**一、 建造者（Builder）模式**

建造者模式可以将一个产品的内部表象与产品的生成过程分割开来，从而可以使一个建造过程生成具有不同的内部表象的产品对象。

**对象性质的建造**

有些情况下，一个对象会有一些重要的性质，在它们没有恰当的值之前，对象不能作为一个完整的产品使用。比如，一个电子邮件有发件人地址、收件人地址、主题、内容、附录等部分，而在最起码的收件人地址未被赋值之前，这个电子邮件不能发出。

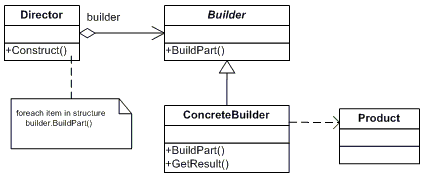
有些情况下，一个对象的一些性质必须按照某个顺序赋值才有意义。在某个性质没有赋值之前，另一个性质则无法赋值。这些情况使得性质本身的建造涉及到复杂的商业逻辑。

这时候，此对象相当于一个有待建造的产品，而对象的这些性质相当于产品的零件，建造产品的过程就是组合零件的过程。由于组合零件的过程很复杂，因此，这些"零件"的组合过程往往被"外部化"到一个称作建造者的对象里，建造者返还给客户端的是一个全部零件都建造完毕的产品对象。

**命名的考虑**

之所以使用"建造者"而没有用"生成器"就是因为用零件生产产品，"建造"更为合适，"创建"或"生成"不太恰当。

**二、 Builder模式的结构：**



建造者（Builder）角色：给出一个抽象接口，以规范产品对象的各个组成成分的建造。一般而言，此接口独立于应用程序的商业逻辑。模式中直接创建产品对象的是具体建造者（ConcreteBuilder）角色。具体建造者类必须实现这个接口所要求的方法：一个是建造方法，另一个是结果返还方法。

具体建造者（Concrete Builder）角色：担任这个角色的是于应用程序紧密相关的类，它们在应用程序调用下创建产品实例。这个角色主要完成的任务包括：

* 实现Builder角色提供的接口，一步一步完成创建产品实例的过程。
* 在建造过程完成后，提供产品的实例。

指导者（Director）角色：担任这个角色的类调用具体建造者角色以创建产品对象。导演者并没有产品类的具体知识，真正拥有产品类的具体知识的是具体建造者对象。

产品（Product）角色：产品便是建造中的复杂对象。

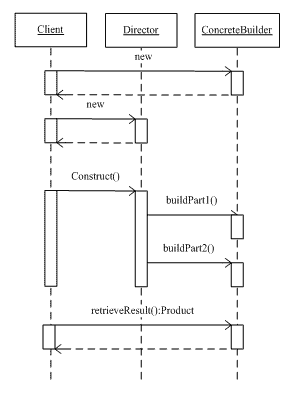
指导者角色是于客户端打交道的角色。导演者角色将客户端创建产品的请求划分为对各个零件的建造请求，再将这些请求委派给具体建造者角色。具体建造者角色是做具体建造工作的，但却不为客户端所知。

