

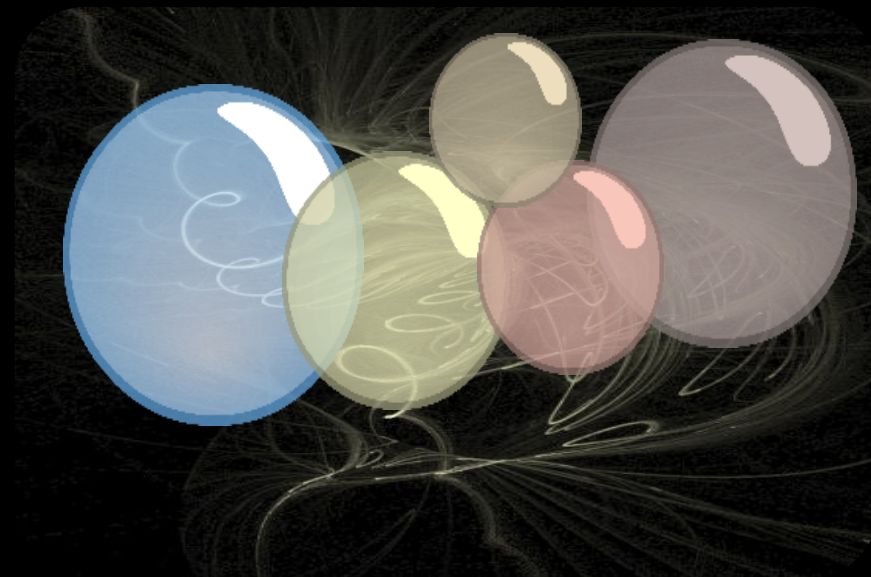
# Шаблони за проектиране при създаване на обекти (Creational Design Patterns)



Учителски екип

Обучение за ИТ кариера

<https://it-kariera.mon.bg/e-learning/>



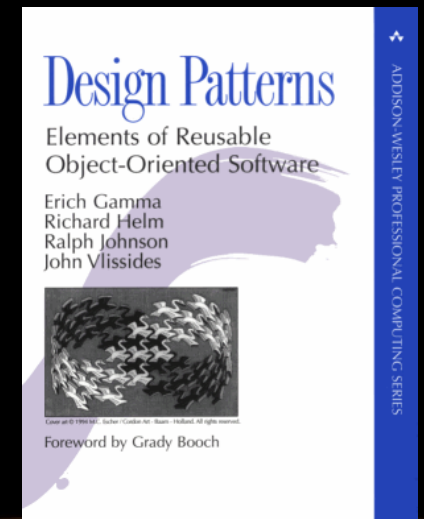
# Съдържание

1. Какво са шаблони в проектирането?
2. Типове шаблони в проектирането
3. Шаблони при създаването на обекти
  - Singleton шаблон
  - Factory шаблон
  - Шаблон на Factory метод




# Какво са шаблоните в проектирането?

- Софтуерните шаблони в проектирането (design patterns)
  - Готови решения на често срещани казуси в софтуерния дизайн
  - Двойка проблем / решение, валидно в даден контекст
  - Шаблон или рецепта за решаване на специфични казуси при проектирането на софтуера
- GoF шаблони
  - Класическа книга за обектно-ориентирани шаблони в проектирането от Gamma, Helm, Johnson, Vlissides 1995
  - Книгата „Шайката на четиримата“ (The Gang of Four)
  - Шаблони за създаване, структура и поведение





# Елементи на шаблоните в проектирането

- Шаблоните в проектирането се описват чрез няколко елемента:
    - Име на шаблона
      - Обогалява речника на дизайнера
    - Проблем
      - Предназначение и контекст на употреба
    - Решение
      - UML структура или абстрактен код
    - Последици
      - Резултат и негативи
- 



# Три основни типа на ОО шаблони в дизайна

- Шаблони за създаване
  - За инициализиране и конфигуриране на класове и обекти
- Шаблони в структурата
  - Начини за групиране на обекти за реализиране на нови функции
  - Сбор от класове или обекти
- Шаблони в поведението
  - За динамичното взаимодействие на общност от класове и обекти
  - Кой за какво отговаря и т.н.

# Шаблони при създаването

- Засягат механизмите на създаване на обекти
- Опитват се да създадат обекти по най-удачния за дадена ситуация начин
  - Вместо "**new SomeClass()**" ползвайте "**pattern.Create()**"
- Комбинация от две основни идеи
  - Капсулиране на знанието кои точно класове ползва системата
  - Скриване на това как екземпляри от тези конкретни класове са създадени и групирани

# Singleton шаблон

- Класът Singleton е такъв, който се предполага да има една-единствена инстанция
  - Обикновено се създава при поискване (lazy loading)
- Понякога **Singleton** е погрешно смятан за глобална променлива
  - Не е!
- Възможни проблеми:
  - Thread-safe

## Singleton

Type: Creational

### What it is:

Ensure a class only has one instance and provide a global point of access to it.

