站点部署，IIS配置优化指南

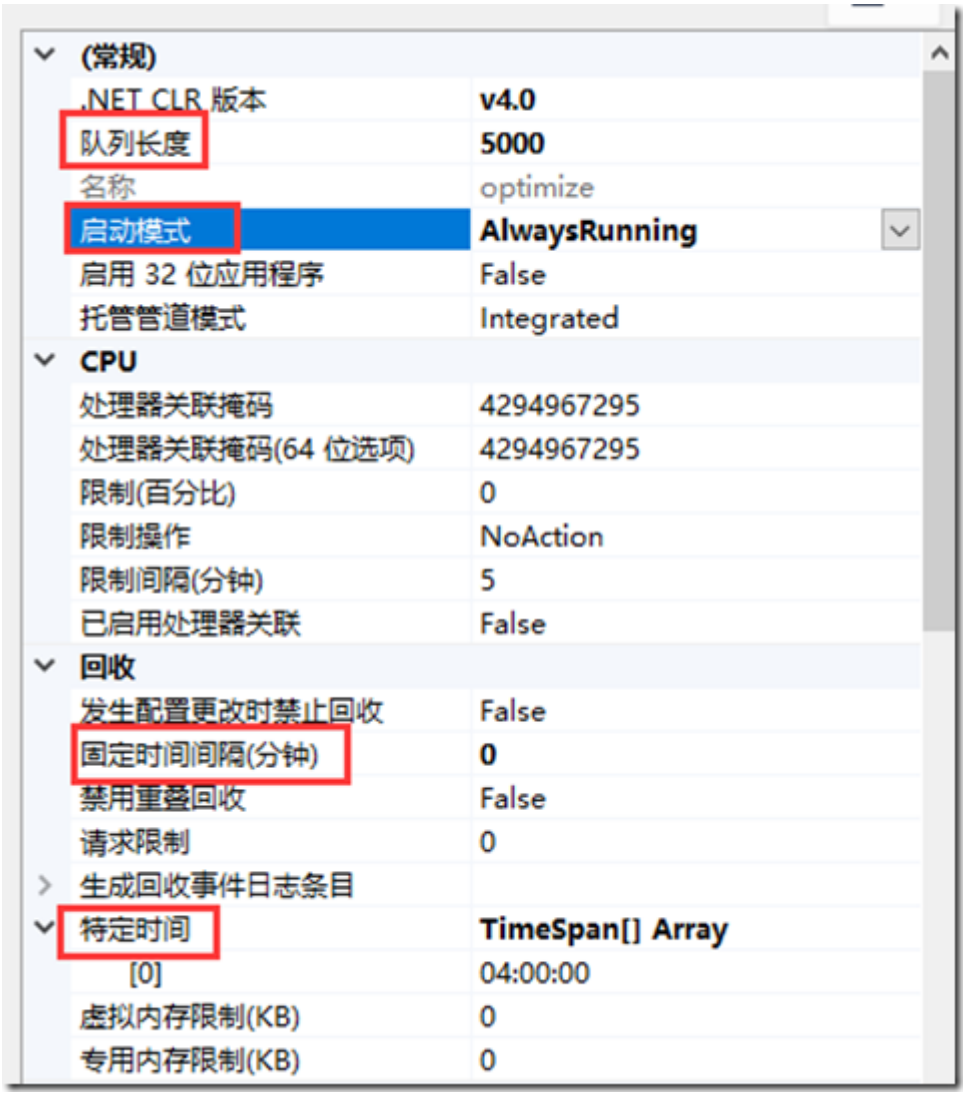
cnblogs.com/heyuquan/p/deploy-iis-set-performance-guide.html

