Building High performance ASP.NET web applications

Presenter: Peter Kellner, San Jose, CA

Blog: http://PeterKellner.net

Peter Kellner, Background

- Consulting With Large and Small Companies building scalable, high performance applications
- 2007,2008 MVP, ASP.NET
- Development including publishing 4 MSDN Articles on ASP.NET 2.0
- Organized 2006 and 2007 SV Code Camps
- Complete Custom Insurance Co. Management s/w to run $200M business.
- 1986 – 2001 President Tufden Inc. Built and Delivered: 500 doctor office turnkey computer systems; University Clinic Scheduling System;
- .Net Forms, Mobile and Web Application
- 1980 – 1986 Lockheed Missiles and Space Co, Senior Research Engineer
- Cornell University BS,MS Engineering

Code Camp 11/8-9 Speakers!!!
www.siliconvalley-codecamp.com

Silicon Valley Codecamp 2008 - Windows Internet Explorer

http://www.siliconvalley-codecamp.com/Sessions.aspx

Links | peterkellner.net › Manage Blogroll — WordPress | Customize Links | Gmail | MX Lookup Tool - Check your DNS MX Records online | iGoogle

Silicon Valley Codecamp 2008

SiliconValley codecamp_08

Login
Forgot Password?

// REGISTER // PROGRAM // NEWS // ABOUT // WIKI

Sessions Speakers For Presenters

For Speakers
My Profile
RSS Feed
FAQ

Sessions

Sort by: Session Title | Hide Descriptions

An Introduction to Comet and Bayeux (Pushing data to the browser) Intermediate Wiki Here
Kevin Nilson Intermediate
Agenda Not Made Yet | Room Unknown

Introduction to Comet and Bayeux. Will discuss AJAX polling, long polling, and Streaming. Examples will be run using Jetty. In depth details of Bayeux will be given.

AJAX
Comet
J2EE
Java
JavaScript

Not Interested | Interested (1) | Will Attend (No Time Set Yet)

Boosting Your Testing Productivity with Groovy Intermediate Wiki Here
Andres Almiray Intermediate
Agenda Not Made Yet | Room Unknown

Groovy
Java
Unit Testing

75 min sesions
Handouts with lots of Q&A time
Hands-on demos or exercises
Chalk talks or full-on slides
Experts sharing their insights
Share with others, etc.

...and free coffee and food!

Attendance is FREE, but space is limited so you need to Register.

Topics

View all topics

Show Top Tags Only

C#(5) Java(6) JavaScript(4) LINQ(4) Web development (6)

Internet | Protected Mode: Off 100%

Goals

- Understand Cornerstones of High Performance
- Understand How Performance Relates to Web Sites
- Hands On Methods and Tools to Improve Web Performance

Not Talking about Security… Maybe another talk later? Not unimportant, just too many things here already to talk about.

Agenda

- ASP.NET Fundamentals Relating to High Performance
- How IIS Basically Works with ASP.NET
- Web Farm Issues (Scale Out)
- SqlServer (Scale Up)
- Examples of Performance Helpers
- Survey of Tools to Help
- Questions

Cornerstones of High Performance

- Minimize Request/Response Processing Time
- Minimize Resources Used on Web Server
- Minimize Cost
- Maximize User Experience (response time)

HttpApplication pipeline

The following events occur:

1. Validate the request – checking for malicious markup
2. Perform URL mapping
3. BeginRequest event
4. AuthenticateRequest event
5. PostAuthenticateRequest event
6. AuthorizeRequest event
7. PostAuthorizeRequest event
8. ResolveRequestCache event
9. PostResolveRequestCache event
10. Find the requested resource, if it is a Page, compile the page
11. PostMapRequestHandler event
12. AcquireRequestState event
13. PostAcquireRequestState event
14. PreRequestHandlerExecute event
15. Call ProcessRequest on the HttpHandler
16. PostRequestHandlerExecute event
17. ReleaseRequestState event
18. PostReleaseRequestState event
19. Response filtering
20. UpdateRequestCache event
21. PostUpdateRequestCache event
22. EndRequest event
23. PreSendRequestHeaders event
24. PreSendRequestContent event

$$R \approx \frac{Payload}{Bandwidth} + RTT + \frac{AppTurns(RTT)}{\text{Concurrent Requests}} + Cs + Cc$$

