

Re: RADIest Client for SQL Server

Source: <http://www.tech-archive.net/Archive/SQL-Server/microsoft.public.sqlserver/2004-11/0087.html>

From: Michael C (*mculley_at_NOSPAMoptushome.com.au*)

Date: 11/18/04

Date: Thu, 18 Nov 2004 11:54:26 +1100

"Mike MacSween" <mike.macsween.nospamplease@btinternet.com> wrote in message news:419be7a6\$0\$221\$5a6aecb4@news.aaisp.net.uk...

> *So were they MDB front ends or ADP front ends? I know about Access as a back end*

The projects I've done all in access have been mdb back and front ends. When I've used SQLServer I've used either vb6 or .net as the front end. But my limited understanding is that ADP is designed for SQLServer so is probably the best method.

BTW, I thought because of the title of your post "RADIest Client for SQL Server" that you were asking which front end to use, not which method to use in access, so I may have gone off the track a little :-)

> *Well make a damn start!! Don't make smart alec comments about how such and such a thing is better if you're just going to keep it to yourself. If .net is so massively better than Access as a client to SQL Server then just tell us the first 10. At least I could look at them and see if it's worth loading my copy of VS.net onto the machine.*

Ok, other's have listed a few reasons already but here's the first 10 things I thought of. Most of these apply to any programming language (eg vb6, delphi etc).

- 1) Access starts off with all functionality and you have to find ways to restrict this it, eg tricks to hide toolbars etc. It's easy to miss some functionality or not know how to restrict it. No huge problem but I prefer to start with nothing and add the functionality that I want myself.
- 2) You have much better control over forms in dot net than access, such as border style and sizing.
- 3) The controls look like real windows controls, access controls have a different look and functionality.
- 4) There is much greater freedom in the way you can program in .net. In access you struggle if you don't want to do it access's way.
- 5) Web services.
- 6) Dlls, ability to componentize/encapsulate code, UserControl
- 7) oop features such as inheritance, constructors, interfaces etc.
- 8) Controls have a windows handle so APIs can be used. General support for

APIs and subclassing is much greater.

9) Works better with aftermarket usercontrols. Access's does support usercontrols but it's a bit clunky.

10) Greater reusability of code, for example a class that calculates interest paid on a loan could be used in a web page and windows app.

I'm not sure how good the report writer that comes with .net is but you'd probably need to purchase Active Reports or Crystal Reports to have something comparable to access reports (IMO, the report writer in access is it's best feature :-).

Regards,
Michael