A Recursive Function to Gather CFC Metadata for Inherited Properties

Posted At : December 21, 2009 1:40 PM | Posted By : Bob Silverberg
Related Categories: ColdFusion, CF ORM Integration, BPO

Charlie Stell made a comment on my last post about my Base Persistent Object project, pointing out that my populate method did not take inherited properties into account. When I first developed it I wasn't using inheritance with ColdFusion ORM, as I'm coming from a Transfer background and Transfer doesn't really support inheritance. On a recent project using CF ORM I did implement an inheritance scheme and decided to update my populate method to support that.

The reason that the current populate method does not support inheritance is that is simply looks at the properties array that is included in the cfc metadata of the current object. That array of properties does not include any inherited properties. To find those you need to look at the extends key in the metadata structure and check for any properties of the object that the current object extends, and, of course any objects that that extends, etc. You could potentially have several levels of parents as you move up the inheritance hierarchy.

I decided that the best approach would be to write a recursive function that would return all of the properties of an object as an array, including all properties of any of the object's parents. In his comment Charlie provided some sample code that he is using, and I took his suggestion and made a couple of changes to it. Here's the code:

```
private array function collectAllProperties(required struct md,array props=ArrayNew(1)) {

 local.prop = 1;

 if (structKeyExists(arguments.md,"properties")) {

  for (local.prop=1; local.prop <= ArrayLen(arguments.md.properties); local.prop++) {

   if (not ArrayContains(arguments.props,arguments.md.properties[local.prop].name)) {

    arrayAppend(arguments.props,arguments.md.properties[local.prop]);

   }

  }

 }

 if (arguments.md.extends.fullname neq "WEB-INF.cftags.component") {

  arguments.props = collectAllProperties(arguments.md.extends,arguments.props);

 }

 return arguments.props;

}
```

To use this function, you simply pass in the metadata of an object into the *md* argument, so if I wanted to get the metadata for all properties for a User object I would do something like:

```
User = EntityNew("User");

userProperties = collectAllProperties(getMetadata(User));
```

The collectAllProperties function then starts by adding all of the User object's properties to an array, after which it checks to see which cfc the User object extends. Let's say in this example that it extends a Person.cfc. Because Person.cfc isn't *WEB-INF.cftags.component* (which all cfcs ultimately extend), it then calls the collectAllProperties() function recursively, passing in the *extends* key, which is in itself a complete package of cfc metadata, and it also passes it the current array of properties that is being built. This process will continue, with each cfc in the inheritance hierarchy being interrogated, adding properties to the array as long as they don't already exist in the array. When the recursion finally comes to a parent that is *WEB-INF.cftags.component* it knows it doesn't have to go any further, so the recursive calls stop and the full array of properties is returned to the caller.

I haven't added this logic to my Base Persistent Object project yet, as there are a few other changes I want to implement as well, but I thought I'd put this simple function out there for anyone that's interested.