**三、 程序举例：**

该程序演示了Builder模式一步一步完成构件复杂产品的过程。用户可以控制生成过程以及生成不同对象。

http://www.cnblogs.com/Images/OutliningIndicators/None.gif// Builder pattern -- Structural example    
http://www.cnblogs.com/Images/OutliningIndicators/None.gif  
http://www.cnblogs.com/Images/OutliningIndicators/None.gifusing System;  
http://www.cnblogs.com/Images/OutliningIndicators/None.gifusing System.Collections;  
http://www.cnblogs.com/Images/OutliningIndicators/None.gif  
http://www.cnblogs.com/Images/OutliningIndicators/None.gif// "Director"  
http://www.cnblogs.com/Images/OutliningIndicators/None.gifclass Director  
http://www.cnblogs.com/Images/OutliningIndicators/ExpandedBlockStart.gif{  
http://www.cnblogs.com/Images/OutliningIndicators/InBlock.gif  // Methods  
http://www.cnblogs.com/Images/OutliningIndicators/InBlock.gif  public void Construct( Builder builder )  
http://www.cnblogs.com/Images/OutliningIndicators/ExpandedSubBlockStart.gif  {  
http://www.cnblogs.com/Images/OutliningIndicators/InBlock.gif    builder.BuildPartA();  
http://www.cnblogs.com/Images/OutliningIndicators/InBlock.gif    builder.BuildPartB();  
http://www.cnblogs.com/Images/OutliningIndicators/ExpandedSubBlockEnd.gif  }  
http://www.cnblogs.com/Images/OutliningIndicators/ExpandedBlockEnd.gif}  
http://www.cnblogs.com/Images/OutliningIndicators/None.gif  
http://www.cnblogs.com/Images/OutliningIndicators/None.gif// "Builder"  
http://www.cnblogs.com/Images/OutliningIndicators/None.gifabstract class Builder  
http://www.cnblogs.com/Images/OutliningIndicators/ExpandedBlockStart.gif{  
http://www.cnblogs.com/Images/OutliningIndicators/InBlock.gif  // Methods  
http://www.cnblogs.com/Images/OutliningIndicators/InBlock.gif  abstract public void BuildPartA();  
http://www.cnblogs.com/Images/OutliningIndicators/InBlock.gif  abstract public void BuildPartB();  
http://www.cnblogs.com/Images/OutliningIndicators/InBlock.gif  abstract public Product GetResult();  
http://www.cnblogs.com/Images/OutliningIndicators/ExpandedBlockEnd.gif}  
http://www.cnblogs.com/Images/OutliningIndicators/None.gif  
http://www.cnblogs.com/Images/OutliningIndicators/None.gif// "ConcreteBuilder1"  
http://www.cnblogs.com/Images/OutliningIndicators/None.gifclass ConcreteBuilder1 : Builder  
http://www.cnblogs.com/Images/OutliningIndicators/ExpandedBlockStart.gif{  
http://www.cnblogs.com/Images/OutliningIndicators/InBlock.gif  // Fields  
http://www.cnblogs.com/Images/OutliningIndicators/InBlock.gif  private Product product;  
http://www.cnblogs.com/Images/OutliningIndicators/InBlock.gif  
http://www.cnblogs.com/Images/OutliningIndicators/InBlock.gif  // Methods  
http://www.cnblogs.com/Images/OutliningIndicators/InBlock.gif  override public void BuildPartA()  
http://www.cnblogs.com/Images/OutliningIndicators/ExpandedSubBlockStart.gif  {  
http://www.cnblogs.com/Images/OutliningIndicators/InBlock.gif    product = new Product();  
http://www.cnblogs.com/Images/OutliningIndicators/InBlock.gif    product.Add( "PartA" );  
http://www.cnblogs.com/Images/OutliningIndicators/ExpandedSubBlockEnd.gif  }  
http://www.cnblogs.com/Images/OutliningIndicators/InBlock.gif  
http://www.cnblogs.com/Images/OutliningIndicators/InBlock.gif  override public void BuildPartB()  
http://www.cnblogs.com/Images/OutliningIndicators/ExpandedSubBlockStart.gif  {  
http://www.cnblogs.com/Images/OutliningIndicators/InBlock.gif    product.Add( "PartB" );  
http://www.cnblogs.com/Images/OutliningIndicators/ExpandedSubBlockEnd.gif  }  
http://www.cnblogs.com/Images/OutliningIndicators/InBlock.gif  
http://www.cnblogs.com/Images/OutliningIndicators/InBlock.gif  override public Product GetResult()  
http://www.cnblogs.com/Images/OutliningIndicators/ExpandedSubBlockStart.gif  {  
http://www.cnblogs.com/Images/OutliningIndicators/InBlock.gif    return product;  
http://www.cnblogs.com/Images/OutliningIndicators/ExpandedSubBlockEnd.gif  }  
http://www.cnblogs.com/Images/OutliningIndicators/ExpandedBlockEnd.gif}  
http://www.cnblogs.com/Images/OutliningIndicators/None.gif  
http://www.cnblogs.com/Images/OutliningIndicators/None.gif// "ConcreteBuilder2"  
http://www.cnblogs.com/Images/OutliningIndicators/None.gifclass ConcreteBuilder2 : Builder  
http://www.cnblogs.com/Images/OutliningIndicators/ExpandedBlockStart.gif{  
http://www.cnblogs.com/Images/OutliningIndicators/InBlock.gif  // Fields  
http://www.cnblogs.com/Images/OutliningIndicators/InBlock.gif  private Product product;  
http://www.cnblogs.com/Images/OutliningIndicators/InBlock.gif  
http://www.cnblogs.com/Images/OutliningIndicators/InBlock.gif  // Methods  
http://www.cnblogs.com/Images/OutliningIndicators/InBlock.gif  override public void BuildPartA()  
http://www.cnblogs.com/Images/OutliningIndicators/ExpandedSubBlockStart.gif  {  