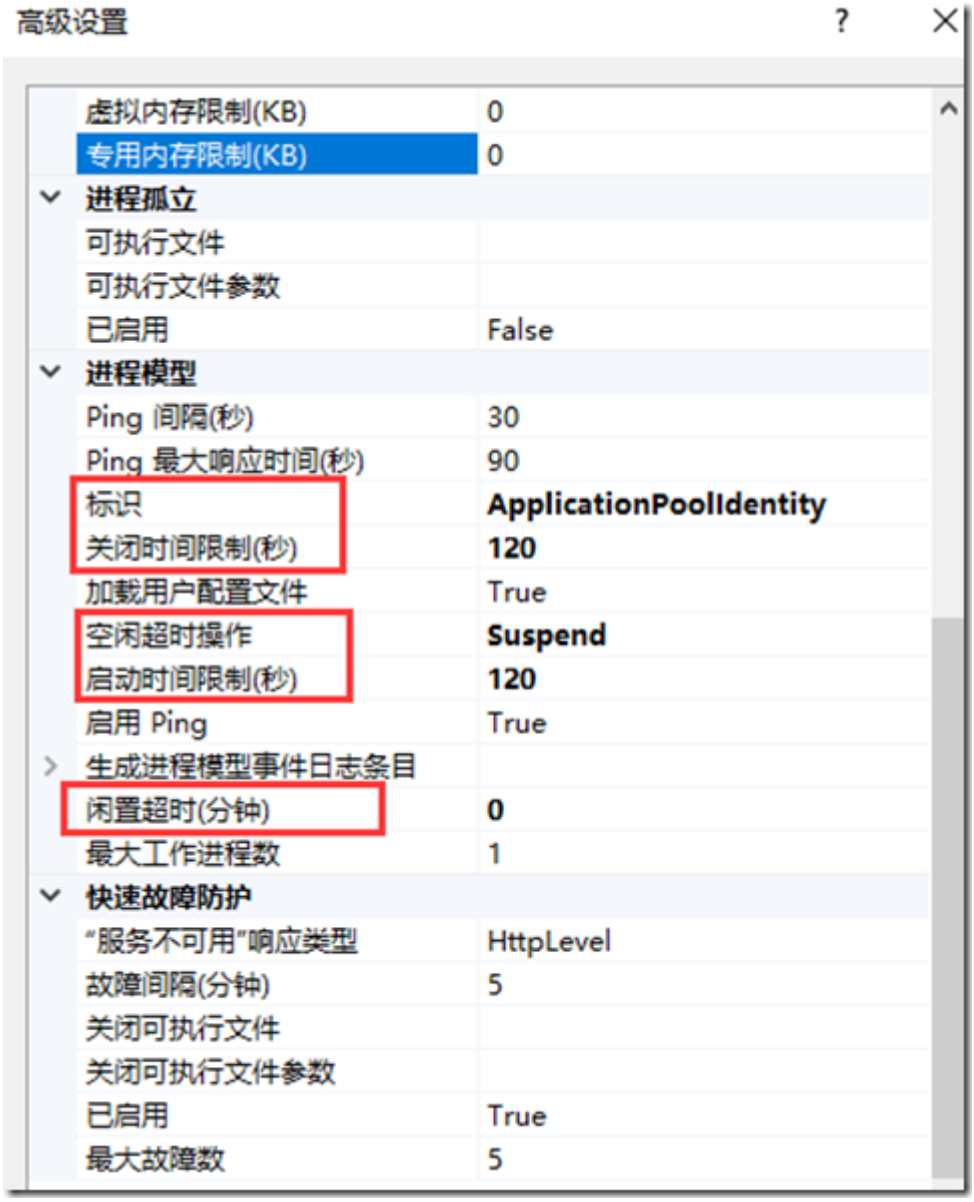
目录

通常把站点发布到IIS上运行正常后，很少会去考虑IIS提供的各种参数，如何配置才是最适合当前站点运行需要的？这篇文章，从基本设置、回收机制、性能、并发、安全性等IIS设置讲解应当如何优化。

先来“IIS应用程序池”优化后的参数配置截图：

图中一些数值限制参数，可以借助一些工具（如：windows性能监控）观察站点运行的指标进行设置，具体后面会介绍到


(常规)
.NET CLR 版本
v4.0
队列长度
5000
名称
optimize
启动模式
AlwaysRunning
启用 32 位应用程序
False
托管管道模式
Integrated
CPU
处理器关联掩码
4294967295
处理器关联掩码(64 位选项)
4294967295
限制(百分比)
0
限制操作
NoAction
限制间隔(分钟)
5
已启用处理器关联
False
回收
发生配置更改时禁止回收
False
固定时间间隔(分钟)
0
禁用重叠回收
False
请求限制
0
生成回收事件日志条目
特定时间
TimeSpan[] Array
[0]
04:00:00
虚拟内存限制(KB)
0
专用内存限制(KB)
0



下面来分别解说下这些参数为什么要这样设置（注：文章中的参数，不是按照应用程序池的设置从上到下排列的，而是按照优化的功能点排列）

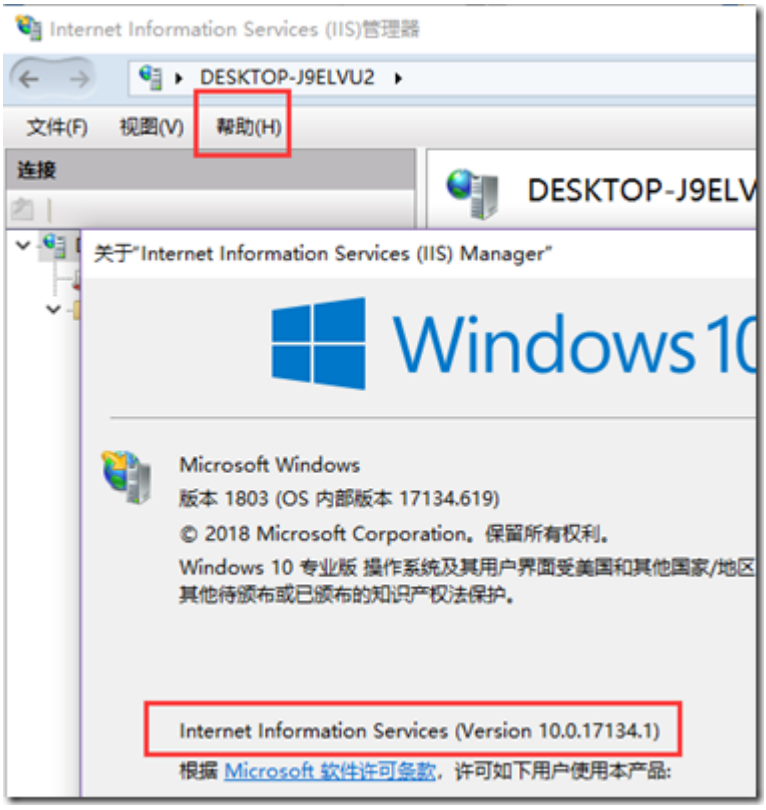
一、 设置应用程序池默认设置

按如下图进行默认参数模板设置，设置后，新建的应用程序池就使用这个默认参数模板。

二、 常规设置

IIS版本号查看

在iis管理器中->帮助->关于Internet信息服务，如下图，版本是IIS10.



