Design Patterns

Tuesday, August 10, 2021 11:56 AM

Design patterns are evolved as reusable solutions to the problems that we encounter every day of programming.

They are generally targeted at solving the problems of object generation and integration.

These generalized patterns act as templates that can be applied to the real- world problems.

Evolution of Design Patterns:

The four authors of the book "Elements of reusable Object-Oriented software" are referred as Gang of Four.

The book is divided into two parts with first part explaining about the pros and cons of Object oriented programming and second part explaining the evolution of 23 classic software design patterns.

From then, Gang of Four design patterns has made a significant role in the software development life cycle.

There are 3 types of Design Patterns

Creational

This type deals with the object creation and initialization. This pattern gives the program more flexibility in deciding which objects need to be created for a given case.

Eg : Singleton, Factory, Abstract Factory .. etc

Structural

This type deals with class and object composition.
This pattern focuses on decoupling interface and implementation of classes and its objects.

Eg : Adapter, Bridge .. Etc

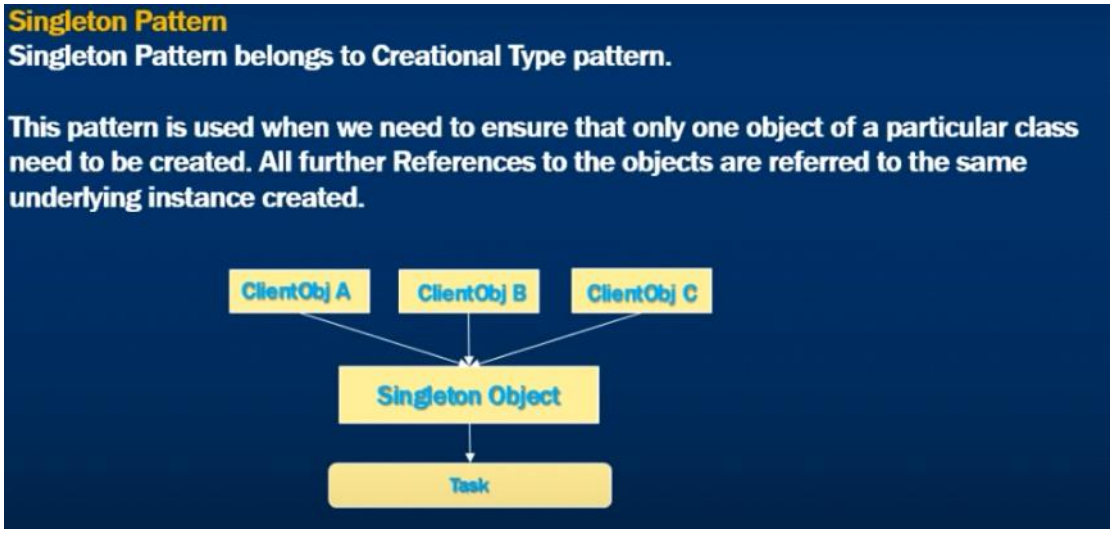
Behavioural

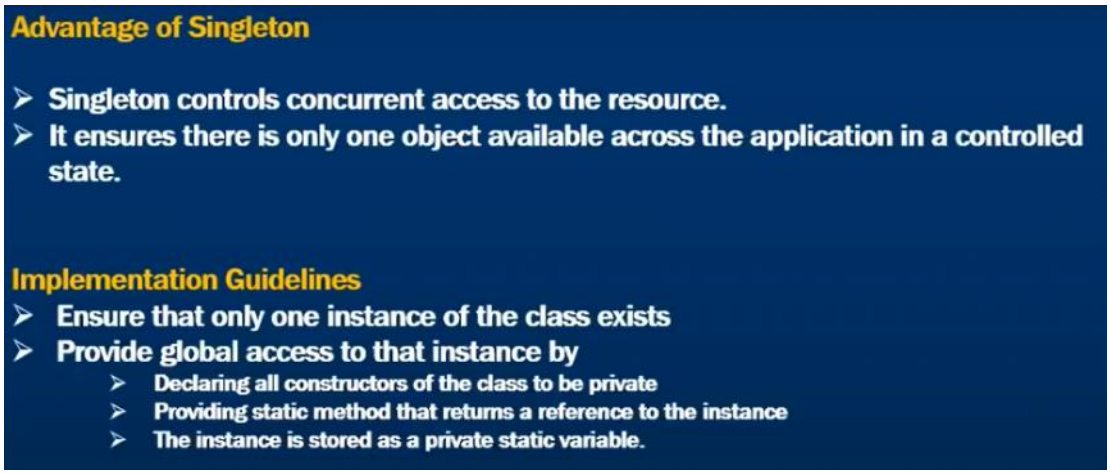
This type deals with the communication between Classes and objects.

