

# 创建型模式

创建型模式(Creational Pattern)对类的实例化过程进行了抽象，能够将软件模块中对象的创建和对象的使用分离。为了使软件的结构更加清晰，外界对于这些对象只需要知道它们共同的接口，而不清楚其具体的实现细节，使整个系统的设计更加符合单一职责原则。

创建型模式在创建什么(What)，由谁创建(Who)，何时创建(When)等方面都为软件设计者提供了尽可能大的灵活性。创建型模式隐藏了类的实例的创建细节，通过隐藏对象如何被创建和组合在一起达到使整个系统独立的目的。

## 包含模式

- 简单工厂模式 (**Simple Factory**)  
重要程度：4 (5为满分)
- 工厂方法模式 (**Factory Method**)  
重要程度：5
- 抽象工厂模式 (**Abstract Factory**)  
重要程度：5
- 建造者模式 (**Builder**)  
重要程度：2
- 原型模式 (**Prototype**)  
重要程度：3
- 单例模式 (**Singleton**)  
重要程度：4

## 目录

- 1. 简单工厂模式( Simple Factory Pattern )
  - [1.1. 模式动机](#)
  - [1.2. 模式定义](#)
  - [1.3. 模式结构](#)
  - [1.4. 时序图](#)
  - [1.5. 代码分析](#)
  - [1.6. 模式分析](#)
  - [1.7. 实例](#)

- 1.8. 简单工厂模式的优点
- 1.9. 简单工厂模式的缺点
- 1.10. 适用环境
- 1.11. 模式应用
- 1.12. 总结
- 2. 工厂方法模式(Factory Method Pattern)
  - 2.1. 模式动机
  - 2.2. 模式定义
  - 2.3. 模式结构
  - 2.4. 时序图
  - 2.5. 代码分析
  - 2.6. 模式分析
  - 2.7. 实例
  - 2.8. 工厂方法模式的优点
  - 2.9. 工厂方法模式的缺点
  - 2.10. 适用环境
  - 2.11. 模式应用
  - 2.12. 模式扩展
  - 2.13. 总结
- 3. 抽象工厂模式(Abstract Factory)
  - 3.1. 模式动机
  - 3.2. 模式定义
  - 3.3. 模式结构
  - 3.4. 时序图
  - 3.5. 代码分析
  - 3.6. 模式分析
  - 3.7. 实例
  - 3.8. 优点
  - 3.9. 缺点
  - 3.10. 适用环境
  - 3.11. 模式应用
  - 3.12. 模式扩展
  - 3.13. 总结
- 4. 建造者模式
  - 4.1. 模式动机
  - 4.2. 模式定义
  - 4.3. 模式结构
  - 4.4. 时序图
  - 4.5. 代码分析
  - 4.6. 模式分析
  - 4.7. 实例
  - 4.8. 优点
  - 4.9. 缺点
  - 4.10. 适用环境
  - 4.11. 模式应用

- 4.12. 模式扩展
  - 4.13. 总结
- 5. 单例模式
  - 5.1. 模式动机
  - 5.2. 模式定义
  - 5.3. 模式结构
  - 5.4. 时序图
  - 5.5. 代码分析
  - 5.6. 模式分析
  - 5.7. 实例
  - 5.8. 优点
  - 5.9. 缺点
  - 5.10. 适用环境
  - 5.11. 模式应用
  - 5.12. 模式扩展
  - 5.13. 总结