**RouteContext**

RouteContext代表路由上下文，其封装了HttpContext

public class RouteContext

{

    // Http上下文

    public HttpContext HttpContext { get; }

    // 路由数据

    public RouteData RouteData { get; set; }

    // 请求处理器

    public RequestDelegate Handler { get; set; }

    public RouteContext (HttpContext httpContext);

}

**RouteData**

RouteData封装了解析到的路由参数

public class RouteData

{

    // 存放请求路径携带的参数

    public RouteValueDictionary     Values { get; }

    // 存放Router对象自身携带的参数

    public RouteValueDictionary     DataTokens { get; }

    // 路由解析并提供路由参数的所有Router对象

    public IList<IRouter>           Routers { get; }

}

在控制器（Controller）中，我们可以很容易获取到RouteData，其保存在RouteData属性中

也可以调用HttpContext的扩展方法HttpContext.GetRouteData();获取

**IRouter接口**

IRouter用于从HttpContext中分析出路由

并向RouteContext中添加路由参数，或

设置RouteContext的请求处理器

public interface IRouter

{

    // 路由分析

    Task RouteAsync(RouteContext context);

    // 获取虚拟路径

    VirtualPathData GetVirtualPath(VirtualPathContext context);

}

RouteAsync的任务：

为RouteContext 的RouteData和Handler赋值

**RouteHandler**

RouteHandler完成了IRouter的“设置RouteContext的请求处理器”的功能

如下是一个简单的示例：

public class RouteHandler : IRouteHandler, IRouter

{

    private readonly RequestDelegate \_requestDelegate;

    public RouteHandler (RequestDelegate requestDelegate) {

        \_requestDelegate = requestDelegate;

}

    public RequestDelegate GetRequestHandler (HttpContext httpContext, RouteData routeData)

    {

        return \_requestDelegate;

}

    public Task RouteAsync (RouteContext context) {

        context.Handler = \_requestDelegate;

        return Task.CompletedTask;

    }

}

**Route**

Route完成了IRouter的“向RouteContext中添加路由参数”的功能

Route继承了RouteBase，RouteBase继承了INamedRouter继承了IRouter

RouteBase

public abstract class RouteBase : INamedRouter

{

    public virtual Task RouteAsync(RouteContext context);

    protected abstract Task OnRouteMatched(RouteContext context);

    …

}

RouteBase实现了RouteAsync，其方法判断给定的条件是否满足自身的路由规则，在规则满足的情况下将解析出来的路由参数保存到RouteContext这个上下文中，然后调用抽象方法OnRouteMatched

Route

Route是RouteBase的默认实现

Route接收一个IRouter target参数，实际这个参数的类型是RouteHandler

public class Route : RouteBase

{

    private readonly IRouter \_target;

    public Route (IRouter target, string routeTemplate, IInlineConstraintResolver inlineConstraintResolver) {

        \_target = target;

}

    protected override Task OnRouteMatched (RouteContext context) {

        context.RouteData.Routers.Add (\_target);

        return \_target.RouteAsync (context);

}

    ...

}

**关系图**

