

e-Genda - Online Calendar System

by: Nick Peeples and Matt Sellars

19th October 2005

Introduction

The e-Genda Online Calendar System is a new type of online web based calendar. By utilizing many different programming languages and applications, we have come up with a full-featured application which allows you to do many things you would expect from your calendar program, and much more. Our system feels more like a desktop application then something that you would get on the world wide web, supporting draggable frames, quick status updates, and seamless interaction with the webserver. Within our application, it is possible to complete many tasks, but the main focus of the application is organizing reminders and sending them to cell-phones, email, and instant messenger clients.

LAMP and AJAX

Within this application, many different programming languages and server applications are used to bring it all together. With a base Gentoo Linux installation, we use the very common Apache Webserver, MySQL database for the information storing, Python for the daemon language, and PHP for the web scripting language. This is one of the most common setups for building dynamic websites, and is often referred to as the LAMP (Linux Apache MySQL PHP or Perl or Python) system. Recently, a new interface for merging several different languages and creating a very powerful utility called AJAX has become very popular in web development. AJAX stands for Asynchronous JavaScript and XML, which seems like just another acronym for linking together very different things. However, this new method for building web applications is very powerful, and becoming widely used in even your favorite online websites. Currently, Google seems to be the leaders in this field, and AJAX can be seen inside of Gmail, Google Maps, Google Earth, and a few of their other online applications. Microsoft is not far behind either, with their new 'Start' incubation project.

So how does AJAX work?

Currently, a web application uses forms to post information to a webserver, which a scripting language may or may not interpret and send information back to the client. The process would go something like this: client request to server, server parses script if applicable, sends requested information to client, client makes another request to the server. This method is slow and quite cumbersome in the long run, every time a request is made, even if it is to the exact same web page, the entire web page must be sent back to the client. This is where AJAX becomes very useful. Instead of the request from the client going to the web server, it goes to a client side script written in JavaScript. JavaScript then creates an open, asynchronous connection to the server and requests the specific information that the client requested, instead of the whole page. Using XML to send and receive the server request, JavaScript then parses out the information and can either edit the current DOM tree, or sit and wait for the next request from the client. This saves a huge amount of time and bandwidth, because a specific task is requested, instead of an entire script.

Conclusion

Our interest in these applications and their uses has become greater the more we did for the project. This new method for developing dynamic web applications with database backends will be a growing market for quite a while. We believe that this could have the potential to some day replace many of the office applications that are currently used exclusively on the desktop, and launch a whole new way that we currently use the Internet.