Zend_Gdata - Bill Karwin

<ac:macro ac:name="unmigrated-inline-wiki-markup"><ac:plain-text-body><![CDATA[

<ac:macro ac:name="unmigrated-inline-wiki-markup"><ac:plain-text-body><![CDATA[

Zend Framework: Zend_Gdata Component Proposal

Proposed Component Name	Zend_Gdata
Developer Notes	http://framework.zend.com/wiki/display/ZFDEV/Zend_Gdata
Proposers	Bill Karwin (lead) John Herren (initial implementation)
Revision	1.1 - 20 August 2006: Initial proposal. (wiki revision: 11)

Table of Contents

1. Overview
2. References
3. Component Requirements, Constraints, and Acceptance Criteria
4. Dependencies on Other Framework Components
5. Theory of Operation
6. Milestones / Tasks
7. Class Index
8. Use Cases
9. Class Skeletons

1. Overview

Zend_Gdata is a set of classes to implement Google's GData protocol. It includes an interface for authentication as well as concrete interfaces to Google's Gdata enabled services such as Calendar, Google Base, Blogger, and Code Search

2. References

- Google Data APIs (Beta) Developer's Guide
- Google Account Authentication

3. Component Requirements, Constraints, and Acceptance Criteria

- This component **will** follow Google's Data API Protocol.
- This component **will** provide an authentication method per Google's AuthSub authentication scheme.
- This component **will** provide an authentication method per Google's ClientLogin authentication scheme.
- This component **will** throw exceptions when an authentication error is received.
- The generic Gdata class **will** allow subclassing.

4. Dependencies on Other Framework Components

- Zend_Exception
- Zend_Feed (and subclasses)
- Zend_Http_Client

5. Theory of Operation

The Gdata module leverages several existing framework modules to facilitate HTTP request/response and feed parsing. The Gdata module serves mainly to provide convenience methods to create queries against the Google webservices. Because Google requires a somewhat complicated authentication scheme, Gdata implements the simplest interface possible to provide authentication to the webservices.

6. Milestones / Tasks

- Milestone 1: Working prototype checked into the incubator supporting use cases #1, #2, #3, #4
- Milestone 2: Better API for return types and namespaced request fields.
- Milestone 3: Unit tests exist, work, provide at least 80% code coverage, and are checked into SVN.
- Milestone 4: Initial API documentation exists.
- Milestone 5: Support for additional Google services: Blogger, Code Search
- Milestone 6: Initial manual documentation exists.

7. Class Index

- Zend_Gdata
- Zend_Gdata_Exception
- Zend_Gdata_AuthSub
- Zend_Gdata_ClientLogin
- Zend_Gdata_Base
- Zend_Gdata_Blogger
- Zend_Gdata_Calendar
- Zend_Gdata_CodeSearch

8. Use Cases

UC-01: manual Google Calendar query/write using Gdata class

```
// Private calendar location
$uri = 'http://www.google.com/calendar/feeds/EMAIL/private/basic';

// Google Calendar URI to post new entries
$postUri ='http://www.google.com/calendar/feeds/default/private/full';

//Retrieve auth token from Google
$client = new Zend_Gdata_ClientLogin(EMAIL, PASS, 'cl');
$auth = $client->getAuth();

//Create gData client
$gdata = new Zend_Gdata($auth);

// We want results in order of occurance, not by most recently updated
$uri .= "?orderby=starttime";

// Retrieve list of calendar entries
$feed = $gdata->getFeed($uri);
foreach ($feed as $entry) {
 echo $entry->title()."<br>";
 echo $entry->content()."<br>";
}

// Search for party events!
$queryuri = $uri . '&q=party';
$feed = $gdata->getFeed($queryuri);
foreach ($feed as $entry) {
 echo $entry->title()."<br>";
 echo $entry->content()."<br>";
}

// Create a new calendar entry ...
$newentry= new Zend_Feed_EntryAtom();
$newentry->title = "Developer Meeting";
Zend_Feed::registerNamespace('gd', 'http://schemas.google.com/g/2005');
$newentry->{'gd:when'}['startTime'] = '2006-10-21';
$newentry->{'gd:where'}['valueString'] = 'The Office';

// ...and save it to Google Calendar ...
if ($gdata->post($newentry->saveXML(), $postUri)->isSuccessful())
 echo "<p><strong>New event added successfully</p></strong>";

// ...and search for it...
$queryuri = $uri . '&q=meeting';
$feed = $gdata->getFeed($queryuri);
foreach ($feed as $entry) {
 echo $entry->title()."<br>";
 echo $entry->content()."<br>";
}
```