http://www.cnblogs.com/Images/OutliningIndicators/InBlock.gif    product = new Product();  
http://www.cnblogs.com/Images/OutliningIndicators/InBlock.gif    product.Add( "PartX" );  
http://www.cnblogs.com/Images/OutliningIndicators/ExpandedSubBlockEnd.gif  }  
http://www.cnblogs.com/Images/OutliningIndicators/InBlock.gif  
http://www.cnblogs.com/Images/OutliningIndicators/InBlock.gif  override public void BuildPartB()  
http://www.cnblogs.com/Images/OutliningIndicators/ExpandedSubBlockStart.gif  {  
http://www.cnblogs.com/Images/OutliningIndicators/InBlock.gif    product.Add( "PartY" );  
http://www.cnblogs.com/Images/OutliningIndicators/ExpandedSubBlockEnd.gif  }  
http://www.cnblogs.com/Images/OutliningIndicators/InBlock.gif  
http://www.cnblogs.com/Images/OutliningIndicators/InBlock.gif  override public Product GetResult()  
http://www.cnblogs.com/Images/OutliningIndicators/ExpandedSubBlockStart.gif  {  
http://www.cnblogs.com/Images/OutliningIndicators/InBlock.gif    return product;  
http://www.cnblogs.com/Images/OutliningIndicators/ExpandedSubBlockEnd.gif  }  
http://www.cnblogs.com/Images/OutliningIndicators/ExpandedBlockEnd.gif}  
http://www.cnblogs.com/Images/OutliningIndicators/None.gif  
http://www.cnblogs.com/Images/OutliningIndicators/None.gif// "Product"  
http://www.cnblogs.com/Images/OutliningIndicators/None.gifclass Product  
http://www.cnblogs.com/Images/OutliningIndicators/ExpandedBlockStart.gif{  
http://www.cnblogs.com/Images/OutliningIndicators/InBlock.gif  // Fields  
http://www.cnblogs.com/Images/OutliningIndicators/InBlock.gif  ArrayList parts = new ArrayList();  
http://www.cnblogs.com/Images/OutliningIndicators/InBlock.gif   
http://www.cnblogs.com/Images/OutliningIndicators/InBlock.gif  // Methods  
http://www.cnblogs.com/Images/OutliningIndicators/InBlock.gif  public void Add( string part )  
http://www.cnblogs.com/Images/OutliningIndicators/ExpandedSubBlockStart.gif  {  
http://www.cnblogs.com/Images/OutliningIndicators/InBlock.gif    parts.Add( part );  
http://www.cnblogs.com/Images/OutliningIndicators/ExpandedSubBlockEnd.gif  }  
http://www.cnblogs.com/Images/OutliningIndicators/InBlock.gif  
http://www.cnblogs.com/Images/OutliningIndicators/InBlock.gif  public void Show()  
http://www.cnblogs.com/Images/OutliningIndicators/ExpandedSubBlockStart.gif  {  
http://www.cnblogs.com/Images/OutliningIndicators/InBlock.gif    Console.WriteLine( " Product Parts -------" );  
http://www.cnblogs.com/Images/OutliningIndicators/InBlock.gif    foreach( string part in parts )  
http://www.cnblogs.com/Images/OutliningIndicators/InBlock.gif      Console.WriteLine( part );  
http://www.cnblogs.com/Images/OutliningIndicators/ExpandedSubBlockEnd.gif  }  
http://www.cnblogs.com/Images/OutliningIndicators/ExpandedBlockEnd.gif}  
http://www.cnblogs.com/Images/OutliningIndicators/None.gif  
http://www.cnblogs.com/Images/OutliningIndicators/ExpandedBlockStart.gif/// <summary>  
http://www.cnblogs.com/Images/OutliningIndicators/InBlock.gif/// Client test  
http://www.cnblogs.com/Images/OutliningIndicators/ExpandedBlockEnd.gif/// </summary>  
http://www.cnblogs.com/Images/OutliningIndicators/None.gifpublic class Client  
http://www.cnblogs.com/Images/OutliningIndicators/ExpandedBlockStart.gif{  
http://www.cnblogs.com/Images/OutliningIndicators/InBlock.gif  public static void Main( string[] args )  
http://www.cnblogs.com/Images/OutliningIndicators/ExpandedSubBlockStart.gif  {  
http://www.cnblogs.com/Images/OutliningIndicators/InBlock.gif    // Create director and builders  
http://www.cnblogs.com/Images/OutliningIndicators/InBlock.gif    Director director = new Director( );  
http://www.cnblogs.com/Images/OutliningIndicators/InBlock.gif  
http://www.cnblogs.com/Images/OutliningIndicators/InBlock.gif    Builder b1 = new ConcreteBuilder1();  
http://www.cnblogs.com/Images/OutliningIndicators/InBlock.gif    Builder b2 = new ConcreteBuilder2();  
http://www.cnblogs.com/Images/OutliningIndicators/InBlock.gif  
http://www.cnblogs.com/Images/OutliningIndicators/InBlock.gif    // Construct two products  
http://www.cnblogs.com/Images/OutliningIndicators/InBlock.gif    director.Construct( b1 );  
http://www.cnblogs.com/Images/OutliningIndicators/InBlock.gif    Product p1 = b1.GetResult();  
http://www.cnblogs.com/Images/OutliningIndicators/InBlock.gif    p1.Show();  
http://www.cnblogs.com/Images/OutliningIndicators/InBlock.gif  
http://www.cnblogs.com/Images/OutliningIndicators/InBlock.gif    director.Construct( b2 );  
http://www.cnblogs.com/Images/OutliningIndicators/InBlock.gif    Product p2 = b2.GetResult();  
http://www.cnblogs.com/Images/OutliningIndicators/InBlock.gif    p2.Show();  
http://www.cnblogs.com/Images/OutliningIndicators/ExpandedSubBlockEnd.gif  }  
http://www.cnblogs.com/Images/OutliningIndicators/ExpandedBlockEnd.gif}

