Cut to the Chase Series
More Walk – Less Talk

zenyan webapp framework

Introduction
What is zenyan?

zenyan webapp framework - Introduction

Table of Contents

A brief Introduction to Zenyan

Welcome to what will be a brief introduction to zenyan in order to keep in concert with my “more walk, less talk” motto. My name is Eric Matthews. I am the chief designer, programmer, and inventor of this framework. I confess here and now that I built this framework first and foremost for myself. I use it for my own web endeavors. I also use it to build commercial applications. Others use it as well. How many I cannot really say.

What I can say is that I am a huge fan and advocate of open source software. I have great respect for it's pioneers, people like Richard Stallman of GNU. I must admit, at first I thought this movement was some form of communism that would doom us all. I remember a guy I really respected telling me, “Eric, when you give something away for free it is worth nothing.” I still believe today that there is a great measure of truth in this statement.

But open source software is not free, it is about freedom. Freedom to choose, to create, to be expressive. Freedom to not always have to follow the pack.

What zenyan is

zenyan, in a nutshell, is a loosely coupled framework that allows me to quickly build web content and web applications. zenyan integrates what I feel are many other cool open source software. zenyan can also integrate and support commercial software as well.

zenyan is based upon and integrates the following browser-side and server-side technologies:

- Php
- Javascript
- JQuery
- CSS
- Html
- Apache
- MySQL
- Oracle
- SQL Server (in reality in can work with any ODBC data source)

The design goals for zenyan are pretty simple and can be summarized as follows

Zenyan Design Goals

- ✓ Do not reinvent the wheel, when you have a wheel that works and does what you need.
- ✓ Do not lock others (and yourself) into a development paradigm that restricts freedom.
- ✓ Functionality and pragmatism must always trump trends and fashion.
- ✓ Spend as little time as possible before going into the work room.

- ✓ Choose technologies and solutions that minimize mindless meetings and the pretense of collaboration.

There are many nice web development frameworks out there. Open source frameworks like Joomla and Drupal are notable examples. I have used them both in the past. But, they always left me wanting and waiting as I felt they forced me into their perspective of application development. I do not mean this as a slam or criticism of them. For me, I just needed more freedom and control over my destiny than they offered.

Most importantly, zenyan is an attempt to wrap up and generalize the process of business application development. There are many generalized technologies and languages that have done this. For example, Php and its inherent context management is a fantastic thing. JQuery is an absolutely browser compliant and productive wrapper (and theology) around Javascript.

But, I still sought something more, something of a higher order in terms of business programming. For instance, it has always bothered me that data access to a dbms only involves two items. 1) Reading data; 2) writing data. And writing data to a database only involve three other functions: 1) adding something new; 2) deleting something; 3) updating something.

Yet every language I have programmed in (C, VB, Cobol, Php, Perl, Java, Blaze) makes data access something that never made much sense to me. Even the invention of what are called “connectors” left me scratching my head. What I longed for were four basic design goals in terms of structured database access. They were:

1. Keeping the technical implementation of data access as simple as it really is at a high level.
2. Being able to decouple the data access code from the host language code.
3. Getting the data back in a format of my choosing though and api to keep me from having to always be responsible for formatting the returned data.
4. Make data access through the host language code totally agnostic to the underlying dbms it is reading and writing to.

This is just one example of the type of application api I have put together in zenyan.

I do not see zenyan as just a web development framework that I have put together using the aforementioned technologies. It certainly is, as I have always been a “roll-up-your-sleaves” and lets get-er done type of person. I also see zenyan as a development philosophy that at its foundational core says, “what business programming functions can we automate in some way, shape, or form, while still maintaining maximum freedom for the developer?” In other words, zenyan is not "take it all and do it my way". I have packaged it up in a way where you can get it all. But, the bottom line is that you choose what you implement and how you wish to write code. Since most is open source you are also free to change or add whatever you feel you need.

Finally, zenyan will always be a work in progress. Since I use it in production instances it is developed with the architectural consideration that if I am going to add something new I must do it in a manner that does NOT break something that already exists. This for me is an important and a foundational principle that is often overlooked. As I like to say, “you get a clean sheet of paper until someone starts using your stuff for the first time!”

What zenyan is not

It is just easier to bullet these items.

- It is not a web development solution that anyone can use to build web applications. In other words, zenyan is an api for experienced programmers based upon the WAMP/LAMP stack and other open source technologies. You need to be a Php programmer to use it.
- It is not a black box that hides the evil details of programming and web development from you. In other words, if you need to get data in and out of a database you still need to understand the database you are working with.
- It is not some form of artificial intelligence. You need to understand the business you are developing your application for.
- It is not a GUI like UML, that claims if you draw a bunch of stick figures and pictures and push a button out pops a working application.
- It is not one way of doing application development. Rather, it employees a wide range of techniques from code generators, static templates, toycode examples, etc... etc... to help you in developing web applications.