UC-02: manual Google Base query using Gdata class

```
// Google Base location
$uri = 'http://www.google.com/base/feeds/snippets';

// Google Calendar URI to post new entries
$postUri ='http://www.google.com/calendar/feeds/default/private/full';

// Developer Key
$key = '...';

//Retrieve auth token from Google
$client = new Zend_Gdata_ClientLogin(EMAIL, PASS);
$authToken = $client->getAuth();

//Create gData client
$gdata = new Zend_Gdata($authToken,$key);

// find 5 digital cameras
$queryuri = $uri . "?bq=digital+camera&max-results=5";

$feed = $gdata->getFeed($queryuri);

foreach ($feed as $entry) {
 echo $entry->title()."<br>";
}

// find 5 digital cameras of Nikon brand
$queryuri = $uri . "?bq=digital+camera+%5Bbrand%3Acanon%5D&max-results=3";

$feed = $gdata->getFeed($queryuri);

foreach ($feed as $entry) {
 echo $entry->title()."<br>";
}
```

UC-03: Google Calendar query/write using Gdata_Calendar class

```
$gcal = new Zend_Gdata_Calendar(EMAIL, PASS);

$gcal->query = 'meeting';
$gcal->startMax = time();

$feed = $gcal->getFeed();

foreach ($feed as $entry) {
 echo $entry->title()."<br>";
 echo $entry->content()."<br>";
 echo "<hr>";
}

// Create a new calendar entry ...
$newentry= new Zend_Feed_EntryAtom();
$newentry->title = "Developer Meeting";
Zend_Feed::registerNamespace('gd', 'http://schemas.google.com/g/2005');
$newentry->{'gd:when'}['startTime'] = '2006-10-21';
$newentry->{'gd:where'}['valueString'] = 'The Office';

// ...and save it to Google Calendar
if ($gcal->post($newentry->saveXML())->isSuccessful())
 echo "<p><strong>New event added successfully</p></strong>";
```

UC-04: Google Base query using Gdata_Base class

```
// Developer Key
$key = '...';

$gcal = new Zend_Gdata_Base(EMAIL, PASS, $key);

$gcal->query = '[item type:products] (ipod | "mp3 player") [price <= 150.0 USD]';
$gcal->maxResults = 5;

$feed = $gcal->getFeed();

foreach ($feed as $entry) {
 echo $entry->title()."<br>";
 echo $entry->content()."<br>";
 echo "<hr>";
}
```