Legend:

R:	Response time
RTT:	Round Trip Time
App Turns:	Http Requests
Concurrent Requests:	# server sockets open by browser
Cs:	Server Side Compute time
Cc:	Client Compute time

Source: Field Guide to Application Delivery Systems, by Peter Sevcik and Rebecca Wetzel, NetForecast / Richard Campbell

State and Why We Care

- Types of State
  - Application – Global to App
  - Session – Per Client
  - Cookie – Per Client
  - View – Across Post Requests

Application State

Not stored across machines

Initialize in global.asax typically

Set:

Application[“MyStuff”] = “Do not forget”;

“Does not Scale well”, use Cache instead

Session State

- Requires Cookies or URL Mangling
- Can Work across Web Farms
- Set:
- Session[“MySessionStuff”] = “save me”;
- Try to avoid, can be a problem using too much memory or database
- Can Store in SqlServer,InProc,StateServer,3rd Party

Cookie

- State stored on Client (if enabled)
- Limited (~4k) Sent back and forth on every request (including images)

Set:

```
HttpCookie myCookie = new
    HttpCookie("Stuff");

myCookie.Values.Add("thing1", "value1");

myCookie.Values.Add("thing2", "value2");

Response.Cookies.Add(myCookie);
```

View State

- Across Same Post
- Works well in Web Farms
- Size can make requests slow

Set:

View[“MyState”] = “Thing to Store”

Reduce View State Size

- By Default EnableViewState="true" (Turn off whenever possible)
- Create PageBase class to Inherit From with Warning if over certain value (policy)
- Look at sites like MySpace

(StrangeLoops Product, Richard Campbell)

Viewstate and AJAX

- What happens in UpdatePanel?
- Demo Code

Cache… Very Important!

- Output Caching
- Page Fragmentation (User Controls)
- Cache Static Class Use … (see next slides)

Is this Correct Cache Use?

```
[DataObjectMethod(DataObjectMethodType.Select, true)]
public List<BusinessObjectItem> GetDataCached1
    (int numberRecordsToReturn, int delayMilliSeconds)
{
    const string cacheName = "CACHE-LIST-BUSINESS";
    List<BusinessObjectItem> listBusinessObjectItems = null;
    if (HttpContext.Current.Cache[cacheName] == null)
    {
        listBusinessObjectItems = GetData1(numberRecordsToReturn,
            delayMilliSeconds);
        HttpContext.Current.Cache.Insert(cacheName, listBusinessObjectItems,
                                null,
                                DateTime.Now.Add(new TimeSpan(0, 0, 5)),
                                TimeSpan.Zero);
    }
    else
    {
        listBusinessObjectItems =
            (List<BusinessObjectItem>) HttpContext.Current.Cache[cacheName];
    }
    return listBusinessObjectItems;
}
```

Then, How about this?

```
[DataObjectMethod(DataObjectMethodType.Select, true)]
public List<BusinessObjectItem> GetDataCached1
    (int numberRecordsToReturn, int delayMilliSeconds)
{
    const string cacheName = "CACHE-LIST-BUSINESS";
    List<BusinessObjectItem> listBusinessObjectItems =
        (List<BusinessObjectItem>) HttpContext.Current.Cache[cacheName];
    if (listBusinessObjectItems == null)
    {
        listBusinessObjectItems = GetData1(numberRecordsToReturn,
            delayMilliSeconds);
        HttpContext.Current.Cache.Insert(cacheName, listBusinessObjectItems,
                                null,
                                DateTime.Now.Add(new TimeSpan(0, 0, 5)),
                                TimeSpan.Zero);
    }
    else
    {
        listBusinessObjectItems =
            (List<BusinessObjectItem>) HttpContext.Current.Cache[cacheName];
    }
    return listBusinessObjectItems;
}
```