常规 > 启动32位应用程序

默认值：False

优化设置：按需设置。如果确认站点依赖一些32位的组件，需将此设置为true。

建议：为 32bit 应用程序的网站单独创建一个应用程序池

参考：

64位系统上iis运行32位的网站程序

常规 > 托管管道模式

IIS7 应用程序池新增的经典模式和集成模式

经典模式：是为了保留和IIS6一样的处理方式，以前开发的代码，可以方便的移植到IIS7上。

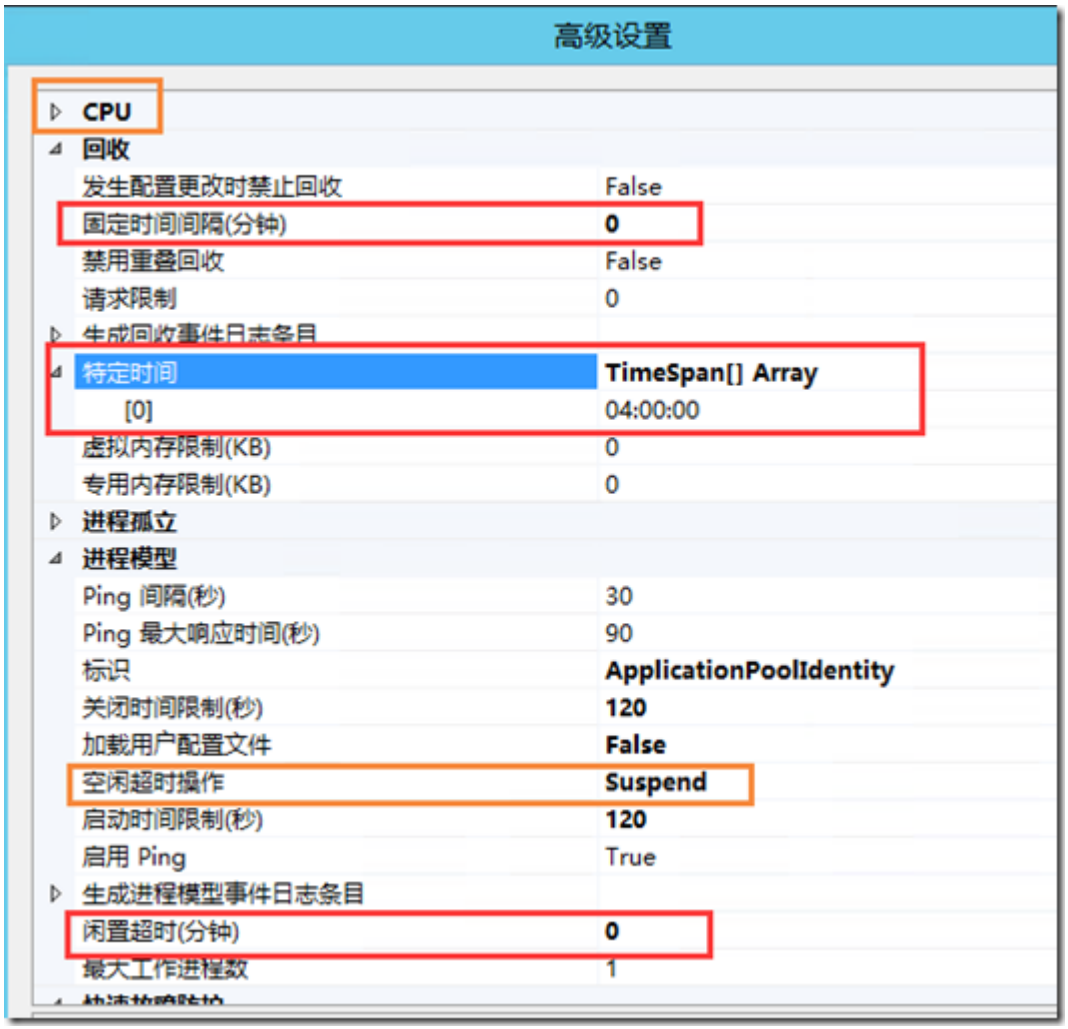
集成模式：将ASP.NET请求管道与IIS核心管道组合在一起，这种模式与操作系统结合更紧密，能够提供更好的性能，能够实现配置和治理的模块化，而且增加了使用托管代码模块扩展IIS时的灵活性。

优化设置： 改为 Integrated（集成模式）

参考：

对IIS7经典模式和集成模式的理解

三、 优化回收策略



回收 > 固定时间间隔（分钟）

一个时间段，超过该时间段，应用程序池将回收。值为 0 ，则应用程序池不会按固定间隔回收

默认值：1740分钟，29小时

优化设置：改为0 。因为无法避免在高峰期发生回收。同时设置“回收 > 特定时间”