9. Class Skeletons

```
<?php

/**
 * Zend Framework
 *
 * LICENSE
 *
 * This source file is subject to the new BSD license that is bundled
 * with this package in the file LICENSE.txt.
 * It is also available through the world-wide-web at this URL:
 * http://framework.zend.com/license/new-bsd
 * If you did not receive a copy of the license and are unable to
 * obtain it through the world-wide-web, please send an email
 * to license@zend.com so we can send you a copy immediately.
 *
 * @category   Zend
 * @package    Zend_Gdata
 * @copyright  Copyright (c) 2006 Zend Technologies USA Inc. (http://www.zend.com)
 * @license    http://framework.zend.com/license/new-bsd     New BSD License
 */

/**
 * Zend_Feed
 */
require_once 'Zend/Feed.php';

/**
 * Zend_Gdata_Exception
 */
require_once 'Zend/Gdata/Exception.php';


/**
 * @category   Zend
 * @package    Zend_Gdata
 * @copyright  Copyright (c) 2006 Zend Technologies USA Inc. (http://www.zend.com)
 * @license    http://framework.zend.com/license/new-bsd     New BSD License
 */
class Zend_Gdata {

    /**
  * Authentication token
```

```
 *
 * @var string
 */
protected $token;

/**
 * Optional developer key
 *
 * @var string
 */
protected $key;

/**
    * Client object used to communicate
    *
    * @var Zend_Http_Client
    */
   protected $client ;

   /**
    * Query parameters.
    *
    * @var array
    */
   protected $params = array();


   /**
    * Create Gdata object
    *
    * @param string $token authorization token
    * @param string $key optional developer key
    */
   public function __construct($token,$key = null) {
       $this->client = new Zend_Http_Client();
       $this->setToken($token);
    if (! isset($this->token))
     throw new Zend_Gdata_Exception("No auth token set.");

    if (isset($key)){
     $this->setKey($key);
    }
   }

   /**
    * Sets authentication token
    *
    * @param string $token
    */
   public function setToken($token){
    $this->token = (string) $token;
    $headers['authorization'] = 'GoogleLogin auth=' . $this->token;
    $this->client->setHeaders($headers);
   }

   /**
    * Sets developer key
    *
    * @param string $key
    */
   public function setKey($token){
    $this->key = (string) $token;
    $headers['X-Google-Key'] = 'key=' . $this->key;
    $this->client->setHeaders($headers);
   }

   protected function getQueryString(){
    if (!count($this->params))
     return '';
```

```
 $querystring = '?';
 foreach ($this->params as $name => $value) {
        $querystring .= '&' . urlencode($name) . '=' . urlencode($value);
    }
    return $querystring;
}

public function resetParams(){
 $this->params = array();
}

/**
 * Retreive feed object
 *
 * @param string $uri
 * @return Zend_Feed
 */
public function getFeed($uri){
 $feed = new Zend_Feed();
 $this->client->resetParameters();
 $feed->setHttpClient($this->client);
 return $feed->import($uri);
}

/**
 * POST xml data to Google with authorization headers set
 *
 * @param string $xml
 * @param string $uri POST URI
 * @return Zend_Http_Response
 */
public function post($xml,$uri){
 if (isset($uri))
  $this->client->setUri($uri);
  $this->client->setMaxRedirects(0);
 $this->client->setRawData($xml,'application/atom+xml');
 $response = $this->client->request('POST');
 if ($response->isRedirect()){
  //this usually happens. Re-POST with redirected URI.
  $this->client->setUri($response->getHeader('Location'));
  $this->client->setRawData($xml,'application/atom+xml');
  $response = $this->client->request('POST');
 }
 if (!$response->isSuccessful()) {
throw new Zend_Gdata_Exception("Post to Google failed.");
 }
 return $response;
}

public function __set($var, $value) {
 $this->params[$var] = $value;
```

```
    }

}
```

```
<?php
/**
 * Zend Framework
 *
 * LICENSE
 *
 * This source file is subject to the new BSD license that is bundled
 * with this package in the file LICENSE.txt.
 * It is also available through the world-wide-web at this URL:
 * http://framework.zend.com/license/new-bsd
 * If you did not receive a copy of the license and are unable to
 * obtain it through the world-wide-web, please send an email
 * to license@zend.com so we can send you a copy immediately.
 *
 * @category   Zend
 * @package    Zend_Gdata
 * @copyright  Copyright (c) 2006 Zend Technologies USA Inc. (http://www.zend.com)
 * @license    http://framework.zend.com/license/new-bsd     New BSD License
 */


/**
 * Zend_Exception
 */
require_once 'Zend/Exception.php';


/**
 * Gdata exceptions
 *
 * Class to represent exceptions that occur during Gdata operations.
 *
 * @category   Zend
 * @package    Zend_Gdata
 * @copyright  Copyright (c) 2006 Zend Technologies USA Inc. (http://www.zend.com)