And Finally, this one…

```
[DataObjectMethod(DataObjectMethodType.Select, true)]
public List<BusinessObjectItem> GetDataCached3(int numberRecordsToReturn,
    int delayMilliSeconds)
{
    const string cacheName = "CACHE-LIST-BUSINESS";
    var listBusinessObjectItemsGeneral =
        (List<BusinessObjectItem>)HttpContext.Current.Cache[cacheName];
    if (listBusinessObjectItemsGeneral == null)
    {
        Monitor.Enter(lockval);
        try
        {
            // check it again in case someone else loaded the cache and we
            // hit the null condition before the cache was fully loaded.
            listBusinessObjectItemsGeneral =
                (List<BusinessObjectItem>)HttpContext.Current.Cache[cacheName];
            if (listBusinessObjectItemsGeneral == null)
            {
                listBusinessObjectItemsGeneral =
                    GetData3(numberRecordsToReturn, delayMilliSeconds);
                HttpContext.Current.Cache.Insert(cacheName,
                    listBusinessObjectItemsGeneral,
                    null,
                    DateTime.Now.Add(new TimeSpan(0, 0, 5)), TimeSpan.Zero);
            }
        }
        finally
        {
            Monitor.Exit(lockval);
        }
    }
    return listBusinessObjectItemsGeneral;
}
```

Track Hits!

```
[DataObjectMethod(DataObjectMethodType.Select, true)]
public List<BusinessObjectItem> GetDataCached3(int numberRecordsToReturn, int del
{
    const string cacheName = "CACHE-LIST-BUSINESS";
    var listBusinessObjectItemsGeneral =
        (List<BusinessObjectItem>)HttpContext.Current.Cache[cacheName];
    if (listBusinessObjectItemsGeneral != null)
    {
        LogStatus(CacheStatus.CachedOnFirstTry);
    }
    else
    {
        Monitor.Enter(lockval);
        try
        {
            // check it again in case someone else loaded the cache
            // and we hit the null condition before the cache was fully loaded.
            listBusinessObjectItemsGeneral = (List<BusinessObjectItem>)HttpContex
            if (listBusinessObjectItemsGeneral != null)
            {
                LogStatus(CacheStatus.CachedOnSecondTry);
            }
            else
            {
                LogStatus(CacheStatus.NotCached);

                listBusinessObjectItemsGeneral = GetData3(numberRecordsToReturn,
                HttpContext.Current.Cache.Insert(cacheName, listBusinessObjectIte
                                                 null, DateTime.Now.Add(new TimeS
            }
        }
        finally
        {
            Monitor.Exit(lockval);
        }
    }
    return listBusinessObjectItemsGeneral;
}
```

Problems With Cache

- Locking Issues slow it down
- Not dependable (Captcha Control Story)
- Creates Lots of Objects in Memory (garbage collection a problem, look for gen2 garbage collection events)
- Easy to Misuse.. Hueristic Cache?
- Invalidation Big Problem

Server Issues

- Thread Pool
- Static File Compression
- Load Balancing (Web Farms)
- Thread Affinity
- SSL Processing