回收 > 特定时间

应用程序池进行回收的一组特定的本地时间（24小时制）

优化设置：固定在低峰期时回收。eg：设定为 04:00 、15:30 等

另外，也可以使用windows计划任务实现iis站点每周六晚定时回收

进程模型 > 闲置超时（分钟）

一个时间段，设定工作进程允许保持闲置状态的最大时间间隔，超过该时间就会自动关闭。

优化设置：改为0，避免内存信息频繁被回收清空。同时设置“回收 > 特定时间”

进程模型 > 空闲超时操作

默认是“Terminate”（另一个选项是“Suspend”）。

Terminate 表示一旦超时就终止服务，并回收工作进程的缓冲区的内存；

Suspend 则悬停等待，暂不回收缓冲区内存。

另外：

CPU超限占用安全方案设置

CPU限制并不是用于控制每个进程的CPU利用率，而是一种处理发生CPU超限的工作进程的安全方案，这样可以避免工作进程占用CPU过久。

参考：

iis7.0 cpu 限制

iis中对cpu限制的操作：

1. 限制：10000 （以百分比*1000计算，10000则表示10%）

2. 限制操作：1、noaction 无操作 2、KillW3wp 删除进程 并在限制时间内重新开启新进程

3. 限制间隔（分钟）：设置时间限制，多久时间内重启和检测

内存超限回收机制

根据实际运行情况设定 "回收 > 虚拟内存限制" 和 "回收 > 专用内存限制"，默认为禁用状态，一般不用为此专门设定。

开启|关闭时间限制

根据实际运行情况设定，默认90秒。如上图，我都设置为了120秒

进程模型 > 关闭时间限制（秒）：为工作进程指定的，完成处理请求并关闭的时间段。如果工作进程超过关闭时间限制，将被终止。

进程模型 > 启动时间限制（秒）：为工作进程指定的，启动并进行初始化的时间段。如果工作进程初始化时间超过启动时间限制，将被终止。

回收 > 禁用重叠回收

默认值 false。应用程序池使用重叠回收方式。在这种方式下，当应用程序池要关闭某个工作进程时，会先创建一个工作进程，直到新的工作进程成功创建后才关闭旧的工作进程；