**四、 建造者模式的活动序列：**



客户端负责创建指导者和具体建造者对象。然后，客户把具体建造者对象交给指导者。客户一声令下，指导者操纵建造者开始创建产品。当产品创建完成后，建造者把产品返还给客户端。

**五、 建造者模式的实现：**

下面的程序代码演示了Shop对象使用VehicleBuilders来建造不同的交通工具。该例子使用了Builder模式顺序建造交通工具的不同部分。

http://www.cnblogs.com/Images/OutliningIndicators/None.gif// Builder pattern -- Real World example    
http://www.cnblogs.com/Images/OutliningIndicators/None.gif  
http://www.cnblogs.com/Images/OutliningIndicators/None.gifusing System;  
http://www.cnblogs.com/Images/OutliningIndicators/None.gifusing System.Collections;  
http://www.cnblogs.com/Images/OutliningIndicators/None.gif  
http://www.cnblogs.com/Images/OutliningIndicators/None.gif// "Director"  
http://www.cnblogs.com/Images/OutliningIndicators/None.gifclass Shop  
http://www.cnblogs.com/Images/OutliningIndicators/ExpandedBlockStart.gif{  
http://www.cnblogs.com/Images/OutliningIndicators/InBlock.gif  // Methods  
http://www.cnblogs.com/Images/OutliningIndicators/InBlock.gif  public void Construct( VehicleBuilder vehicleBuilder )  
http://www.cnblogs.com/Images/OutliningIndicators/ExpandedSubBlockStart.gif  {  
http://www.cnblogs.com/Images/OutliningIndicators/InBlock.gif    vehicleBuilder.BuildFrame();  
http://www.cnblogs.com/Images/OutliningIndicators/InBlock.gif    vehicleBuilder.BuildEngine();  
http://www.cnblogs.com/Images/OutliningIndicators/InBlock.gif    vehicleBuilder.BuildWheels();  
http://www.cnblogs.com/Images/OutliningIndicators/InBlock.gif    vehicleBuilder.BuildDoors();  
http://www.cnblogs.com/Images/OutliningIndicators/ExpandedSubBlockEnd.gif  }  
http://www.cnblogs.com/Images/OutliningIndicators/ExpandedBlockEnd.gif}  
http://www.cnblogs.com/Images/OutliningIndicators/None.gif  
http://www.cnblogs.com/Images/OutliningIndicators/None.gif// "Builder"  
http://www.cnblogs.com/Images/OutliningIndicators/None.gifabstract class VehicleBuilder  
http://www.cnblogs.com/Images/OutliningIndicators/ExpandedBlockStart.gif{  
http://www.cnblogs.com/Images/OutliningIndicators/InBlock.gif  // Fields  
http://www.cnblogs.com/Images/OutliningIndicators/InBlock.gif  protected Vehicle vehicle;  
http://www.cnblogs.com/Images/OutliningIndicators/InBlock.gif  
http://www.cnblogs.com/Images/OutliningIndicators/InBlock.gif  // Properties  
http://www.cnblogs.com/Images/OutliningIndicators/InBlock.gif  public Vehicle Vehicle  
http://www.cnblogs.com/Images/OutliningIndicators/ExpandedSubBlockStart.gif  {  
http://www.cnblogs.com/Images/OutliningIndicators/ExpandedSubBlockStart.gif    get{ return vehicle; }  
http://www.cnblogs.com/Images/OutliningIndicators/ExpandedSubBlockEnd.gif  }  
http://www.cnblogs.com/Images/OutliningIndicators/InBlock.gif  
http://www.cnblogs.com/Images/OutliningIndicators/InBlock.gif  // Methods  
http://www.cnblogs.com/Images/OutliningIndicators/InBlock.gif  abstract public void BuildFrame();  
http://www.cnblogs.com/Images/OutliningIndicators/InBlock.gif  abstract public void BuildEngine();  
http://www.cnblogs.com/Images/OutliningIndicators/InBlock.gif  abstract public void BuildWheels();  
http://www.cnblogs.com/Images/OutliningIndicators/InBlock.gif  abstract public void BuildDoors();  
http://www.cnblogs.com/Images/OutliningIndicators/ExpandedBlockEnd.gif}  
http://www.cnblogs.com/Images/OutliningIndicators/None.gif  
http://www.cnblogs.com/Images/OutliningIndicators/None.gif// "ConcreteBuilder1"  
http://www.cnblogs.com/Images/OutliningIndicators/None.gifclass MotorCycleBuilder : VehicleBuilder  
http://www.cnblogs.com/Images/OutliningIndicators/ExpandedBlockStart.gif{  
http://www.cnblogs.com/Images/OutliningIndicators/InBlock.gif  // Methods  
http://www.cnblogs.com/Images/OutliningIndicators/InBlock.gif  override public void BuildFrame()  
http://www.cnblogs.com/Images/OutliningIndicators/ExpandedSubBlockStart.gif  {  
http://www.cnblogs.com/Images/OutliningIndicators/InBlock.gif    vehicle = new Vehicle( "MotorCycle" );  
http://www.cnblogs.com/Images/OutliningIndicators/InBlock.gif    vehicle[ "frame" ] = "MotorCycle Frame";  
http://www.cnblogs.com/Images/OutliningIndicators/ExpandedSubBlockEnd.gif  }  
http://www.cnblogs.com/Images/OutliningIndicators/InBlock.gif  
http://www.cnblogs.com/Images/OutliningIndicators/InBlock.gif  override public void BuildEngine()  
http://www.cnblogs.com/Images/OutliningIndicators/ExpandedSubBlockStart.gif  {  
http://www.cnblogs.com/Images/OutliningIndicators/InBlock.gif    vehicle[ "engine" ] = "500 cc";  
http://www.cnblogs.com/Images/OutliningIndicators/ExpandedSubBlockEnd.gif  }  
http://www.cnblogs.com/Images/OutliningIndicators/InBlock.gif  
http://www.cnblogs.com/Images/OutliningIndicators/InBlock.gif  override public void BuildWheels()  
http://www.cnblogs.com/Images/OutliningIndicators/ExpandedSubBlockStart.gif  {  
http://www.cnblogs.com/Images/OutliningIndicators/InBlock.gif    vehicle[ "wheels" ] = "2";  
http://www.cnblogs.com/Images/OutliningIndicators/ExpandedSubBlockEnd.gif  }  
