Zend_Mail_Transport_Queue - Simon Mundy

<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_Mail_Transport_Queue Component Proposal

Proposed Component Name	Zend_Mail_Transport_Queue
Developer Notes	http://framework.zend.com/wiki/display/ZFDEV/Zend_Mail_Transport_Queue
Proposers	• Simon Mundy • Alexander Veremyev • (Zend-liaison)
Revision	0.1 - 1 October 2007: Initial proposal. (wiki revision: 4)

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
UC 01: Simple usage
UC 02: Advanced usage with DB and Logger
9. Class Skeletons

1. Overview

Zend_Mail_Transport_Queue is designed to manage the delivery of a large volume of mail. It provides a number of backends by which to store the mail and then allows deferred delivery from the storage container using concrete transport methods from a separate script.

2. References

- PEAR Mail_Queue

3. Component Requirements, Constraints, and Acceptance Criteria

- This component **will** provide an interface to allow the creation of multiple backends.
- This component **will not** send mail after a developer-specified amount of retries.
- This component **will** allow the developer to pass any valid transport component to re-send mail.

- This component **will** provide a means for passing log instances.

4. Dependencies on Other Framework Components

- Zend_Mail
- Zend_Mail_Transport_Abstract

5. Theory of Operation

This transport type replaces an existing SMTP or Sendmail transport instance. A concrete storage adapter is instantiated (e.g. Zend_Mail_Transport_Queue_Db or Zend_Mail_Transport_Queue_File) and then used as the transport to dispatch mail in the usual manner.

A secondary script (run as a cron job, scheduled task or as a trigger during another script's execution) will then access the transport's static process method. During this time mail is processed and directed to a specified transport (e.g. SMTP or Sendmail) and sent as normal. Processing will be limited to an amount configurable by the developer. If a mail instance cannot be sent (i.e. throws a Zend_Mail_Transport_Exception or Zend_Mail_Protocol_Exception exception) the mail will be requeued for a specified amount of retries (default is 10).

When mail is processed (success or failure) Zend_Mail_Transport_Queue will also log the attempt if a Zend_Log adapter is made available. The content and format of the log message is also developer-configurable.

6. Milestones / Tasks

- Milestone 1: design notes will be published here
- Milestone 2: Working prototype checked into the incubator supporting use cases #1, #2, ...
- Milestone 3: Working prototype checked into the incubator supporting use cases #3 and #4.
- Milestone 4: Unit tests exist, work, and are checked into SVN.
- Milestone 5: Initial documentation exists.

7. Class Index

- Zend_Mail_Transport_Queue_Interface
- Zend_Mail_Transport_Queue_Abstract
- Zend_Mail_Transport_Queue_Db
- Zend_Mail_Transport_Queue_File

8. Use Cases

9. Class Skeletons

```
abstract class Zend_Mail_Transport_Queue_Abstract extends Zend_Mail_Transport_Abstract
{
    /**
     * Config options for authentication
```

```
     *
     * @var array
     */
    protected $_config = array('groupId'             => 'default',
                               'limit'               => 10,
                               'retry'               => 10,
                               'setDeliverTimestamp' => false,
                               'archiveDelivered'    => false,
                               'deliverTimestamp'    => 'Y-m-d H:i:s',
                               'logMessageDeliver'   => '[%s] delivered: subject=%s to=%s from=%s
id=%s',
                               'logMessageError'     => '[%s] failed:    subject=%s to=%s from=%s
id=%s');


    /**
     * Send an email to the database queue
     *
     * Every individual recipient is inserted as a separate row. These rows accumulate in the
     * queue until they are manually processed.
     *
     * @return void
     */
    abstract public function _sendMail();



    /**
     * Process rows in the database queue
     *
     * @return int Number of rows successfully processed
     */
    static public function process($containerId, $options, $transport = null, $log = null) {}
}

class Zend_Mail_Transport_Queue_Db extends Zend_Mail_Transport_Queue_Abstract
{
    const FIELD_ID        = 'id';
    const FIELD_GROUPID   = 'groupid';
    const FIELD_SENDER    = 'sender';
    const FIELD_RECIPIENT = 'recipient';
    const FIELD_SUBJECT   = 'subject';
    const FIELD_MESSAGE   = 'message';
    const FIELD_ATTEMPTS  = 'attempts';
    const FIELD_SCHEDULE  = 'schedule';
    const FIELD_STATUS    = 'status';
    const FIELD_SENT      = 'sent';

    /**
     * Database connection
     *
     * @var string
     */
    protected $_db;


    /**
     * Config options for authentication
     *
     * @var array
     */
    protected $_schema = array(self::FIELD_ID        => 'id',
                               self::FIELD_GROUPID   => 'groupid',
                               self::FIELD_SENDER    => 'sender',
                               self::FIELD_RECIPIENT => 'recipient',
                               self::FIELD_SUBJECT   => 'subject',
                               self::FIELD_MESSAGE   => 'message',
                               self::FIELD_ATTEMPTS  => 'attempts',
                               self::FIELD_SENT      => 'sent',
                               self::FIELD_STATUS    => 'status');
```

```
    /**
     * Constructor.
     *
     * @param  Zend_Db_Adapter_Abstract $db
     * @param  string $table
     * @param  array|null $config OPTIONAL (Default: array)
     * @return void
     */
    public function __construct(Zend_Db_Adapter_Abstract $db, Array $config = array()) {}


    /**
     * Process rows in the database queue
     *
     * @return int Number of rows successfully processed
     */
    static public function process(Zend_Db_Adapter_Abstract $db, Array $config = array(),
                                   Zend_Mail_Transport_Abstract $transport = null,
```

```
                                        Zend_Log $log = null) {}

  }
```

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