Zend_Service_Technorati - Simone Carletti

<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_Service_Technorati Component Proposal

Proposed Component Name	Zend_Service_Technorati
Developer Notes	http://framework.zend.com/wiki/display/ZFDEV/Zend_Service_Technorati
Proposers	Simone Carletti Darby Felton, Zend liaison
Revision	1.0 - 4 February 2007: Proposal draft. 2.0 - 13 November 2007: Proposal approved, updated details. (wiki revision: 12)

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_Service_Technorati provides an easy, intuitive and object oriented interface for using Technorati APIs.

Each API call has a 1:1 method in the Zend_Service_Technorati class.
Convenient methods may be created in order to make developer's life easier.

2. References

- Technorati API
- Technorati API (old wiki pages)

3. Component Requirements, Constraints, and Acceptance Criteria

- This component **will** implement all available Technorati API calls with 1:1 naming convention.
- This component **will** implement not-documented API calls from useful API services, for example KeyInfo.
- This component **will** implement convenient calls to access complex data or properties.
- This component **will** parse and return server responses as intuitive data objects.

4. Dependencies on Other Framework Components

- Zend_Service_Exception
- Zend_Rest_Client
- Zend_Uri
- Zend_Date

5. Theory of Operation

An instance of Zend_Service_Technorati object is created, with an unique Technorati API Key.
After that, you can use this instance to query Technorati search index with one of available Zend_Service_Technorati API methods.

Depending on query type, developer **MUST** provide mandatory parameters required by Technorati.
Optional parameters **MAY** be provided using an associative array.

6. Milestones / Tasks

Milestone 0: [DONE] develop release 0.1 alpha for testing
Milestone 1: [DONE] write component proposal
Milestone 2: [DONE] proposal acceptance
Milestone 3: [DONE] import release 0.1 alpha to incubator/lab svn
Milestone 4: [DONE] class development and improvements
Milestone 5: [DONE] write unit tests and debugging
Milestone 6: [DONE] write documentation

7. Class Index

- Zend_Service_Technorati - core class
- Zend_Service_Technorati_ResultSet - a base result set
- Zend_Service_Technorati_Result - a base result
- Zend_Service_Technorati_CosmosResultSet - a Cosmos query result set
- Zend_Service_Technorati_CosmosResult - a Cosmos query result
- ...
- Zend_Service_Technorati_Author - each weblog has zero or more authors
- Zend_Service_Technorati_Weblog - some queries returns weblog object along with other data
- Zend_Service_Technorati_Key - a Technorati key, used for KeyInfo() query

Note. Due to Technorati API responses, each query should have a specific result set and result class.

8. Use Cases

UC-01

Create a Zend_Service_Technorati instance

```
include 'Zend/Service/Technorati.php';

$t = new Zend_Service_Technorati('YOUR_API_KEY');
```

UC-02

Retrieve key info

```
include 'Zend/Service/Technorati.php';

$t = new Zend_Service_Technorati('YOUR_API_KEY');
$myKey = $t->keyInfo();

print(  'You used ' . $myKey->apiQueries .
        ' of your ' . $myKey->maxQueries . ' queries per day.'
        );
```

UC-03

Retrieve Cosmos with default options, URL www.simonecarletti.com/blog

```
include 'Zend/Service/Technorati.php';

$t = new Zend_Service_Technorati('YOUR_API_KEY');
$results = $t->cosmos('www.simonecarletti.com/blog');

print('Cosmos query returned ' . $results->totalResultsReturned . ' results.');
foreach ($results as $result) {
    // print_r($result);
    print("<h1>{$result->weblog->name}</h1>");
    print("<p>{$result->excerpt}</p>");
}
```

UC-04

Retrieve Cosmos with custom options, URL www.simonecarletti.com/blog

```
include 'Zend/Service/Technorati.php';

$t = new Zend_Service_Technorati('YOUR_API_KEY');
$results = $t->cosmos('www.simonecarletti.com/blog',
                      array('limit' => 3, 'type' => 'link', 'claim' => '1'));

print('Cosmos query returned ' . $results->totalResultsReturned . ' results.');
foreach ($results as $result) {
    print_r($result);
    print("<h1>{$result->weblog->name}</h1>");
    print("<p>{$result->excerpt}</p>");
}
```

UC-05

Retrieve Cosmos XML response (useful for debugging purpose) with custom options

```
include 'Zend/Service/Technorati.php';

$t = new Zend_Service_Technorati('YOUR_API_KEY');
$results = $t->cosmos('www.simonecarletti.com/blog',
                      array('limit' => 3, 'type' => 'link', 'claim' => '1'));

print('Cosmos query returned ' . $results->totalResultsReturned . ' results.');

print('<h1>XML response</h1>');
print('<pre>');
print_r(htmlentities($results->getXML()));
print('</pre>')
```

