**简单工厂**：提供专门的工厂类用于创建对象，将对象的创建和对象的使用分离开，一个方法，根据类型创建多个对象。

**工厂方法**模式：将简单工厂方法的判断丢到不同的工厂的具体实现中，而不变化的部分抽象为一个工厂接口。实质：推迟到子类去实例化对象。

**抽象工厂**模式：在工厂方法的单一产品线的基础上加上其他系列的产品线。

**建造者模式**：一步步构建一个包含多个组成部件的完整对象，使用相同的构建过程构建不同的产品

**适配器模式**：将现有接口转化为客户类所期望的接口，实现了对现有类的复用

**桥接模式**是一种很实用的结构型设计模式，如果软件系统中某个类存在两个独立变化的维度，通过该模式可以将这两个维度分离出来，使两者可以独立扩展，让系统更加符合“单一职责原则

**组合模式：**使用面向对象的思想来实现树形结构的构建与处理，描述了如何将容器对象和叶子对象进行递归组合，实现简单，灵活性好。

**装饰模式(Decorator Pattern)：**动态地给一个对象增加一些额外的职责，就增加对象功能来说，装饰模式比生成子类实现更为灵活。装饰模式是一种对象结构型模式。

**外观模式**：它通过引入一个外观角色来简化客户端与子系统之间的交互，为复杂的子系统调用提供一个统一的入口，使子系统与客户端的耦合度降低，且客户端调用非常方便。

**享元模式**：通过共享技术实现相同或相似的细粒度对象的复用，从而节约了内存空间，提高了系统性能

**代理模式**：给某一个对象提供一个代理或占位符，并由代理对象来控制对原对象的访问。当无法直接访问某个对象或访问某个对象存在困难时可以通过一个代理对象来间接访问，为了保证客户端使用的透明性，所访问的真实对象与代理对象需要实现相同的接口。

**职责链模式**(Chain of Responsibility  Pattern)：避免请求发送者与接收者耦合在一起，让多个对象都有可能接收请求，将这些对象连接成一条链，并且沿着这条链传递请求，直到有对象处理它为止。职责链模式是一种对象行为型模式。

命令模式(Command Pattern)：将一个请求封装为一个对象，从而让我们可用不同的请求对客户进行参数化；对请求排队或者记录请求日志，以及支持可撤销的操作。命令模式是一种对象行为型模式，其别名为动作(Action)模式或事务(Transaction)模式。命令模式可以将请求发送者和接收者完全解耦，发送者与接收者之间没有直接引用关系，发送请求的对象只需要知道如何发送请求，而不必知道如何完成请求。

**解释器模式(Interpreter Pattern)：**定义一个语言的文法，并且建立一个解释器来解释该语言中的句子，这里的“语言”是指使用规定格式和语法的代码。解释器模式是一种类行为型模式. 解释器模式为自定义语言的设计和实现提供了一种解决方案，它用于定义一组文法规则并通过这组文法规则来解释语言中的句子。虽然解释器模式的使用频率不是特别高，但是它在正则表达式、XML文档解释等领域还是得到了广泛使用。

**迭代器模式**：迭代器提供一种方法顺序访问一个聚合对象中各个元素, 而又无须暴露该对象的内部表示。遍历集合中元素的职能从集合本身转移到迭代器对象，其提供了一个用于访问集合元素并记录当前元素的接口，不同的迭代器可以执行不同的遍历策略。

**中介者模式(Mediator Pattern)：用一个中介对象（中介者）来封装一系列的对象交互，中介者使各对象不需要显式地相互引用，从而使其耦合松散，而且可以独立地改变它们之间的交互。中介者模式又称为调停者模式，它是一种对象行为型模式。**

**备忘录模式(Memento Pattern)：在不破坏封装的前提下，捕获一个对象的内部状态，并在该对象之外保存这个状态，这样可以在以后将对象恢复到原先保存的状态。它是一种对象行为型模式，其别名为Token。**

观察者模式(Observer Pattern)：定义对象之间的一种一对多依赖关系，使得每当一个对象状态发生改变时，其相关依赖对象皆得到通知并被自动更新。观察者模式的别名包括发布-订阅（Publish/Subscribe）模式、模型-视图（Model/View）模式、源-监听器（Source/Listener）模式或从属者（Dependents）模式。观察者模式是一种对象行为型模式。

**状态模式(State Pattern)：允许一个对象在其内部状态改变时改变它的行为，对象看起来似乎修改了它的类。其别名为状态对象(Objects for States)，状态模式是一种对象行为型模式。**状态模式将一个对象在不同状态下的不同行为封装在一个个状态类中，通过设置不同的状态对象可以让环境对象拥有不同的行为，而状态转换的细节对于客户端而言是透明的，方便了客户端的使用

**策略模式(Strategy Pattern)：**定义一系列算法类，将每一个算法封装起来，并让它们可以相互替换，策略模式让算法独立于使用它的客户而变化，也称为政策模式(Policy)。策略模式是一种对象行为型模式。

**模板方法模式：**定义一个操作中算法的框架，而将一些步骤延迟到子类中。模板方法模式使得子类可以不改变一个算法的结构即可重定义该算法的某些特定步骤，模板方法模式是一种基于继承的代码复用技术，它是一种类行为型模式。模板方法模式是结构最简单的行为型设计模式，在其结构中只存在父类与子类之间的继承关系。

**访问者模式(Visitor Pattern)**:提供一个作用于某对象结构中的各元素的操作表示，它使我们可以在不改变各元素的类的前提下定义作用于这些元素的新操作。访问者模式是一种对象行为型模式。