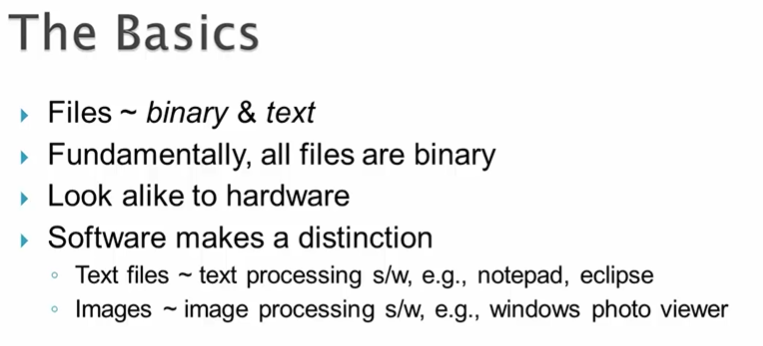
Before we do any programming that involves processing of text, it is important to know about character encoding.



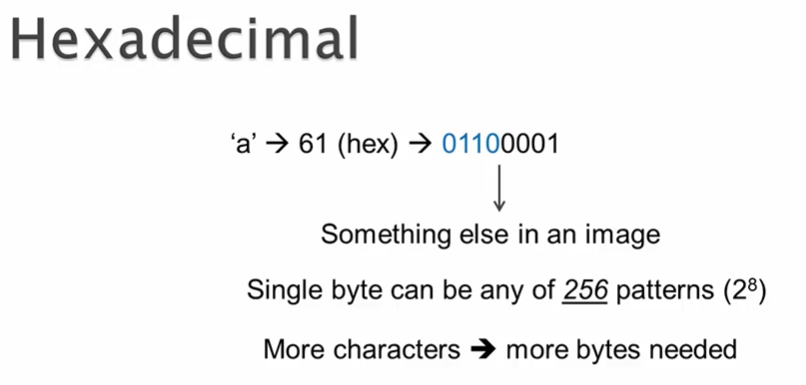
Most people classify files into 2 categories :- binary & text

Binary would include content like images, videos & audios while text is about file with characters. However, fundamentally, all files are binary that a sequence of bytes where each byte is a group of 8 bits and a bit is either 0 or 1.

So, text files are also binary files, it is just that they are stored in a certain way in binary.

All files whether they are binary or text, would look alike to computers hardware. However, software’s would make a difference.

For instance, bytes representing text are only handle by text processing s/w like notepad, wordpad or editors like vi, eclipse…Similarly, bytes representing images are only handled by image processing s/w like windows photo viewer or paint application.



Computer use hexadecimal numbers to represent bytes.

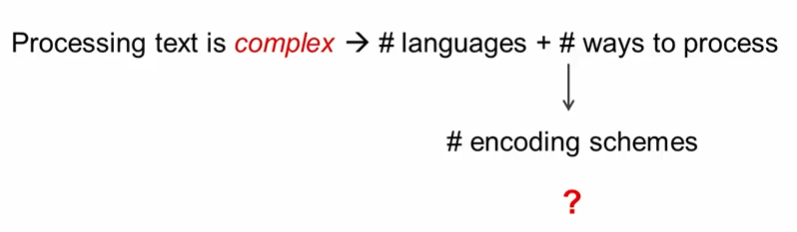
For ex : character ‘a’ is represented using the hexadecimal number 61, which is stored in binary as the byte of 01100001, where 6 corresponds to the first 4 bits while 1 for the last 4 bits. Now, the same bit pattern would represent something else in an image file. Also, note that single byte can be any of 256 bit patterns.

It means we can represent as many as 256 characters using different variations of a byte.

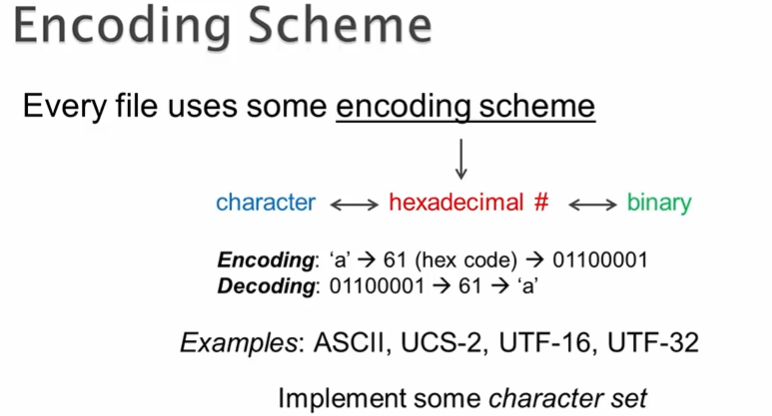
But to represent more characters, we need to use more than one byte and that’s where different encoding schemes comes into play.

