Assembly Equivalent

As you can see, the high level programming language version is much easier to read:

1. `return if number < 97 || number > 122`. It reads: return if number given is less than 97 or greater than 122.
2. Otherwise the command `number - 32` is executed.

In this course we are going to learn 2 high level programming languages:

1. JavaScript
2. Ruby

Statically Typed Compiled Languages

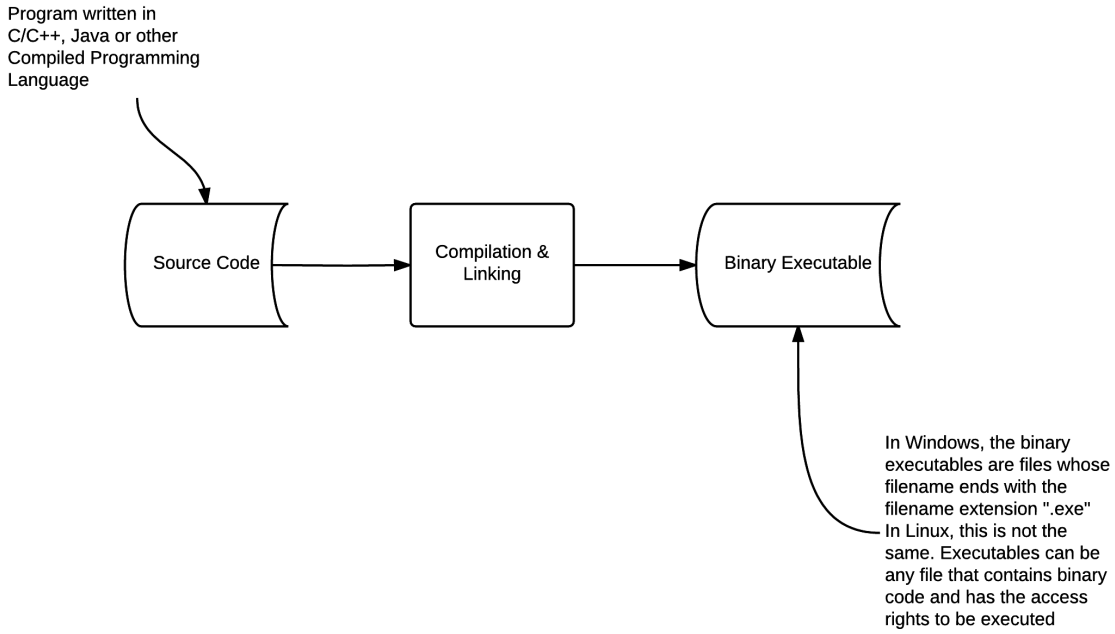
There is a big family of programming languages that includes the statically typed compiled languages. For example:

1. C
2. C++
3. Java

They are `statically typed` because the developer needs to declare the type of the data a RAM position is allowed to hold. For example, whether the position is going to hold a number or a

character. Don't worry if you that does n't make too much sense to you right now. We will explain that later.

Also, they are `compiled` languages because the program that a developer writes (called `source code`) goes through a process, which is called compilation and linking, and which generates the final program deliverable, in binary machine executable format, that is shipped or deployed to the computer machine that will execute it. So, the actual source code is never delivered and it is not necessary for the program to run. Developer keeps the source code on its development machine (or other backup storage), and usually is kept private.



Compilation & Linking Process

Advantages

Statically typed and compiled languages have a lot of advantages:

1. They produce programs that are very fast and optimized.
2. They catch a lot of programming errors at compilation time. So, they protect the developer from introducing bugs at run-time. However, this is true only for the programming errors that can be detected by the compilation and linking phase. Not errors and bugs that have to do with program logic.
3. The source code is more descriptive with regards to the signature of the methods, functions and procedures.
4. They protect the source code from becoming open and public, since source code does not need to be delivered / deployed on the computer machine.

Disadvantages

As always, there are some disadvantages too:

1. The programs are quite verbose. Even for simple things, might need to write a lot of code.
2. Requires that developers know in advance about the requirements of the type of the data. Sometimes this is not feasible and definitely is restrictive when requirements change.

Example source code

Below, you can see same comparison program that we start with this chapter, written in Java:

```
int compare(int number) {
  if (number < 97 || number > 122) {
    return;
  }
  else {
    return number - 32;
  }
}
```

(the above code snippet online)

As you can see above, we need to tell that our `number` variable is going to be an integer (see: `int`).

Dynamically Typed Script Languages

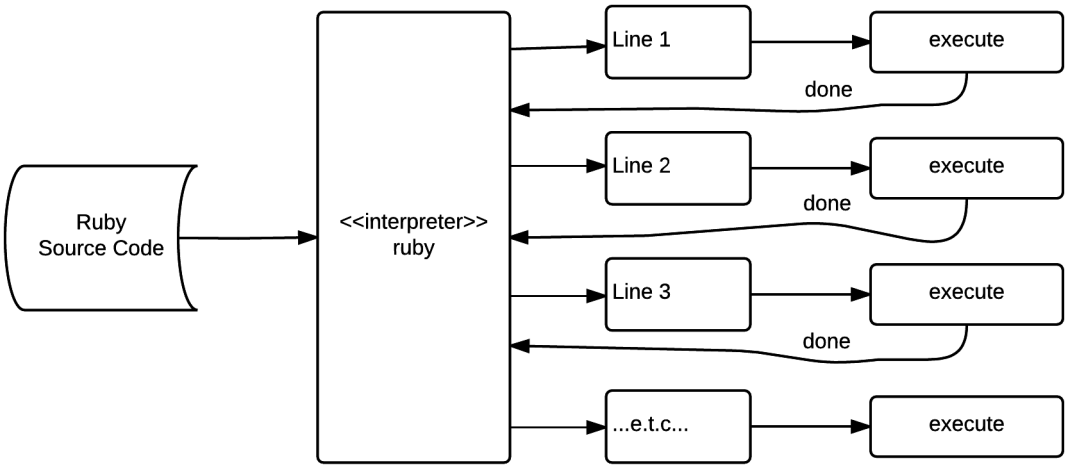
Another very popular family of programming languages is the dynamically typed script languages. Like:

1. Python
2. Ruby
3. JavaScript
4. PHP

They are dynamically typed because the developer does not have to specify the type of the RAM positions that will be holding program data.

They are script languages because they require another program, the interpreter, that would take the source code line-by-line, and execute each statement, one-by-one.

When we deploy a program written using a script language, we have to deploy the source code as well and make sure that the correct interpreter is already installed on the machine we are going to do the deployment. The source code does not go through any compilation & linking phase. It is immediately fed to the language interpreter which parses the program line-by-line. For each line, it converts it to machine executable code and executes it, before doing the same for the next line.



Interpreting a Ruby Program

Advantages

Dynamically typed script languages have a lot of advantages:

1. They do not require the developer to specify the type of the RAM positions in advance. The language interpreter itself identifies internally what is the correct type of data to attach to each value stored in a RAM position.
2. You can easily read the source code and change the program to fix any bugs or enhance it with new features. And in extreme cases, you can even do that on the computer that runs the application.
3. The source code size is much smaller than the one written with a statically typed compiled language. This means that for the same behaviour the developer writes much less code.
4. They are very easy and handy to quickly build a prototype.
5. They are usually open source and have huge community support for free.

Disadvantages

But still, there are some disadvantages too:

1. The programs are generally much slower than the programs generated by the compiled programming languages.
2. The developers need to disclose their source code. This is because the actual source code needs to be deployed on the computer machine the application is going to run.
3. It is more difficult to build programmer-friendly IDEs (Integrated Development Environments like RubyMine).
4. The fact that there is no type checking at development time, might hide errors that will only appear while the program will run.

Example source code

Below, you can see same comparison program that we start with this chapter, written in Ruby:

```ruby
def compare(number)
  return if number < 97 || number > 122
  number - 32
end
```

(the above code snippet online)

If you compare that version to the one with Java language you get a first impression of what writing less mean.

Tasks and Quizzes

> **Before you continue, you may want to know that**: You can sign up to Tech Career Booster and have a mentor evaluate your tasks, your quizzes and, generally, your progress in becoming a Web Developer. Or you can sign up and get access to Tech Career Booster Slack channel. In that channel, there are a lot of people that can answer your questions and give you valuable feedback.

Quiz

The quiz for this chapter can be found here

2 - Numbering Systems

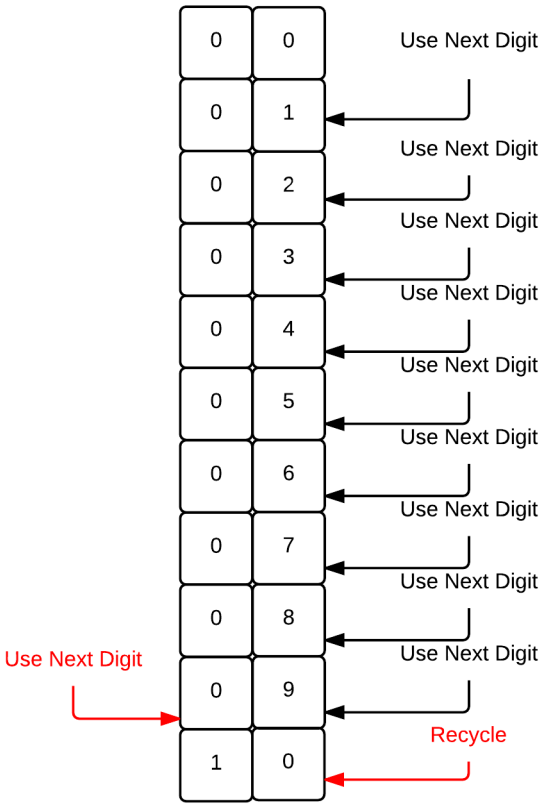
Summary

In this chapter we take a deeper dive into numbering systems. We study the

- Decimal,
- Binary,
- Octal, and
- Hexadecimal

numbering systems.

We explain in detail how the number of available digits is calculated. Also, we explain the logic behind counting and how we can use an algorithm to count on any numbering system.