Thread Pool

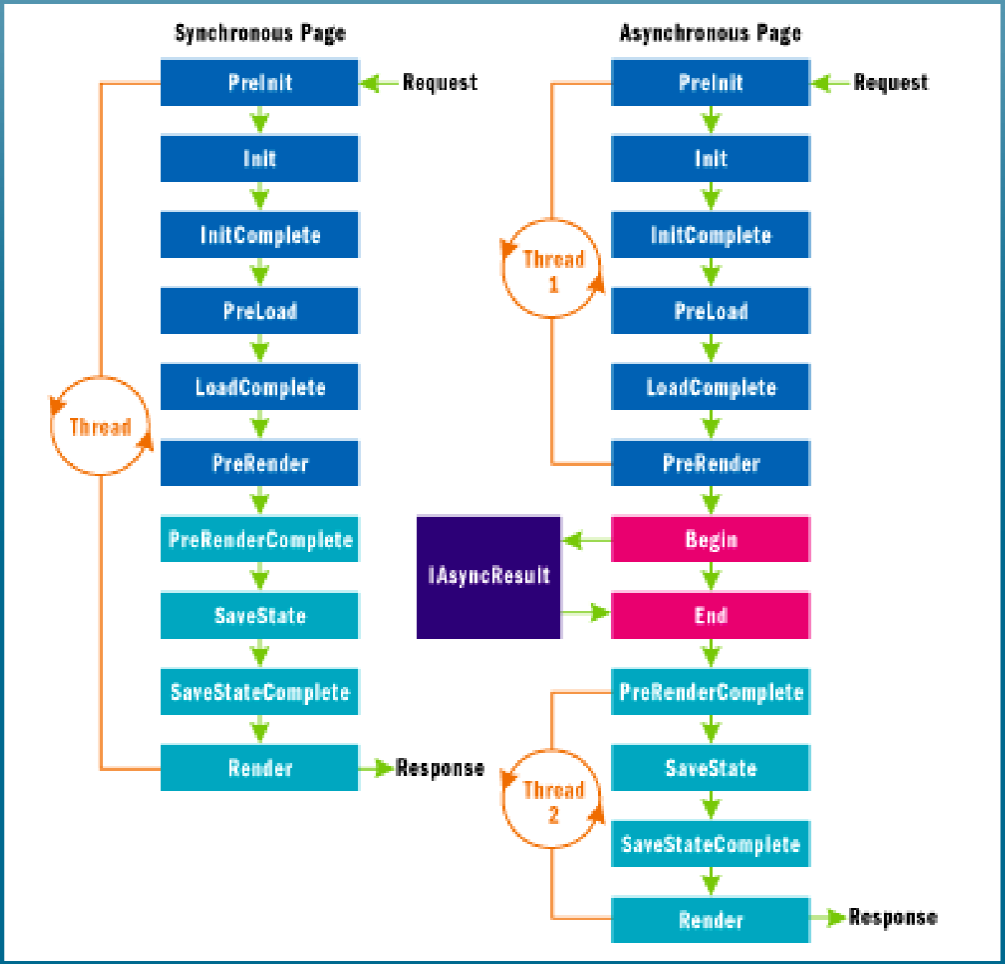
- Why do We have one?
- What Happens when we run out?
- Async Processing Demo (when request is paused, thread is tied up if not async)

Trace Information

Category	Message	From First(s)	From Last(s)
aspx.page	Begin PreInit		
aspx.page	End PreInit	0.00107835978252515	0.001078
aspx.page	Begin Init	0.00111958109024805	0.000041
aspx.page	End Init	0.00115643285113265	0.000037
aspx.page	Begin InitComplete	0.00117433208689637	0.000018
aspx.page	End InitComplete	0.00119148813879894	0.000017
aspx.page	Begin PreLoad	0.00120748931506117	0.000016
aspx.page	End PreLoad	0.00122324454558517	0.000016
aspx.page	Begin Load	0.00123912943228639	0.000016
aspx.page	End Load	0.00164951662974139	0.000410
aspx.page	Begin LoadComplete	0.00168384210016274	0.000034
aspx.page	End LoadComplete	0.00170126682105108	0.000017
aspx.page	Begin PreRender	0.00171810341082629	0.000017
aspx.page	End PreRender	0.00174685500228903	0.000029
	BeginAsyncOperation	0.00251726933397493	0.000770
	EndAsyncOperation	0.127788568201488	0.125271
aspx.page	Begin PreRenderComplete	0.342791961985344	0.215003
aspx.page	End PreRenderComplete	0.342842915526327	0.000051
aspx.page	Begin SaveState	0.343636202811668	0.000793
aspx.page	End SaveState	0.344381573451227	0.000745
aspx.page	Begin SaveStateComplete	0.344413917989126	0.000032
aspx.page	End SaveStateComplete	0.344430878888432	0.000017
aspx.page	Begin Render	0.344449847453493	0.000019
aspx.page	End Render	0.345032347211223	0.000582

