Chapter 16 Important Notes : Serialization and file I/O

- In java program we have lots of options for how to save the state of your java program and that depends on your goal for example if you will use your data in only java programs so you can use SEREIALIZATION and that means you write a file that holds the state values of the objects and the relations between the objects if it is existed and convert them into bytes and wrote them in this file and this process called serialized and when you need to read your data the reverse thing happened but this happened in only the java programs can do this
- The second thing is to use a plain text file and it is like the file we use instead of the database to save our data on it and we use delimiters that other programs can parse which makes your file can be used in other programs like excel or anything else not just the java programs and it is very closer to the data base it consists of rows and attributes and its readable for anyone can read it easily
- And this is not only the existed options there is another options
- The serialized is much harder for humans to read but it is much faster for your program to read it rather than read the objects values from the plain text file

- to write a serialized object to a file you can make a fileoutputstream object and it will be your file and make pbjectoutputstream and it will be used to writing the serialized objects into the file by it and then you will need to close the file at the end of the serialization process

- there is difference between the object in the heap and the serialized object , the first one we can think about it like alive we can make changes on It and can calling methods and can make behaviors by using this one but the second one is like a snapshot for specific time for this object values we can not make anything to this changing or calling methods by using it this can not happened at all when we make deserialized it turns to new object and in this time it becomes new object with the same values In the heap

- But what about if the instance variables for the object is another object? Or array of objects ? → When an object is serialized, all the objects references through its instance variables are also serialized. Then, all the objects those referenced objects point to are serialized as well. In fact, the entire object graph is serialized automatically — and the best part is, it all happens without you having to do anything!

- Do you remember in the serializable process when we write object into the object stream output? To can make this without any failure we need to ensure the object we will serialize it is implements serializable interface cause when we make that it tells the JVM it is ok to serialize objects from this type and be careful about the i/o operations can it can throw exceptions so do not forget the exception handling

- Very important note about making the class implements serializable if the object we will serialize is pound and inside it , it has another object instance from type cat for example we must ensure that both classes cat and pound both implements the serializable interface not just the pound , and know that the serialization is all or nothing like the transaction in the data base concepts

- What about if you do not want to save all or serializable all the instance variables for the object ? you can use the keyword TRANSIENT before the variable this one make this variable be skipped in the serialization process , be careful about using transient cause this make the value of this value not saved so when you make deserialized for the object the transient variable will having in the new object null or the default value of the type and you will need to handle this

- now we will start talking about the deserialization the reverse process and the reading from the file we will need to define fileinputstream object and then objectinputstream object and connect it to the file object and then using readobject method to read the objects existed in the file but here we will read the data in OBJECT type so we will need to cast this objects into their original objects and in the end close the file and here it is input cause we will take an input from the file to read it

- static variables not been serialized ,

- **transient → non-transient = ممكن يسبب فقدان بيانات قديمة**
- **non-transient → transient = آمن، بس البيانات الجديدة مش هتتحفظ**
- If a class that implements Serializable might change over time, put a static final long serialVersionUID on that class. This version ID should be changed when the serialized variables in that class change.
- There is difference between the filewriter (which it is send each time you write content in the file in the disk which make overhead) and the second thing which is

the buffered file which like a temporary place to store all the objects contents on it and when the buffer full it goes just one time and put all of them in the file in the disk which make the process faster and less time and if you want to make the buffer sends all of its content without waiting until it become full you can use FLUSH method