http://www.cnblogs.com/Images/OutliningIndicators/InBlock.gif  
http://www.cnblogs.com/Images/OutliningIndicators/InBlock.gif  override public void BuildDoors()  
http://www.cnblogs.com/Images/OutliningIndicators/ExpandedSubBlockStart.gif  {  
http://www.cnblogs.com/Images/OutliningIndicators/InBlock.gif    vehicle[ "doors" ] = "0";  
http://www.cnblogs.com/Images/OutliningIndicators/ExpandedSubBlockEnd.gif  }  
http://www.cnblogs.com/Images/OutliningIndicators/ExpandedBlockEnd.gif}  
http://www.cnblogs.com/Images/OutliningIndicators/None.gif  
http://www.cnblogs.com/Images/OutliningIndicators/None.gif// "ConcreteBuilder2"  
http://www.cnblogs.com/Images/OutliningIndicators/None.gifclass CarBuilder : VehicleBuilder  
http://www.cnblogs.com/Images/OutliningIndicators/ExpandedBlockStart.gif{  
http://www.cnblogs.com/Images/OutliningIndicators/InBlock.gif  // Methods  
http://www.cnblogs.com/Images/OutliningIndicators/InBlock.gif  override public void BuildFrame()  
http://www.cnblogs.com/Images/OutliningIndicators/ExpandedSubBlockStart.gif  {  
http://www.cnblogs.com/Images/OutliningIndicators/InBlock.gif    vehicle = new Vehicle( "Car" );  
http://www.cnblogs.com/Images/OutliningIndicators/InBlock.gif    vehicle[ "frame" ] = "Car Frame";  
http://www.cnblogs.com/Images/OutliningIndicators/ExpandedSubBlockEnd.gif  }  
http://www.cnblogs.com/Images/OutliningIndicators/InBlock.gif  
http://www.cnblogs.com/Images/OutliningIndicators/InBlock.gif  override public void BuildEngine()  
http://www.cnblogs.com/Images/OutliningIndicators/ExpandedSubBlockStart.gif  {  
http://www.cnblogs.com/Images/OutliningIndicators/InBlock.gif    vehicle[ "engine" ] = "2500 cc";  
http://www.cnblogs.com/Images/OutliningIndicators/ExpandedSubBlockEnd.gif  }  
http://www.cnblogs.com/Images/OutliningIndicators/InBlock.gif  
http://www.cnblogs.com/Images/OutliningIndicators/InBlock.gif  override public void BuildWheels()  
http://www.cnblogs.com/Images/OutliningIndicators/ExpandedSubBlockStart.gif  {  
http://www.cnblogs.com/Images/OutliningIndicators/InBlock.gif    vehicle[ "wheels" ] = "4";  
http://www.cnblogs.com/Images/OutliningIndicators/ExpandedSubBlockEnd.gif  }  
http://www.cnblogs.com/Images/OutliningIndicators/InBlock.gif  
http://www.cnblogs.com/Images/OutliningIndicators/InBlock.gif  override public void BuildDoors()  
http://www.cnblogs.com/Images/OutliningIndicators/ExpandedSubBlockStart.gif  {  
http://www.cnblogs.com/Images/OutliningIndicators/InBlock.gif    vehicle[ "doors" ] = "4";  
http://www.cnblogs.com/Images/OutliningIndicators/ExpandedSubBlockEnd.gif  }  
http://www.cnblogs.com/Images/OutliningIndicators/ExpandedBlockEnd.gif}  
http://www.cnblogs.com/Images/OutliningIndicators/None.gif  
http://www.cnblogs.com/Images/OutliningIndicators/None.gif// "ConcreteBuilder3"  
http://www.cnblogs.com/Images/OutliningIndicators/None.gifclass ScooterBuilder : VehicleBuilder  
http://www.cnblogs.com/Images/OutliningIndicators/ExpandedBlockStart.gif{  
http://www.cnblogs.com/Images/OutliningIndicators/InBlock.gif  // Methods  
http://www.cnblogs.com/Images/OutliningIndicators/InBlock.gif  override public void BuildFrame()  
http://www.cnblogs.com/Images/OutliningIndicators/ExpandedSubBlockStart.gif  {  
http://www.cnblogs.com/Images/OutliningIndicators/InBlock.gif    vehicle = new Vehicle( "Scooter" );  
http://www.cnblogs.com/Images/OutliningIndicators/InBlock.gif    vehicle[ "frame" ] = "Scooter Frame";  
http://www.cnblogs.com/Images/OutliningIndicators/ExpandedSubBlockEnd.gif  }  
http://www.cnblogs.com/Images/OutliningIndicators/InBlock.gif  
http://www.cnblogs.com/Images/OutliningIndicators/InBlock.gif  override public void BuildEngine()  
http://www.cnblogs.com/Images/OutliningIndicators/ExpandedSubBlockStart.gif  {  
http://www.cnblogs.com/Images/OutliningIndicators/InBlock.gif    vehicle[ "engine" ] = "none";  
http://www.cnblogs.com/Images/OutliningIndicators/ExpandedSubBlockEnd.gif  }  
http://www.cnblogs.com/Images/OutliningIndicators/InBlock.gif  
http://www.cnblogs.com/Images/OutliningIndicators/InBlock.gif  override public void BuildWheels()  
http://www.cnblogs.com/Images/OutliningIndicators/ExpandedSubBlockStart.gif  {  
http://www.cnblogs.com/Images/OutliningIndicators/InBlock.gif    vehicle[ "wheels" ] = "2";  
http://www.cnblogs.com/Images/OutliningIndicators/ExpandedSubBlockEnd.gif  }  
http://www.cnblogs.com/Images/OutliningIndicators/InBlock.gif  
http://www.cnblogs.com/Images/OutliningIndicators/InBlock.gif  override public void BuildDoors()  
http://www.cnblogs.com/Images/OutliningIndicators/ExpandedSubBlockStart.gif  {  
http://www.cnblogs.com/Images/OutliningIndicators/InBlock.gif    vehicle[ "doors" ] = "0";  
http://www.cnblogs.com/Images/OutliningIndicators/ExpandedSubBlockEnd.gif  }  
http://www.cnblogs.com/Images/OutliningIndicators/ExpandedBlockEnd.gif}  
http://www.cnblogs.com/Images/OutliningIndicators/None.gif  