9. Class Skeletons

Core Object

```
/** Zend_Service_Exception */
require_once 'Zend/Service/Exception.php';

/** Zend_Service_Technorati_Result */
require_once 'Zend/Service/Technorati/Result.php';

/** Zend_Service_Technorati_ResultSet */
require_once 'Zend/Service/Technorati/ResultSet.php';

/** Zend_Service_Technorati_CosmosResult */
require_once 'Zend/Service/Technorati/CosmosResult.php';

/** Zend_Service_Technorati_CosmosResultSet */
require_once 'Zend/Service/Technorati/CosmosResultSet.php';

/** Zend_Service_Technorati_Author */
require_once 'Zend/Service/Technorati/Author.php';

/** Zend_Service_Technorati_Key */
require_once 'Zend/Service/Technorati/Key.php';

/** Zend_Service_Technorati_Weblog */
require_once 'Zend/Service/Technorati/Weblog.php';


class Zend_Service_Technorati
{
    /** Base Technorati API Uri for Requests */
    const URI_BASE = 'http://api.technorati.com';

    /**
     * Your Technorati API key
     *
     * @var     string
     * @access  protected
     */
    protected $_apiKey;

    /**
     * Zend_Rest_Client instance
     *
     * @var     Zend_Rest_Client
     * @access  protected
     */
    protected $_restClient;


    /**
     * Zend_Service_Technorati Constructor, setup character encoding
     *
     * @param   string $apiKey  Your Technorati API key
     */
    public function __construct($apiKey)
    {}


    /**
     * Cosmos Query
     *
     * Cosmos lets you see what blogs are linking to a given URL.
     * On the Technorati site, you can enter a URL in the searchbox and
     * it will return a list of blogs linking to it.
     * The API version allows more features and gives you a way
```

```
     * to use the cosmos on your own site.
     *
     * Query options include:
     *
     * 'type'       => (link|weblog)
     *      optional - A value of link returns the freshest links referencing your target URL.
     *      A value of weblog returns the last set of unique weblogs referencing your target URL.
     * 'limit'      => (int)
     *      optional - adjust the size of your result from the default value of 20
     *      to between 1 and 100 results.
     * 'start'      => (int)
     *      optional - adjust the range of your result set.
     *      Set this number to larger than zero and you will receive
     *      the portion of Technorati's total result set ranging from start to start+limit.
     *      The default start value is 1.
     * 'current'    => (yes|no)
     *      optional - the default setting of true
     *      Technorati returns links that are currently on a weblog's homepage.
     *      Set this parameter to false if you would like to receive all links
     *      to the given URL regardless of their current placement on the source blog.
     * 'claim'      => (int)
     *      optional - the default setting of 0 returns no user information
     *      about each weblog included in the result set when available.
     *      Set this parameter to 1 to include Technorati member data
     *      in the result set when a weblog in your result set
     *      has been successfully claimed by a member of Technorati.
     * 'highlight'  => (int)
     *      optional - the default setting of 1
     *      highlights the citation of the given URL within the weblog excerpt.
     *      Set this parameter to 0 to apply no special markup to the blog excerpt.
     *
     * @param   string  $url      Target URL. Prefixes http:// and www. are optional.
     * @param   array   $options  Additional parameters to refine your query
     * @return  Zend_Service_Technorati_CosmosResultSet Cosmos resultset
     * @link    http://technorati.com/developers/api/cosmos.html Technorati API: Cosmos Query
reference
     */
    public function cosmos($url, $options = null)
    {}

    /**
     * Search Query
     *
     * @todo    docblock
     * @todo    develop function
     *
     * @param   string  $query      Set this to the words you are searching for.
     *                              Separate words with '+' as usual.
     * @param   array   $options    Additional parameters to refine your query
     * @return  Zend_Service_Technorati_SearchResultSet Search resultset
     * @link    http://technorati.com/developers/api/search.html Technorati API: Search Query
reference
     */
    public function search($query, $options = null)
    {}

    /**
     * Tag Query
     *
     * @todo    docblock
     * @todo    develop function
     *
     * @param   string  $tag        Set this to the tag term you are searching posts for.
     * @param   array   $options    Additional parameters to refine your query
     * @return  Zend_Service_Technorati_TagResultSet Tag resultset
     * @link    http://technorati.com/developers/api/tag.html Technorati API: Tag Query reference
     */
    public function tag($tag, $options = null)
    {}
```

```
    /**
     * KeyInfo Query
     *
     * @todo docblock
     */
    public function keyInfo()
    {}


    /**
     * Return Technorati API key
     *
     * @return string   Technorati API key
     */
    public function getApiKey()
    {}

    /**
     * Set Technorati API key
     *
     * @param   string $value   Technorati API Key
     * @link    http://technorati.com/developers/apikey.html How to get your Technorati API Key
     * @todo    Should this function validate the key?
     */
    public function setApiKey($value)
    {}


    /**
     * Validate Cosmos Options
     *
     * @param   array   $options
     * @throws  Zend_Service_Exception
     */
    protected function _validateCosmos($options)
    {}

    /**
     * Validate Search Options
     *
     * @param   array   $options
     * @throws  Zend_Service_Exception
     * @todo    develop function
     */
    protected function _validateSearch($options)
    {}

    /**
     * Validate Tag Options
     *
     * @param   array   $options
     * @throws  Zend_Service_Exception
     * @todo    develop function
     */
    protected function _validateTag($options)
    {}

    /**
     * Validate that an option is in a given array
     *
     * @param   string $name    Option name
     * @param   string $value   Option value
     * @param   array $array    Array in which to check for the option
     * @throws  Zend_Service_Exception
     */
    protected function _validateInArray($name, $value, $array)
    {}

    /**
     * Check Result for errors
```

```
     *
     * @param   DomDocument $dom    API response DOM document
     * @throws  Zend_Service_Exception
     * @link    http://technorati.com/developers/api/error.html Technorati API: Error response
     */
    protected static function _checkErrors(DomDocument $dom)
    {}

    /**
     * Check ReST response for errors
     *
     * @param   Zend_Http_Response  $response   ReST response
     * @throws  Zend_Service_Exception
     */
    protected static function _checkResponseErrors(Zend_Http_Response $response)
    {}

    /**
     * Checks whether the user options are valid
     *
     * @param   array $options        User options
     * @param   array $validOptions   Valid options
     * @throws  Zend_Service_Exception
     */
    protected function _compareOptions($options, $validOptions)
    {}

    /**
     * Prepare options for the request
     *
     * @param   array $options        User options
     * @param   array $defaultOptions Default options
     * @return  array Merged array of user and default/required options.
     */
```

```
    protected function _prepareOptions($options, $defaultOptions)
    {}
}
```

