ASP.NET Application Performance Tuning and Scalability Improvement (Part – 1)

By Md. Mahedee Hasan. Microsoft MVP, ASP.NET/IIS, Software Architect, Leadsoft Bangladesh Ltd.
Source: http://mahedee.net/asp-net-application-performance-tuning-and-scalability-improvement-part-1/

Every developers and customer wants his/her application performed well. But it is very tricky to improve performance of a web application. It's actually depends on different parameter like HTML client, HTTP network, Web server, middle-tier components, database components, resource-management components, TCP/IP networks, and database servers. Sometimes performance of the application increases drastically, if you change a single parameter. Sometimes you have to change multiple parameters. Here are the some performance optimization tips for ASP.NET web applications which may help you to increase performance of ASP.NET Application.

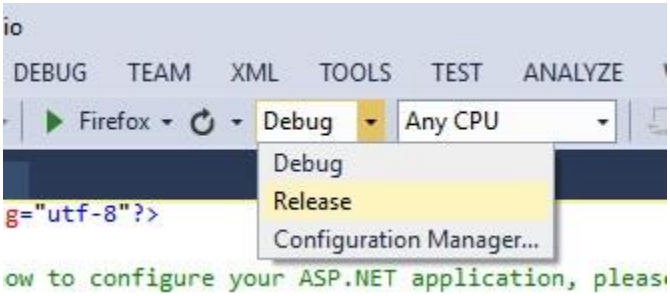
Prepare release with release mode
When prepare release for deployment, select release mode instead of debug mode.

How affect in performance?

- If you choose debug mode instead of release mode, you are creating pdb (program database – uses for debugging) files which creates extra overhead. For this reason you may face timeout problem.

Best Practice

- Always prepare release in release mode.



In Web.Config, Set debug="false"
Set debug="false" in web.config as follows after deploying your application. By default debug="true" when you create a web application. It is necessary to set debug="true" in development environment.

```
<system.web>
  <compilation debug="false" targetFramework="4.5.1" />
</system.web>
```

How affect in performance?

- If you set debug = “true”, application requires the pdb information to be inserted into the file and that results in a comparatively larger file and hence processing will be slow.

Best Practice

- In deployment server, set debug = “false” in web.config

Turn off Tracing unless until required

Sometimes developers need to trace the application to monitor the executions of application or a pages. It requires for application diagnostic purposes.

How affect in performance?

- When trace is enabled it loaded extra information to the pages which degrades performances.

Best Practice

- Always set trace enabled = “false” unless or until you required to monitor a page’s executions. Set trace enable = “false” as follows in web.config.

```
<system.web>
    <trace    enabled="true"    pageOutput="true"    requestLimit="10"    localOnly="false"
mostRecent="true" traceMode="SortByTime"/>
</system.web>
```

Carefully manage session state

Session state is a very useful feature of asp.net. Though, ASP.NET manages session state by default, we must pay attention of session memory management.

How affect in performance?

- When you store your data in in-process or on a state server or in a SQL Database, session state, it requires memory.
- It is also time consuming when you store or retrieve data in-process or state server or SQL server.

Best Practice

- If your page is static, it is recommended not to use session state. In such cases where you don’t need to use session state, disable it on your web form using the following directive:
  <@%Page EnableSessionState="false"%>

- In case you use the session state only to retrieve data and not to update, make the session state read-only using the following directive.
  ```
  <@%Page EnableSessionState ="ReadOnly"%>
  ```

- If your application session state is out of process then consider carefully whether there is a need of the state server or SQL Server mode.
- SQL Server session mode provides lower performance than state server mode.
- Try to avoid keeping object in session. Since it requires serializing then de-serializing which affected in performance.
- Use client-side state management than server side.

Disable View State of a page if not required

- It stores data in the generated HTML using hidden field not on the server.
- View State provides page level state management
- As long as the user is on the current page, state is available and the user redirects to the next page and the current page state is lost
- View State can store any type of data because it is object type but it is preferable not to store a complex type of data due to the need for serialization and deserialization on each post back

How affect in performance?

- It increases the total payload of a page when submitted and when serving request.
- Serialization and deserialization of the data is required when submitting data and gets requested data.
- View state increases the memory allocations on the server.

Best Practice

- Pages that do not have any server postback events can have the view state turned off.
- The default behaviour of the View State property is enabled, but if you don't need it, you can turn it off at the control or page level.
- Within a control, simply set the EnableViewState property to false, or set it globally within the page using this setting.

```
<%@ Page EnableViewState="false" %>
```

Use finally block to release resources

We always uses try, catch and finally block for exception handling. Finally block executes whether any exception occurs or not.

How affect in performance?

- Sometimes application occupy resources where as it doesn't need it. It is occur due to bad programming.

Best Practice

- Always use a finally block to release resources like closing database connections, closing files, disposing objects and other resources.

Avoid unnecessary round trips to the server

How affect in performance?

- Unnecessary round trips significantly effect on web application performance.
- It increases network latency and downstream server latency.

- Many data-driven Web sites heavily access the database for every user request. While connection pooling helps, the increased network traffic and processing load on the database server can adversely affect performance.

Best Practice

- Keep round trips as minimum as possible
- Use Ajax or partial page load instead of full page reload or refresh.

Choose low cost authentication

Authentication is a main factor for a secured applications. You must take decision which authentication will you use? Keep in mind, passport authentication is slower than form-base authentication which is slower than Windows authentication.

Use paging in grid view

In asp.net web application to show data in tabular format, generally we use grid view. Besides this we also uses DataGrid, JQgrid, Telerik grid, Kendo Grid etc. For huge number of data we cannot think general way because it takes huge time to load.

Best Practice

- To load grid view faster take advantages of paging, it shows small subsets of data at a time. JQGrid is faster than asp.net grid view because it does everything in client side.

Minimizes number of web server control

How affect in performance?

- The uses of web server controls increase the response time.
- Web server controls go to the server executes all of its life cycle and then rendered on the client side.

Best Practice

- Don't use server control unless until required. Use HTML elements where suited.