设置为 true，则先关闭旧的工作进程，然后再创建新的工作进程。 如果Web 应用程序不支持多实例运行，那么你必须配置应用程序池禁止使用重叠回收方式。

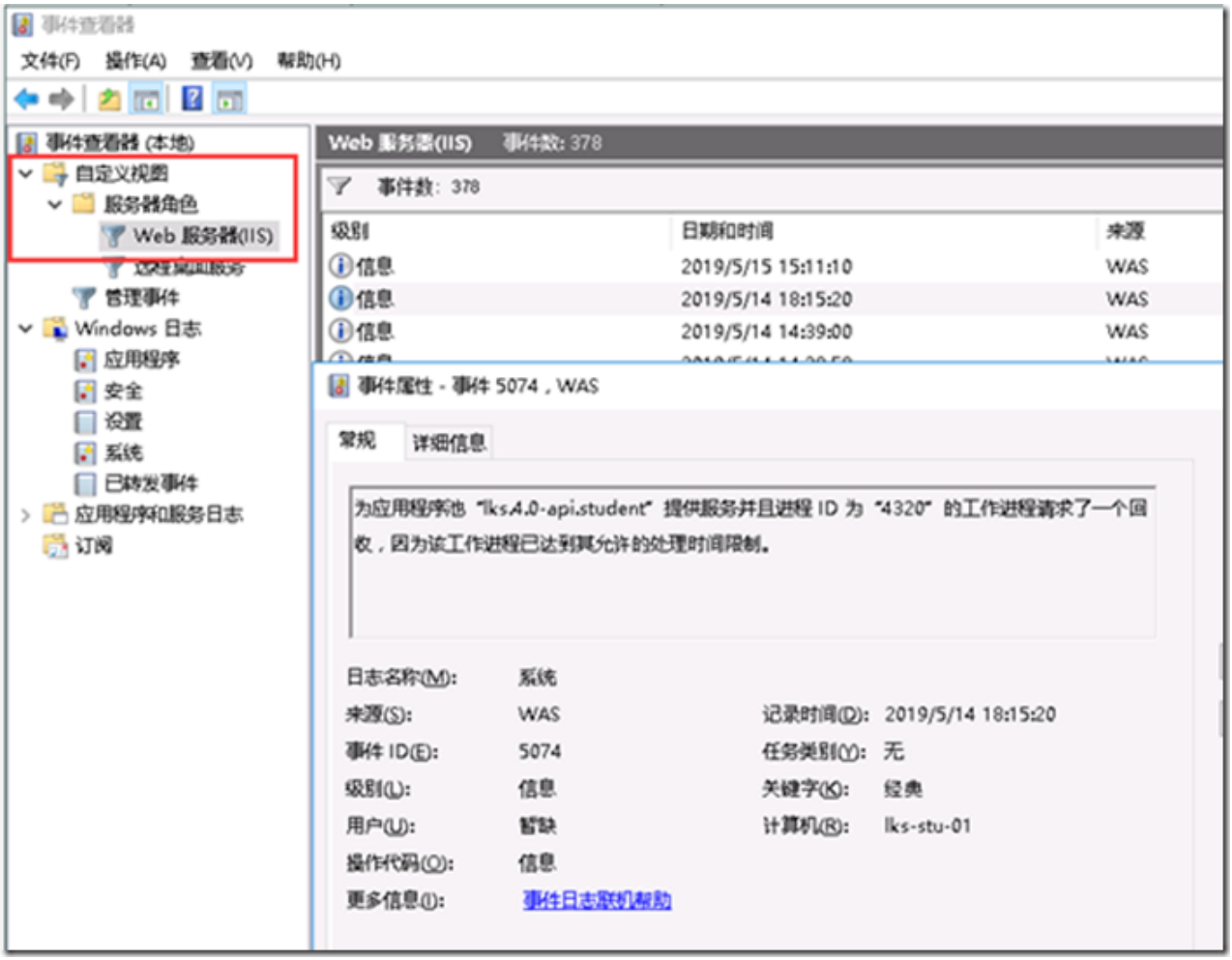
回收 > 生成回收事件条目

IIS事件查看器

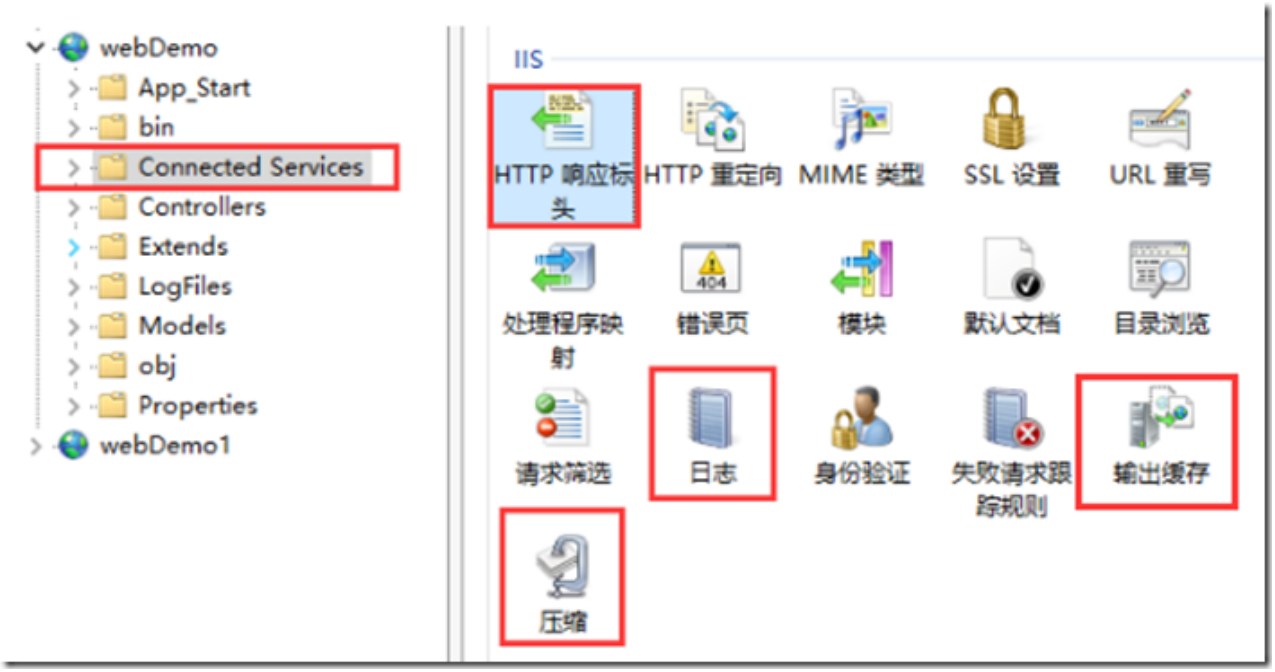
方法一：点击“开始→运行”，输入eventvwr，点击“确定”，就可以打开事件查看器。

方法二：单击“开始”-“设置”-“控制面板”-“管理工具”-“事件查看器”，开事件查看器窗口。

方法三：在“运行”对话框中手工键入“%SystemRoot%/system32/eventvwr.msc /s”打开事件查看器窗口。



四、 性能



关闭IIS日志

当开启记录功能后，IIS会事无巨细地忠实记录所有的web访问记录。这些记录文件的内容是非常庞杂的，比如访问时间、客户端IP、从哪个链接访问、 Cookies等，另外还包括Method(方法), UserAgent(用户代理)等。这些记录不但占用大量的磁盘空间还大大地影响了web服务器的性能。有人做过评测，停止访问记录可以提升5%到8%的web性能。

启用内容过期（客户端缓存）

对于静态文件启用内容过期可以提高访问性能。

1. 首先网站的目录要划分合理，图片、CSS、JavaScript均放在单独目录下

2. 然后在IIS中选择要缓存的目录 > HTTP 响应标头 > 设置常用标头 > 设置"web内容过期"策略

如上图webDemo站点，这样，用户浏览器将比较当前日期和截止日期，以便决定是显示缓存页还是从服务器请求更新的页，由于图片、CSS、JS通常变化较少，因此基本上都从本地缓存读取，从而加快显示速度。

