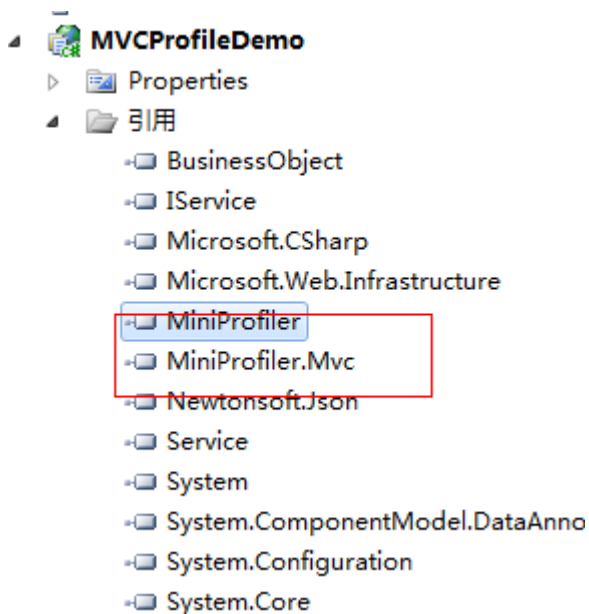


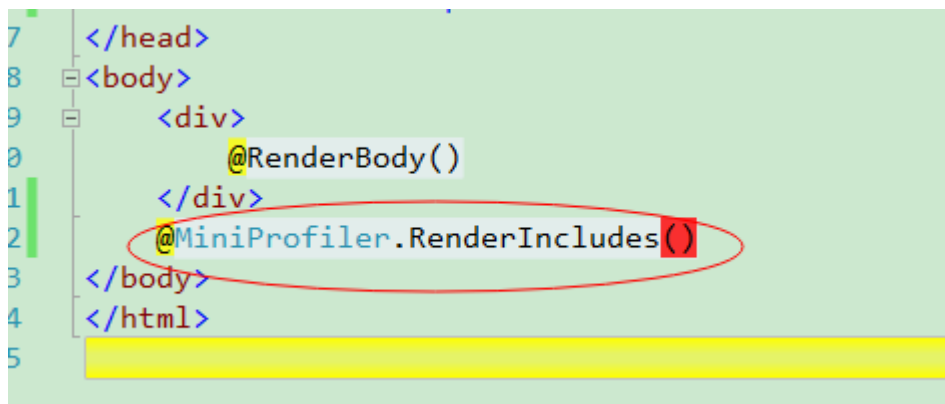
MiniProfile 监控工具在 MVC 框架下的配置和使用

1 添加核心库和支持库

使用 nuget 对 mvc 项目添加 Miniprofile 的两个库文件，核心库 Miniprofile.dll 和 MVC 监控扩展库 Miniprofile.MVC



2.修改母版页 Layout



在 body 标签结束前添加 Miniprofile.RenderIncludes()
将监控信息渲染至此

3.修改 global.asax 文件

```
protected void Application_BeginRequest()
{
    if (Request.IsLocal)
    {
        MiniProfiler.Start();
        //指定sql格式化器, 输出带实参sql
        MiniProfiler.Settings.SqlFormatter = new
StackExchange.Profiling.SqlFormatters.InlineFormatter();
    }
}

protected void Application_EndRequest()
{
    MiniProfiler.Stop();
}
}
```

监控单元默认为一个 Request 请求的周期。

实例化以上两个管道事件，并分别在 BeginRequest 中启动 Miniprofile 的监控，在 EndRequest 的事件中结束监控

在 Start 时，指定 sql 格式化器，输出带实参 sql，默认输出 sql 带形参。

4 监控 ado.net 数据库访问，调整 DAO

以 dapper 为例

```
public static DbConnection GetMySQLConnection()
{
    var connection = new MySqlConnection(ConfigurationManager.ConnectionStrings["huatongdb"].ConnectionString);
    //connection.Open();
    //return connection;

    var profiledConn= new StackExchange.Profiling.Data.ProfiledDbConnection(connection,MiniProfiler.Current);
    profiledConn.Open();
    return profiledConn;
}
#endregion
```

注意，如果是 mvc 项目的监控，创建包装对象时，第二个参数用 MiniProfile.Current,而不是用 MiniProfile.Start()

Mvc 项目中，监控单元默认是整个 Request 周期，Profile 监控实例保持为一个，On Session Per

Request,在 Nhibernate 的使用中提及过,不多说。

Service 中监控 ADO.NET 数据库访问,调整代码如下

```
//创建数据库访问监控单元
var service = new TicketLineOrderService();
using (profiler.Step("数据库访问监控", ProfileLevel.Info))
{
    orderBo = service.GetTicketLineOrderByOrderNumOverWrite("2b758401400c0800");
}
// ViewBag.Title = "Index";
return View(orderBo);
}
```

5 自定义监控单元

```
{
    TicketLineOrderBo orderBo = null;
    var profiler = MiniProfiler.Current;
    //创建监控单元,并命名
    using (profiler.Step("设置page title耗时监控"))
    {
        ViewBag.Title = "title耗时监控";
    }
}
```

通过 current 读取到当前监控实例。

通过扩展方法 step,使用 using 块手动创建监控单元。

注意,MVC 项目监控中,Request 是最顶层的监控单元,但是仍可在 Request 监控单元内,创建子监控单元。也可手动创建父子监控单元,如下

```
//创建父子监控单元
//执行时间超过1ms则记录,true标识父监控单元的监控时间包含子监控单元的执行时间
using (profiler.StepIf("监控测试",1,true))
{
    using (profiler.Step("Children1"))
    {
        Thread.Sleep(100);
    }
    using (profiler.Step("Children2"))
    {
        Thread.Sleep(250);
    }
}
```

