Getting Started with VT and Transfer - Part I - What You Need

Posted At : March 11, 2009 7:58 AM | Posted By : Bob Silverberg
Related Categories: ColdFusion, ValidateThis

Update: The information in this post is no longer correct due to changes to the framework. A new series is available via the Getting Started with VT category of my blog.

Original content of the article follows:

In this initial post in my series about getting started with Transfer and ValidateThis!, my validation framework for ColdFusion objects, we're going to look at the setup for a simple demo application.

First off, we'll need the following ColdFusion frameworks:

- Transfer ORM, at least version 1, the latest of which can be found here.
- Coldspring, version 1.2, which can be found here.
- ValidateThis!, the latest of which can be found here.

The sample application that we're going to work through is a scaled down version of the demo that can be found at www.validatethis.org. We're going to start with a simple, one screen app that doesn't include any validations, and then integrate VT into the model and define a few validation business rules. The example will not be using an MVC framework, but will be using Coldspring to wire the dependencies together.

In addition to the above mentioned frameworks, we're going to need a database table. Here's a script to generate the User table (MS SQL Server):

```
CREATE TABLE dbo.tblUser (

 [UserId] [int] IDENTITY(1,1) NOT NULL PRIMARY KEY,

 [UserName] [varchar](100) NOT NULL,

 [UserPass] [varchar](50) NOT NULL,

 [Nickname] [varchar](50) NULL

)
```

And, of course, we'll need a transfer.xml file to define our User Business Object:

```
<?xml version="1.0" encoding="UTF-8"?>

<transfer xsi:noNamespaceSchemaLocation="transfer.xsd" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">

 <objectDefinitions>

  <package name="user">

   <object name="User" table="tblUser">

    <id name="UserId" type="numeric" />

    <property name="UserName" type="string" />

    <property name="UserPass" type="string" />

    <property name="Nickname" type="string" nullable="true" />

   </object>

  </package>

 </objectDefinitions>

</transfer>
```

To allow Transfer to find our database table, we're going to need a datasource.xml file as well:

```
<?xml version="1.0" encoding="UTF-8"?>

<datasource xsi:noNamespaceSchemaLocation="datasource.xsd" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">

 <name>VTAndTransfer</name>

 <username></username>

 <password></password>

</datasource>
```

That means that to run this sample app you'll need a DSN defined to ColdFusion called *VTAndTransfer* that points to the database containing the tblUser table.

Finally, we need a coldspring.xml config file in which we'll include the setup for Transfer, as well as defining the bean for our Service Object:

```
<!DOCTYPE beans PUBLIC "-//SPRING//DTD BEAN//EN" "http://www.springframework.org/dtd/spring-beans.dtd">

<beans default-autowire="byName">


 <!-- Transfer beans-->


 <bean id="transferFactory" class="transfer.TransferFactory">

  <constructor-arg name="datasourcePath">

   <value>/model/config/datasource.xml.cfm</value>

  </constructor-arg>
```

```
  <constructor-arg name="configPath">

   <value>/model/config/transfer.xml.cfm</value>

  </constructor-arg>

  <constructor-arg name="definitionPath">

   <value>/TransferTemp</value>

  </constructor-arg>

 </bean>


 <bean id="transfer"

  factory-bean="transferFactory" factory-method="getTransfer" />


 <!-- Service beans -->


 <bean id="UserService" class="model.service.UserService" />


</beans>
```

For anyone already using Transfer the above configuration files should be straightforward. In the interest of keeping these posts short I'm going to stop here for now.

I really want to keep this series focused on simply integrating VT into an existing Transfer application, so the sample app that we'll be working with is not reflective of the way that I actually use Transfer. My goal is to create the simplest possible app that uses Transfer and demonstrate how to integrate ValidateThis into that app.

If you are interested in how I structure and code my real world applications with Transfer, I've written extensively about it in the past. You can find those posts here.

In my next post I will go over the code that comprises the initial sample app, which will give us a starting point for integration.