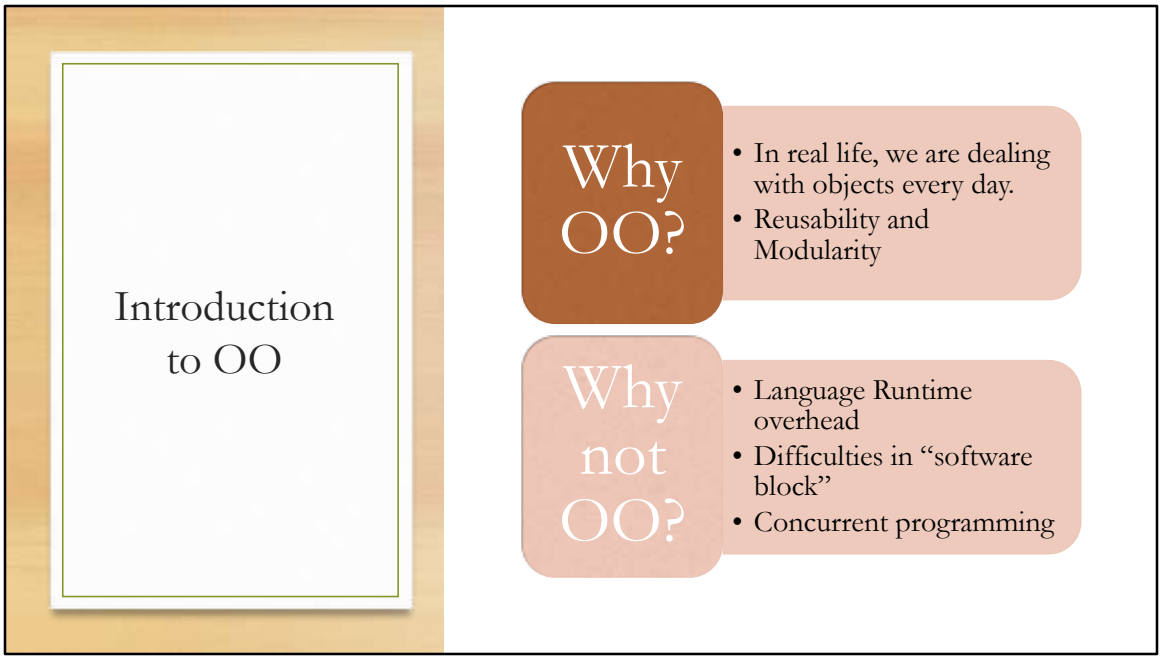
Object Oriented Design

Cecil Sheng

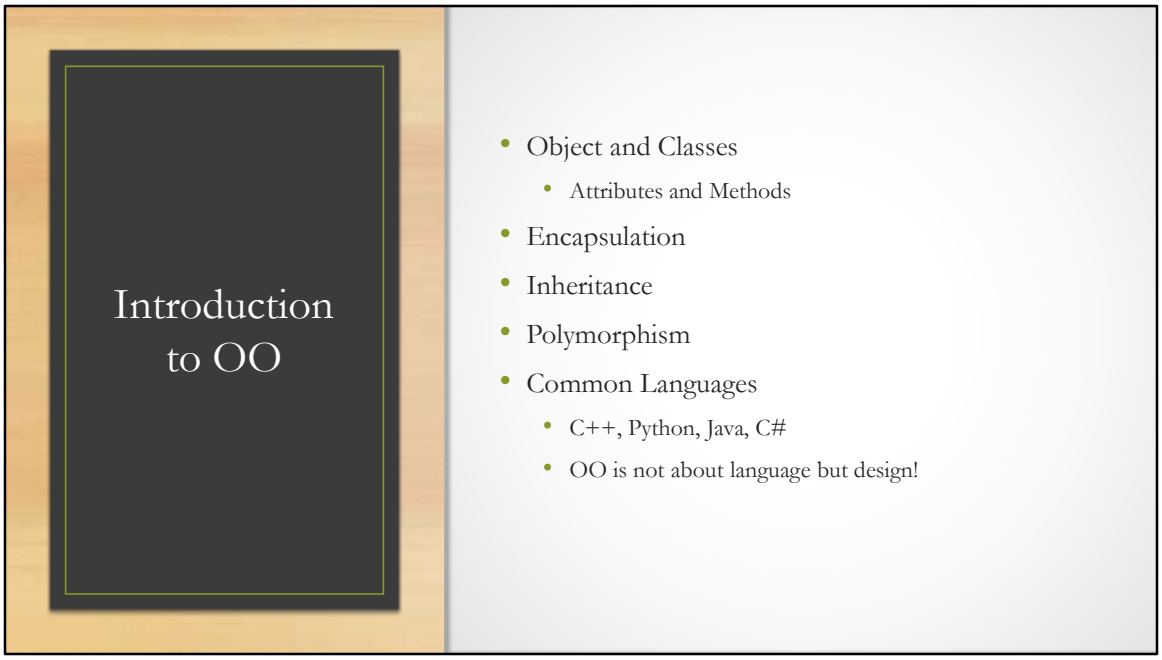
DellEMC Taiwan

Topics

- Introduction to OO
- Handling Interview Questions
- Design Patterns
- Sample Questions
- Q & A