Eg :- Chain of Responsibility, Command, Interpreter .. etc.

Singleton Design Pattern

Tuesday, August 10, 2021 12:02 PM





Lazy vs Eager initialization

Lazy Loading

- Improves the performance
- Avoids unnecessary load till the point object is accessed
- Reduces the memory footprint on the start-up
- Faster application load

Non-Lazy or Eager Loading

- Pre-Instantiation of the object
- Commonly used in lower memory footprints

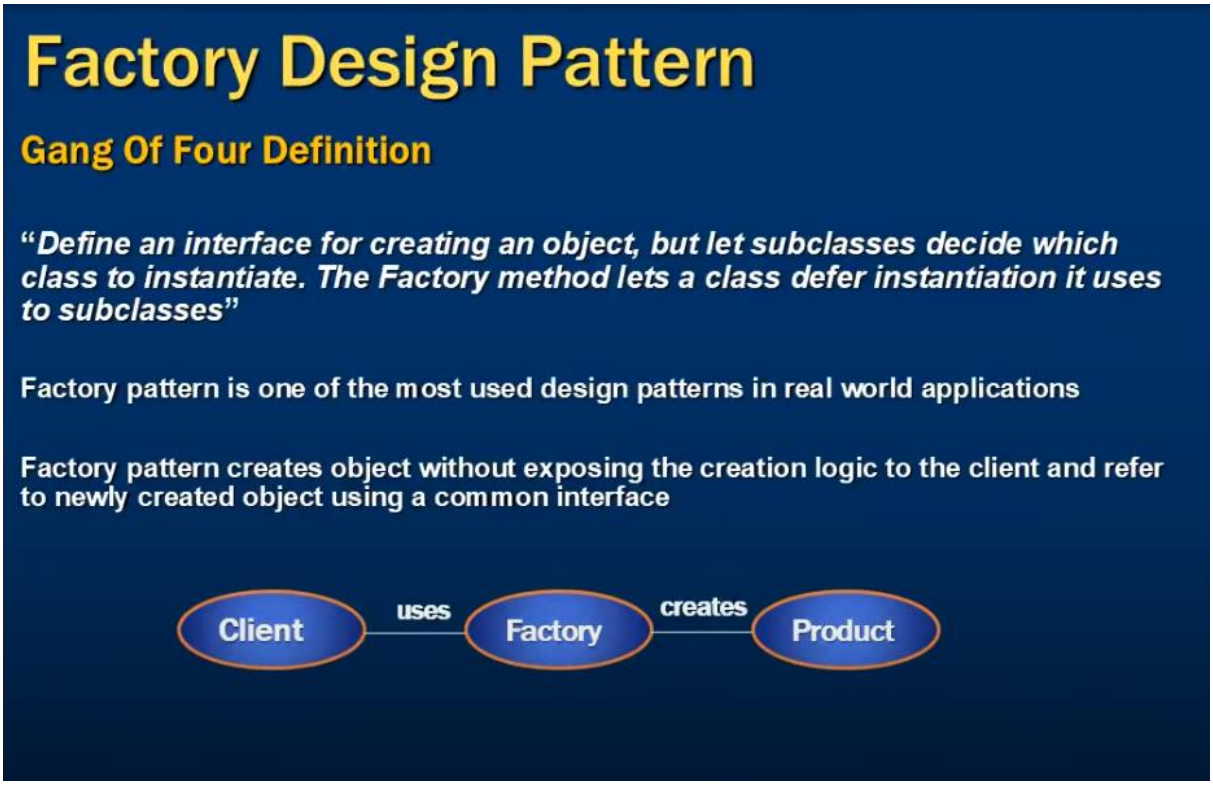
Static vs Singleton

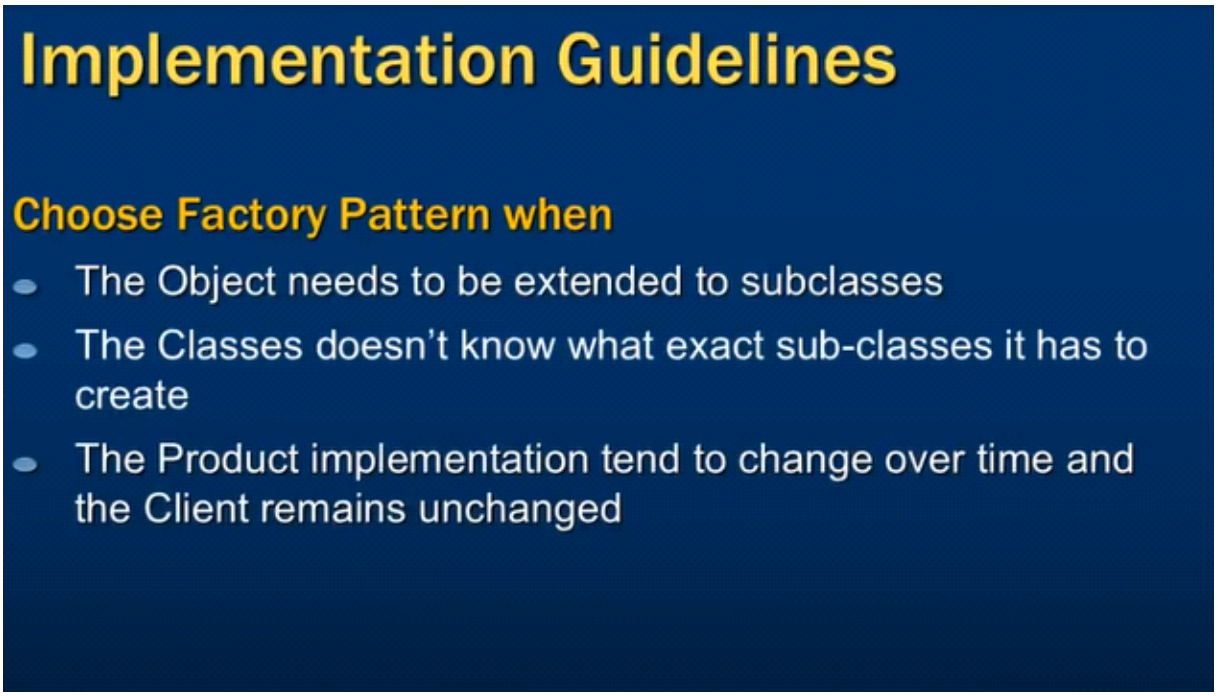
- Static is a keyword and Singleton is a design pattern
- Static classes can contain only static members.
- Singleton is an object creational pattern with one instance of the class.
- Singleton can implement interfaces, inherit from other classes and it aligns with the OOPS concepts.
- Singleton object can be passed as a reference
- Singleton supports object disposal
- Singleton object is stored on heap
- Singleton objects can be cloned


Singleton Real World Usages
Logging
Managing a connection or a pool of connections to Database.
Printer spooling
File
Configuration
Cache
Session based Shopping cart