参考：

IIS7禁用单个静态文件的客户端缓存

服务器验证缓存

IIS自动机制，会在访问css、js等静态文件时，返回给浏览器Last-Modified和Etag标记

参考：

浏览器缓存之Last-Modified

服务端的缓存验证 Last-Modified和Etag

启用Gzip压缩

IIS 压缩功能使用Gzip算法

gzip是HTTP的一种压缩算法，HTTP压缩是在Web服务器和浏览器间传输压缩文本内容的方法。HTTP压缩采用通用的压缩算法如gzip等压缩HTML、JavaScript或 CSS文件。压缩的最大好处就是降低了网络传输的数据量，从而提高客户端浏览器的访问速度。当然，同时也会增加一点点服务器的负担。Gzip是比较常见的一种HTTP压缩算法。

五、 IIS初始化（预加载），解决（被回收后）第一次访问慢

参考：https://www.cnblogs.com/teamblog/p/6195078.html

设置之后，什么时候会自动初始化？

（比如初始化执行 Global.Application_Start 初始化函数）

1）会 - 应用程序池启动、应用程序池回收、cmd->iisreset （w3wp的PID会变）

2）不会 - 站点重启（IIS站点右键 > 管理网站 > 重新启动）、站点启动

3）不会 - web.config更改引起的应用程序池回收

在IIS10版本上测试是上面行为。另外有人IIS8.5上使用也是同样的行为，参考文章。

步骤一、安装IIS应用程序初始化功能

步骤二、设置IIS上应用程序池启动模式

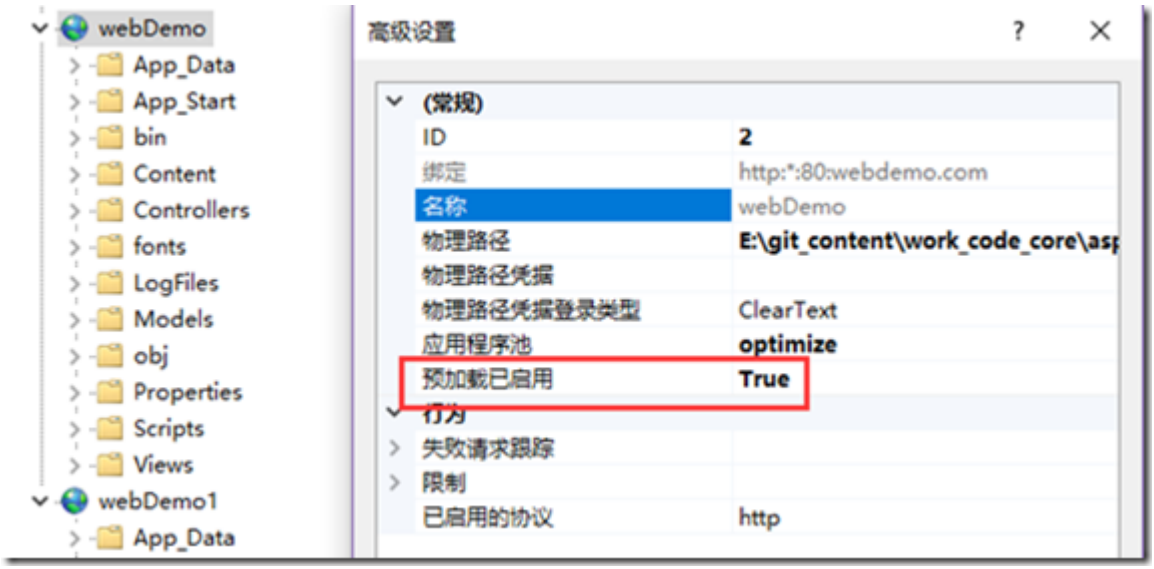
常规 > 启动模式

默认值：OnDemand（按需运行模式），另外值AlwaysRuning（始终运行模式）

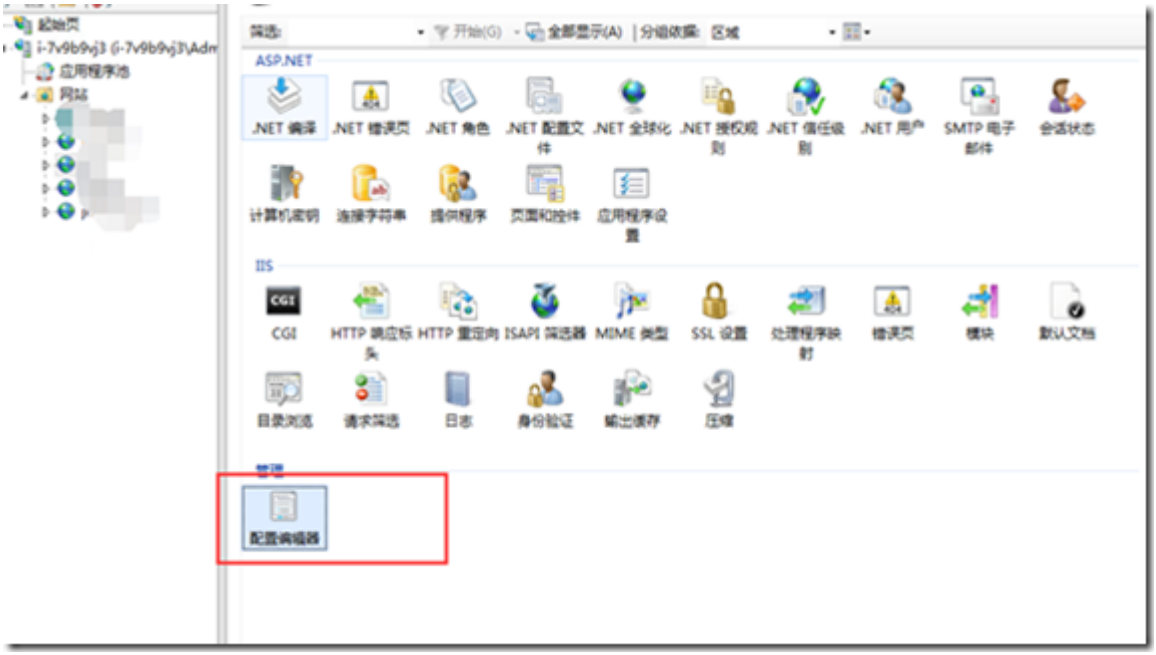
优化设置：改为 AlwaysRunning（始终运行）

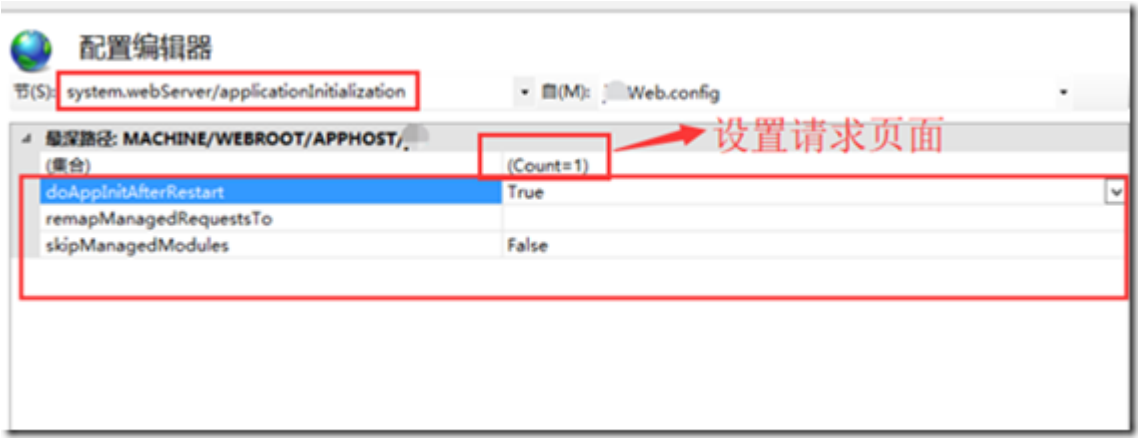
步骤三、设置站点预加载

在IIS上站点右键 > 管理网站 > 高级设置，把【预加载已启用】设置为true。



步骤四、配置站点 web.config ，添加站点重启后预加载请求的页面





eg：地址：http://webdemo.com/home/about

这样操作保存后，IIS会修改 web.config 添加如下内容

```
<system.webServer>
……
<applicationInitialization doAppInitAfterRestart="true">
<add initializationPage="home/about" hostName="" />