Author Object

```
class Zend_Service_Technorati_Author
{
    /**
     * Author first name
     *
     * @var string
     */
    public $firsName;

    /**
     * Author last name
     *
     * @var string
     */
    public $lastName;

    /**
     * Technorati account description
     *
     * @var string
     */
    public $description;

    /**
     * Technorati account biography
     *
     * @var string
     */
    public $bio;

    /**
     * Technorati account thumbnail picture URL, if any
     *
     * @var null|Zend_Uri_Http
     */
    public $thumbnailPicture;


    /**
     * Parse the given Author Element
     *
     * @param   DomElement $dom The ReST fragment for this author object
     * @todo    Check which elements are optional
     */
    public function __construct(DomElement $dom)
    {}
}
```

Weblog Object

```


class Zend_Service_Technorati_Weblog
{
    /**
     * Blog name as written in the feed
     *
     * @var string
     */
    public $name;
```

```
    /**
     * Base blog URL
     *
     * @var Zend_Uri_Http
     */
    public $url;

    /**
     * RSS feed URL, if any
     *
     * @var null|Zend_Uri_Http
     */
    public $rssUrl;

    /**
     * Atom feed URL, if any
     *
     * @var null|Zend_Uri_Http
     */
    public $atomUrl;

    /**
     * Number of unique blogs linking this blog
     *
     * @var int
     */
    public $inboundBlogs;

    /**
     * Number of incoming links to this blog
     *
     * @var int
     */
    public $inboundLinks;

    /**
     * Technorati rank value for this blog
     *
     * @var int
     */
    public $rank;

    /**
     * Last blog update timestamp
     *
     * @var int
     */
    public $lastUpdate;

    /**
     * Blog latitude coordinate
     *
     * @var float
     */
    public $lat;

    /**
     * Blog longitude coordinate
     *
     * @var float
     */
    public $lon;

    /**
     * Whether the author who claimed this weblog has a photo
     *
     * @var int
     * @see Zend_Service_Technorati_Author::$thumbnailPicture
     */
```

```
    public $hasPhoto;

    /**
     * A list of Zend_Service_Technorati_Author who claimed this blog
     *
     * @var array
     */
    public $authors = array();


    /**
     * Parse the given Weblog Element
     *
     * @param   DomElement $dom The ReST fragment for this weblog object
     */
```

```
    public function __construct(DomElement $dom)
    {}
}
```

Key Object

```
class Zend_Service_Technorati_Key
{
    /**
     * Technorati API key
     *
     * @var     string
     */
    public $apiKey;

    /**
     * Number of queries used today
     *
     * @var     string
     */
    public $apiQueries;

    /**
     * Total number of available queries per day
     *
     * @var     string
     */
    public $maxQueries;


    /**
     * Technorati API response document
     *
     * @var     DomDocument
     * @access  protected
     */
    protected $_dom;

    /**
     * Object for $this->_dom
     *
     * @var     DOMXpath
     * @access  protected
     */
    protected $_xpath;


    /**
     * Parse the given Weblog Element
     *
     * @param   DomElement $dom
     * @param   Zend_Service_Technorati $technorati Object that generated the request
     */
    public function __construct(DomDocument $dom, $technorati)
    {}

    /**
     * Return the response XML document
     *
     * @return string   The response document converted into XML format
     */
    function getXML()
    {}
}
```

