David Roossien
CS 658
2/26/2009

Photography Website Framework Design

Introduction

In the past I have made extensive use of objects on the web server and modeling them after the table structure in the database. Consequently, I identified object orientation and object exchange as key feature that needed to be provided by the web service. Some research eventually led me to two possible options for messaging, namely JavaScript Object Notation (JSON) and the Action Message Format (AMF). Both appeared to be a convenient method for object exchange, but AMF intrigued me because of its support for PHP and its reported higher performance. In AMF, each PHP object on the server can become an ActionScript object on the client. Alternatively, Remote Objects on the client can be defined to reach into the server through the gateway and access class methods.

PHP as the Server Language

PHP (v5.X) was selected as the server language because most inexpensive web hosting services provide PHP/MySQL server technology. Other server languages like JSP and ASP are less common on smaller websites and generally more complex to design. PHP was also selected because I have a legacy system that uses PHP/MySQL. Fortunately, the existing website provides several objects that can be modified to support the new design. For example, the existing website has a photo and catalog table that provides corresponding photo and catalog classes for extracting and supplying data to the Flex client.

Prototyping

Prototyping was used to validate several important concepts. It also revealed that the Zend Framework could only be used with PHP 5.2 or later.

Prototyping included the following steps:

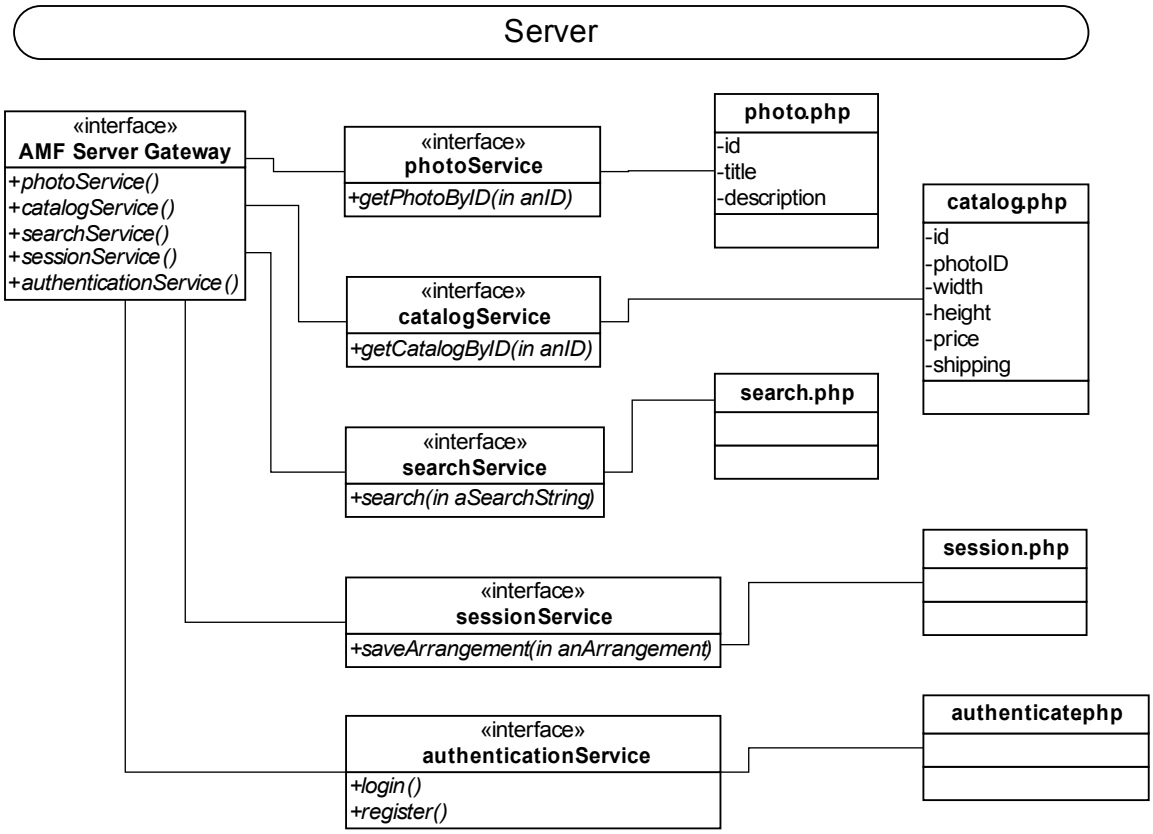
- Installing Flex Development Studio and obtaining a student license
- Installing the Zend Framework v1.75. Zend Framework provides a PHP Server gateway for exchanging AMF messages with a Flex/Flash application.