http://www.cnblogs.com/Images/OutliningIndicators/None.gif// "Product"  
http://www.cnblogs.com/Images/OutliningIndicators/None.gifclass Vehicle  
http://www.cnblogs.com/Images/OutliningIndicators/ExpandedBlockStart.gif{  
http://www.cnblogs.com/Images/OutliningIndicators/InBlock.gif  // Fields  
http://www.cnblogs.com/Images/OutliningIndicators/InBlock.gif  private string type;  
http://www.cnblogs.com/Images/OutliningIndicators/InBlock.gif  private Hashtable parts = new Hashtable();  
http://www.cnblogs.com/Images/OutliningIndicators/InBlock.gif  
http://www.cnblogs.com/Images/OutliningIndicators/InBlock.gif  // Constructors  
http://www.cnblogs.com/Images/OutliningIndicators/InBlock.gif  public Vehicle( string type )  
http://www.cnblogs.com/Images/OutliningIndicators/ExpandedSubBlockStart.gif  {  
http://www.cnblogs.com/Images/OutliningIndicators/InBlock.gif    this.type = type;  
http://www.cnblogs.com/Images/OutliningIndicators/ExpandedSubBlockEnd.gif  }  
http://www.cnblogs.com/Images/OutliningIndicators/InBlock.gif  
http://www.cnblogs.com/Images/OutliningIndicators/InBlock.gif  // Indexers  
http://www.cnblogs.com/Images/OutliningIndicators/InBlock.gif  public object this[ string key ]  
http://www.cnblogs.com/Images/OutliningIndicators/ExpandedSubBlockStart.gif  {  
http://www.cnblogs.com/Images/OutliningIndicators/ExpandedSubBlockStart.gif    get{ return parts[ key ]; }  
http://www.cnblogs.com/Images/OutliningIndicators/ExpandedSubBlockStart.gif    set{ parts[ key ] = value; }  
http://www.cnblogs.com/Images/OutliningIndicators/ExpandedSubBlockEnd.gif  }  
http://www.cnblogs.com/Images/OutliningIndicators/InBlock.gif  
http://www.cnblogs.com/Images/OutliningIndicators/InBlock.gif  // Methods  
http://www.cnblogs.com/Images/OutliningIndicators/InBlock.gif  public void Show()  
http://www.cnblogs.com/Images/OutliningIndicators/ExpandedSubBlockStart.gif  {  
http://www.cnblogs.com/Images/OutliningIndicators/InBlock.gif    Console.WriteLine( " ---------------------------");  
http://www.cnblogs.com/Images/OutliningIndicators/InBlock.gif    Console.WriteLine( "Vehicle Type: "+ type );  
http://www.cnblogs.com/Images/OutliningIndicators/InBlock.gif    Console.WriteLine( " Frame : " + parts[ "frame" ] );  
http://www.cnblogs.com/Images/OutliningIndicators/InBlock.gif    Console.WriteLine( " Engine : "+ parts[ "engine"] );  
http://www.cnblogs.com/Images/OutliningIndicators/InBlock.gif    Console.WriteLine( " #Wheels: "+ parts[ "wheels"] );  
http://www.cnblogs.com/Images/OutliningIndicators/InBlock.gif    Console.WriteLine( " #Doors : "+ parts[ "doors" ] );  
http://www.cnblogs.com/Images/OutliningIndicators/ExpandedSubBlockEnd.gif  }  
http://www.cnblogs.com/Images/OutliningIndicators/ExpandedBlockEnd.gif}  
http://www.cnblogs.com/Images/OutliningIndicators/None.gif  
http://www.cnblogs.com/Images/OutliningIndicators/ExpandedBlockStart.gif/// <summary>  
http://www.cnblogs.com/Images/OutliningIndicators/InBlock.gif/// BuilderApp test  
http://www.cnblogs.com/Images/OutliningIndicators/ExpandedBlockEnd.gif/// </summary>  
http://www.cnblogs.com/Images/OutliningIndicators/None.gifpublic class BuilderApp  
http://www.cnblogs.com/Images/OutliningIndicators/ExpandedBlockStart.gif{  
http://www.cnblogs.com/Images/OutliningIndicators/InBlock.gif  public static void Main( string[] args )  
http://www.cnblogs.com/Images/OutliningIndicators/ExpandedSubBlockStart.gif  {  
http://www.cnblogs.com/Images/OutliningIndicators/InBlock.gif    // Create shop and vehicle builders  
http://www.cnblogs.com/Images/OutliningIndicators/InBlock.gif    Shop shop = new Shop();  
http://www.cnblogs.com/Images/OutliningIndicators/InBlock.gif    VehicleBuilder b1 = new ScooterBuilder();  
http://www.cnblogs.com/Images/OutliningIndicators/InBlock.gif    VehicleBuilder b2 = new CarBuilder();  
http://www.cnblogs.com/Images/OutliningIndicators/InBlock.gif    VehicleBuilder b3 = new MotorCycleBuilder();  
http://www.cnblogs.com/Images/OutliningIndicators/InBlock.gif  
http://www.cnblogs.com/Images/OutliningIndicators/InBlock.gif    // Construct and display vehicles  
http://www.cnblogs.com/Images/OutliningIndicators/InBlock.gif    shop.Construct( b1 );  
http://www.cnblogs.com/Images/OutliningIndicators/InBlock.gif    b1.Vehicle.Show();  
http://www.cnblogs.com/Images/OutliningIndicators/InBlock.gif  
http://www.cnblogs.com/Images/OutliningIndicators/InBlock.gif    shop.Construct( b2 );  
http://www.cnblogs.com/Images/OutliningIndicators/InBlock.gif    b2.Vehicle.Show();  
http://www.cnblogs.com/Images/OutliningIndicators/InBlock.gif  
http://www.cnblogs.com/Images/OutliningIndicators/InBlock.gif    shop.Construct( b3 );  
http://www.cnblogs.com/Images/OutliningIndicators/InBlock.gif    b3.Vehicle.Show();  
http://www.cnblogs.com/Images/OutliningIndicators/ExpandedSubBlockEnd.gif  }  
http://www.cnblogs.com/Images/OutliningIndicators/ExpandedBlockEnd.gif}

