|  |
| --- |
| // Chain of Responsibility pattern -- Structural example |
| using System;    namespace DoFactory.GangOfFour.Chain.Structural  {    /// <summary>    /// MainApp startup class for Structural    /// Chain of Responsibility Design Pattern.    /// </summary>    class MainApp    {      /// <summary>      /// Entry point into console application.      /// </summary>      static void Main()      {        // Setup Chain of Responsibility        Handler h1 = new ConcreteHandler1();        Handler h2 = new ConcreteHandler2();        Handler h3 = new ConcreteHandler3();        h1.SetSuccessor(h2);        h2.SetSuccessor(h3);          // Generate and process request        int[] requests = { 2, 5, 14, 22, 18, 3, 27, 20 };          foreach (int request in requests)        {          h1.HandleRequest(request);        }          // Wait for user        Console.ReadKey();      }    }      /// <summary>    /// The 'Handler' abstract class    /// </summary>    abstract class Handler    {      protected Handler successor;        public void SetSuccessor(Handler successor)      {        this.successor = successor;      }        public abstract void HandleRequest(int request);    }      /// <summary>    /// The 'ConcreteHandler1' class    /// </summary>    class ConcreteHandler1 : Handler    {      public override void HandleRequest(int request)      {        if (request >= 0 && request < 10)        {          Console.WriteLine("{0} handled request {1}",            this.GetType().Name, request);        }        else if (successor != null)        {          successor.HandleRequest(request);        }      }    }      /// <summary>    /// The 'ConcreteHandler2' class    /// </summary>    class ConcreteHandler2 : Handler    {      public override void HandleRequest(int request)      {        if (request >= 10 && request < 20)        {          Console.WriteLine("{0} handled request {1}",            this.GetType().Name, request);        }        else if (successor != null)        {          successor.HandleRequest(request);        }      }    }      /// <summary>    /// The 'ConcreteHandler3' class    /// </summary>    class ConcreteHandler3 : Handler    {      public override void HandleRequest(int request)      {        if (request >= 20 && request < 30)        {          Console.WriteLine("{0} handled request {1}",            this.GetType().Name, request);        }        else if (successor != null)        {          successor.HandleRequest(request);        }      }    }  } |