Counting

We also explain how you can quickly draw the table with the first 16 numbers of the binary system, and the pattern behind it.

1	0	0	0	0
2	0	0	0	1
3	0	0	1	0
4	0	0	1	1
5	0	1	0	0
6	0	1	0	1
7	0	1	1	0
8	0	1	1	1
9	1	0	0	0
10	1	0	0	1
11	1	0	1	0
12	1	0	1	1
13	1	1	0	0
14	1	1	0	1
15	1	1	1	0
16	1	1	1	1

First 16 Binary Numbers

Finally, we teach you why the number `10` is not always the number ten.

Learning Goals

1. Learn which are the most popular numbering systems in the computer world.
2. Learn about the `base` of a numbering system.
3. Learn why the decimal numbering system is the most popular numbering system in the world.
4. Learn how to find out the maximum digit that can be used in a specific number system, based on its base.
5. Learn about the digits of the decimal numbering system.
6. Learn about the binary numbering system, its base and its digits.
7. Learn about the hexadecimal numbering system, its base and its digits.
8. Learn the logic behind counting and learn to count on all popular numbering systems.
9. Learn to build the table with first 16 binary numbers quickly.

Introduction

In this chapter, we are going to elaborate more on Numbering Systems.

The most popular numbering system in the computer world are:

- Decimal
- Binary
- Octal
- Hexadecimal

Decimal Numbering System

All numbering system has a **base**, which equals to the number of digits that are used to build up numbers in this system.

The decimal numbering system has base equal to 10. This means that has 10 digits. It is very popular in the human world, because its base is equal to the number of fingers that humans have. Fingers have always been used to carry out calculations.

When we know the base of a numbering system, we can immediately tell which digits are part of this particular numbering system. It is all the digits that start from 0 up to the digit that is equal to base minus 1 (base - 1).

Hence, the decimal numbering system, which has base equal to 10, has all the digits from `0` up to `9`, since `9` is the result of `base - 1`, a.k.a. `10 - 1`.

Decimal Numbering System Digits: `0`, `1`, `2`, `3`, `4`, `5`, `6`, `7`, `8`, `9`.

Binary Numbering System

Binary is very popular numbering system in the digital world, the world of computers and digital devices.

Binary numbering system base is 2. This means that it has 2 digits: `0`, `1` (since `1` is the result of `base - 1`, a.k.a. `2 - 1`).

Octal Numbering System

There are cases in which the octal numbering system is useful in the computers and programming world.

Octal numbering system has base equal to 8. This means that it has 8 digits.

Octal Numbering System Digits: `0`, `1`, `2`, `3`, `4`, `5`, `6`, `7`.

We start again from `0` and we go up to the `base - 1`, a.k.a. `8 - 1`, `7`.

Hexadecimal Numbering System

This is another popular numbering system in the programming world. It has base 16. This means that it has 16 digits (`base -1`).

First digit is `0`, second digit is `1`, and so on up until 10th digit which is `9`. After that, the 11th digit is represented with the letter `A`, the 12th digit with the letter `B`, the 13th digit with the letter `C`, the 14th digit with the letter `D`, the 15th digit with the letter `E` and the 16th digit with the letter `F`. We decided to use the first letters for the English alphabet for the 11th digit and above (for the digit that represents the number 10 essentially and above) because we wanted a single symbol for number. This is the full list of digits again:

Digit Index	Decimal Equivalent	Hex Digit
1	0	0
2	1	1
3	2	2
4	3	3
5	4	4
6	5	5
7	6	6
8	7	7
9	8	8
10	9	9
11	10	A
12	11	B
13	12	C
14	13	D
15	14	E
16	15	F

Hexadecimal Digits

As you can see we start from `0` and we end up to `15` decimal which we decide to symbolize with `F`.

Counting

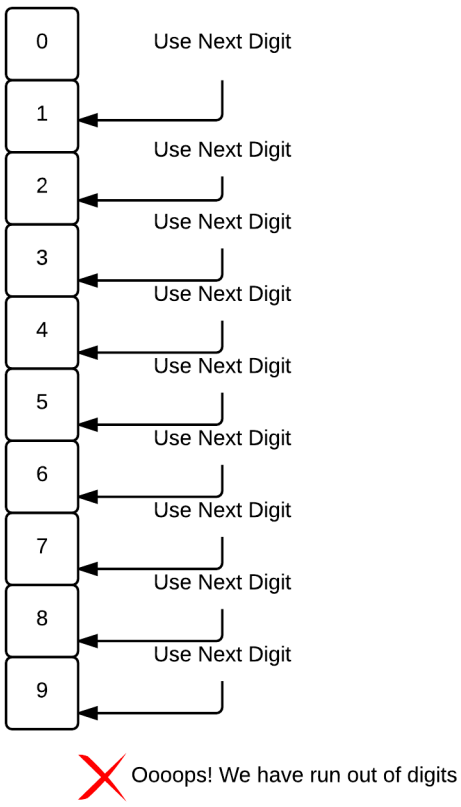
We all know how to count using the decimal numbering system. But how do we count using other numbering systems?

We count using the same logic.

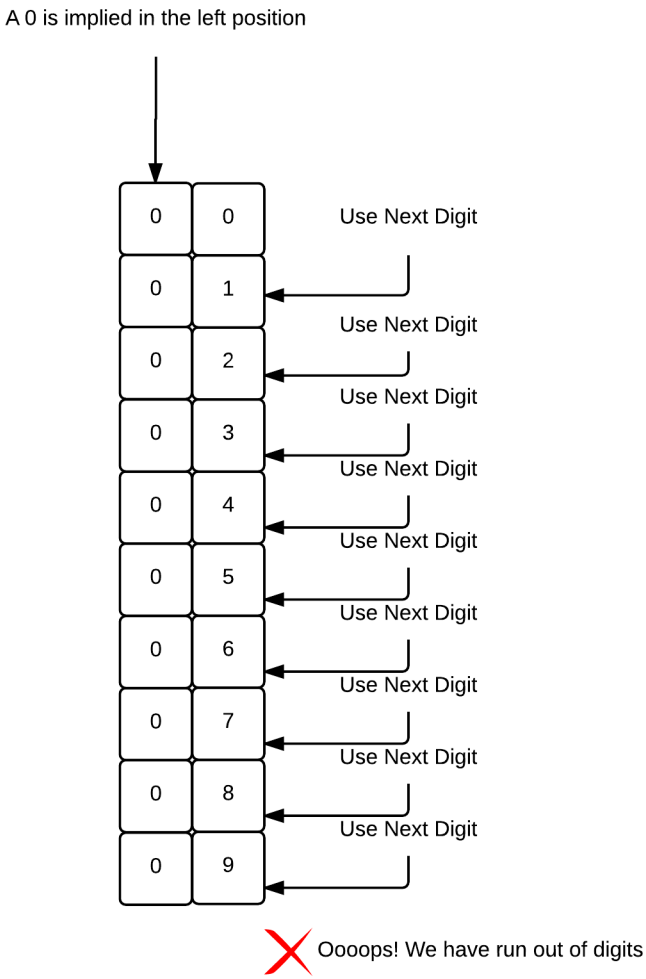
Counting on Decimal Numbering System

Let's refresh our memory about how we count using the decimal numbering system.

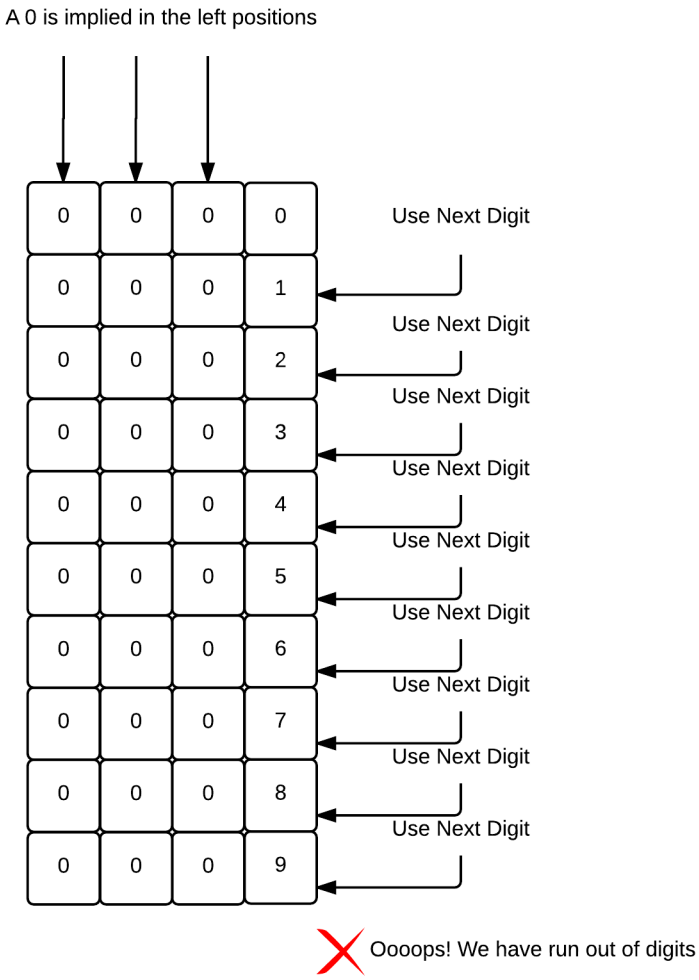
1. We start from 0.
2. We then use the next digit. Which is 1.
3. We then use the next digit. Which is 2.
4. We then use the next digit. Which is 3.
5. We then use the next digit. Which is 4.
6. We then use the next digit. Which is 5.
7. We then use the next digit. Which is 6.
8. We then use the next digit. Which is 7.
9. We then use the next digit. Which is 8.
10. We then use the next digit. Which is 9.
11. We then use the next digit. Ooooop!. There is no next digit to use. In decimal numbering system, the last, maximum digit, is 9.



