Introduction

This system, code named PostIt, is a new look on the old form management concept. It was designed as a form administration tool that developers could expand on. To accomplish this, forms are created in the Survey Manager interface and stored as XML templates. Developers leverage these XML templates to display forms in any fashion. These formats could be basic HTML, XML-compliant JavaScript Frameworks, or Flex to name a few.

For this demonstration, I've built a front-end tool that displays surveys with simple CSS & JavaScript. This includes a few new CF9 JavaScript features. I've also constructed a processing engine that tracks survey results in a MySQL/PGSQL database. The results can be viewed in the Survey Manager tool under the Results tab. Several CF9 features are used throughout this. The examples were intentionally kept simple to show just how powerful they can be in context of an application.

Installation

Unzip the package and place the "qa" directory it in your web root. Then navigate to the following:

http://<yourserver>/qa/install

Once you have completed the onscreen instructions, you can begin using the tool. To fully understand where & how new CF9 features are being used, it helps to understand the application architecture. I've included a section on this as well as a breakdown (as best I can remember) of where CF9 features appear.

License

This application can be used for learning/review purposes only. After more testing has been done, I may release both an open source and commercial version of the tool.

Table of Contents

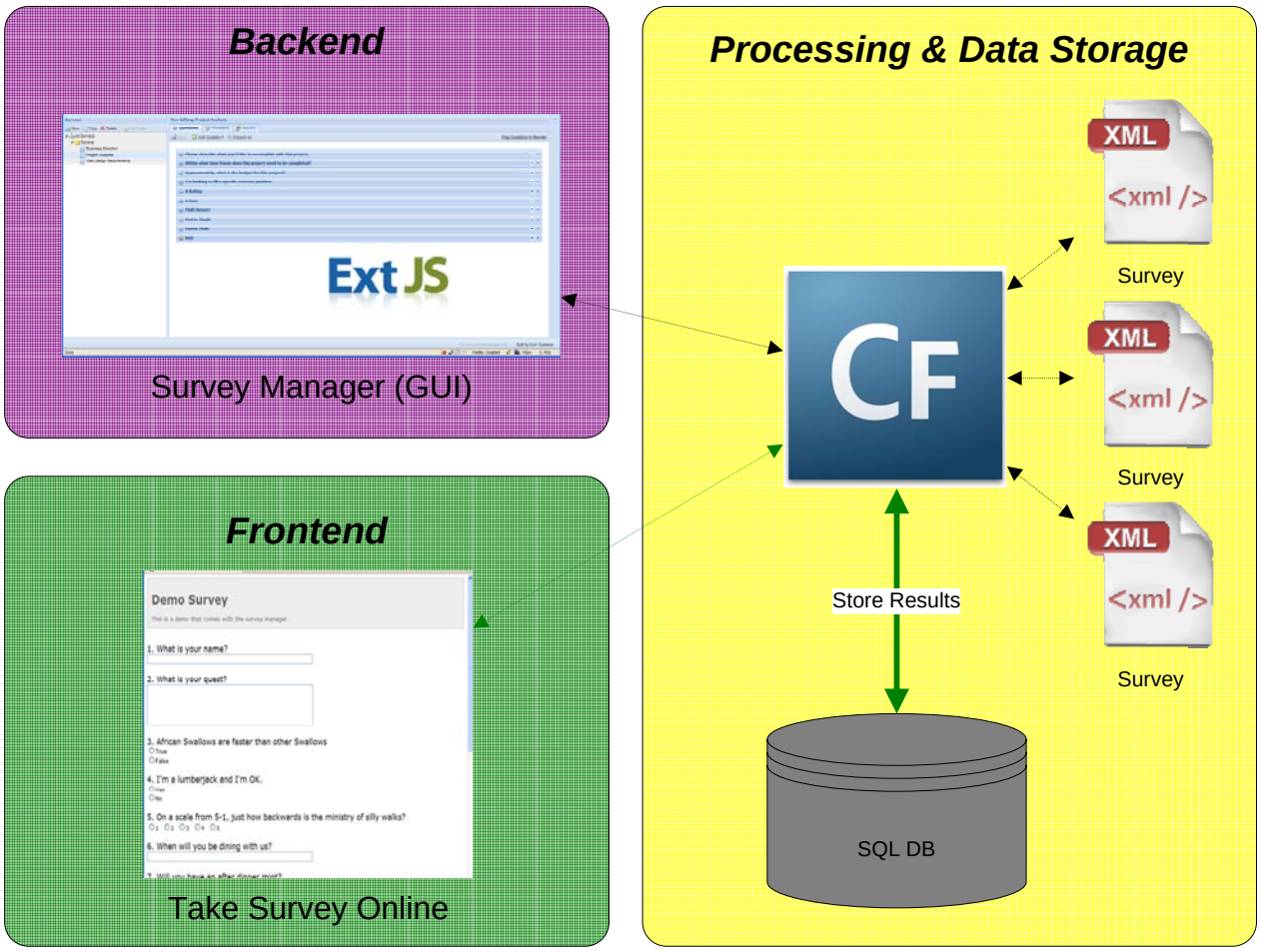
Use of CF9

Notes: Appending ?restart to URL will flush the entire application.

