Chapter 13 Important Notes : Risky Behavior

- ✓ There is in java Java Sound API and it is collection of classes and interfaces added to java and there is specific type of it called MIDI and it is file having information about how the song should be played it does not have any actual sound data it is like sheet music instructions for the some one who will play the music and it can be for any type music piano or ..
- ✓ Methods in java use exceptions to tell the calling code that something bad happened I failed (سيم يا ميثود)
- ✓ So that means you need to write risky code that will dealing with the risky methods and you need to wrap it in try / catch cause making the compiler relax
- ✓ So the explanation of try / catch blocks easily is that I am gonna try this risky thing and I am gonna catch myself if I fail
- ✓ An exception is an object of type exception , and what you write in the catch block depends on the exception that was thrown so if the server is down and you catch that you need to write inside the catch block that to try another server for example to can know the reason of the exception
- ✓ It is important to know which methods will throw the exception and which one will catch it
- ✓ So the known is that there is one method will catch what another method throws , an exception is always thrown back to the caller and the method that throws has to declare that it might throw the exception
- ✓ There is types of exceptions that not are subclass of runtime exception and that means there are checked by the compiler and cause that they are called checked exceptions , compiler checks everything except runtime exceptions , run time exceptions does not have to be wrapped in a try / catch block
- ✓ There is a finally block we can add it to our try/ catch block and the finally block is where you put code that must run regardless of an exception it means if there is exception happen the finally block will work and if there is no exception happened the finally block will also work
- ✓ if i not put finally the code after the catch may not working cause there is may another unhandled exception happened will stop the whole code and the finally will solve this problem cause even if there is unhandled exception sothe finally will be worked in all cases
- ✓ there is a thinking of what about just making catch and it will catch object from exception class so that will make it easily catch any exception and I will not needed to handle or worry about all of them that's right BUT you use the catch block to

catch the exception and based on that you will make specific decisions to recovery or to handle the situations and if you just make the catch with exception object you make it too general how and each thing need its specific way to be handled and that how you can make it you do not even know what specific exception you catch to can handle it so here we come with the multiple catch blocks each one for specific exception and you must put the super , biggest exception class in the last catch , multiple catch blocks must be ordered from smallest to biggest

- ✓ okay now we have 2 ways to satisfy the compiler when you call a risky method the first way is to handle it and that by using try catch blocks or to declare that your method(caller) throws the same exceptions as the risky method you are calling will throw and this is calling DUCK IT and if you will try this way you will need to defining the whole chain until the main and make the main too throws the same exception to can shut down without any problems
- ✓ remember it is not enough to have a catch block no you have to catch the thing being thrown
- ✓ this is some important roles you must be careful about it :
  - you can not have a catch or finally without a try
  - you can not put code between the try and catch
  - a try must be followed by catch or finally
  - if you make try and finally without a catch it is still not handle the exception