ResultSet Object

```
class Zend_Service_Technorati_ResultSet implements SeekableIterator
{
    /**
     * The total number of results available
     *
     * @var     int
     */
    public $totalResultsAvailable;

    /**
     * The number of results in this result set
     *
     * @var     int
     */
    public $totalResultsReturned;

    /**
     * The offset in the total result set of this search set
     *
     * @var     int
     */
    public $firstResultPosition;


    /**
     * A DomNodeList of results
     *
     * @var     DomNodeList
     * @access  protected
     */
    protected $_results;

    /**
     * Technorati API response document
     *
     * @var     DomDocument
     * @access  protected
     */
    protected $_dom;

    /**
     * Object for $this->_dom
     *
     * @var     DOMXpath
     * @access  protected
     */
    protected $_xpath;

    /**
     * Current Item
     *
     * @var     int
     * @access  protected
     */
    protected $_currentItem = 0;


    /**
     * Parse the search response and retrieve the results for iteration
     *
     * @param   DomDocument $dom    The ReST fragment for this object
     * @param   array $options      Query options as associative array
     */
    public function __construct(DomDocument $dom, $options = array())
    {}


    /**
```

```
     * Total Number of results returned
     *
     * @return  int     Total number of results returned
     */
    public function totalResults()
    {}


    /**
     * Implement SeekableIterator::current
     *
     * @return  void
     * @throws  Zend_Service_Exception
     */
    public function current()
    {}


    /**
     * Implement SeekableIterator::key
     *
     * @return  int
     */
    public function key()
    {}


    /**
     * Implement SeekableIterator::next
     *
     * @return  void
     */
    public function next()
    {}


    /**
     * Implement SeekableIterator::rewind
     *
     * @return  bool
     */
    public function rewind()
    {}


    /**
     * Implement SeekableIterator::seek
     *
     * @param   int     $item
     * @return  Zend_Service_Technorati_Result
     * @throws  Zend_Service_Exception
     */
    public function seek($item)
    {}


    /**
     * Implement SeekableIterator::valid
     *
     * @param   int     $item
     * @return  bool
     */
    public function valid($item = null)
    {}

    /**
     * Return the response XML document
     *
     * @return string   The response document converted into XML format
     */
```

```
    function getXML()
    {}
}
```

Result Object

```
class Zend_Service_Technorati_Result
{
    /**
     * An associative array of 'fieldName' => 'xmlfieldtag'
     *
     * @var     array
     * @access  protected
     */
    protected $_fields;

    /**
     * The ReST fragment for this result object
     *
     * @var     DomElement
     * @access  protected
     */
    protected $_result;

    /**
     * Object for $this->_dom
     *
     * @var     DOMXpath
     * @access  protected
     */
    protected $_xpath;


    /**
     *
     *
     * @param   DomElement $result  The ReST fragment for this object
     * @todo docblock
     */
    public function __construct(DomElement $result)
    {}
}
```

Cosmos ResultSet Object (each service will have its own class)

```
class Zend_Service_Technorati_CosmosResultSet extends Zend_Service_Technorati_ResultSet
{
    /**
     * Parse the search response and retrieve the results for iteration
     *
     * @param   DomDocument $dom    The ReST fragment for this object
     * @param   array $options      Query options as associative array
     */
    public function __construct(DomDocument $dom, $options = array())
    {}


    /**
     * Implement SeekableIterator::current and
     * overwrites Zend_Service_Technorati_ResultSet::current()
     *
     * @return Zend_Service_Technorati_CosmosResult Current Zend_Service_Technorati_Cosmos result
     */
    public function current()
    {}
}
```

Cosmos Result Object (each service will have its own class)

```
class Zend_Service_Technorati_CosmosResult extends Zend_Service_Technorati_Result
{
    /**
     * @var     Zend_Service_Technorati_Blog
     * @todo docblock
     */
    public $weblog;

    /**
     * @var
     * @todo docblock
     */
    public $nearestPermalink;

    /**
     * @var
     * @todo docblock
     */
    public $excerpt;

    /**
     * @var
     * @todo docblock
     */
    public $linkCreated;

    /**
     * @var
     * @todo docblock
     */
    public $linkUrl;


    /**
     *
     *
     * @param   DomElement $result  The ReST fragment for this object
     * @todo docblock
     */
    public function __construct(DomElement $result)
    {}
}
```

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