Title:
Optimizing Your Asp.Net Pages for Faster Loading and Better Performance.

Word Count:
928

Summary:
If you read the internet and all of the websites dedicated to Asp.Net you will inevitably read about the wonders of the DataGrid, DataList, and Repeater controls. While each of these has its place, if you are only displaying data there is a much faster and more efficient means to do so.

Keywords:
asp.net, web custom control, performance, caching, cache, asp.net controls, sqldatareader, sql server, classes, arraylist

Article Body:


If you read the internet and all of the websites dedicated to Asp.Net you will inevitably read about the wonders of the DataGrid, DataList, and Repeater controls. While each of these has its place, if you are only displaying data there is a much faster and more efficient means to do so.

Let's say you have a page that displays articles based on a query string. Take my article pages for instance. Each article is stored in a database and displayed on the page based on the unique id of the article as stored in the database.

A normal asp page execution procedure goes something like this. The code queries the database based on the Article I.D. and then brings back that information to the page where you display it in the fashion that you would like. This is a fairly straight forward approach with asp and is done all the time.

So how do we speed up our asp.net pages?

Number 1: Use Asp.Net Caching!

This is a no-brainer, and I won't go into the brilliance or details of asp.net caching here because at the time of this writing Google has 2,780,000 articles on the topic. Basically instead of querying the database each time the page is

loaded you only query the database once and load that result into the system cache. Subsequent calls to load the page retrieve the data from the cache as opposed to the database which gives you an instant and considerable performance boost. You can then set the cache for how long the cache should store the information as well as many other features. If you are not using the cache, you should be whenever possible!

Number 2: If possible, do NOT use the standard Asp.Net controls.

That's right. The standard asp.net controls are designed for rapid development and not page performance. They allow you to design pages that grab and display data very quickly but their actual performance suffers because of the extra overhead which is there for ease and speed of development time and not page execution speed.

Instead, create either a User Control or even better yet a Web Custom Control which is by far the fastest performance wise and really quite easy to create and use.

Number 3: Use an SqlDataReader or even better yet use a set based command for Sql Server data retrieval and simply execute that one command against the database.

An asp.net SqlDataReader is a fast forward only datareader that closes the connection after it reads the last set of results. Now for my article pages we are only returning 1 particular result. In this case we would opt for the set based command. If you had more than 1 result returned, in your table of contents for instance, you would use the SqlDataReader because you are returning multiple results.

Set based commands are stored procedures that bring back data through parameters as opposed to a result set which then in turn needs to be looped through to obtain your data. So instead of writing your stored procedure like the following which brings back 1 result set:

```
Select Title, Body, Author
From Articles
Where ArtID = 215
```

We can write it using a set based command like this.

```
Create Procedure mysp_GetArticle

@Title varchar(200) Output,
```

```
@Body varchar(8000) Output,
@Author varchar(500) Output

As

Select @Title = Title, @Body = Body, @Author = Author
From Articles
Where ArtID = 215

GO
```

The above query will return only the three parameters called for and not a result or record set so you don't have to then walk through the returned record set that has only 1 result in it anyway. This second little process of work decreases your performance so do not use it if you can. Combine this technique with the asp.net cache.

Number 4: Use Classes and ArrayLists as opposed to returning an SqlDataReader.

Create a class and then if there are more than one set of results store those results into individual instantiations of that class. Finally store each of those classes into an ArrayList. You can then store only that ArrayList into the asp.net cache. So instead of getting the results back from a SqlDataReader when loading your page you get them from the ArrayList which is stored in the cache. Nice huh?

Finally... you want to incorporate all of these techniques into your final results which would be performed in the following manner and sequence.

On the first time the page loads, query the database and return all of your data storing it into individual classes. Then store each of those classes into an ArrayList. If you only have one single result you may store only the class into the cache. Then take your ArrayList and store it into the cache.

Next create a Web Custom Control and pass the cached ArrayList to the custom control and loop out your data using the HtmlTextWriter which is very fast. Remember each subsequent call to load the page will be called from the cache which stores your ArraList of classes or your single class.

Certainly it takes a significant amount of additional coding to do it in this fashion, especially when you take proper error handling into consideration, but if you follow this approach your pages will be screeching fast, you will immediately notice the difference, and your asp.net pages will execute in the

proper sequence - Data handling in the Page_Load function and the html display in the Page_Render function. Further, you will be glad you did and so will your visitors.

Happy Programming!