 * @license    http://framework.zend.com/license/new-bsd     New BSD License
 */
class Zend_GData_Exception extends Zend_Exception
{}
```

```
<?php

/**
 * Zend Framework
 *
 * LICENSE
 *
 * This source file is subject to the new BSD license that is bundled
 * with this package in the file LICENSE.txt.
 * It is also available through the world-wide-web at this URL:
 * http://framework.zend.com/license/new-bsd
 * If you did not receive a copy of the license and are unable to
 * obtain it through the world-wide-web, please send an email
 * to license@zend.com so we can send you a copy immediately.
 *
 * @category   Zend
 * @package    Zend_Gdata
 * @copyright  Copyright (c) 2006 Zend Technologies USA Inc. (http://www.zend.com)
 * @license    http://framework.zend.com/license/new-bsd     New BSD License
```

```
 */

/**
 * Zend
 */
require_once 'Zend.php';

/**
 * Zend_Http_Exception
 */
require_once 'Zend/Http/Exception.php';

/**
 * Zend_Http_Client
 */
require_once 'Zend/Http/Client.php';

/**
 * Wrapper around Zend_Http_Client to facilitate Google's "Account Authentication
 * for Installed Applications".
 * @see http://code.google.com/apis/accounts/AuthForInstalledApps.html
 *
 * @category   Zend
 * @package    Zend_Gdata
 * @copyright  Copyright (c) 2006 Zend Technologies USA Inc. (http://www.zend.com)
 * @license    http://framework.zend.com/license/new-bsd     New BSD License
 */
class Zend_Gdata_ClientLogin extends Zend_Http_Client
{
 /**
  * The Google client login URI
  *
  */
 const CLIENTLOGIN_URI = 'https://www.google.com/accounts/ClientLogin';

 /**
  * The default 'source' parameter to send to Google
  *
  */
 const DEFAULT_SOURCE = 'Zend-ZendFramework-0.1.5';

 /**
  * Authentication token array
  *
  * @var array
  */
 protected $token = array();

 /**
  * User's e-mail (username)
  *
  * @var string
  */
 protected $email;

 /**
  * User's password
  *
  * @var string
  */
 protected $password;

 /**
  * Google service to authenticate to (eg. 'cl' for Calendar)
  *
  * @var string
  */
 protected $service;

 /**
```

```
 * Source string, ie. companyName-applicationName-versionID
 *
 * @var string
 */
protected $source;

/**
 * Set Google authentication data. Must be done before accessing any Google
 * service
 *
 * @param string $email
 * @param string $password
 * @param string $service
 * @param string $source
 */
public function __construct($email, $password, $service = 'xapi',
 $source = self::DEFAULT_SOURCE)
{
 $this->email = (string) $email;
 $this->password = (string) $password;
 $this->service = (string) $service;
 $this->source = (string) $source;

 if (! ($this->email && $this->password))
  throw new Zend_Http_Exception('Please set your Google credentials ' .
   'before trying to authenticate');

 // Build the HTTP client for authentication
 $client = new Zend_Http_Client(self::CLIENTLOGIN_URI);
 $client->setMaxRedirects(0);
 $client->setParameterPost('Email', $this->email);
 $client->setParameterPost('Passwd', $this->password);
 $client->setParameterPost('service', $this->service);
 $client->setParameterPost('source', $this->source);

 // Send the authentication request
 // For some reason Google's server causes an SSL error. We use the
 // output buffer to supress an error from being shown. Ugly - but works!
 ob_start();
 $response = $client->request('POST');
 ob_end_clean();

 // Parse Google's response
 $goog_resp = array();
 foreach (explode("\n", $response->getBody()) as $l) {
  $l = chop($l);
  if ($l) {
   list($key, $val) = explode('=', chop($l), 2);
   $goog_resp[$key] = $val;
  }
 }

 if ($response->getStatus() == 200) {
  $this->token = $goog_resp;
 } elseif ($response->getStatus() == 403) {
  throw new Zend_Http_Exception("Authentication with Google failed. Reason: " .
   (isset($goog_resp['Error']) ? $goog_resp['Error'] : 'Unspecified.'));
 }
}


   /**
    * Send the HTTP request authenticating to Google and return a response
 *
    * @return string
    */
   public function getAuth(){
    if (! isset($this->token['Auth'])) $this->authenticate();
    if (! isset($this->token['Auth']))
     throw new Zend_Http_Exception("Unable to authenticate with Google for an unknown reason");
```

```
        return $this->token['Auth'];
```

```
    }

}
```

```
<?php

/**
 * Zend Framework
 *
 * LICENSE
 *
 * This source file is subject to the new BSD license that is bundled
 * with this package in the file LICENSE.txt.
 * It is also available through the world-wide-web at this URL:
 * http://framework.zend.com/license/new-bsd
 * If you did not receive a copy of the license and are unable to
 * obtain it through the world-wide-web, please send an email
