

Google Data APIs Protocol (GData Module)

Michael E. Cotterell
University of Georgia, CSCI Major
mepcotterell@gmail.com

Abstract

This proposal aims to create a module that implements the GData protocol specification in Drupal. The Google data APIs provide a simple standard protocol, called "GData", for reading and writing data on the web using either of two standard XML-based syndication formats: Atom or RSS. This module will handle all the basic stuff a developer shouldn't have to worry about when developing modules that extend its functionality such as data transfer, protocol adherence, and authentication. It will expose its own API which will allow other developers to create modules that easily interact with information provided by Google's many service APIs. Using this module, module developers who need access to this information need only concern themselves with what's important: the data.

Detailed Description

DRUPAL.ORG USERNAME:

mepcotterell

LINK TO PROPOSAL DISCUSSION:

<http://groups.drupal.org/node/10142>

BENEFITS TO DRUPAL/OPEN SOURCE COMMUNITY:

This module will benefit Drupal by providing developers with an easy way to interact with information provided by Google's many service APIs without worrying about all the behind-the-scenes stuff that goes into accessing and modifying that information. This opens up a world of possibilities for integrating Google applications into Drupal. Some ideas that could be more easily implemented using the API provided by a GData module include, but are not limited to, the following:

- Using Google's Apps API to integrate various Google applications into Drupal, allowing it to be a competitor with other online groupware suites such as Microsoft SharePoint.
- Integrating Google Base Data into dynamic Drupal applications using Google's Base Data API.
- Creating a Drupal front-end to Google's Blogger service through its Blogger Data API, allowing for the administration of one or more Blogger sites through a Drupal interface.
- Allow users with a Google account to add their contacts to Drupal through Google's Contacts API, enabling, beyond the simple importation of such information, the ability for social networks built on Drupal to connect users using their Contacts data.

- Easily share calendars with users or between sites using Google's Calendar API.
- Enable users to search through Google Code through a Drupal interface with Google's Code Search API.
- Allow users to share documents and notebooks using Google's Document List, Notebook Data, and Spreadsheet Data APIs.
- Allow users to import and export their Drupal settings to their Google accounts via database abstraction of Google Spreadsheets through their Spreadsheets Data APIs.
- Create a community image hosting application in Drupal leveraging the power of Google's Picasa Web Albums Data API.
- Create a community video hosting application in Drupal leveraging the power of Google and YouTube using Google's YouTube API.

Many people will benefit from this module. Developers who need to harness the powers of GData can do so with relative ease. Users who use contributed modules that make use of the API exposed by the GData will benefit by knowing that very interesting things are being done with their Google services while still maintaining a high level of security and without exposing any personal information except that information that they approve.

The likely user-base for this module is Drupal developers. Currently, there is not single way to access Google APIs in Drupal and so developers are coming up with varying implementations of accessing GData on a module-by-module basis. This module is an improvement by providing a single, maintainable, and easy-to-use interface for Drupal developers to get what they need out of Google services.

PROJECT DETAILS:

Develop a module using the Zend Google Data library to implement a REST client in Drupal. This module will adhere to Google's protocol reference. Authentication will support both the ClientLogin method, which requires a username and password, and the newer AuthSub method, which allows a web-based application to access a Google service on behalf of a user using an authentication token instead of handling the user's account login information.

- The Zend Google Data Client [<http://framework.zend.com/download/gdata/>]
- Google Data APIs Protocol Reference [<http://code.google.com/apis/gdata/reference.html>]

DELIVERABLES:

- Drupal Module: GData
 - includes a REST client implementation for interacting with the Google Data APIs
 - exposes a Drupal-like API enabling module developers to create code specific to a particular Google Data API
 - includes inline documentation
- Drupal Module: GDataContacts
 - depends on the GData Module
 - is an example of how the GData module makes interacting with the Google Data APIs easy in Drupal

PROJECT SCHEDULE:

This is my proposed schedule. It's not final, and more than likely will need to be tweaked.

- May 26 - 31
 - Begin preliminary development of the GData module
 - Outline the specification for the Drupal-like API that the module provides.
 - Keeping in mind that it needs to:
 - provide all the functionality provided by the Zend Google Data Client.
 - be syntactically consistent with other Drupal APIs.
 - be relatively easy to extend and use in dependent modules.
 - Try to determine potential snags and whether:
 - they can be solved with implementations found in readily available GPL code. (we love Google Code Search!)
 - they need to be implemented from scratch.
 - Begin coding the GData module based on the preliminary research.
- June 1 - 7
 - Outline and fix bugs
- June 8 - 14
 - Milestone 1 - GData Protocol Implemented
 - At this point in time, the GData module should be able to privately interact with the Google Data APIs
 - Outline and fix bugs
- June 15 - 21
 - Outline and fix bugs
- June 22 - 28
 - Outline and fix bugs
- June 29 - July 2
 - Milestone 2 - GData Module API Implemented
 - At this point in time, the GData module should be able to provide the Drupal-like API spec'd in the preliminary research.
 - Outline and fix bugs
- July 3 - 6
 - Independence Day break (...right)
- July 7 - 14
 - GData Module "Semi-Code Freeze" begins.
 - Only vital changes are made
 - Submit mid-term evaluations.
 - Begin preliminary development of the GDataContacts module.
 - Goals:
 - Import contacts from Google into Drupal
 - Export contacts from Drupal into Google
 - Determine an interesting goal, which may or may not include:
 - Letting users see if any of their contacts already have accounts on that Drupal site
 - Letting users send invitations to their contacts that don't have an account on that Drupal site
 - Note possible changes that need to be made to the GData module
 - Try to determine potential snags and whether:

- they can be solved with implementations found in readily available GPL code. (we love Google Code Search!)
 - they need to be implemented from scratch.
- Outline bugs derived from GData module dependency.
- Begin coding the GDataContacts module based on the preliminary research.
- July 15 - 19
 - Outline and fix bugs
- July 20 - 26
 - Outline and fix bugs
- July 27 - August 1
 - Outline and fix bugs
- August 2 - 5
 - Outline and fix bugs
- August 6 - 10
 - Outline and fix bugs
- August 11 - 17
 - GDataContacts Module "Semi-Code Freeze" begins
 - only vital changes are made
 - Test code on multiple Drupal platforms
 - address platform specific bugs
 - address dependency problems
- August 18
 - Code Completion
 - Begin submitting final evaluations.

BIO:

My name is Michael Cotterell, I am twenty years old and am currently an undergraduate Computer Science student at the University of Georgia. I am currently seeking a B.S. in my major. Besides my major classes, I'm also taking Latin, which I have found has helped me approach programming problems from a different and often better perspective. I have been working with PHP for the last five years and feel comfortable implementing ideas with the language. My experience with Drupal module development only goes back a year or so. However, in that time, I have come to believe that Drupal is an excellent code-base, not only for content management systems, but as a platform for web-applications. I do the bulk of my development with the aid of the Eclipse platform running on an Ubuntu Linux machine. My C++ experience has lead me to outline my programming ideas first and work on them incrementally. I try to separate my interfaces from my implementation. And, I'm a big fan returning information from a function and not passing it by a reference variable. Although, I've never worked on a project this big before, I don't believe the task is daunting. I believe I can do it.