注意,创建父子监控单元时,需要使用 StepIf 扩展方法,指定三个参数,

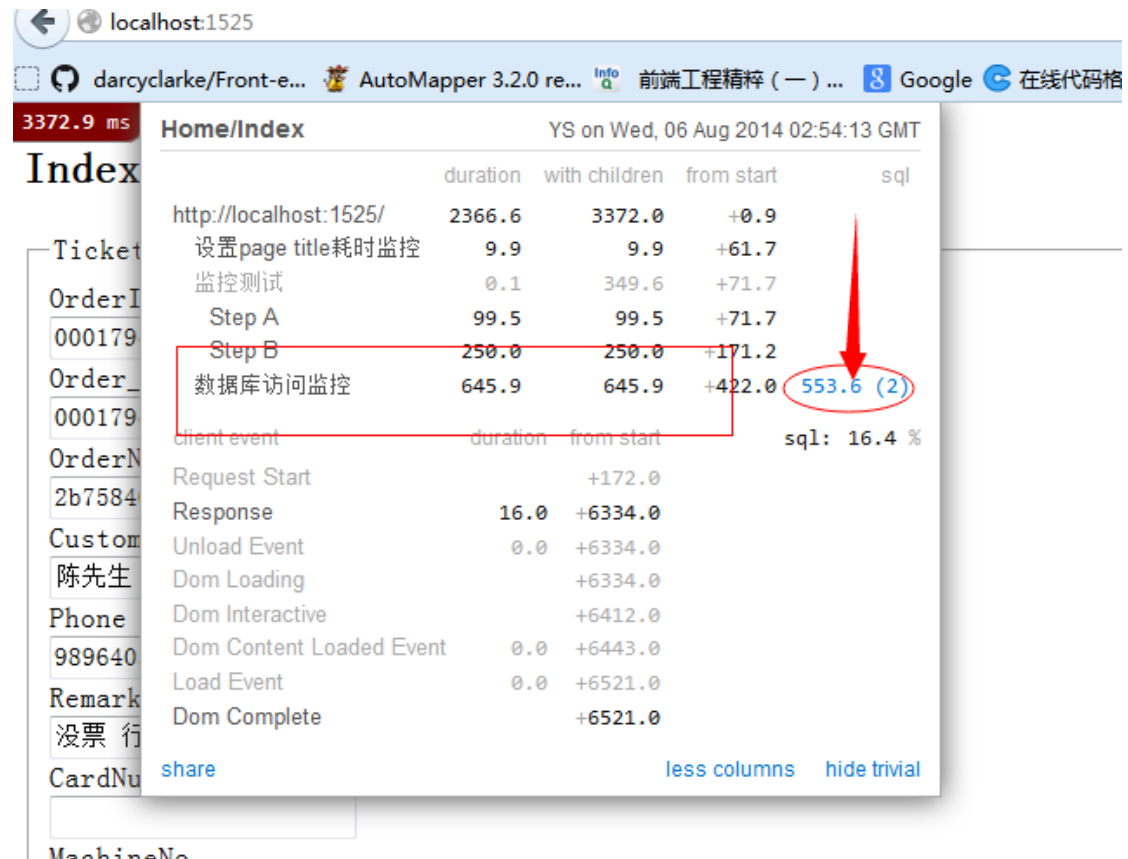
1.name 为监控单元命名,

2,第二个参数为监控时间限度,意思是超过这个设置的时间,相关监控信息才会被记录,

未超过这个时间，执行时间等信息不会被监视器统计和输出

3. 第三个 bool 参数，父监控单元监控，是否计入子监控单元的执行时间。如果是 true，那么父监控单元会记录两个子监控单元 Children1 和 Children2 的执行时间，并且会在 Request 结束后，输出这两个子监控单元的监控信息，否则不输出。

6 监视效果



	duration	with children	from start	sql
http://localhost:1525/	2366.6	3372.0	+0.9	
设置page title耗时监控	9.9	9.9	+61.7	
监控测试	0.1	349.6	+71.7	
Step A	99.5	99.5	+71.7	
Step B	250.0	250.0	+171.2	
数据库访问监控	645.9	645.9	+422.0	553.6 (2)
Client event	duration	from start		sql: 16.4 %
Request Start		+172.0		
Response	16.0	+6334.0		
Unload Event	0.0	+6334.0		
Dom Loading		+6334.0		
Dom Interactive		+6412.0		
Dom Content Loaded Event	0.0	+6443.0		
Load Event	0.0	+6521.0		
Dom Complete		+6521.0		

查看数据库访问的 sql 语句

3372.9 ms	Home/Index	YS on Wed, 06 Aug 2014 02:54:13 GMT
	step time from start call type duration	call stack call
	512.00 ms	Step B — 250.00 ms
	数据库访问监控 T+512.0 ms sql - Reader 540.6 ms	GetTicketLineOrderByOrderNumOverWrite Index <code>select * from tb_ticketlineorder where OrderNum = @orderNum</code>
	数据库访问监控 T+1054.0 ms sql - Reader 13.0 ms	GetTicketLineOrderByOrderNumOverWrite Index <code>select * from tb_orderticket where order_id =@orderId</code>
	2305.00 ms	http://localhost:1525/ — 2304.10 ms

[show trivial gaps](#)

关于 Miniprofile 在单元测试中的应用，见这里 <http://192.168.10.30/index.php?doc-view-2904>
有问题私下找我

2014-8-6