 * to license@zend.com so we can send you a copy immediately.
 *
 * @category   Zend
 * @package    Zend_Gdata
 * @copyright  Copyright (c) 2006 Zend Technologies USA Inc. (http://www.zend.com)
 * @license    http://framework.zend.com/license/new-bsd     New BSD License
 */

/**
 * Zend_Gdata
 */
require_once 'Zend/Gdata.php';

/**
 * Zend_Gdata_ClientLogin
 */
require_once 'Zend/Gdata/ClientLogin.php';

/**
 * @category   Zend
 * @package    Zend_Gdata
 * @copyright  Copyright (c) 2006 Zend Technologies USA Inc. (http://www.zend.com)
 * @license    http://framework.zend.com/license/new-bsd     New BSD License
 */
class Zend_Gdata_Base extends Zend_Gdata  {

 const BASE_URI ='http://www.google.com/base/feeds/snippets';

    /**
     * Create Gdata_Calendar object
     *
     * @param string $email
  * @param string $password
     *
     */
    public function __construct($email,$password,$key) {
     $client = new Zend_Gdata_ClientLogin($email, $password);
  $this->token = $client->getAuth();
  parent::__construct($this->token,$key);
    }

    /**
     * Retreive feed object
     *
     * @return Zend_Feed
     */
    public function getFeed(){
     return parent::getFeed(Zend_Gdata_Base::BASE_URI . $this->getQueryString());
```

```
}

/**
 * POST xml data to Google with authorization headers set
 *
 * @param string $xml
 * @return Zend_Http_Response
 */
public function post($xml){
 return parent::post($xml,Zend_Gdata_Base::BASE_URI);
}

public function __set($var,$value){
 switch ($var) {
    case 'query':
     $var = 'bq';
     break;
    case 'maxResults':
     $var = 'max-results';
     break;
    case 'startIndex':
     $var = 'start-index';
     break;
 }

    parent::__set($var,$value);
```

```
    }

}
```

```
<?php

/**
 * Zend Framework
 *
 * LICENSE
 *
 * This source file is subject to the new BSD license that is bundled
 * with this package in the file LICENSE.txt.
 * It is also available through the world-wide-web at this URL:
 * http://framework.zend.com/license/new-bsd
 * If you did not receive a copy of the license and are unable to
 * obtain it through the world-wide-web, please send an email
 * to license@zend.com so we can send you a copy immediately.
 *
 * @category   Zend
 * @package    Zend_Gdata
 * @copyright  Copyright (c) 2006 Zend Technologies USA Inc. (http://www.zend.com)
 * @license    http://framework.zend.com/license/new-bsd     New BSD License
 */

/**
 * Zend_Gdata
 */
require_once 'Zend/Gdata.php';

/**
 * Zend_Gdata_ClientLogin
 */
require_once 'Zend/Gdata/ClientLogin.php';

/**
 * @category   Zend
 * @package    Zend_Gdata
 * @copyright  Copyright (c) 2006 Zend Technologies USA Inc. (http://www.zend.com)
 * @license    http://framework.zend.com/license/new-bsd     New BSD License
 */
class Zend_Gdata_Calendar extends Zend_Gdata  {

 const POST_URI = 'http://www.google.com/calendar/feeds/default/private/full';

 /**
  * Email address used for uri creation
  *
  * @var string
  */
 protected $email;

    /**
     * Create Gdata_Calendar object
     *
     * @param string $email
  * @param string $password
     *
     */
    public function __construct($email,$password) {
     $client = new Zend_Gdata_ClientLogin($email, $password,'cl');
  $this->token = $client->getAuth();
  $this->email = (string) $email;
  parent::__construct($this->token);
    }
```

```
 /**
  * Retreive feed object
  *
  * @return Zend_Feed
  */
 public function getFeed(){
  $uri = 'http://www.google.com/calendar/feeds/' . $this->email . '/private/basic';
  return parent::getFeed($uri . $this->getQueryString());
 }

 /**
  * POST xml data to Google with authorization headers set
  *
  * @param string $xml
  * @return Zend_Http_Response
  */
 public function post($xml){
  return parent::post($xml,Zend_Gdata_Calendar::POST_URI);
 }

 public function __set($var,$value){
  switch ($var) {
     case 'query':
      $var = 'q';
      break;
     case 'startMin':
      $var = 'start-min';
      $value = $this->formatTimestamp($value);
      break;
     case 'startMax':
      $var = 'start-max';
      $value = $this->formatTimestamp($value);
      break;
     case 'updatedMin':
      $var = 'updated-min';
      $value = $this->formatTimestamp($value);
      break;
     case 'updatedMax':
      $var = 'updated-max';
      $value = $this->formatTimestamp($value);
      break;
  }

     parent::__set($var,$value);

 }
 /**
  *  Convert timestamp into RFC 3339 date string.
  *  2005-04-19T15:30:00
  *
  * @param int $timestamp
  */
 private function formatTimestamp($timestamp){
  if (ctype_digit($timestamp)) {
return date('Y-m-d\TH:i:s', $timestamp);
  }
```

```
        }

    }
```

]]></ac:plain-text-body></ac:macro>
]]></ac:plain-text-body></ac:macro>