**六、 建造者模式的演化**

建造者模式在使用的过程中可以演化出多种形式。

**省略抽象建造者角色**

如果系统中只需要一个具体建造者的话，可以省略掉抽象建造者。这时代码可能如下：

http://www.cnblogs.com/Images/OutliningIndicators/None.gif// "Director"  
http://www.cnblogs.com/Images/OutliningIndicators/None.gifclass Director  
http://www.cnblogs.com/Images/OutliningIndicators/ExpandedBlockStart.gif{  
http://www.cnblogs.com/Images/OutliningIndicators/InBlock.gif  private ConcreteBuilder builder;  
http://www.cnblogs.com/Images/OutliningIndicators/InBlock.gif  
http://www.cnblogs.com/Images/OutliningIndicators/InBlock.gif  // Methods  
http://www.cnblogs.com/Images/OutliningIndicators/InBlock.gif  public void Construct()  
http://www.cnblogs.com/Images/OutliningIndicators/ExpandedSubBlockStart.gif  {  
http://www.cnblogs.com/Images/OutliningIndicators/InBlock.gif    builder.BuildPartA();  
http://www.cnblogs.com/Images/OutliningIndicators/InBlock.gif    builder.BuildPartB();  
http://www.cnblogs.com/Images/OutliningIndicators/ExpandedSubBlockEnd.gif  }  
http://www.cnblogs.com/Images/OutliningIndicators/ExpandedBlockEnd.gif}