File	CF9 Highlights	Purpose
Server.cfc	Shows a *really basic* use case for the new OnServerStart method.	Our Service.cfc uses the OnServerStart method to identify whether the code is running in production or development. This is not critical to the execution of this application, but I use it to simplify deployment/error management. In production all errors are automatically logged to a Bugzilla instance (see bugconnect.riaforge.com) whereas they show up onscreen in development. We also use it to set certain default values used across all applications (like pointing to LDAP auth servers).
Application.cfc	Default DSN & Email Settings	Leverages the new datasource & smtpserversettings properties of Application.cfc to define default values for CFQUERY & CFMAIL.
Survey.cfc	CFSCRIPT used to create the component.	This represents a survey object that does NOT use ORM. It reflects data stored in an XML file. This was converted from a standard CFC written for CF8 to show how easy it is to get used to the new syntax as well as a highlight of a few nice new features.
Survey.cfc	Use of VFS	Every time a survey is taken by an end user, the Survey CFC object reads the questionnaire (i.e. XML file) unless caching is used (which it is in these examples). For example sake, this illustrates how reading from RAM is faster than reading from disk. This would minimize processing times for surveys with a lot of participants. Consider this example as an alternative to Ehcache.
cache.cfm	Some new CFSCRIPT functions.	Simple use of FileDelete (VFS) and WriteDump functions.
Take.cfm	cacheGet, cachePut	Displays a survey as an HTML form. Leverages the new Ehcache capabilities to store objects and/or entire HTML pages for quick rendering.
Take.cfm	CFMESSAGEBOX extension	Shows how some Ajax components can be "extended"[1]. The JS submit function on this page is called by the CFMESSAGEBOX callbackhandler. Since CF uses the ExtJS library, it loads ExtJS 3.0 and allows simple use of some regular ExtJS objects. This example uses a basic ExtJS loading mask along with a CF Ajax submission, all tied together with multiple CFMESSAGEBOX tags.
Download.cfm	CFSPREADSHEET	Uses several of the new spreadsheet functions to create a downloadable version of a survey.
report.cfm, result.cfc	ORM	Uses EntityLoad() function to access survey results on an individual question level. Also leverages caching when applicable.
Map.cfm	CFMAP	A simple example of CFMAP used to show where users are taking surveys from. Due to the fact my geolocation DB is behind my firewall, I've created some hard-coded lat/long values stored at the server level (see Server.cfc). These values are randomly assigned whenever a survey is taken.

[1] This is not a true "extension" of a CF component, but rather a means of integrating additional 3rd party technology that's already directly included in CF9. A key take-away here is CF9 has once again simplified and extended another technology that new developers can use without learning a new framework, all without restricting the creativity/options for more advanced developers.

Application Architecture



The backend was written entirely in JavaScript using the ExtJS framework (non-CF). Using AJAX, the backend communicates with the processing and storage engine. ColdFusion is responsible for physically making changes to the XML files (each of which represents a survey) & managing the cache.

***Key Highlights*:**
Use CF to support a non-CF application. Shows off some AJAX data functionality capabilities, as well as new cache control (VFS/Ehcache) features of CF9. The survey CFC was originally written with CF8 in a tag-based syntax. It is now written in a CFSCRIPT-based syntax, showing how simple it is to upgrade (took about 5 minutes with learning curve).

The frontend is an extension to the core application. The whole purpose of this tool was to allow developers deep control over the display and processing of surveys/forms created by business users. The frontend was written specifically for the Best of ColdFusion 9 contest, leveraging CF9 features throughout.

***Key Highlights*:**
Use of caching is illustrated from the front-end (i.e. get), as well as illustration of several of the CF9 JavaScript-oriented tags. CFMESSAGEBOX is “extended” using features of ExtJS that are not documented in CF but are documented in ExtJS. Several other new tags are illustrated, such as CFSPREADSHEET for downloading forms, CFMAP for some survey results, and CFaaS for report generation.

ADDITIONAL ARCHITECTURE HIGHLIGHTS

Many new CF9 features target configuration at an application level.

OnServerStart() (Server.cfc)
This new functionality is used to create an array of location structs. This array can be used to map a pretty name like “Chicago” to it’s corresponding latitude/longitude.