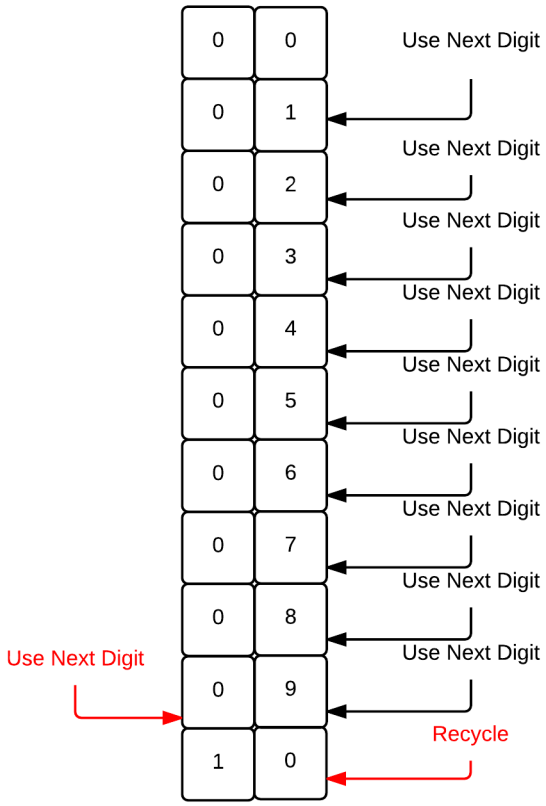
12. Now that we have run out of digits on the current position, we recycle the right most digit and we use the next digit for its left position digit. Note that when we work to increase the digit on one position, the left position may be blank, but in fact, it holds the digit `0`. So, we have to increase it by `1`. From `0` to `1`. We will try to explain that using the following pictures. Firstly, you see that the digit `0` is implied on the left-to-the-current position.



Actually, all the left positions are supposed to have `0`. Think about it in another way. `2` is equal to `02` and to `002` and to `0002` and so on.

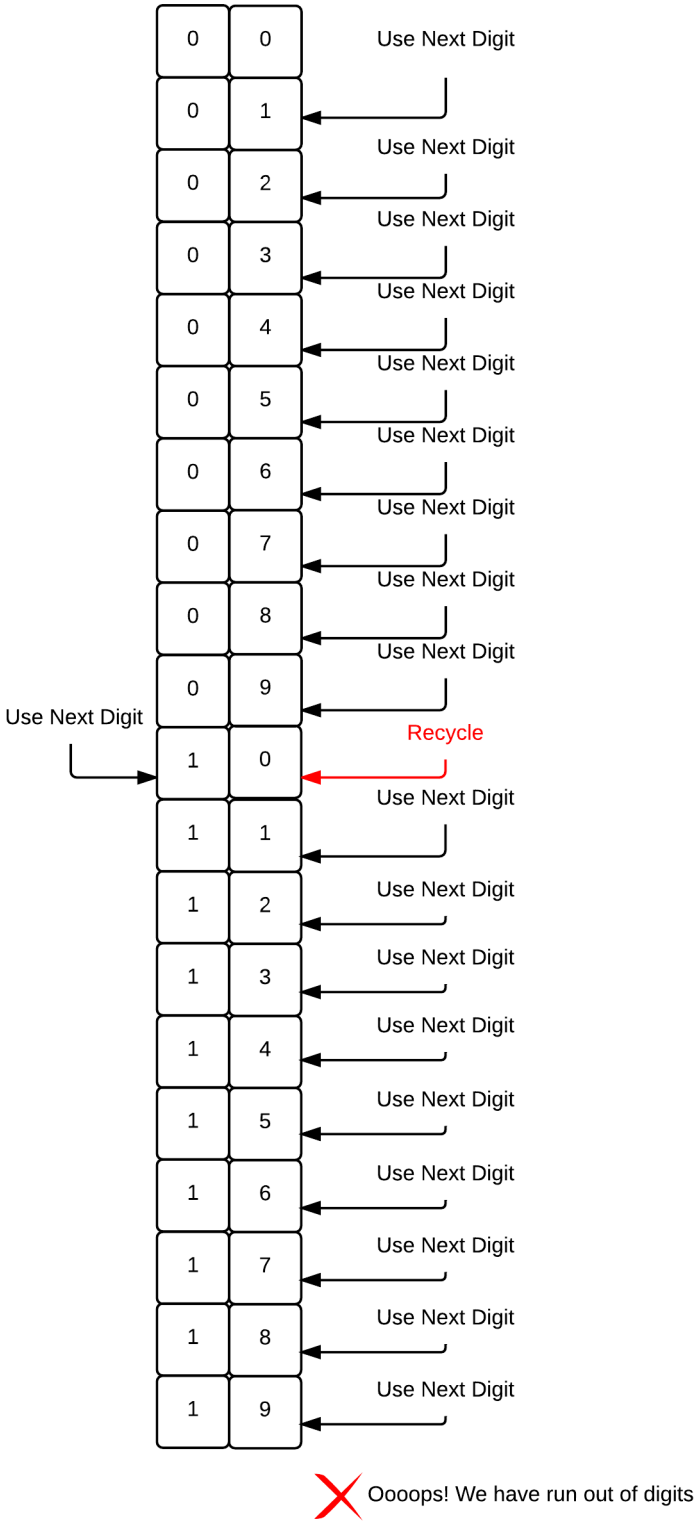


Now that we have run out of digits on the current column, we need to start using the left column. What we do, is that we recycle the current column back to `0` and we use the next digit available from the left column. Hence, we have the number `10`.

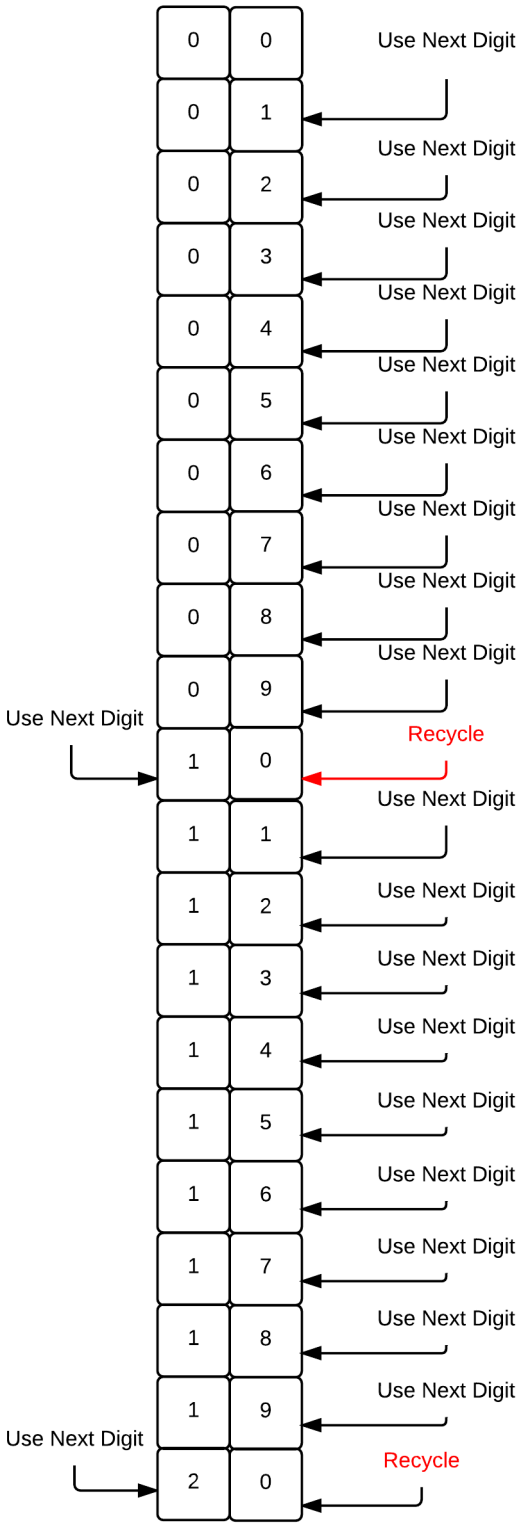


13. We then use the next digit. Which gives `11`.
14. We then use the next digit. Which gives `12`.
15. ...and so on...until `19`.

16. At this point, we again run out of digits on the right most position.



17. But we deal with that as we did earlier. We will recycle the right position back to `0` and we will take the next digit on the left position, a.k.a, from `1` to `2`, having `20` as result.



18. We then take the next digit available on the right position. We have `21`.
19. We then take the next digit available on the right position. We have `22`.
20. And the same goes on until `29`.
21. On `29` we have run out of digits on right position. We do what we did in similar cases earlier. We recycle `9` back to `0` and we take the next digit on the left position. Hence we have `30`.
22. We continue with `31`.

23. Then `32`.
24. We follow the same "take next digit and, if run out of digits, recycle" approach up until `99`. In that case we have run out of digits on both positions. We recycle both of them to `0` and we take the next digit on one more position to the left. Hence we have `100`.
25. Then `101`. E.t.c.

It seems that we have found the pattern on how we count from `0` up to whatever number in the decimal numbering system. This pattern is enough to tell us which is the number that follows this:

```
1838472874091238099999
```

(the above code snippet online)

Spend some time to do this exercise on your own. You should be able to use the above pattern and find out that the next number is:

```
1838472874091238100000
```

(the above code snippet online)

Counting on Binary Numbering System

The way we count on binary numbering system is exactly the same like we count on decimal numbering system. We use the same pattern, the same algorithm.

1. We start from 0.
2. We take the next digit. We have 1.
3. We have run out of digits. Remember that binary numbering system only has 2 digits. 0 and 1. Hence, we recycle right position to `0` and we take the next digit on the left position. Hence, we get the number 10.
4. We take the next digit on the right most position. We have 11.
5. We have run out of digits on the right most position. We have also run out of digits on the left position too. So, we recycle them both to `0` and we take the next digit on the left most position, the third position. Hence we have the number 100.
6. We take the next digit on the right most position. We have 101.
7. We have run out of digits and we recycle to `0` taking the next available digit on the left. Hence, we have 110.
8. We take the next digit on the right most position. We have 111.

The next table displays the first 17 binary numbers, counting from `0`.