Factory Design Pattern

Thursday, August 12, 2021 12:38 PM



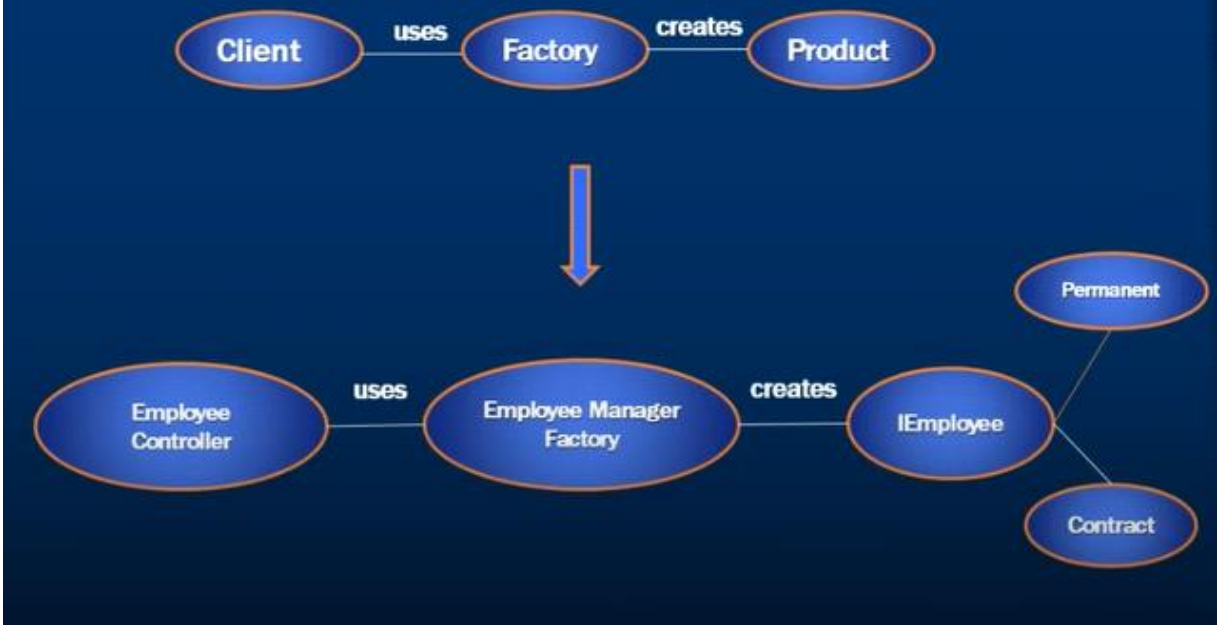


Simple Factory Example

Business Requirement

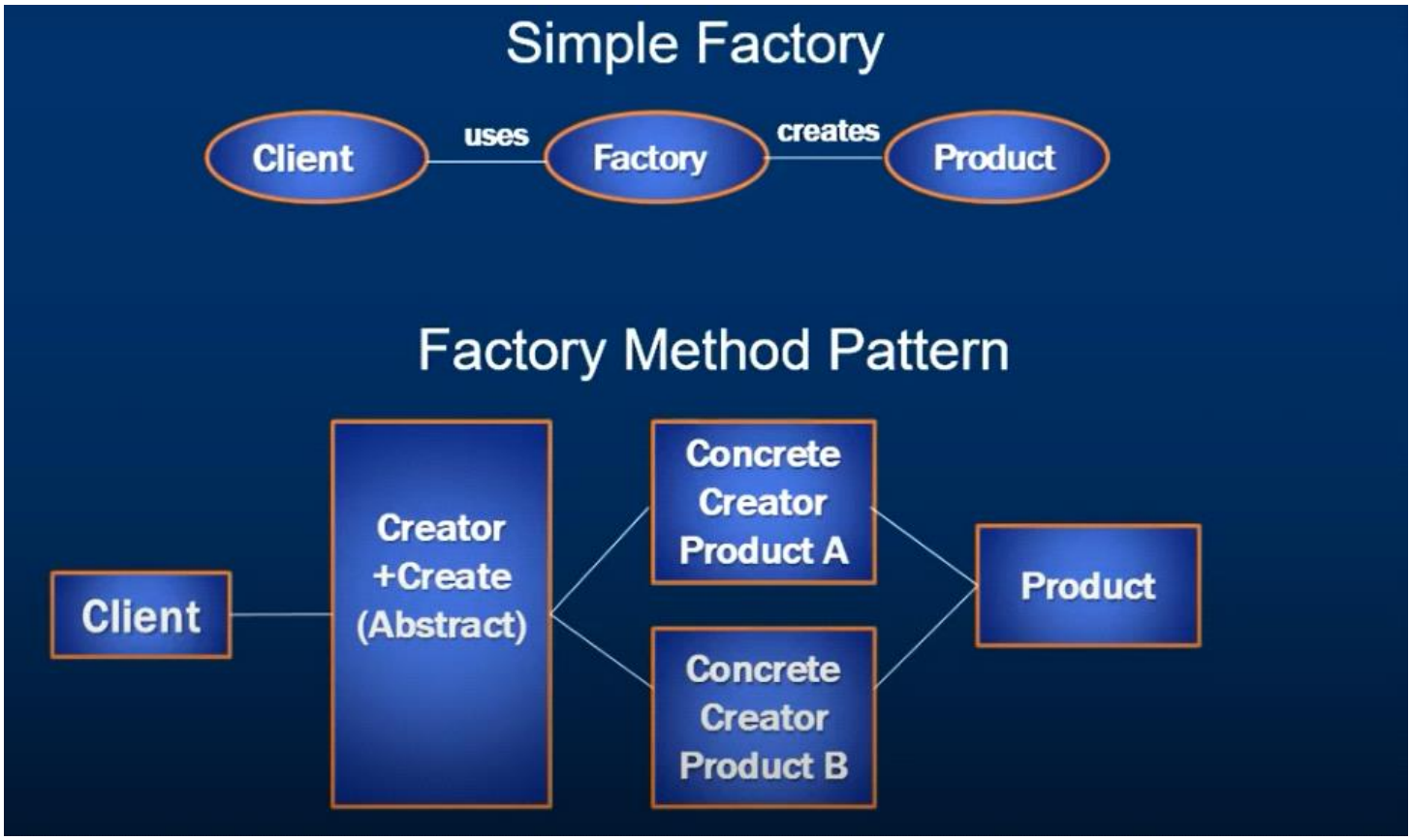
Differentiate employees as permanent and contract and segregate their pay scales as well as bonus based on their employee types