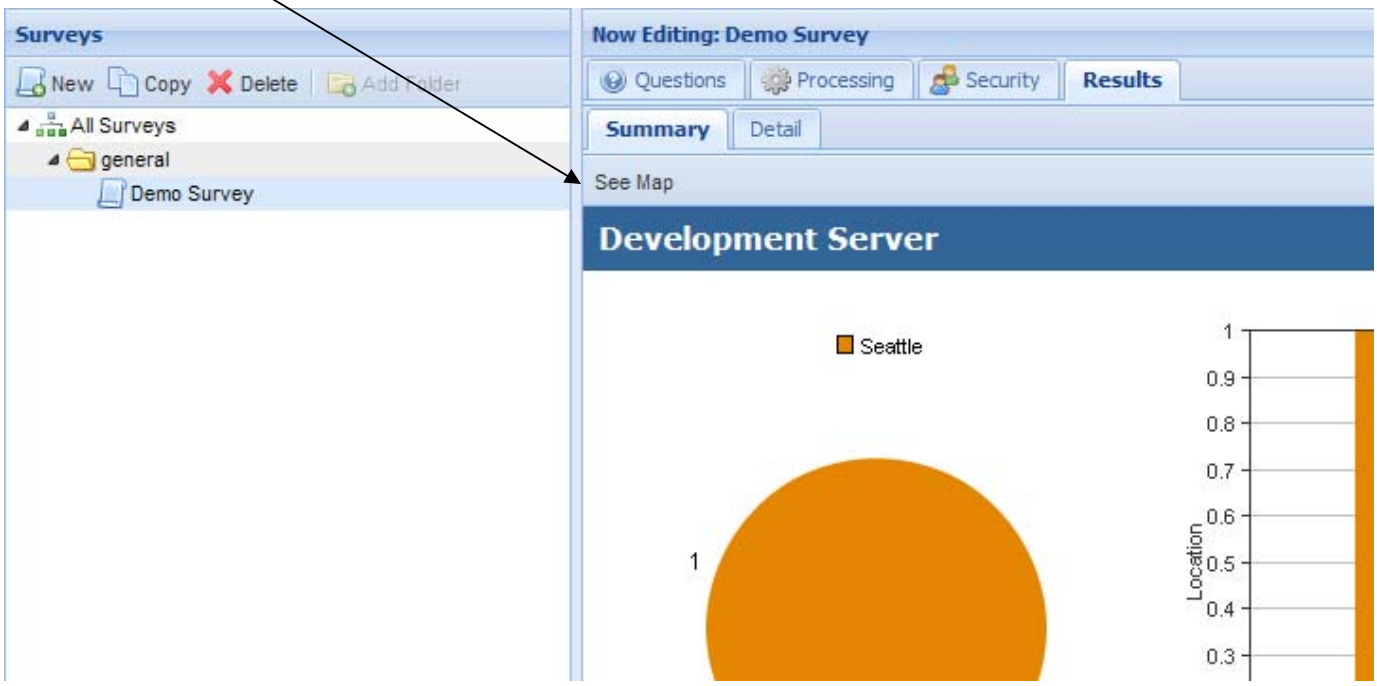
ORM & custom SMTP settings
ORM is used when generating reports and results for surveys. The SMTP settings are stored within Application.cfc and are used to send survey results to a specified email address.

Demo the System

1. Take the demo survey.
The system comes with a default survey. The first time it loads, it will be a little slow. It should be faster on subsequent loads. Please note there is a known issue with performance when using CF9’s JavaScript tags and caching page content. Also note that the survey can be downloaded as an rudimentary Excel file.

Don’t worry too much about the answers. Not all of them will be saved. This was done to avoid some known issues and will be completed after the contest.

2. See the results.
Open the Survey Manager and navigate to the form you just completed. Double click on it to open it. The results tab. The summary page shows some basic charts while the detail page shows answers to each survey. Notice the “See Map” button. Click it to see a Google Map of where surveys were taken.



Known Issues

1. Encoding Issue – Survey Editor

Don't use special characters in any of the survey questions/answers. Only use alphanumeric values to avoid an AJAX encoding issue related to character escaping that will be resolved when this application is released.

Incorrect: I'm a CF Developer
Correct: I am a CF Developer

2. Permissions – Survey Editor

Deleting/updating the demo survey may cause a JavaScript error depending on the file permissions. This system was developed on a remote server where CF runs under its own system account. However; the files are copied to the server under an administrative account. If your environment is setup similarly and the application is installed similarly, ColdFusion may be denied write access to the survey file created with an administrative account.

3. Performance – Taking a Survey

The HTML of the displayed surveys are assigned to a variable name via a <cfsavecontent/> block, which is cached using EhCache. However; the surveys utilize <cfmessagebox> & some use <cfslider>. These tags generate JavaScript that's inserted into the head *after* all CF tags are processed. Therefore, the useful JavaScript created by them is not saved within the <cfsavecontent/> block, and is therefore not cached. To workaround this issue for the purposes of this demonstration, the JS code was copied and inserted into the document using a <cfhtmlhead> tag. If a "rate" question exists in the survey, the slider will show up twice the first time the survey is viewed.

In a production environment, it would be easier to simply not use these tags or use an alternative. However; for the purposes of this demo, the <cfmessagebox> tags are necessary to demonstrate integration/extension of ExtJS.