Decimal	Binary				
0					0
1					1
2				1	0
3				1	1
4			1	0	0
5			1	0	1
6			1	1	0
7			1	1	1
8		1	0	0	0
9		1	0	0	1
10		1	0	1	0
11		1	0	1	1
12		1	1	0	0
13		1	1	0	1
14		1	1	1	0
15		1	1	1	1
16	1	0	0	0	0

First 17 Binary Numbers - Comparison to their Decimal Equivalent

Usually, when we display a table of binary numbers, we fill in the left positions with 0s, when 0 is implied. Here is the table of first 16 binary numbers filled in that way.

Decimal	Binary			
0	0	0	0	0
1	0	0	0	1
2	0	0	1	0
3	0	0	1	1
4	0	1	0	0
5	0	1	0	1
6	0	1	1	0
7	0	1	1	1
8	1	0	0	0
9	1	0	0	1
10	1	0	1	0
11	1	0	1	1
12	1	1	0	0
13	1	1	0	1
14	1	1	1	0
15	1	1	1	1

First 16 Binary Numbers With Leading 0s Printed

If you look more carefully on how this table is constructed, you will see the following pattern:

1. The first column is composed of consecutive `01` chunks.

2. The second column is composed of consecutive `0011` chunks.
3. The third column is composed of consecutive `00001111` chunks.
4. The fourth column is composed of consecutive `0000000011111111` chunks.

This is a nice mnemonic trick that would allow you to either build this table quickly or even then verify its correctness. Watch the following video to see how we construct this table very quickly using this technique.

How title tag affects the name of the tab of the page

And here is a colour map that highlights again this pattern.

1	0	0	0	0
2	0	0	0	1
3	0	0	1	0
4	0	0	1	1
5	0	1	0	0
6	0	1	0	1
7	0	1	1	0
8	0	1	1	1
9	1	0	0	0
10	1	0	0	1
11	1	0	1	0
12	1	0	1	1
13	1	1	0	0
14	1	1	0	1
15	1	1	1	0
16	1	1	1	1

First 16 Binary Numbers And The Patterns on the Table With Colors

That was a good start on understanding how we count in binary system. And of course, we now know what is the next number of a given binary number. For example, try to find the next number of the following binary number:

```
1010010001001001111
```

(the above code snippet online)

Think about that the way we did that earlier. Try to find whether you can increase the right most digit. If not, then recycle it to `0` and increase the left one. If you follow the pattern and the logic to increase a binary number, you will see that the next one is:

```
1 1010010001001010000
```

(the above code snippet online)

Counting on Octal Numbering System

Octal numbering system has `8` digits, starting from `0` up to `7`. Counting follows the same logic. Spend some time to study the following table. It has the first `16` numbers in the octal numbering system. Practice yourself to be in position to quickly draw this table. The counting follows the same pattern as we did for decimal and binary numbering system.

Decimal	Octal
0	0 0
1	0 1
2	0 2
3	0 3
4	0 4
5	0 5
6	0 6
7	0 7
8	1 0
9	1 1
10	1 2
11	1 3
12	1 4
13	1 5
14	1 6
15	1 7

First 16 Octal Numbers

Counting on Hexadecimal Numbering System

Hexadecimal numbering system has 16 digits. `0`, `1`,..., `9`, `A`, `B`, `C`, `D`, `E`, `F`. We follow the same pattern to count from `0` and increase one by one. Here is the table with the first 64 hexadecimal numbers:

Decimal	Hexadecimal		Decimal	Hexadecimal	
0	0	0	32	2	0
1	0	1	33	2	1
2	0	2	34	2	2
3	0	3	35	2	3
4	0	4	36	2	4
5	0	5	37	2	5
6	0	6	38	2	6
7	0	7	39	2	7
8	0	8	40	2	8
9	0	9	41	2	9
10	0	A	42	2	A
11	0	B	43	2	B
12	0	C	44	2	C
13	0	D	45	2	D
14	0	E	46	2	E
15	0	F	47	2	F
16	1	0	48	3	0
17	1	1	49	3	1
18	1	2	50	3	2
19	1	3	51	3	3
20	1	4	52	3	4
21	1	5	53	3	5
22	1	6	54	3	6
23	1	7	55	3	7
24	1	8	56	3	8
25	1	9	57	3	9
26	1	A	58	3	A
27	1	B	59	3	B
28	1	C	60	3	C
29	1	D	61	3	D
30	1	E	62	3	E
31	1	F	63	3	F

First 64 Hexadecimal Numbers

Pay attention how the next number following `29`, for example, is not `30`, but, instead, it is `2A`, since, being on `29` we have not run out of digits yet. After `9` we still have `A`.

Practice with the following. What is the next number following this?

```
38A3892BD482DF
```

(the above code snippet online)

If you think about that and you understand what we have done so far, the next number is:

```
38A3892BD482E0
```

(the above code snippet online)

Closing

It is not difficult to count in non-decimal numbering systems, as long as you understand what is the logic behind. Besides, that, you now know that if somebody asks you which number is this?

```
10
```

(the above code snippet online)

you should be clever enough to tell: “it depends on the numbering system”

Because, according to the numbering system, this number is different.

1. On decimal numbering system, this number is the number ten.
2. On binary numbering system, this number is the number two.
3. On octal numbering system, this number is the number eight.
4. On hexadecimal numbering system, this number is the number sixteen.

Tasks and Quizzes

> **Before you continue, you may want to know that**: You can sign up to Tech Career Booster and have a mentor evaluate your tasks, your quizzes and, generally, your progress in becoming a Web Developer. Or you can sign up and get access to Tech Career Booster Slack channel. In that channel, there are a lot of people that can answer your questions and give you valuable feedback.

Quiz

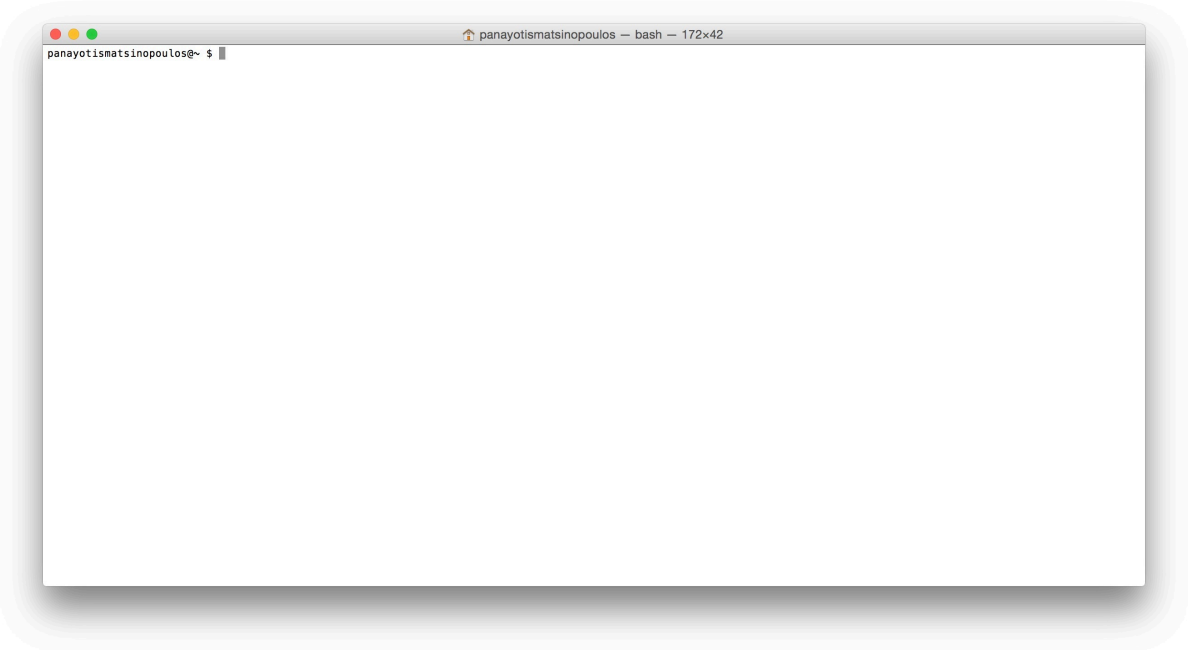
The quiz for this chapter can be found here

3 - Getting Familiar with Linux Terminal Commands

Summary

This chapter will give you some basic knowledge on using the terminal commands on Linux. This knowledge is important for every developer.

You will learn to work on the terminal:



Terminal Session On Mac

and you will be able to get a list of the contents of a folder like this:

```
panayotismatsinopoulos@~ $ ls -l
total 40
drwx------    5 panayotismatsinopoulos  staff   170 Sep  6  2015 Applications
drwx------+  15 panayotismatsinopoulos  staff   510 Apr 10 20:22 Desktop
drwx------+  85 panayotismatsinopoulos  staff  2890 Apr 10 16:38 Documents
drwx------+ 287 panayotismatsinopoulos  staff  9758 Apr 10 17:19 Downloads
drwx------@  54 panayotismatsinopoulos  staff  1836 Feb 11 08:56 Library
drwx------+  20 panayotismatsinopoulos  staff   680 Apr  9 21:37 Movies
drwx------+  21 panayotismatsinopoulos  staff   714 Mar 20 15:42 Music
drwx------+ 125 panayotismatsinopoulos  staff  4250 Apr  8 18:30 Pictures
drwxr-xr-x+   5 panayotismatsinopoulos  staff   170 Aug  6  2015 Public
drwxr-xr-x    5 panayotismatsinopoulos  staff   170 Feb 22 20:18 PycharmProjects
drwxr-xr-x    2 panayotismatsinopoulos  staff    68 Feb 22 15:09 Sites
drwxr-xr-x    3 panayotismatsinopoulos  staff   102 Feb  7 07:53 VirtualBox VMs
-rw-------    1 panayotismatsinopoulos  staff  1671 Jan 25 19:56 e-travel-root.pem
-rw-r--r--    1 panayotismatsinopoulos  staff   192 Feb 20 05:52 main.rb
```

