Performance penalty on putting classes in aspx and ascx codebehinds

Asked 16 years, 2 months ago Modified 16 years, 2 months ago

Viewed 794 times

2

What's the performance penalty on defining classes in an aspx/ascx codebehind rather than compiling them into a dll beforehand? I know that this isn't a best practice and that there are numerous problems with this (e.g. difficult to unit test, code is not reusable, etc.), but it does come in very handy when you're dealing with classes that need to be modified on the fly several times a day since those modifications will not require any sort of app restart (e.g. App_Code changes, updating dlls in bin folder).

asp.net performance

Share

Improve this question

Follow

asked Sep 25, 2008 at 0:33

Kevin Pang

41.4k ● 38 ● 122 ● 173

5 Answers

Sorted by: Highest score (default)

"None." The codebehind classes are compiled into a DLL on the fly, and then that DLL is kept around. So basically the first time you load the page there will be a short delay, but afterwards the speed should be the same as with precompiled classes.

8

Share Improve this answer

Follow

answered Sep 25, 2008 at 0:36

CodeRedick

7,415 ● 8 ● 48 ● 75

> Read my post again, I said there would be a performance penalty the first time you load the page. Afterwards it's compiled though, and you're fine. – CodeRedick Sep 26, 2008 at 14:59

You should see no performance issue after the initial compile. It sounds as though you have business logic that is changing frequently, and not necessarily the web pages.

1

Share Improve this answer

Follow

answered Sep 25, 2008 at 0:54

David Robbins

10k ● 7 ● 54 ● 83

The choice of whether to use dynamic compilation or compiled DLLs really has to do with how organized your release process is. If your application is tightly compiled into DLLs than you can expect that you've tested for build

1

errors and expect things to be more sturdy when you release. With dynamic compilation you have the ability to swap out .cs files on the fly (e.g. drag & drop, ftp). This means you may be more agile, but you might not have that extra step of assurance that helps you know you're keeping the build intact.

Share Improve this answer

Follow

answered Sep 25, 2008 at 1:20

devlord

4,154 ● 4 ● 41 ● 56

1

Collateral damage - session resets

From personal experience, users are much more likely to complain about session reset caused by App Domain recycling than about slight performance hit. So if you can shift your changes from code to data and avoid code updates altogether, by all means do it. This will improve your users' performance :)

Share Improve this answer

Follow

answered Sep 25, 2008 at 8:35

Constantin

28.1k ● 10 ● 63 ● 79

0

I don't believe there really is a performance penalty after the initial dynamic compilation (which will occur on the first hit to the page whose code-behind was modified). How did you end up having to change classes several times a day? That would suck!

EDIT: I should've added that this shouldn't affect unit tests or the code-reusability like you stated. There's nothing stopping you from deploying a non-pre-compiled site for maintainability purposes while still being able to run unit tests, deploying compiled assemblies for other projects (if needed), etc. during a check-in/build.

However, if you're not using source control and don't have an automated build, then there's a whole new problem. Our team members used to edit CODE files directly on production servers. *shivers*

Share Improve this answer Follow

edited Sep 25, 2008 at 0:55

answered Sep 25, 2008 at 0:37

brock.holum

3,213 ● 2 ● 22 ● 15

Happens to me on our production site constantly, some customer will make some bizarre request that just "has" to be in place by the end of the day... – CodeRedick Sep 25, 2008 at 0:44