http://msdn.microsoft.com/en-us/magazine/cc163725.aspx

Async Processing Chart



http://msdn.microsoft.com/en-us/magazine/cc163725.aspx

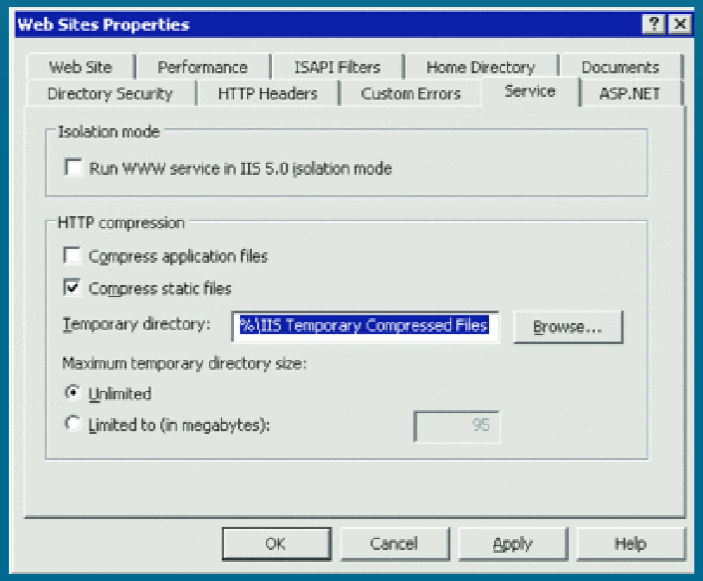
Async asp.net 2.0 Example*

```
<%@ Page Language="C#" Async="true" CompileWith="SlowWSAsync.aspx.cs"
            ClassName="PS.Samples.SlowWSAsync_aspx" %>

SlowWsAsync.aspx.cs:

using System;

namespace PS.Samples
{
  public partial class SlowWSAsync_aspx
  {
    Slow slowWebservice = new Slow();
    void Page_Load(object sender, EventArgs e)
    {
      BeginEventHandler bh = new BeginEventHandler(this.BeginGetAsyncData);
      EndEventHandler eh = new EndEventHandler(this.EndGetAsyncData);
      AddOnPreRenderCompleteAsync(bh, eh);
    }

    IAsyncResult BeginGetAsyncData(Object src, EventArgs args, AsyncCallback cb, Object state)
    {
      // Note - this is serviced on the same thread as Page_Load
      // but a different thread is used to service EndGetAsyncData
      //
      return slowWebservice.BeginHelloWorld(cb, state);
    }

    void EndGetAsyncData(IAsyncResult ar)
    {
      string ret = slowWebservice.EndHelloWorld(ar);
      Response.Write(AppDomain.GetCurrentThreadId());
    }
  }
}
```

*From Fritz Onion Example
http://www.pluralsight.com/blogs/fritz/archive/2004/10/19/2892.aspx

IIS Static File Compression

- Good for things like jpg's, css,e tc.
- Adds processing overhead
- Minimizes download (payload) size



http://msdn.microsoft.com/en-us/magazine/cc500561(printer).aspx?pr=blog

Web Farms / Load Balancing

- Two Types

Network Load Balancing (NLB) – Comes with win2003 Server. Rumor has it webtest will not balance.

Hardware (BigIP for example). Buy 2. works based on system loads.

Thread Affinity

Definition – Once Request starts on server, stays on server (State is “InProc”)

Avoid Because:

- Hard to Maintain (can’t shutdown)
- Hard to balance performance
- Data Loss – Losing server, loses session

Benefits of Losing Affinity

- Getting Session out-of-process reduces memory pressure on server
- Garbage collection easier for server

Microsoft Solutions:

SqlServer or State Server

Look at Third Party like NetCache, etc.