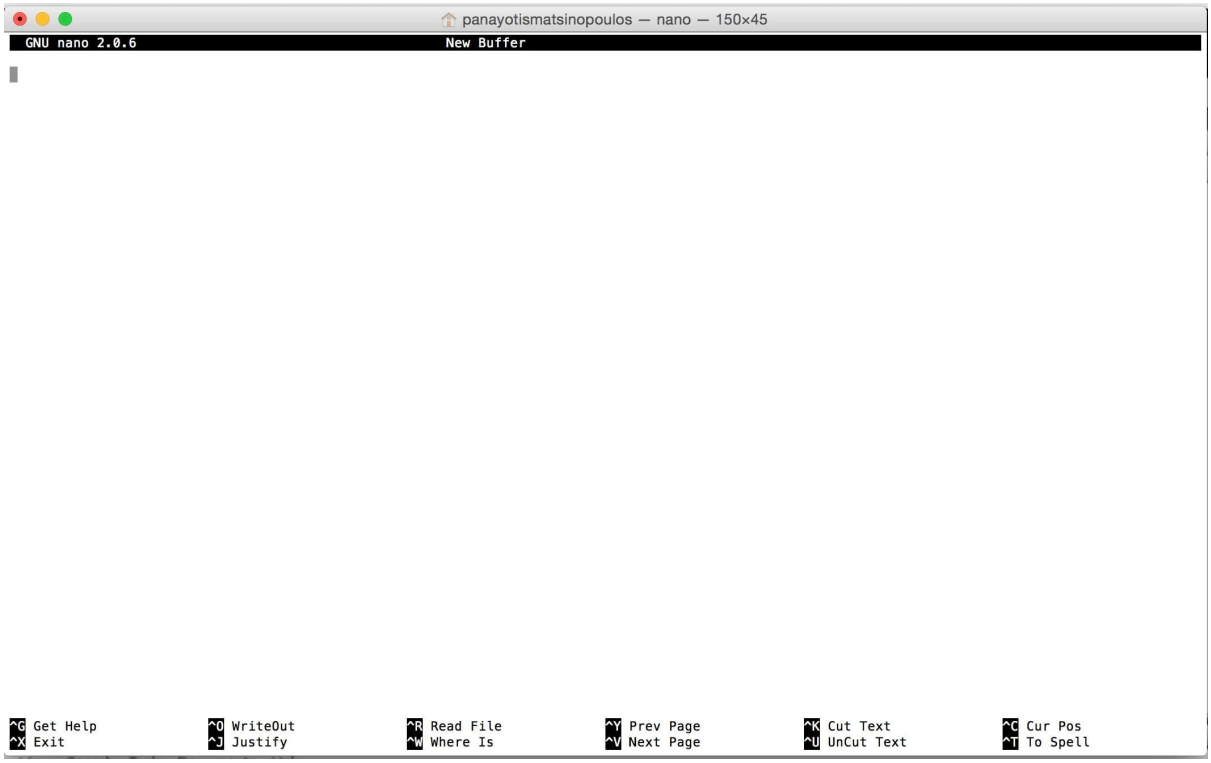
```
17 lrwxr-xr-x    1 panayotismatsinopoulos  staff    76 Sep  6  2015 openvpn -> /User\
18 s/panayotismatsinopoulos/Applications/openvpn-2.3.8/src/openvpn/openvpn
19 drwxr-xr-x@   6 panayotismatsinopoulos  staff   204 Sep  6  2015 openvpnkeys
20 -rw-r-----@   1 panayotismatsinopoulos  staff  5250 Sep  6  2015 openvpnkeys.tar.\
21 gz
22 panayotismatsinopoulos@~ $
```

(the above code snippet online)

You will also learn various commands like:

`ls`, `cd`, `mkdir`, `rm` and more.

Finally, you will learn about a common text editor in Linux and Mac, the `nano` text editor:



Nano Editor

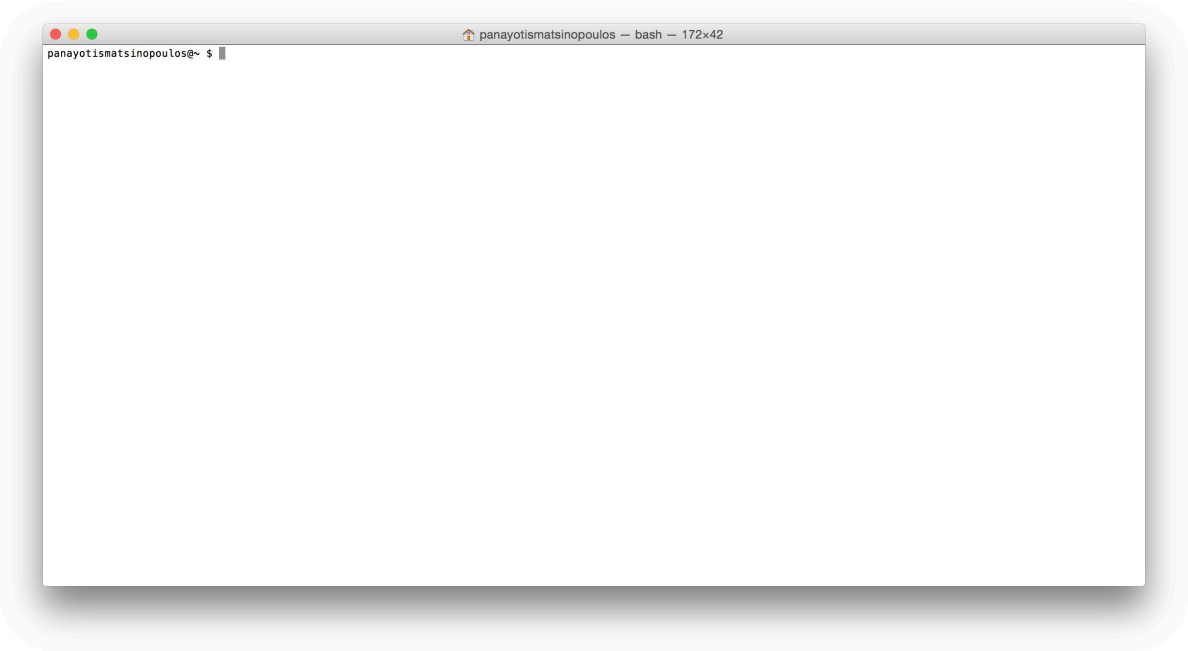
Learning Goals

1. Learn to start a terminal with an Operating System prompt to issue commands.
2. Learn which is the symbol that usually indicates a command prompt.
3. Learn about the command `pwd`.
4. Learn how the directories are constructing the folder/directory tree.
5. Learn about the parent and children directories.
6. Learn what kind of contents a directory can have.
7. Learn about the command `ls`.
8. Learn about the command `ls -l`.
9. Learn about the command `clear`.

10. Learn about the command `mkdir`.
11. Learn about the command `ls -ltr`.
12. Learn about the command `cd`.
13. Learn about the user home directory and the symbol ~.
14. Learn to navigate through the directories using the `cd` command.
15. Learn how to edit files with a basic editor like `nano`.
16. Learn about the command `cat`.
17. Learn about the command `rm`.
18. Learn about the command `rm -R`.

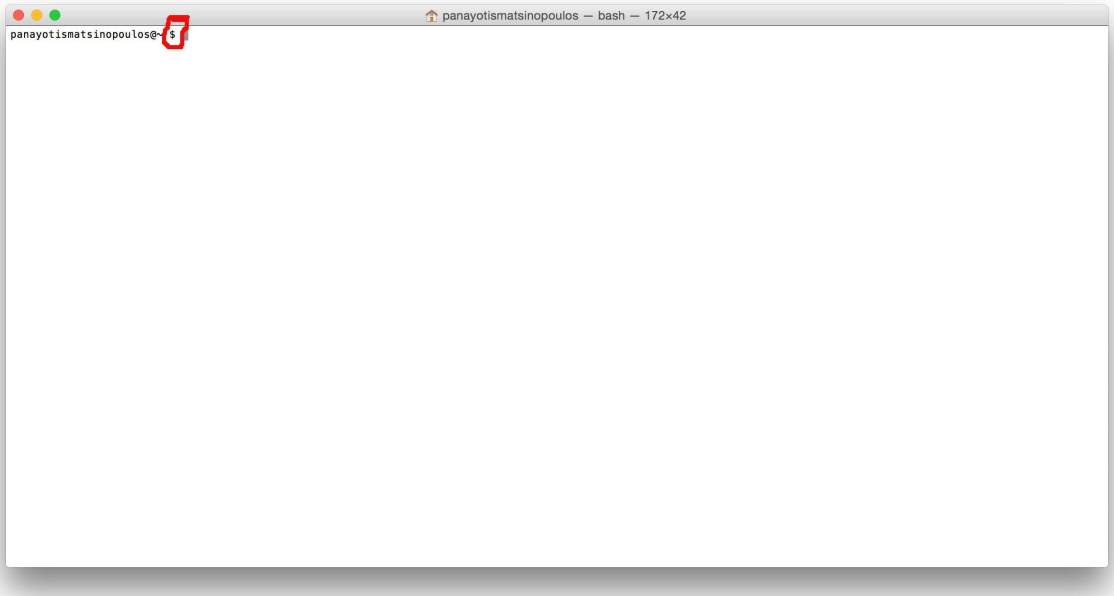
Let's start with some basic Linux commands that need to be given from the terminal. But first, open a terminal session:

Starting Terminal



Terminal Session On Mac

On the previous screenshot, you can see a new terminal session started on Mac. Find a way to start the program "Terminal". This will open a window like that and a prompt for you to type in Linux commands.



Prompt To Enter Linux Commands

Usually, the $ symbol is the prompt symbol used to prompt the user enter Linux commands.

So, the Linux operating system is waiting for the user to input a command. We give the command by typing the command and then pressing the key `Return` or `Enter`.

Some useful commands

`pwd` - Present Working Directory

The command `pwd` prints the Present Working Directory. Let's try that. Give the command:

```
1 pwd
```

(the above code snippet online)

You will get something like that:

```
1 panayotismatsinopoulos@~ $ pwd
2 /Users/panayotismatsinopoulos
3 panayotismatsinopoulos@~ $
```

(the above code snippet online)

which means that the command will return something like `/Users/panayotismatsinopoulos` and then, the operating system, will prompt you to give the next command.