Processing text is very complex due to the number of languages involved and also due the number of ways in which characters in those languages can be represented. Number of ways would be the different encoding schemes.

So a given text follow a encoding scheme and if a software ( for eg : browser ) doesn’t understand that character encoding used then You would end up seeing familiar weird character such as below.



This happens usually for international characters.

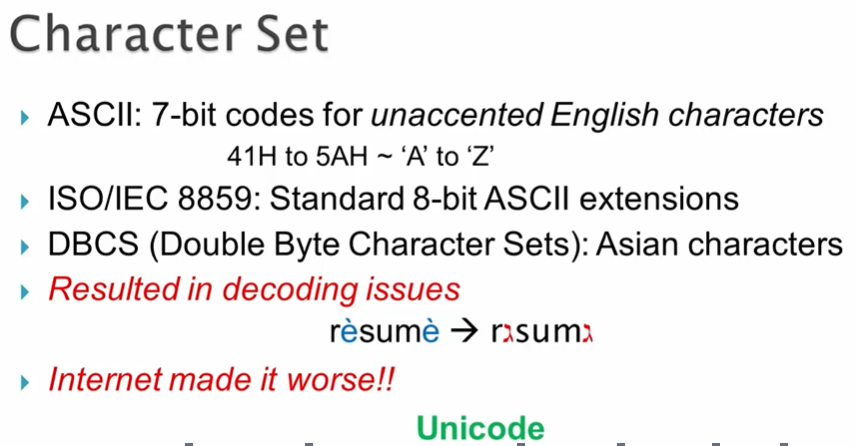


Every file uses some **encoding scheme** to represent its content . Each encoding schemes ( ASCII, UTF… ) is basically an implementation of some character set.

For instance, UTR-8. UTF-16, UTF-32 are all implementation of Unicode character set.

Terms encoding and character set are frequently inter mixed but they are different.

With regards to ASCII, it represent both as character set and encoding scheme.



Initially, only character that mattered were unaccented English characters and for that 7 bit ASCII code were used.

But since we have many other languages, soon tother 8 bit ASCII extensions ( ISO 8859 ) came up. Infact, ISO 8859 had 15 different variations for 15 different regions in the world.

In these extensions, the first one 128 codes are always identical and would correspond to the 7 bit ASCII while the remaining 128 codes is where we see the differences based on the regions.

Now Asians languages have thousands of letters and 8 bit wasn’t sufficient for them, so they discovered another encoding scheme called as DBCS.

Due to all these variations, when document were mailed from one country to another, then they were not getting decoded properly.

For instance, hexadecimal code of a particular character in one country would corresponds to a completely different character in another country.

To address this, Unicode character set was invented, which aims to cover all languages in the world.

It is backward compatible with 7 bit ASCII so first 128 characters are same. Now, initial assumption was : 16 bits which would represent 65536 chars would suffice to cover all languages and these chars together as a group are referred to as BMP. As a result, UCS-2 which was a fixed length encoding scheme with 16bit was created. Since its fixed length, it means it use exactly 16 bits to represent any character.

But soon it was realized that there were more characters. These characters were especially important for Asian markets. These would also include things like Smiley graphics symbols which we see in various social networking applications, as a result other encoding schemes like UCS-4, UTF-16 were created to accommodate new characters but UCS-4 was not favored as it required 4 bytes for every character. So it would be a lot of disk space and memory.

According to WIKI, as of today Unicode covers around 120K characters from around 129 scripts. So Unicode is really universal.

So Unicode character set has 2 parts.

BMP ( 65536 characters ) , NON BMP ( 55K characters ), that is as of today and we may expect more characters to be added to NON BMP set and the hexadecimal code that represent each character is referred to as **Code Point** and to represent a single BNP character one 16 bit code point is used, however for NON BNP, two 16bit code points are used.