</applicationInitialization>
</system.webServer>
```

如果只是初始化（比如只执行 Global.Application_Start 初始化函数），不需要访问特定API进行额外资源的初始化，则不需要 <add initializationPage="**" /> 子节点

六、 并发性

常规 > 队列长度

HTTP.sys 将针对应用程序池排队的最大请求数。默认值1000，最大值65535。

如果设置太大则会消耗大量的系统资源 ，而设置太小会导致客户端访问时频繁出现"503服务不可用"响应。

优化设置：可先改为 5000（设置为预期最多并发用户数的1.5倍，官方参考

使用windows性能监控（性能监控：cmd->perfmon.msc），添加“HTTP Service Request Queues/CurrentQueueSize”指标，观察某个应用程序池当前队列中请求的个数。

启用Web园（Web Garden），进程模型 > 最大工作进程数

在Web园中你可以配置此应用程序池所使用的最大工作进程数，默认为1，最大可以设置为4000000； 配置使用多个工作进程可以提高该应用程序池处理请求的性能，但是在设置为使用多个工作进程之前，请考虑以下两点：

1、每一个工作进程都会消耗系统资源和CPU占用率；太多的工作进程会导致系统资源和CPU利用率的急剧消耗；

2、每一个工作进程都具有自己的状态数据，如果Web应用程序依赖于工作进程保存状态数据，那么可能不支持使用多个工作进程。

这样设置，增加了处理进程数，相当于集群，避免大量请求处于排队状态

参考：

IIS并发优化

> 文章介绍：使用windows性能监控：cmd->perfmon.msc。监控IIS应用运行情况，再根据需要进行iis参数设置
> Web Service/Current Connections 监控某个应用程序池来指示当前该应用程序池的连接的数量。
> ASP.NET Apps v4.0.30319/Requests Executing 监控所有的 ASP.Net 4.0 正在处理中的请求数量。
> ASP.NET v4.0.30319/Requests Current 与上述类似用于监控 Asp.Net 4.0 正在处理中的请求数量。
> HTTP Service Request Queues/CurrentQueueSize 用来监控某个应用程序池当前队列中请求的个数。

调整支持并发请求的数量

默认支持并发请求数量为：5000

超出此并发数，会报异常

HTTP Error 503.2 - Service Unavailable

The serverRuntime@appConcurrentRequestLimit setting is being exceeded.