This command shows that the current logged in user, the one that gives the commands, has a terminal open at the particular branch of the folders tree. The name of the current folder

is `panayotismatsinopoulos`, which is a child folder of the parent folder `Users`. The full path, `/Users/panayotismatsinopoulos` uses `/` to separate one part of the branch from the next.

A directory, like `Users`, can have many child directories. But, on the other hand, a child directory, like `panayotismatsinopoulos`, can only have `1` parent directory. This does not exclude the fact that any folder can be both a child and a parent. Only the root of the folders tree does not have any parent. The root is represented by the starting `/`.

Hence, in order to be more specific, the full path `/Users/panayotismatsinopoulos` denotes that `Users` is a child folder belonging to the root folder `/`. Also, it denotes that `panayotismatsinopoulos` is one of the children of the `Users` folder.

`ls` - List command

A directory can contain other directories or files. If we want to see the content of the present working directory, or in other words, if we want to see a *list* of the contents of the present working directory, we type in the command `ls`. Try it. Give the following command on your command prompt:

```
ls
```

(the above code snippet online)

This is what I get on my computer:

```
panayotismatsinopoulos@~ $ ls
Applications                Downloads               Music                           PycharmProjects
Desktop                        Library                         Pictures                Sites
Documents                Movies                    Public                          VirtualBox V
panayotismatsinopoulos@~ $
```

(the above code snippet online)

This is the list of the contents inside the current directory. This is a compact version of the list. You can get a long list if you use the command switch `-l`. Let's try that:

```
ls -l
```

(the above code snippet online)

On my terminal, this returns this one:

```
panayotismatsinopoulos@~ $ ls -l
total 40
drwx------    5 panayotismatsinopoulos  staff   170 Sep  6  2015 Applications
drwx------+  15 panayotismatsinopoulos  staff   510 Apr 10 20:22 Desktop
drwx------+  85 panayotismatsinopoulos  staff  2890 Apr 10 16:38 Documents
drwx------+ 287 panayotismatsinopoulos  staff  9758 Apr 10 17:19 Downloads
drwx------@  54 panayotismatsinopoulos  staff  1836 Feb 11 08:56 Library
drwx------+  20 panayotismatsinopoulos  staff   680 Apr  9 21:37 Movies
drwx------+  21 panayotismatsinopoulos  staff   714 Mar 20 15:42 Music
drwx------+ 125 panayotismatsinopoulos  staff  4250 Apr  8 18:30 Pictures
drwxr-xr-x+   5 panayotismatsinopoulos  staff   170 Aug  6  2015 Public
drwxr-xr-x    5 panayotismatsinopoulos  staff   170 Feb 22 20:18 PycharmProjects
drwxr-xr-x    2 panayotismatsinopoulos  staff    68 Feb 22 15:09 Sites
drwxr-xr-x    3 panayotismatsinopoulos  staff   102 Feb  7 07:53 VirtualBox VMs
-rw-------    1 panayotismatsinopoulos  staff  1671 Jan 25 19:56 e-travel-root.pem
-rw-r--r--    1 panayotismatsinopoulos  staff   192 Feb 20 05:52 main.rb
lrwxr-xr-x    1 panayotismatsinopoulos  staff    76 Sep  6  2015 openvpn -> /User\
s/panayotismatsinopoulos/Applications/openvpn-2.3.8/src/openvpn/openvpn
drwxr-xr-x@   6 panayotismatsinopoulos  staff   204 Sep  6  2015 openvpnkeys
-rw-r-----@   1 panayotismatsinopoulos  staff  5250 Sep  6  2015 openvpnkeys.tar.\
gz
panayotismatsinopoulos@~ $
```

(the above code snippet online)

This is very useful because it does not only print the contents, but for each item in the list it gives you a lot of extra useful information. For example, you can see a `d` as the first character in some entries. This indicates that the corresponding entry is a directory and not a file. The files have the first character being a dash `-`. You can also see the size of each entry in bytes. For example, the size of file `main.rb` in the above entry is `192` bytes.

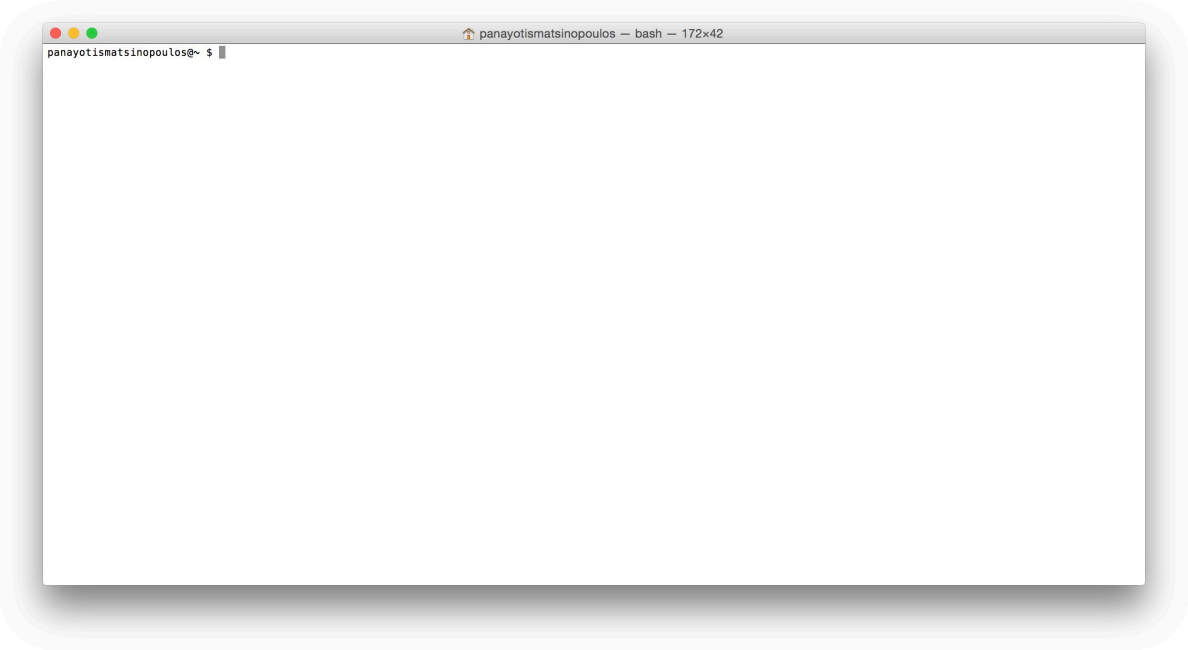
`clear`

If you want to clear the contents of the terminal, you can issue the command `clear`. Try that, give the following command on your terminal:

```
clear
```

(the above code snippet online)

You will see that the contents of the terminal disappear and you are back on top line:



Terminal Contents Have Been Cleared

`mkdir` - Make Directory

If you want to create a new directory inside the current working directory, you need to use the command `mkdir`, which stands for `MaKe DIRectory`. Let's try that. Give the following command on your command prompt:

```
1 mkdir introduction
```

(the above code snippet online)

You will get back something like that:

```
1 panayotismatsinopoulos@~ $ mkdir introduction
2 panayotismatsinopoulos@~ $
```

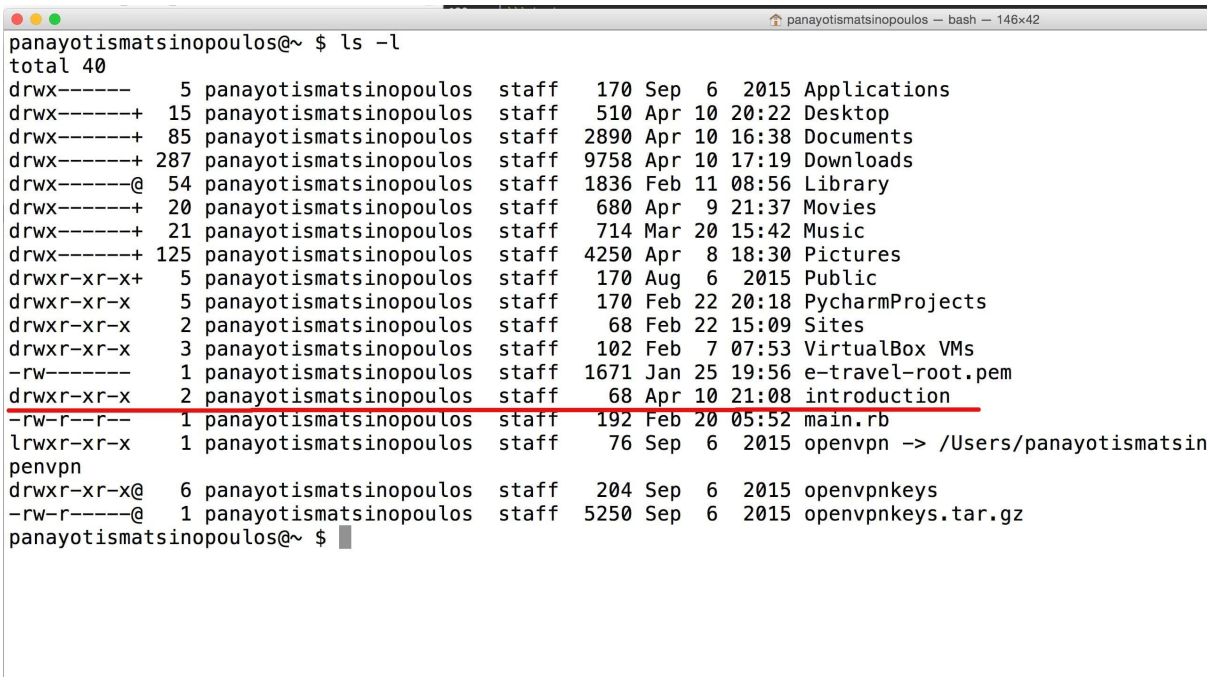
(the above code snippet online)

If everything goes well, you will just be prompted for the next command. In order to double check whether the directory has been created or not, you will have to list the contents of the current directory and see whether `introduction` is part of the list. Let's do that:

```
1 ls -l
```

(the above code snippet online)

If you do that, you will get a long list with the entries of the current directory, like this:



Introduction In the List of Entries

One of the entries is the directory entry `introduction`.