### Singleton

-static uniqueInstance
-singletonData
+static instance()
+SingletonOperation()

# Singleton – пример

```
public sealed class Singleton
{
    private Singleton() { }

    private static readonly Singleton instance = new Singleton();

    public static Singleton Instance
    {
        get
        {
            return instance;
        }
    }
}
```



# Factory шаблон

- В ООП Factory е обект за създаване на други обекти (алтернативен конструктор)
  - Това не е GoF шаблон; често е бъркан с **Factory метода**
- Традиционно създаване на обекти: **new** + викаме конструктор

```
DateTime t = new DateTime(2014, 10, 16);
```

- Създаване чрез **factory** (обикновено това е статичен метод):

```
DateTime t = DateTime.Now;
```

```
Color c = Color.FromArgb(120, 255, 0, 0);
```

# Factory – пример

```
public class Complex
{
    private double real;
    private double imaginary;

    public static Complex FromPolarFactory(double modulus, double angle)
    {
        return new Complex(
            modulus * Math.Cos(angle), modulus * Math.Sin(angle));
    }

    private Complex(double real, double imaginary)
    {
        this.real = real;
        this.imaginary = imaginary;
    }
}

Complex complexNum = Complex.FromPolarFactory(1, Math.PI / 3);
```

# Factory – варианти

- Factory шаблоните може да имат много варианти
  - Статични / нестатични методи за създаване на продукти
  - Връщащи класа на продукта или негов наследник
  - Factory във или извън класа на продукта
- Например:
  - Клас **Coffee** – съдържа смес от кафе и мляко
  - Клас **CoffeeFactory** – създава кафе, капучино / макиато
    - В зависимост от поръчания тип кафе

# Шаблон на Factory метод

## ■ Factory метод

- Създава обекти, без да указва точния им клас
- Създава обекти от някои от **подкласовете**, но връща базовия абстрактен клас или интерфейс

## ■ Ползи

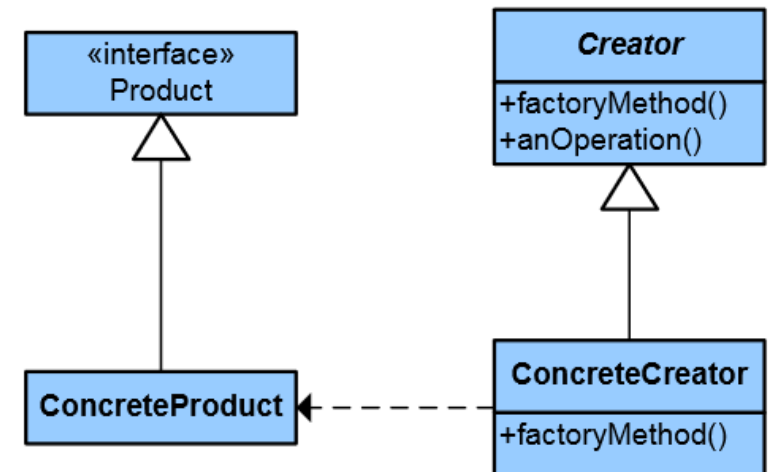
- Добавяне след време на нови подкласове
- Лесна разширяемост
- Лесна поддръжка

### Factory Method

Type: Creational

#### What it is:

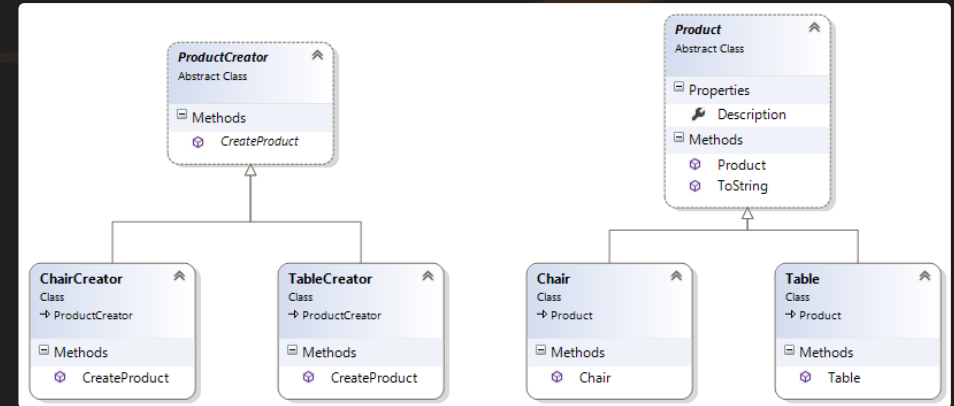
Define an interface for creating an object, but let subclasses decide which class to instantiate. Lets a class defer instantiation to subclasses.





# Factory метод – пример

```
public abstract class Product { ... }  
public class Chair : Product { ... }  
public class Table : Product { ... }  
public abstract class ProductCreator  
{  
    public abstract Product CreateProduct();  
}  
public class TableCreator : ProductCreator  
{  
    public override Product CreateProduct() { return new Table(...); }  
}  
public class ChairCreator : ProductCreator  
{  
    public override Product CreateProduct() { return new Chair(...); }  
}
```



# Обобщение

- Шаблоните в проектирането
  - Готови решения на често срещани казуси в ООП дизайна
- Шаблини за създаване
  - Singleton, Factory, Factory Method



# Шаблони за проектиране при създаване на обекти



Въпроси?



# Министерство на образованието и науката (МОН)

- Настоящият курс (презентации, примери, задачи, упражнения и др.) е разработен за нуждите на Национална програма "**Обучение за ИТ кариера**" на МОН за подготовка по професия "Приложен програмист"



Министерство  
на образованието  
и науката



Национална  
програма  
„Обучение за  
ИТ кариера“

- Курсът е базиран на учебно съдържание и методика, предоставени от **фондация "Софтуерен университет"** и се разпространява под свободен лиценз **CC-BY-NC-SA**



SoftUni  
Foundation