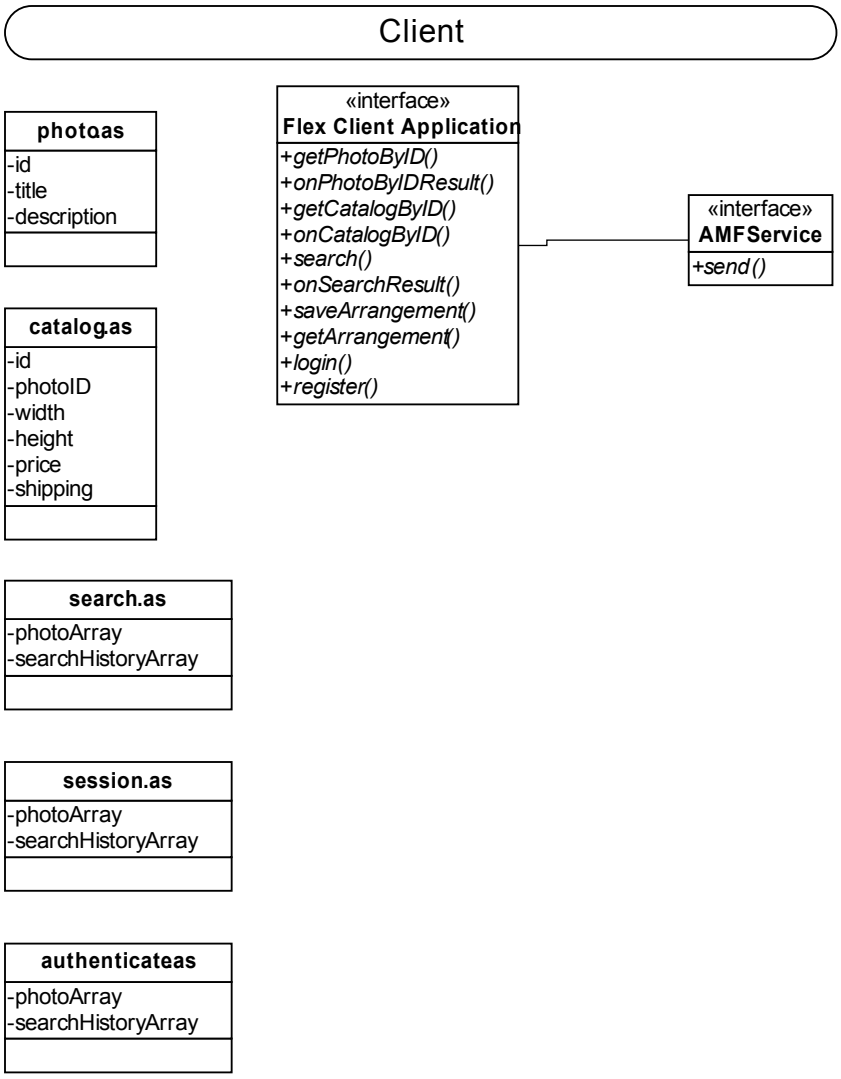
- Designing and testing an application using the Zend Framework for exchanging PHP objects with a Flex client using the AMF protocol. The PHP server objects retrieve their data from a MySQL 5.X database.
- Designing and testing an application for making Remote Object calls from the client and getting return data from the server.
- Testing a simple SESSION management interface.

Prototyping also revealed that errors on the server side can be difficult to troubleshoot. For example, if there is an error on the Server, a generic error occurs on the client that says "NetConnection error". Any error on the gateway, even the slightest mistake can become a huge problem. Even with the Zend Framework call $server->setProduction(false) on the server, fault status messaging was often difficult to interpret and no reference was to be found. As a result, each PHP server object will need to be unit tested independently from the client.

Design

All presentation decisions will be made on the client. Most data will be provided in the form of objects or lists of objects. Objects will be exposed through an interface or gateway. This will simplify the presentation logic. The below diagram shows some of the objects that will be implemented:


Server
«interface»
AMF Server Gateway
+photoService()
+catalogService()
+searchService()
+sessionService()
+authenticationService()
«interface»
photoService
+getPhotoByID(in anID)
photo.php
-id
-title
-description
«interface»
catalogService
+getCatalogByID(in anID)
catalog.php
-id
-photoID
-width
-height
-price
-shipping
«interface»
searchService
+search(in aSearchString)
search.php
«interface»
sessionService
+saveArrangement(in anArrangement)
session.php
«interface»
authenticationService
+login()
+register()
authenticatephp



As you can see from the above diagrams, the same classes are provided on the Server in PHP and on the client on ActionScript. Not all classes will be exposed in this way, but this will be the primary method for information exchange. MXML elements on the client use the AMFService to call the gateway on the server. The gateway routes the call and the parameters to the PHP interface/service for each object type. The interface will instantiate an object using the parameters that were passed by the client and return the object to the result handler on the client (i.e. onSearchResult). The result handler will instantiate the corresponding ActionScript object and copy the received PHP object receives onto it. The client will use the ActionScript object to update the display.

Security / Design Concerns

Clients will use an SSL connection (https) for loading the SWF and for all communication between the client and server. Authentication will be tied to the server

SESSION at the gateway. The final implementation of the design should provide access to some web services without authentication, but others, such as user account information services will be protected using authentication and session.

One of the powerful aspects of the design is that it allows object exchange, but that's also one of the most dangerous features because it exposes all of the data in the objects to the SWF file. There are tools for reverse engineering SWF files. If someone were to reverse engineer the SWF file, then it would be possible to gain access to some of the more sensitive private data classes. For example, once authenticated, a hacker could write an application to retrieve the full contents of my catalog tables. The web services class design should provide access to information, but not make it too easy to retrieve all the contents of the database. A special case is the server SESSION information that will need to be shared with the Flex client. Some SESSION data will likely be public information and used for maintaining the state of the client on the server. However, other data and state information may be private information that should never be shared with a client, or allow the client to control it. The web server will need to retain control over some private methods and data that cannot be accessed at the client.

The application will follow best practices for flex application security are found here:

http://livedocs.adobe.com/flex/3/html/help.html?content=security2_16.html#138204

Search Engine Optimization

My current photography site is designed so that if a user navigates to a URL with a parameter, webPageName = 'lakemichigan' then the user is presented with a view of Lake Michigan images. I plan to extend this into Flex and make some of the content index-able by search engines i.e. Search Engine Optimized. Although I do not fully understand the concept, deep links ("pseudo-anchors") will be used to link specific URL's into "states" of the application.