Alternatively, you can list for the particular entry. Try this command:

```
ls -ld introduction
```

(the above code snippet online)

This will give you the following:

```
panayotismatsinopoulos@~ $ ls -ld introduction
drwxr-xr-x  2 panayotismatsinopoulos  staff  68 Apr 10 21:08 introduction
panayotismatsinopoulos@~ $
```

(the above code snippet online)

which basically verifies that the `introduction` has been successfully created.

Note that the `ls -l introduction` will list the contents of the `introduction` directory. If you want to get information about an entry in the list of entries of the current directory, and that entry is a directory, then you have to use the switch `-d` (alongside the `-l` if you want a long list). If you ommit that switch (`-d`) you will get the contents of the directory itself, rather than the details of this directory as child of the current working directory.

Try that:

```
ls -l introduction
```

(the above code snippet online)

You will see that it does not return anything:

```
panayotismatsinopoulos@~ $ ls -l introduction
panayotismatsinopoulos@~ $
```

(the above code snippet online)

This is because the `introduction` directory has just been created and hence does not have any entries.

`ls -ltr` - List - Long Time Ordered, Reverse

There is a very useful variation of `ls`. It is the `ls -ltr`. This give a long list output but with contents ordered by time created in reverse order. This means that it will print the most recent ones last, hence at the bottom of the list, exactly above your next command prompt.

Try that:

```
ls -ltr
```

(the above code snippet online)

This will give the following:

```
panayotismatsinopoulos@~ $ ls -ltr
total 40
drwxr-xr-x+   5 panayotismatsinopoulos  staff   170 Aug  6  2015 Public
drwx------    5 panayotismatsinopoulos  staff   170 Sep  6  2015 Applications
lrwxr-xr-x    1 panayotismatsinopoulos  staff    76 Sep  6  2015 openvpn -> /Users/panayotismatsinc
pn
-rw-r-----@   1 panayotismatsinopoulos  staff  5250 Sep  6  2015 openvpnkeys.tar.gz
drwxr-xr-x@   6 panayotismatsinopoulos  staff   204 Sep  6  2015 openvpnkeys
-rw-------    1 panayotismatsinopoulos  staff  1671 Jan 25 19:56 e-travel-root.pem
drwxr-xr-x    3 panayotismatsinopoulos  staff   102 Feb  7 07:53 VirtualBox VMs
drwx------@  54 panayotismatsinopoulos  staff  1836 Feb 11 08:56 Library
-rw-r--r--    1 panayotismatsinopoulos  staff   192 Feb 20 05:52 main.rb
drwxr-xr-x    2 panayotismatsinopoulos  staff    68 Feb 22 15:09 Sites
drwxr-xr-x    5 panayotismatsinopoulos  staff   170 Feb 22 20:18 PycharmProjects
drwx------+  21 panayotismatsinopoulos  staff   714 Mar 20 15:42 Music
drwx------+ 125 panayotismatsinopoulos  staff  4250 Apr  8 18:30 Pictures
drwx------+  20 panayotismatsinopoulos  staff   680 Apr  9 21:37 Movies
drwx------+  85 panayotismatsinopoulos  staff  2890 Apr 10 16:38 Documents
drwx------+ 287 panayotismatsinopoulos  staff  9758 Apr 10 17:19 Downloads
drwx------+  16 panayotismatsinopoulos  staff   544 Apr 10 21:09 Desktop
drwxr-xr-x    2 panayotismatsinopoulos  staff    68 Apr 10 21:43 introduction
panayotismatsinopoulos@~ $
```

Listing Contents In Reverse Chronological Order

As you can see the most recent item, the newly created `introduction` folder is at the bottom of the list, and hence it can be easily located.

`cd` - Change Directory

Now, we are in the current directory `/Users/panayotismatsinopoulos` and there is a directory, a child directory, with name `introduction`, which has just been created. What if we want to go into the `introduction` directory. This is a Change Directory action that we want to do and it is accomplished with the command `cd` (from `C`hange `D`irectory).

Let's do that. Give the following command:

```
cd introduction
```

(the above code snippet online)

What you will get is this:

```
panayotismatsinopoulos@~ $ cd introduction
panayotismatsinopoulos@~/introduction $
```

(the above code snippet online)

As you can see the prompt has changed to include the name `introduction` just before the `$` sign which denotes the command prompt.

Give the command `pwd`:

```
pwd
```

(the above code snippet online)

This will output the full path to the present working directory:

```
panayotismatsinopoulos@~/introduction $ pwd
/Users/panayotismatsinopoulos/introduction
panayotismatsinopoulos@~/introduction $
```

(the above code snippet online)

which proves that we are now living inside the `introduction` directory (which is child of `/Users/panayotismatsinopoulos`).

~ - Home Directory

As you can see in the prompt above, there is the symbol ~ as part of the prompt, and actually, it seems to be part of a path specification. It is indeed. The `~/introduction` is equivalent to `/Users/panayotismatsinopoulos/introduction`. The ~ is a shorthand to refer to the home directory of the currently logged in user.

Hence, ~ is equal to `/Users/panayotismatsinopoulos` (or whatever is your home directory) and they can be used interchangeably.

For example, type the following command:

```
cd ~
```

(the above code snippet online)

This will change your current working directory to be the home directory of the currently logged in user. In my case `/Users/panayotismatsinopoulos`.

```
panayotismatsinopoulos@~/introduction $ cd ~
panayotismatsinopoulos@~ $
```

(the above code snippet online)

If I do now `pwd` I will see this:

```
panayotismatsinopoulos@~ $ pwd
/Users/panayotismatsinopoulos
panayotismatsinopoulos@~ $
```

(the above code snippet online)

which means that I am no longer inside the `introduction` directory. I have been moved to my home directory.

> *Note:* The symbol ~ is called `tilde`.

Navigating Through Directories

Now, let's move back to the `introduction` directory:

```
cd introduction
```

(the above code snippet online)

And let's create another directory, this time, as child of the `introduction` directory. Execute the following command:

```
mkdir chapter1
```

(the above code snippet online)

The above command creates the directory `chapter1` inside the `introduction` directory. Let's confirm that with the following command:

```
ls -l
```

(the above code snippet online)

This will give the following output:

```
panayotismatsinopoulos@~/introduction $ ls -l
total 0
drwxr-xr-x  2 panayotismatsinopoulos  staff  68 Apr 10 22:03 chapter1
```

(the above code snippet online)

which proves that the directory `chapter1` has been successfully created.

Let's move now into the `chapter1` directory:

```
cd chapter1
```

(the above code snippet online)

We get the following output:

```
panayotismatsinopoulos@~/introduction $ cd chapter1
panayotismatsinopoulos@~/introduction/chapter1 $
```

(the above code snippet online)

As you can see the prompt has now changed in order to indicate that we are inside the `~/introduction/chapter1` folder.

I can go back again to the home directory using the tilde:

```
cd ~
```

(the above code snippet online)

You will see that the new prompt shows that I am on the home directory:

```
panayotismatsinopoulos@~/introduction/chapter1 $ cd ~
panayotismatsinopoulos@~ $
```

(the above code snippet online)

Now, I can go back to the `chapter1` directory with a single `cd` command that would combine the two directories:

```
cd introduction/chapter1
```

(the above code snippet online)

and I am back:

```
panayotismatsinopoulos@~ $ cd introduction/chapter1
panayotismatsinopoulos@~/introduction/chapter1 $
```

(the above code snippet online)

And if I want to go to the parent directory of `chapter1` without actually specifying its name, I can refer to it with a double full stop notation . . . Give the following command:

```
cd ..
```

(the above code snippet online)

This changes the current directory to be the parent one.

```
panayotismatsinopoulos@~/introduction/chapter1 $ cd ..
panayotismatsinopoulos@~/introduction $
```

(the above code snippet online)

In other words, the `..` means the parent directory of the present working directory.

Now, if I give the command:

```
cd ..
```

(the above code snippet online)

then the present working directory is going to be ~, which is the parent directory of `introduction`.

```
panayotismatsinopoulos@~/introduction $ cd ..
panayotismatsinopoulos@~ $
```

(the above code snippet online)

Editing Text Files

All distributions of Linux, Windows and Mac contain a program that will allow you to edit text files. One very popular on Linux and Mac is the `nano` program.