SSL Processing

- Try to do as little SSL as possible
- If Web Farm, put SSL on it's own servers if it makes sense

Database / SqlServer Issues

- Scale Up / Bottle Neck
- Create Readonly replications
- Can use for Session, Scary thought
- Connections

  open and close fast

  use using syntax

Sql Connection, Old Way

```
public static Dictionary<int, int> GetDictionaryOfInterestLevelByAttendee(string username)
{
    var dict = new Dictionary<int, int>();

    var sqlConnection =
        new SqlConnection
            (ConfigurationManager.ConnectionStrings["CodeCampSV06"].ConnectionString);
    sqlConnection.Open();
    SqlDataReader reader = null;
    try
    {
        var command =
            new SqlCommand(
                @"SELECT sessions_id,interestlevel
                  FROM SessionAttendee
                  WHERE attendee_username=@attendee_username",
                sqlConnection);
        reader = command.ExecuteReader();
        while (reader.Read())
        {
            int idSession = reader.IsDBNull(0) ? -1 : reader.GetInt32(0);
            int interestLevel = reader.IsDBNull(1) ? -1 : reader.GetInt32(1);
            dict.Add(idSession, interestLevel);
        }
    }
    catch (Exception eee1)
    {
        throw new ApplicationException(eee1.ToString());
    }
    finally
    {
        if (reader != null) reader.Dispose();
    }
    sqlConnection.Close();
    return dict;
}
```

Sql Connection, New Way (Using)

```
public static Dictionary<int, int> GetDictionaryOfInterestLevelByAttendee(string username)
{
    var dict = new Dictionary<int, int>();

    using (var sqlConnection =
        new SqlConnection(ConfigurationManager.ConnectionStrings["CodeCampSV06"].ConnectionString))
    {
        sqlConnection.Open();
        var command =
                   new SqlCommand(
                       @"SELECT sessions_id,interestlevel
                  FROM SessionAttendee
                  WHERE attendee_username=@attendee_username",
                       sqlConnection);

        using (SqlDataReader reader = command.ExecuteReader())
        {
            try
            {

                while (reader.Read())
                {
                    int idSession = reader.IsDBNull(0) ? -1 : reader.GetInt32(0);
                    int interestLevel = reader.IsDBNull(1) ? -1 : reader.GetInt32(1);
                    dict.Add(idSession, interestLevel);
                }
            }
            catch (Exception eee1)
            {
                throw new ApplicationException(eee1.ToString());
            }
        }
    }
    return dict;
}
```

ReSharper 4.0 helps with Using Syntax

Code Optimizers

- Redgate ANTS Profile
- Visual Studio Team Edition
- Hint: Use Generics!

Source code

Method: int Mandelbrot.Algorithm.EvaluateUsingComplexNumbers(double x, double y)

File name: C:\Program Files\Red Gate\ANTS Profiler 3\Tutorials\CS\Mandelbrot\Algorithm.cs

Algorithm.EvaluateUsingComplexNumbers(double x, double y)

Line	Hit count	Time (sec.)	
53			int EvaluateUsingComplexNumbers(double x, double y)
54			{
55	115104	0.0131	int iterations = 0;
56	115104	0.0494	Complex z = new Complex(x, y);
57	115104	0.0663	Complex z0 = new Complex(z);
58	1277980	0.248	while(iterations < m_MaxIterations && z.ModSquared < 4)
59			{
60	1162876	0.130	iterations++;
61			// this is the Mandelbrot algorithm in complex for
62	1162876	1.75	z = z * z + z0;
63			}
64			
65	115104	0.0067	

Source code | Hierarchy

Client Side Tools

15 Tools To help you make pages faster

http://sixrevisions.com/tools/faster_web_page/

Fiddler, Yslow, Firebug, Cuzillion,…

15. Site-Perf.com

Requests	162
Errors	1
Documents total size	3 027 393 B
Headers total size	58 416 B
Average download speed	8 765 Kbit/s

resolve	9.7 s	1
connect	1.16 s	
headers	5.8 s	
first_byte	24 ms	
body	2.9 s	

Id	Timings	Type
	0 1 2 3 4 5 s	
1		sixrevisions.com/

Load Testing

- Visual Studio Load Testing
- Mercury Interactive
- Lots!

Finding Problems

- Divide and Conquer
- Collect Data
- Isolate