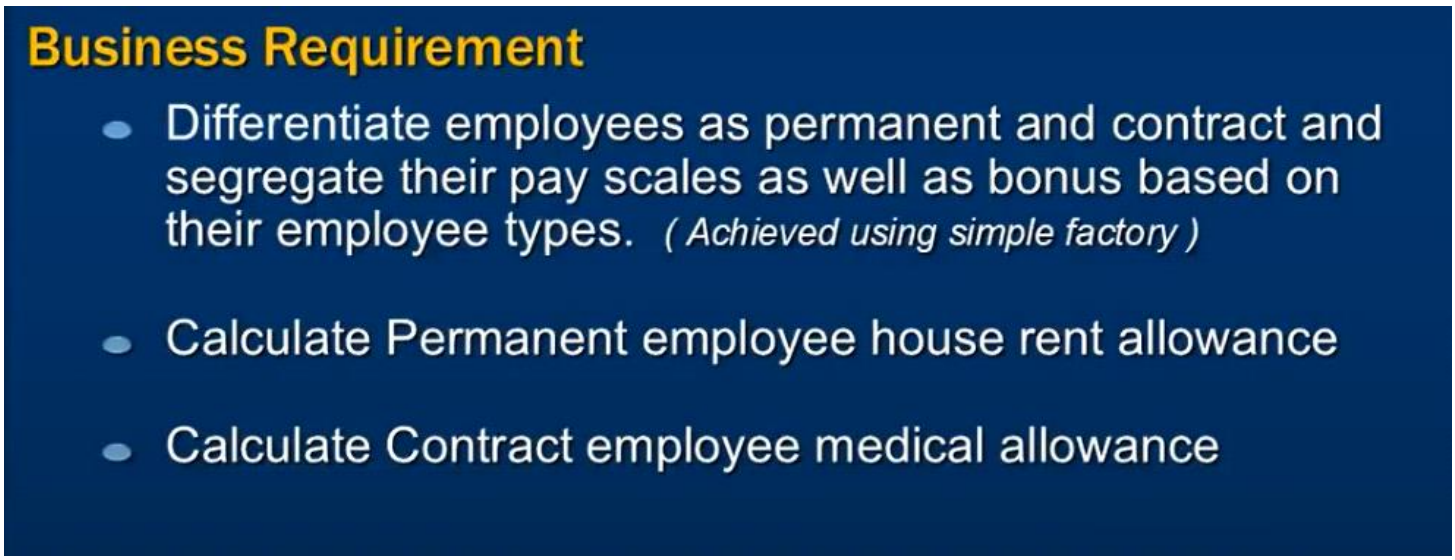
Simple Factory Representation

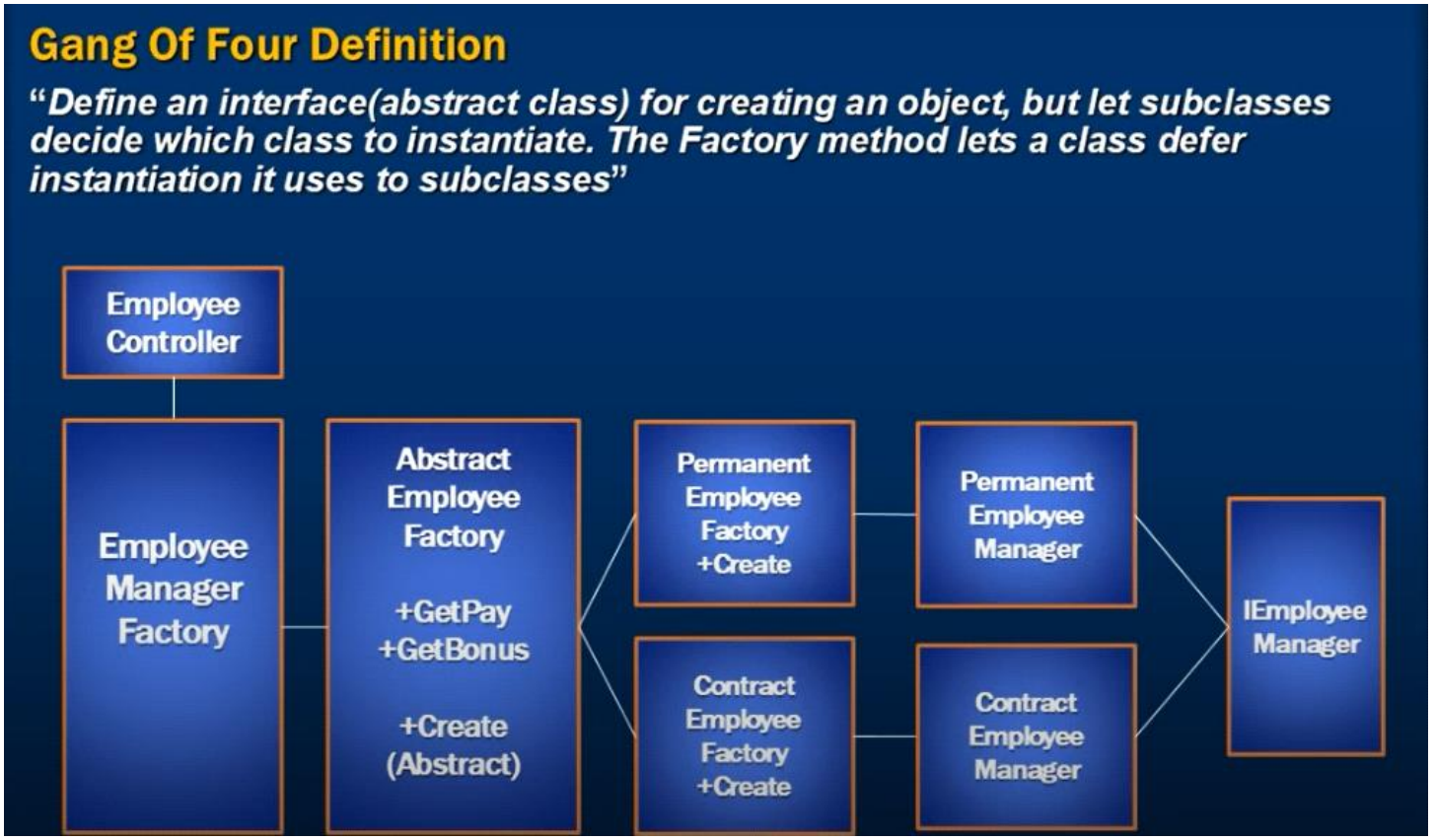


Factory Method Design Pattern

Friday, August 13, 2021 12:13 AM

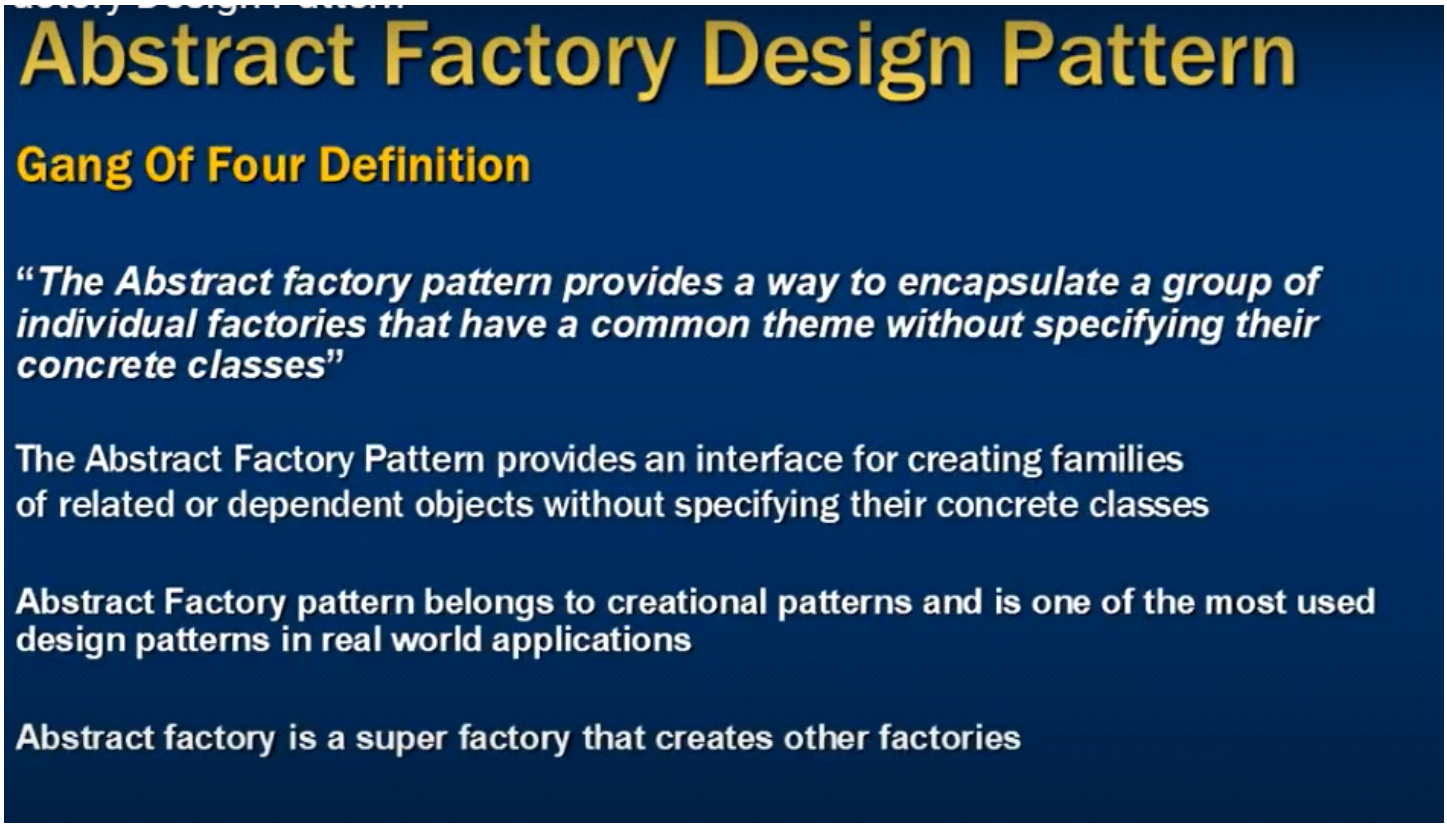