Let's try that. Give the following command on your command prompt:

```
nano
```

(the above code snippet online)

This command will start the nano editor which looks like this:

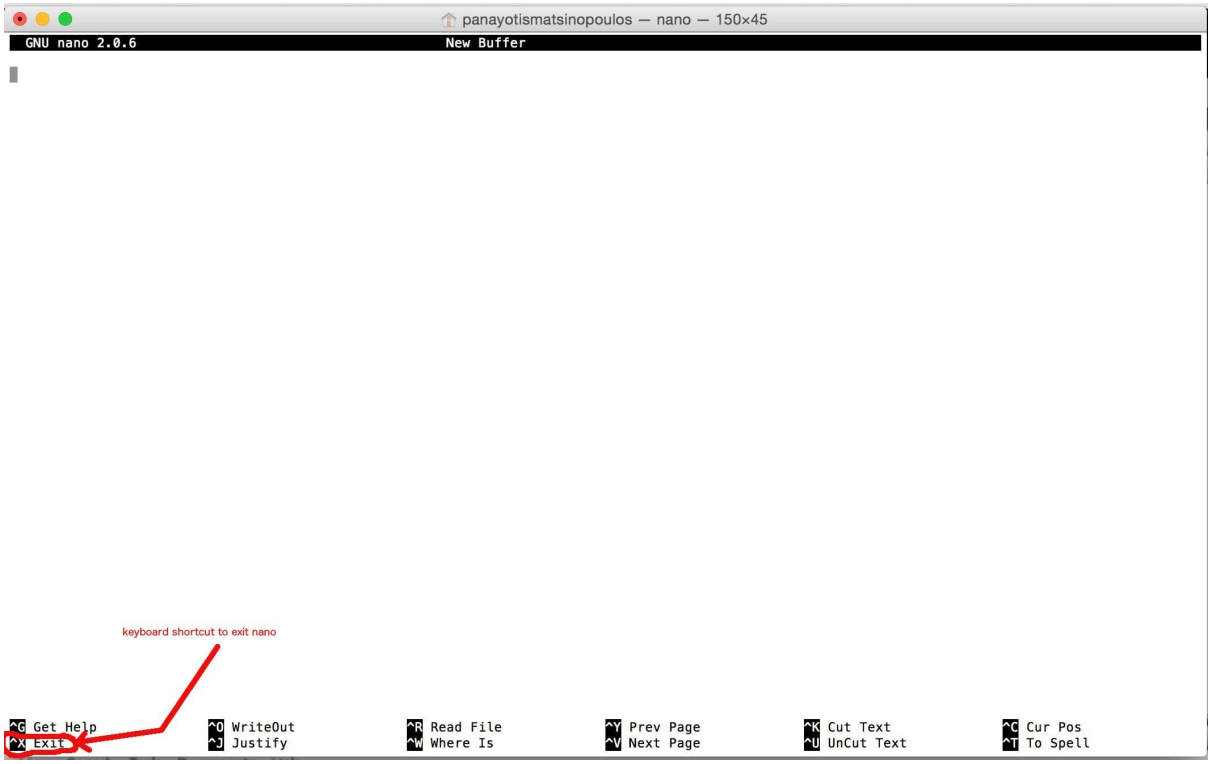
Nano Editor

At the bottom of the nano environment, you can see some of the most common nano commands and how you can activate them.

Nano Commands And How You Can Activate Them

Note: the symbol `^` in the command keyboard shortcuts corresponds to the `Ctrl` key.

Usually the nano commands are activated pressing on the `Ctrl` key and then, without releasing, on the corresponding command key. For example, in order to exit nano, you need to press on `Ctrl + X`.



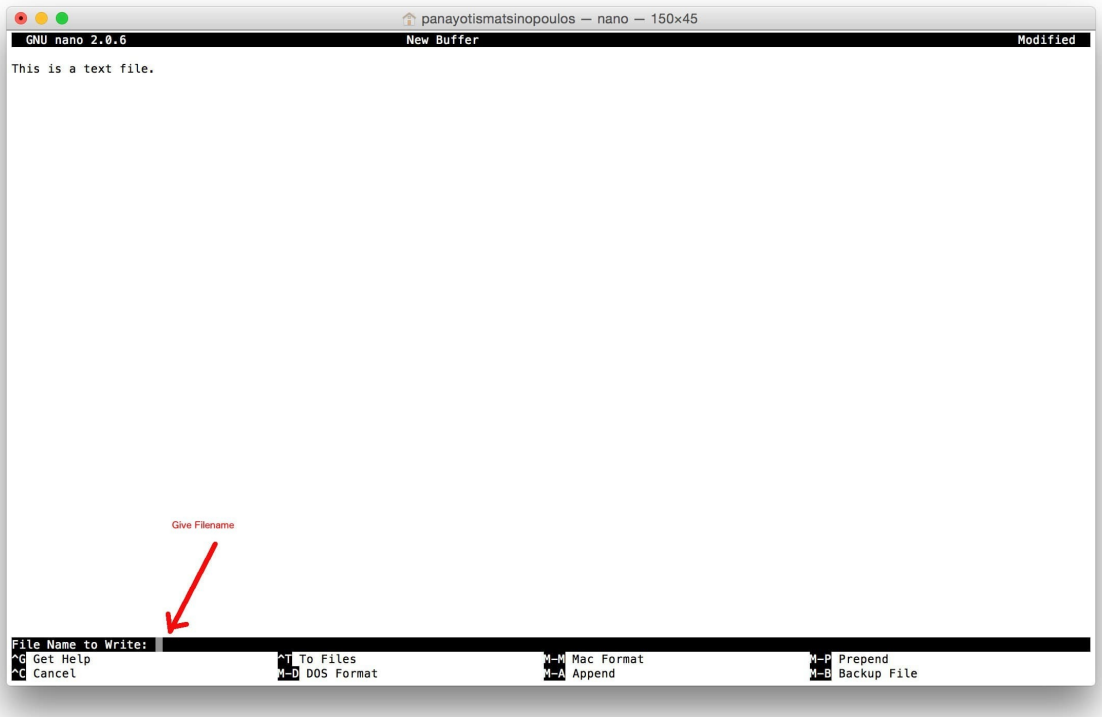
Keyboard Shortcut To Exit Nano

Let's type some content inside the nano text editing area:

Typed In Some Text Inside The Text Editing Area

We typed the text: `This is a text file`.

In order to save the content of this file into disk, you need to use the `WriteOut` command: `Ctrl + O`. You will be prompted to give a name to your first-time saved file:



Need To Give Name In Order to Save File

Let's give the name `sample.txt`. We type that in and we hit the `Enter` key.

So, this file should exist inside the home directory. Let's exit nano (`Ctrl + X`) and execute the following command:

```
ls -ltr
```

(the above code snippet online)

You will see that, at the bottom, there is the newly created file `sample.txt`:

```
panayotismatsinopoulos@~ $ ls -ltr
total 48
drwxr-xr-x+   5 panayotismatsinopoulos  staff   170 Aug  6  2015 Public
drwx------    5 panayotismatsinopoulos  staff   170 Sep  6  2015 Applications
lrwxr-xr-x    1 panayotismatsinopoulos  staff    76 Sep  6  2015 openvpn -> /User\
s/panayotismatsinopoulos/Applications/openvpn-2.3.8/src/openvpn/openvpn
-rw-r-----@   1 panayotismatsinopoulos  staff  5250 Sep  6  2015 openvpnkeys.tar.\
gz
drwxr-xr-x@   6 panayotismatsinopoulos  staff   204 Sep  6  2015 openvpnkeys
-rw-------    1 panayotismatsinopoulos  staff  1671 Jan 25 19:56 e-travel-root.pem
drwxr-xr-x    3 panayotismatsinopoulos  staff   102 Feb  7 07:53 VirtualBox VMs
drwx------@  54 panayotismatsinopoulos  staff  1836 Feb 11 08:56 Library
-rw-r--r--    1 panayotismatsinopoulos  staff   192 Feb 20 05:52 main.rb
drwxr-xr-x    2 panayotismatsinopoulos  staff    68 Feb 22 15:09 Sites
```

```
drwxr-xr-x    5 panayotismatsinopoulos  staff   170 Feb 22 20:18 PycharmProjects
drwx------+  21 panayotismatsinopoulos  staff   714 Mar 20 15:42 Music
drwx------+ 125 panayotismatsinopoulos  staff  4250 Apr  8 18:30 Pictures
drwx------+  20 panayotismatsinopoulos  staff   680 Apr  9 21:37 Movies
drwx------+  85 panayotismatsinopoulos  staff  2890 Apr 10 16:38 Documents
drwx------+ 287 panayotismatsinopoulos  staff  9758 Apr 10 17:19 Downloads
drwxr-xr-x    3 panayotismatsinopoulos  staff   102 Apr 10 22:03 introduction
drwx------+  22 panayotismatsinopoulos  staff   748 Apr 10 22:51 Desktop
-rw-r--r--    1 panayotismatsinopoulos  staff    21 Apr 10 22:54 sample.txt
```

(the above code snippet online)

`cat`

If we want to see the contents of a text file, we can call the `cat` command. Let's do that for the `sample.txt` file:

```
cat sample.txt
```

(the above code snippet online)

You will get this:

```
panayotismatsinopoulos@~ $ cat sample.txt
This is a text file.
panayotismatsinopoulos@~ $
```

(the above code snippet online)

You can see the contents of the file given as input argument to the `cat` command to be printed on the output.

`rm` - Remove a file

What is the command that we should use, if we want to delete a file. It is the `rm` command. Let's do that. Let's delete the file `sample.txt`. Give the following command:

```
rm sample.txt
```

(the above code snippet online)

If you do that, and then you will execute the `ls -ltr` file, you will not see the `sample.txt` file anymore in the list.

`rm -R` - Remove Directory

Let's now go to the `introduction` directory again:

```
panayotismatsinopoulos@~ $ cd introduction
panayotismatsinopoulos@~/introduction $
```

(the above code snippet online)

This one has the directory `chapter1` inside. What if we want to remove the directory `chapter1`. Try the following command:

```
rm chapter1
```

(the above code snippet online)

You will get an error that `chapter1` is a directory:

```
panayotismatsinopoulos@~/introduction $ rm chapter1
rm: chapter1: is a directory
panayotismatsinopoulos@~/introduction $
```

(the above code snippet online)

and the directory will not be removed. If you want to remove a directory, you can still use the command `rm`, but you have to do that with the use of the switch `-R`. Let's do that:

```
rm -R chapter1
```

(the above code snippet online)

This will not output anything, if all goes well. You can see that the directory has been removed, if you do `ls -l` and check whether the `chapter1` is returned as entry in the list.

After having removed the `chapter1` directory, let's go back to the parent directory:

```
cd ..
```

(the above code snippet online)

The situation is now this:

```
panayotismatsinopoulos@~/introduction $ cd ..
panayotismatsinopoulos@~ $
```

(the above code snippet online)

Which means that we can now try to remove the `introduction` directory:

```
rm -R introduction
```

(the above code snippet online)

This will remove `introduction` directory and it will not appear any more in `ls` commands on the home directory.

Tasks and Quizzes

Before you continue, you may want to know that: You can sign up to Tech Career Booster and have a mentor evaluate your tasks, your quizzes and, generally, your progress in becoming a Web Developer. Or you can sign up and get access to Tech Career Booster Slack channel. In that channel, there are a lot of people that can answer your questions and give you valuable feedback.

Quiz

The quiz for this chapter can be found here