**省略指导者角色**

在具体建造者只有一个的情况下，如果抽象建造者角色已经被省略掉，那么还可以省略掉指导者角色。让Builder角色自己扮演指导者与建造者双重角色。这时代码可能如下：

http://www.cnblogs.com/Images/OutliningIndicators/None.gifpublic class Builder  
http://www.cnblogs.com/Images/OutliningIndicators/ExpandedBlockStart.gif{  
http://www.cnblogs.com/Images/OutliningIndicators/InBlock.gif  private Product product = new Product();  
http://www.cnblogs.com/Images/OutliningIndicators/InBlock.gif  
http://www.cnblogs.com/Images/OutliningIndicators/InBlock.gif  public void BuildPartA()  
http://www.cnblogs.com/Images/OutliningIndicators/ExpandedSubBlockStart.gif  {   
http://www.cnblogs.com/Images/OutliningIndicators/InBlock.gif    //Some code here  
http://www.cnblogs.com/Images/OutliningIndicators/ExpandedSubBlockEnd.gif  }  
http://www.cnblogs.com/Images/OutliningIndicators/InBlock.gif  
http://www.cnblogs.com/Images/OutliningIndicators/InBlock.gif  public void BuildPartB()  
http://www.cnblogs.com/Images/OutliningIndicators/ExpandedSubBlockStart.gif  {  
http://www.cnblogs.com/Images/OutliningIndicators/InBlock.gif    //Some code here  
http://www.cnblogs.com/Images/OutliningIndicators/ExpandedSubBlockEnd.gif  }  
http://www.cnblogs.com/Images/OutliningIndicators/InBlock.gif  
http://www.cnblogs.com/Images/OutliningIndicators/InBlock.gif  public Product GetResult()  
http://www.cnblogs.com/Images/OutliningIndicators/ExpandedSubBlockStart.gif  {  
http://www.cnblogs.com/Images/OutliningIndicators/InBlock.gif    return product;  
http://www.cnblogs.com/Images/OutliningIndicators/ExpandedSubBlockEnd.gif  }  
http://www.cnblogs.com/Images/OutliningIndicators/InBlock.gif  
http://www.cnblogs.com/Images/OutliningIndicators/InBlock.gif  public void Construct()  
http://www.cnblogs.com/Images/OutliningIndicators/ExpandedSubBlockStart.gif  {  
http://www.cnblogs.com/Images/OutliningIndicators/InBlock.gif    BuildPartA();  
http://www.cnblogs.com/Images/OutliningIndicators/InBlock.gif    BuildPartB();  
http://www.cnblogs.com/Images/OutliningIndicators/ExpandedSubBlockEnd.gif  }  
http://www.cnblogs.com/Images/OutliningIndicators/ExpandedBlockEnd.gif}

同时，客户端也需要进行相应的调整，如下：

http://www.cnblogs.com/Images/OutliningIndicators/None.gifpublic class Client  
http://www.cnblogs.com/Images/OutliningIndicators/ExpandedBlockStart.gif{  
http://www.cnblogs.com/Images/OutliningIndicators/InBlock.gif  private static Builder builder;  
http://www.cnblogs.com/Images/OutliningIndicators/InBlock.gif  
http://www.cnblogs.com/Images/OutliningIndicators/InBlock.gif  public static void Main()  
http://www.cnblogs.com/Images/OutliningIndicators/ExpandedSubBlockStart.gif  {  
http://www.cnblogs.com/Images/OutliningIndicators/InBlock.gif    builder = new Builder();  
http://www.cnblogs.com/Images/OutliningIndicators/InBlock.gif    builder.Construct();  
http://www.cnblogs.com/Images/OutliningIndicators/InBlock.gif    Product product = builder.GetResult();  
http://www.cnblogs.com/Images/OutliningIndicators/ExpandedSubBlockEnd.gif  }  
http://www.cnblogs.com/Images/OutliningIndicators/ExpandedBlockEnd.gif}

C#中的StringBuilder就是这样一个例子。

**七、 在什么情况下使用建造者模式**

以下情况应当使用建造者模式：

1、 需要生成的产品对象有复杂的内部结构。  
2、 需要生成的产品对象的属性相互依赖，建造者模式可以强迫生成顺序。  
3、 在对象创建过程中会使用到系统中的一些其它对象，这些对象在产品对象的创建过程中不易得到。

使用建造者模式主要有以下效果：

1、 建造模式的使用使得产品的内部表象可以独立的变化。使用建造者模式可以使客户端不必知道产品内部组成的细节。  
2、 每一个Builder都相对独立，而与其它的Builder无关。  
3、 模式所建造的最终产品更易于控制。