OO? OI? Oy Vay!

Posted At : July 17, 2009 8:15 AM | Posted By : Steve
Related Categories: ColdFusion, OO Principles

Clark Valberg recently convinced **Hal Helms**, **Brian Kotek** and **Ben Nadel** to do a recording together discussing **OO programming in ColdFusion**.

The discussion reminded me of an **NPR** debate I listened to several months ago on whether we should bomb Iran to prevent them from acquiring nuclear weapons (unfortunately, I don't remember the specific wording of the question). The debate had three experts on either side of the question. All six seemed to agree that we would almost certainly never need to bomb Iran because so many better options exist but that we should remove the option from the table (just in case). From there it quickly moved to a semantics debate about the wording of the question on the table.

In the ColdFusion OO discussion, Brian Kotek described the use of a CFC service layer (which is what I do) as Object Oriented Programming, just not "Full blown OO". Hal seemed to agree that this was a reasonable approach, but not OO.

This is part of an interesting pattern that I am seeing lately of loosely (or tightly) defining "OO". It reminds me of Matt Williams " **Sip the OO Kool-Aid**" remark recently that I loved so much (part of the "**OO Nearly Destroyed my Business**" conversation).

In part this makes me think that the words we use are getting in the way of our understanding. This is a point I tried (ineloquently) to make in a recent blog post. If OO can mean so many different things (including, even, a development style that needn't have actual objects), then no wonder it is hard to grasp - slippery sucker.

In the **ColdFusion OO Google group list** (set up by Clark Valberg to accompany the audio), **Brian Meloche** coined the term "OI" for "Object Influenced" programming. On one hand, I am skeptical that introducing a new term can solve the problem of the ambiguity of the existing term On the other hand, I can think of nothing better.

Moreover, I like this idea. Part of what happens in discussions about OO is that it is compared to "Procedural Programming" which is often a straw man for "Spaghetti Code" (Functional Programming gets ignored). Those of us who follow OO Principles without doing "Full Blown OO" can find a lot of time wasted in discussions because of the assumption that anything that isn't OO is spaghetti code.

Here is my definition for OI (Brian, feel free to jump in if you would have defined it differently):

> Objected Influenced Programming is a programming strategy that uses many of the principles of Object Oriented Programming, but need not use all of them. Specifically, the use of "objects" is not required.

The debate, for me, is whether certain principles much be (at least mostly) followed in order for code to meet even the very loose definition of OI. For example, I would think that encapsulation and decoupling would be essential.

Personally, I think I do a sort of "SOP" (Service Oriented Programming - as long as we are making up terms) that follows most principles of OO, but specifically uses services as the API (which themselves may, or may not, use objects).

Even so, I think a discussion of OI in ColdFusion might be helpful. That is, how can we use OOP principles without going to "Full blown OO". Hopefully my " **OO Principles**" series will help with that. On top of that, I think it is about time for me to start publicly discussing the framework I have been using for my development for the past few years. I just need to figure out the best way to start that discussion...

Anyway, a big thanks to Brian Meloche for coining the term "OI". Hopefully we can have more enlightening conversations about this in the near future.