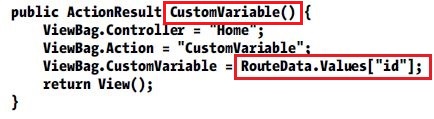
1. **Routing System**
   1. Routing System有兩個主要的功能
      1. Inbound：檢查請求的URL。(根據已定義的路由規則，決定該請求由哪個Controller底下的Action來負責處理)
      2. Outbound：輸出URL超連結到頁面 (View)上。
2. **URL Pattern**
   1. 組成要素
      1. 不包含主機名稱、埠號、查詢字串
      2. 由所謂的Segment組成 (把它理解為有幾個區塊就可以)

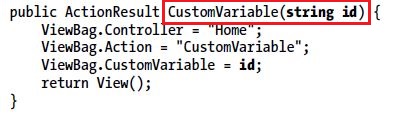


* 1. URL的比對，預設會以該請求中所包含的Segment個數去逐一檢查是否有符合的Route規則。如果沒有找不到符合的Route規則，就會檢查第一條Route是否有定義Default值，若沒有定義就逐一檢查下去，最後找到符合條件的Route規則或是拋出404錯誤
  2. URL Pattern Segment的變化包含以下幾種可能
     1. 變數：例如 http://mysite.com/{controller}/{action}
     2. 文字：例如 http://mysite.com/Public/{controller}/{action}
     3. 變數與文字的組合：例如 http://mysite.com/X{controller}/{action}
  3. Segment的變數名稱可以自己定義，內建的名稱有controller、action、area。可以透過以下寫法在action裡面取得變數的值。

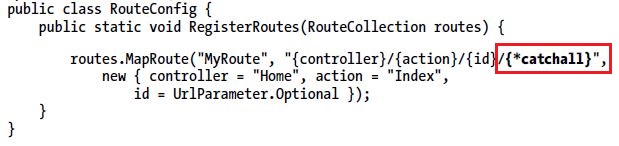
\* 透過RouteData.Values取得變數值



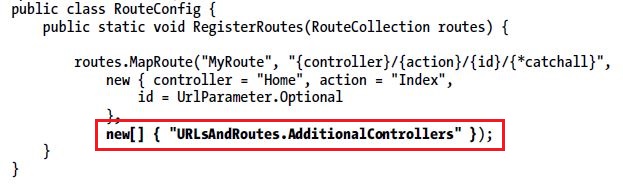
\* 或是透過Model Binding的方式也可以



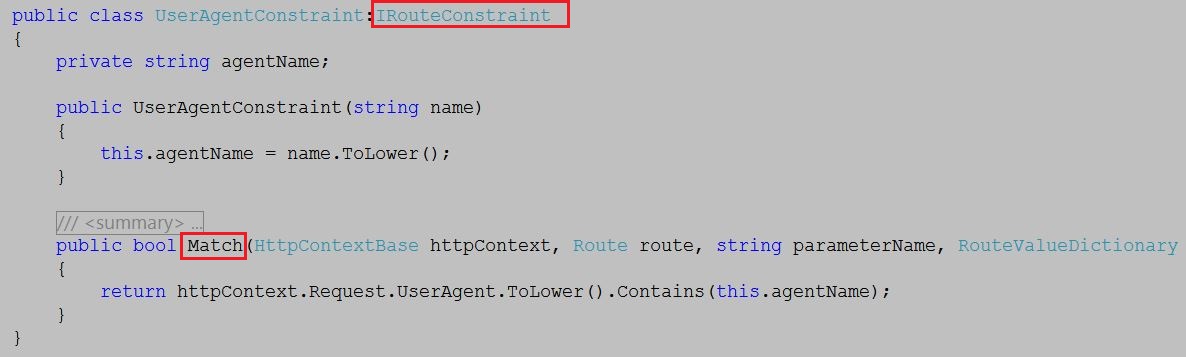
* 1. 不限個數的Segment，可以透過以下寫法來接受數目較多的Segment (在變數前面加一個星號\*)

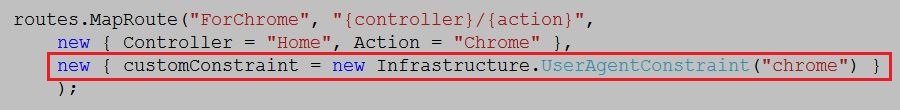


* 1. 若有相同名稱的Controller，可以透過設定Namespace來做區分。相同名稱的Controller可以在同一個專案或不同專案底下存在。

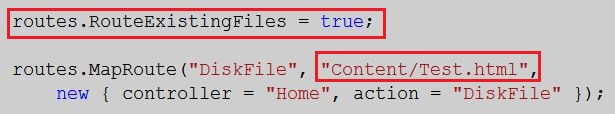


* 1. 也可以設定符合某些限定條件才進行該路由，例如檢查瀏覽器是否是Chrome，是的話則路由到特定的action。主要透過實作IRouteConstraint介面來達成





1. **實體檔案的路由**
   1. 在進行URL Routing之前，Routing System預設會先檢查URL對應的實體檔案是否存在
   2. 若要把實體檔案也加入路由，可以透過以下步驟設定
      1. 把RouteCollection物件的RouteExistingFiles屬性改為true
      2. 加入針對實體檔案的路由規則，例如以下範例



* + 1. 更改Web.config的設定 (先移除再新增，然後把preCondition設定為空字串)