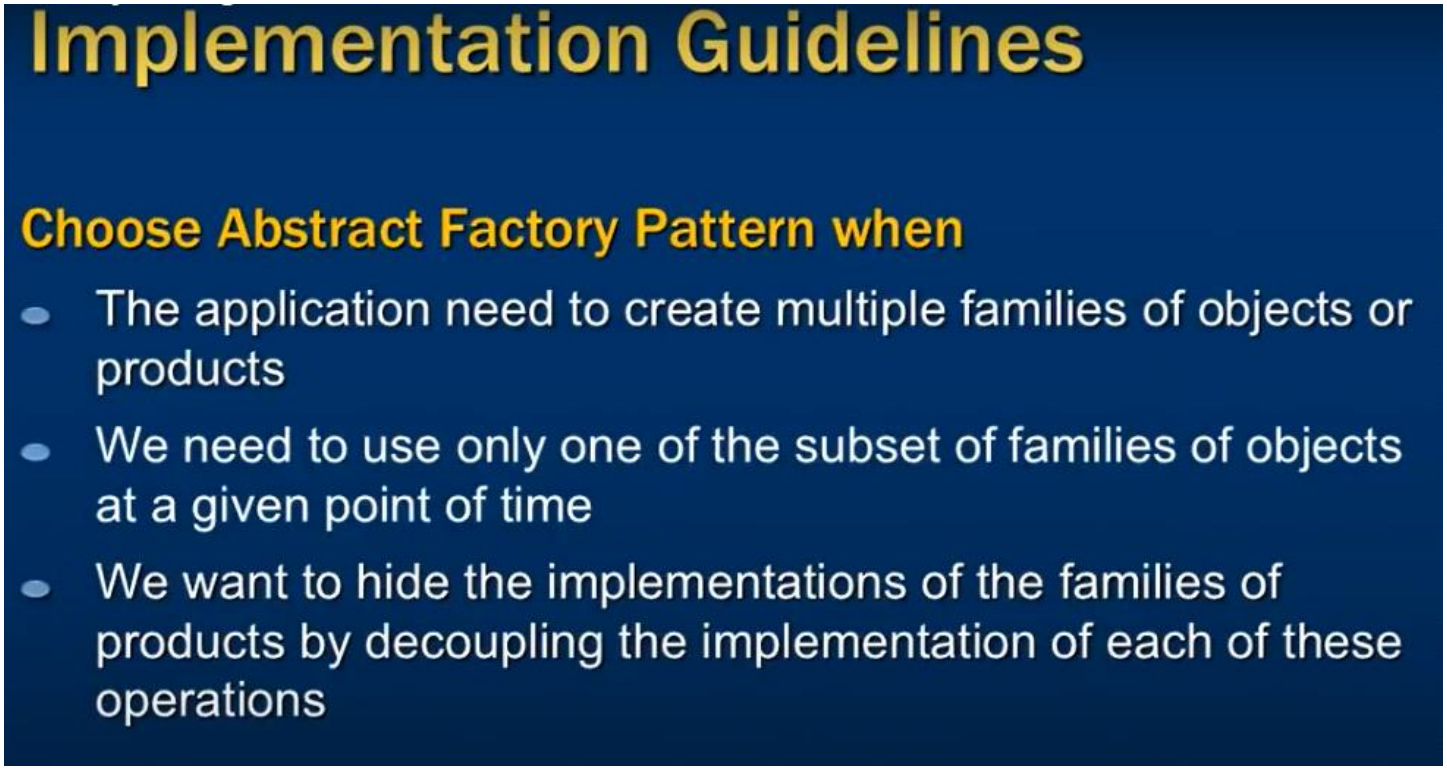



Gang Of Four Definition
"Define an interface(abstract class) for creating an object, but let subclasses decide which class to instantiate. The Factory method lets a class defer instantiation it uses to subclasses"
Employee Controller
Employee Manager Factory
Abstract Employee Factory
+GetPay
+GetBonus
+Create (Abstract)
Permanent Employee Factory +Create
Contract Employee Factory +Create
Permanent Employee Manager
Contract Employee Manager
IEmployee Manager