参考：

IIS 并发请求设置如何设置？

站点最大并发连接数

右键站点 > 高级设置 > 限制 > 最大并发连接数

设置站点线程数：minWorkerThreads、maxWorkerThreads、maxIoThreads

（感谢园友 @ runliuv 提供的新姿势）

maxWorkerThreads默认20，maxIoThreads默认20，minWorkerThreads默认1，minIoThreads默认1 (eg：8核，默认分别就是160, 160, 8, 8)

1、配置文件：
C:\Windows\Microsoft.NET\Framework64\v4.0.30319\Config\machine.config

2、修改参数： <processModel autoConfig="false" maxWorkerThreads="100" maxIoThreads="100" minWorkerThreads="50" minIoThreads="50" />

其中：minWorkerThreads = maxWorkerThreads / 2 ； minIoThreads = maxIoThreads / 2

参数具体值如何设置，还需要各自对站点进行压力测试中调整

参考：

博客园"黑色30秒"事件

排查“黑色30秒”问题-为什么请求会排队

[解决]从ASP.NET线程角度对“黑色30秒”问题的全新分析

IIS7.5优化--提高线程数来适应高并发

processModel 元素（ASP.NET 设置架构）

Improving ASP.NET Performance （微软文档中给出了推荐值，如下图）

Table 6.1: Recommended Threading Settings for Reducing Contention

Configuration setting	Default value (.NET Framework 1.1)	Recommended value
maxconnection	2	12 * #CPUs
maxIoThreads	20	100
maxWorkerThreads	20	100
minFreeThreads	8	88 * #CPUs
minLocalRequestFreeThreads	4	76 * #CPUs

七、 安全性

为不同工作进程指定应用程序池（工作进程隔离模式）

一台服务器上有非常多的Web站点。如何才能做到各个站点之间相互独立，不因某些Web站点出现故障而影响其他站点呢?--为不同工作进程指定应用程序池是个很好的解决办法。

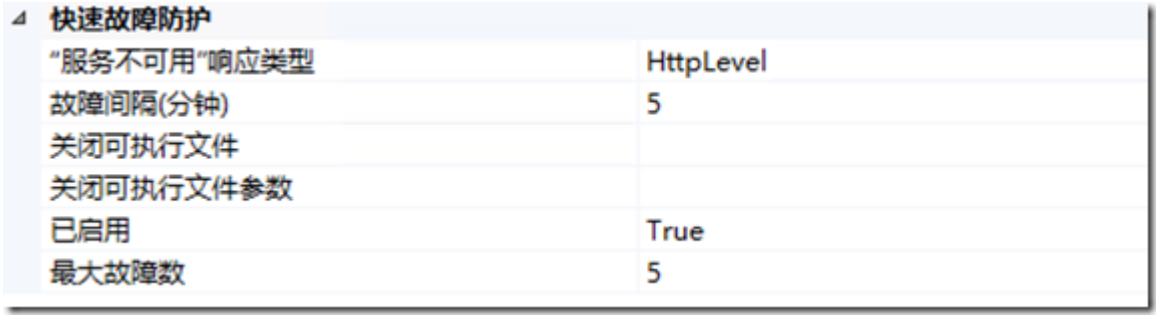
进程模型 > 标识，使用ApplicationPoolIdentity虚拟账户

ApplicationPoolIdentity – 默认情况下，选择“应用程序池标识”帐户。启动应用程序池时动态创建“应用程序池标识”帐户，因此，此帐户对于您的应用程序来说是最安全的。（这样，每个应用程序池都有各自的账户，就避免了木马上传到其中一个池下站点，会对另一个池的文件夹有操作权限）

参考：

IIS7.5中神秘的ApplicationPoolIdentity

启用快速失败保护



如果Web应用程序代码编写有问题，它可能会导致工作进程持续出现问题。默认情况下应用程序池配置为启用快速失败保护，当工作进程在配置的时间段（默认为5分钟）内发生的失败次数超过了配置的值（默认为5次），则禁用此应用程序池。

八、 多服务器IIS集中化管理web

Microsoft IIS Administration 微软提供，管理IIS配置的REST API 和集中化IIS管理WEB UI。

l 支持绝大部分IIS配置项管理

l 支持管理远程IIS，实现集中化IIS配置管理。

l 支持REST API，方便集成到自研系统。

l 支持IIS配置访问安全性设置

详细查看：多服务器IIS集中化管理web和编程访问IIS

其他阅读：

IIS 开启和关闭详细报错信息（404,500,502等）

==========

over，谢谢查阅，觉得文章对你有收获，请多帮推荐。欢迎提供更好的资料信息。

作者：滴答的雨
出处：http://www.cnblogs.com/heyuquan/

欢迎园友讨论下自己的见解，及向我推荐更好的资料。
本文如对您有帮助，还请多帮 **【推荐】** 下此文。
谢谢！！！ (*^_^*)
技术群：185718116（广深莞·NET技术），欢迎你的加入
技术群： 加入QQ群 （广西IT技术交流），欢迎你的加入

分类: dotnet, DevOps, 工具
« 上一篇： .NET Core开源：IIS集中化Web管理工具
» 下一篇： .NET项目迁移到.NET Core操作指南