Zend_Service_Mollom - Ruben Vermeersch

<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_Mollom Component Proposal

Proposed Component Name	Zend_Service_Mollom
Developer Notes	http://framework.zend.com/wiki/display/ZFDEV/Zend_Service_Mollom
Proposers	Ruben Vermeersch
Zend Liaison	TBD
Revision	1.0 - 21 December 2008: Initial Draft. (wiki revision: 7)

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_Mollom is a Zend Framework component for calling Mollom, a spam filtering and CAPTCHA service for blog comments and forum posts.

This component is currently released at http://ruben.savanne.be/articles/zend-service-mollom-documentation and has been reviewed (and approved) by the Mollom team. It is feature complete: it implements the full Mollom API and has unit tests that cover most of the feature set.

The caching API has been modeled similar to the one in Zend_Locale and Zend_Translate.

2. References

- Zend_Service_Mollom Homepage
- Mollom.com

3. Component Requirements, Constraints, and Acceptance Criteria

- This component **will** allow users to call the Mollom API using a Zend_Service component.

- This component **will** optionally support caching by means of a Zend_Cache_Core instance.

4. Dependencies on Other Framework Components

- Zend_Cache_Core (optional)
- Zend_Service_Abstract
- Zend_Service_Exception
- Zend_XmlRpc_Client
- Zend_XmlRpc_Request

5. Theory of Operation

A user guide is available at http://ruben.savanne.be/articles/zend-service-mollom-documentation

6. Milestones / Tasks

- Milestone 1: [DONE] Initial version, with test coverage (release 1.0.0 and 1.1.0, December 07)
- Milestone 2: [DONE] Additional release, with optional caching support and additional test coverage (1.2.0, December 21)

7. Class Index

- Zend_Service_Mollom
- Zend_Service_Mollom_UnavailableException

8. Use Cases

Please check the documentation at http://ruben.savanne.be/articles/zend-service-mollom-documentation, which has been formatted nicely and explains the usage of this component.

9. Class Skeletons

```
class Zend_Service_Mollom_UnavailableException extends Zend_Service_Exception {}

class Zend_Service_Mollom {
    /**
     * The API version supported by this service.
     */
    const API_VERSION = '1.0';

    /**
```

```
     * Fault codes
     */
    const FAULT_REFRESH = 1100;
    const FAULT_BUSY = 1200;

    /**
     * Classification labels
     */
    const CLASSIFICATION_HAM = 'ham';
    const CLASSIFICATION_SPAM = 'spam';
    const CLASSIFICATION_UNSURE = 'unsure';

    /**
     * Feedback types
     */
    const FEEDBACK_SPAM = 'spam';
    const FEEDBACK_PROFANITY = 'profanity';
    const FEEDBACK_LOWQUALITY = 'low-quality';
    const FEEDBACK_UNWANTED = 'unwanted';

    /**
     * Statistics types
     */
    const STATISTICS_TOTAL_DAYS         = 'total_days';
    const STATISTICS_TOTAL_ACCEPTED     = 'total_accepted';
    const STATISTICS_TOTAL_REJECTED     = 'total_rejected';
    const STATISTICS_YESTERDAY_ACCEPTED = 'yesterday_accepted';
    const STATISTICS_YESTERDAY_REJECTED = 'yesterday_rejected';
    const STATISTICS_TODAY_ACCEPTED     = 'today_accepted';
    const STATISTICS_TODAY_REJECTED     = 'today_rejected';

    /**
     * Constructor
     *
     * @param string $publicKey The public key
     * @param string $privateKey The private key
     */
    public function __construct($publicKey, $privateKey)

    /**
     * Get the public key used for making API calls
     *
     * @return string The public key
     */
    public function getPublicKey()

    /**
     * Set the public key used for making API calls
     *
     * @param string $publicKey The public key
     * @return Zend_Service_Mollom
     */
    public function setPublicKey($publicKey)

    /**
     * Get the private key used for making API calls
     *
     * @return string The private key
     */
```

```
    public function getPrivateKey()

    /**
     * Set the private key used for making API calls
     *
     * @param string $privateKey The private key
     * @return Zend_Service_Mollom
     */
    public function setPrivateKey($privateKey)

    /**
     * Returns the set cache
     *
     * @return Zend_Cache_Core The set cache
     */
    public static function getCache()

    /**
     * Set a cache for Zend_Service_Mollom
     *
     * @param Zend_Cache_Core $cache A cache frontend
     */
    public static function setCache(Zend_Cache_Core $cache)

    /**
     * Returns true when a cache is set
     *
     * @return boolean
     */
    public static function hasCache()

    /**
     * Removes any set cache
     *
     * @return void
     */
    public static function removeCache()

    /**
     * Clears all set cache data
     *
     * @return void
     */
    public static function clearCache()

    /**
     * Validate a captcha.
     *
     * @param string $session_id The ID of the solved captcha
     * @param string $solution The solution of the captcha
     * @return boolean Whether it was solved correctly or not
     */
    public function checkCaptcha($session_id, $solution)

    /**
     * Check the content to see whether it's spam or not.
     *
     * The $params array can have any of the following keys:
     *      - session_id
```

```
     *      - post_title
     *      - post_body
     *      - author_name
     *      - author_url
     *      - author_mail
     *      - author_openid
     *      - author_ip
     *      - author_id
     *
     * The meaning of these parameters is available in the Mollom API documentation:
     * http://mollom.com/api
     *
     * @param array $params A key/value pair containing the content of the post to be
checked.
     * @return array An array with a classification label, a rating and a session id.
     */
    public function checkContent(array $params)

   /**
     * Get an audio captcha for a given session id
     *
     * @param string $session_id The session id, will be generated if omitted
     * @param string $author_ip The IP of the content author (the person that will
need to solve the captcha)
     * @return array An array containing a session_id and a captcha url.
     */
    public function getAudioCaptcha($session_id = null, $author_ip = null)

    /**
     * Get an image captcha for a given session id
     *
     * @param string $session_id The session id, will be generated if omitted
     * @param string $author_ip The IP of the content author (the person that will
need to solve the captcha)
     * @return array An array containing a session_id and a captcha url.
     */
    public function getImageCaptcha($session_id = null, $author_ip = null)

    /**
     * Get the list of servers used to make API calls
     *
     * @return array An array of strings containing server URLs
     */
    public function getServers()

    /**
     * Retrieve statistics about your mollom usage
     *
     * @param string $type The desired statistic type (check the STATISTICS_*
constants for possible values).
     * @return int The value of the requested statistic
     */
    public function getStatistics($type)

    /**
     * Send feedback to the Mollom service, used to classify unsure content manually.
     *
     * @param string $session_id The object to which the feedback relates
     * @param string $feedback A feedback rating (see the FEEDBACK_* constants)
```

```
     */
    public function sendFeedback($session_id, $feedback)

    /**
     * Verify the current key.
     *
     * Will not throw an exception when valid.
     */
```

```
    public function verifyKey()
}
```

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