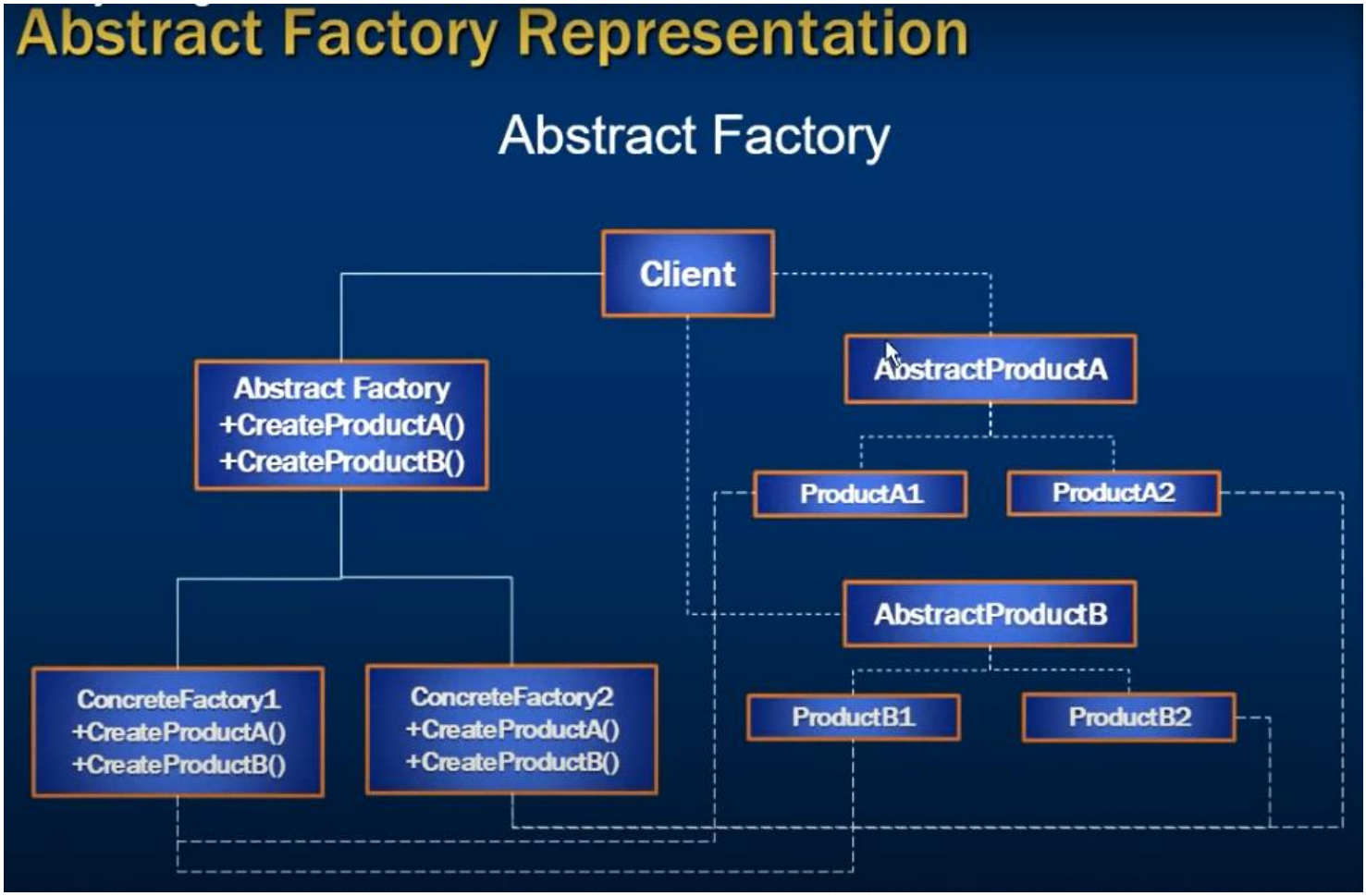
Abstract Factory Design Pattern

Friday, August 13, 2021 7:40 PM





Abstract Factory Representation



Abstract Factory Pattern Example

Business Requirement

- Handout computers to Contract and Permanent employees based on the designation and employee type with below specifications
- *Permanent Employee*
  - *Managerial Position is eligible for Apple MAC Book Laptop*
  - *Non Managerial Position is eligible for Apple IMac desktop*
- *Contract Employee*
  - *Managerial Position is eligible for Dell Laptop*
  - *Non Managerial Position is eligible for Dell desktop*