OO is good for design but difficult for implementation. Both in code files layout (header files, source files = software block) and language runtime. The reason why it's not good at concurrent programming is that it's difficult to module "time".
References:
https://en.wikipedia.org/wiki/Object-oriented_programming
https://www.kingstone.com.tw/book/book_page.asp?kmcode=2014712860683 if you want to know more about OO

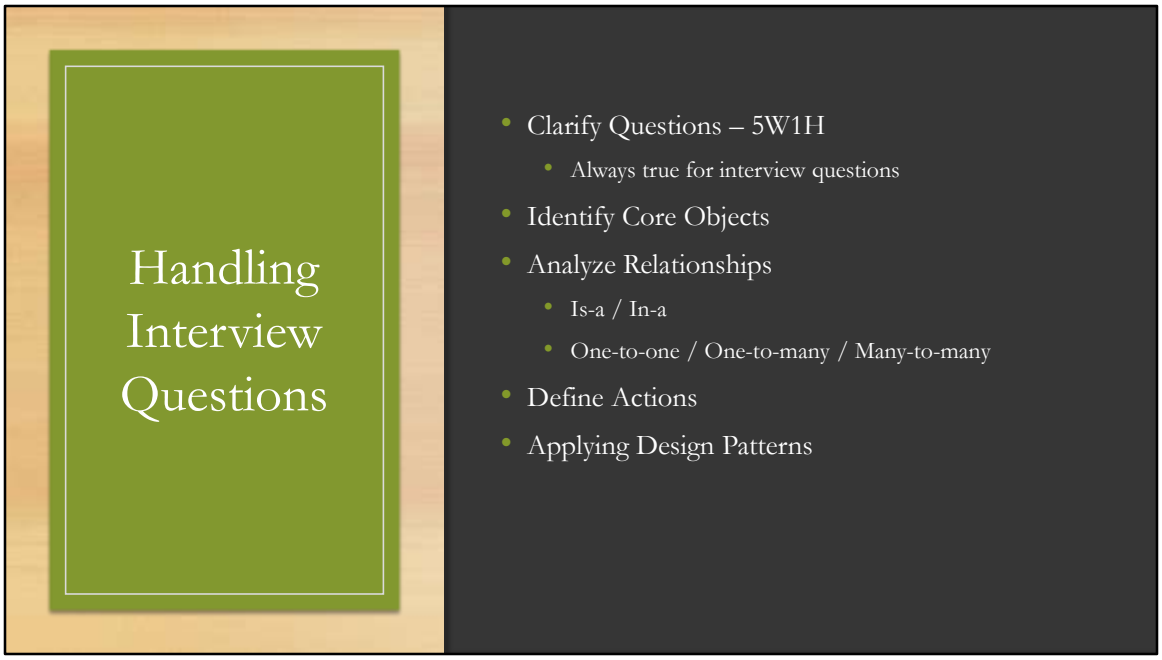


In the real world, there are many objects we interact with everyday. We tend to classify objects into functional groups so that our brain spends less resources on common behaviors. For example, there are various kinds of cups (objects), but we can treat them all the same way (the cup class). A cup is something we can Drink (a method) from, and can store some liquid (attributes, what liquid a cup stores, current amount of the liquid, and the maximum capacity of the cup).

The three primary principals of OO.
Encapsulation – this is good for general design also. A software unit should keep the smallest interface possible to interact with outside world. It's like when we watch a TV, we don't need to know how it works inside, just push the power button and switch channels.
Inheritance – it's about classes. Some classes are more general the others. For example, a Cup is also a Container. Or a cat and a dog are both mammals. In OO's way, we say the Cat class and the Dog class both inherit the Mammal class. Which means the child classes have all the attributes and methods of the parent class.
Polymorphism – When we interact with a parent class, we don't actually know/need to know what the real object is. When we want to drink a cup of water, we can use anything object which is a cup. It can be any cup as long as it shares all the attributes

and methods a cup has. This allows us to change the real implementation later on without breaking original execution flow.

So called OO languages provides the syntax and tools to help you implement OO software. But OO is a way of design things, with the principals in mind you can use any language to implement it (though may be more complicated.)



Design questions tend to be vague. We need to check the limitations and primary concerns of the question before start designing things.
First we need to decide the core objects (classes). Then their relationships, for example,

Design Patterns

- Originally from a book
  - ***Design Patterns: Elements of Reusable Object-Oriented Software***
- Examples
  - Singleton
  - Factory Method
- Pick up a book you can understand
- Apply patterns to problems, not problems to patterns

Othello
Piece – black and white as property
Board - hold pieces.
Player
Game

File System
Entry
File, Directory extends Entry

Q & A