Zend_Mobile_Push - Mike Willbanks

<ac:macro ac:name="info"><ac:parameter ac:name="title">Zend_Mobile_Push</ac:parameter></ac:macro>

<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_Mobile_Push Component Proposal

Proposed Component Name	Zend_Mobile_Push
Developer Notes	http://framework.zend.com/wiki/display/ZFDEV/Zend_Mobile_Push
Proposers	Mike Willbanks
Zend Liaison	TBD
Revision	1.1 - 28 November 2011: Initial Draft. (wiki revision: 8)

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_Mobile_Push is a component for implementing push notifications for the 3 major push notification platforms (Apple (Apns), Google (C2dm), and Microsoft (Mpns). The actual implementation is based on mostly abstraction.

2. References

Specifications / Implementation Guides

- Windows Phone
- Apple Push Notification Service
- Android Cloud to Device Messaging

Other Implementations

APNS

- ApnsPHP
- PHP APNS
- EASY APNS

C2DM

- Zend_Service_Google_C2dm (mine only in github)
- https://github.com/lytsing/c2dm-php

MPNS

- Send Push Notifications to Windows Phone through PHP article

3. Component Requirements, Constraints, and Acceptance Criteria

- This component **will** implement base interfaces along with an abstract class to attempt to bundle common behaviors
- This component **will** implement the current specifications for APNS, C2DM and MPNS.
- This component **will** throw exceptions that should be caught (InvalidAuth, InvalidToken, etc)
- This component **will not** handle the storage of tokens or ID's (no implementation of a persistent storage state).
- This component **will not** attempt to handle re-connections or exponential back off.
- This component **will not** provide a server component that runs in the background.
- This component **will not** interface with a logger.

4. Dependencies on Other Framework Components

Zend_Http_Client - for implementation of sending messages to C2DM and MPNS.
Zend_Filter_Alpha - for filtering out dynamic text to ensure alpha characters.
Zend_Loader - for dynamically loading exception classes based on responses.

5. Theory of Operation

The operation of this component is very simple. It is based simply off of interfaces and abstract classes to implement the server and messages. All common operations have been stated there. The instances of each type of message does not make sense in the context of an abstract factory or a factory at all since each must be implemented in a specific way. Each individual instance must be explicitly stated such as "new Zend_Mobile_Push_C2dm()" including the type of message "new Zend_Mobile_Push_Message_C2dm".

Generally an instance of the client will be created followed by a message that will be passed to the send method of the client. Every attempt was made to standardize some of the handling of each individual item.

6. Milestones / Tasks

- Milestone 1: Finish proposal and architectural description. [Done]
- Milestone 2: Working prototype. [Done]
- Milestone 3: Unit tests exist, work, and are checked into github. [Partially Done]
- Milestone 4: Refinements [Not Done]
- Milestone 5: Initial documentation exists. [Not Done]

7. Class Index

- Zend_Mobile_Exception
- Zend_Mobile_Push_Exception
- Zend_Mobile_Push_Exception_MessageTooBig
- Zend_Mobile_Push_Exception_InvalidRegistration

- Zend_Mobile_Push_Exception_InvalidToken
- Zend_Mobile_Push_Exception_DeviceQuotaExceeded
- Zend_Mobile_Push_Exception_NotRegistered
- Zend_Mobile_Push_Exception_MissingCollapseKey
- Zend_Mobile_Push_Exception_InvalidAuthToken
- Zend_Mobile_Push_Exception_InvalidTopic
- Zend_Mobile_Push_Exception_ServerUnavailable
- Zend_Mobile_Push_Exception_MismatchSenderId
- Zend_Mobile_Push_Exception_QuotaExceeded
- Zend_Mobile_Push_Exception_MissingRegistration
- Zend_Mobile_Push_Apns
- Zend_Mobile_Push_C2dm
- Zend_Mobile_Push_Mpns
- Zend_Mobile_Push_Message_Exception
- Zend_Mobile_Push_Message_Mpns_Tile
- Zend_Mobile_Push_Message_Mpns_Raw
- Zend_Mobile_Push_Message_Mpns_Toast
- Zend_Mobile_Push_Message_Apns
- Zend_Mobile_Push_Message_C2dm
- Zend_Mobile_Push_Message_Mpns
- Zend_Mobile_Push_Message_Interface
- Zend_Mobile_Push_Message_Abstract
- Zend_Mobile_Push_Interface
- Zend_Mobile_Push_Abstract

8. Use Cases

UC-01

Sending an APNS Message

```
require_once 'Zend/Mobile/Push/Apns.php';

$apns = new Zend_Mobile_Push_Apns();
$apns->setCertificate(dirname(__FILE__) . '/mycert.pem'); // REPLACE WITH YOUR CERT

$message = new Zend_Mobile_Push_Message_Apns();
$message->setToken('some-type-of-token'); // REPLACE WITH A APNS TOKEN
$message->setId(time());
$message->setAlert('This is a test message!');
$message->setBadge(1);
$message->setSound('somesound');
$apns->send($message);
```

UC-02

Feedback of an APNS Message

```
require_once 'Zend/Mobile/Push/Apns.php';

$apns = new Zend_Mobile_Push_Apns();
$apns->setCertificate(dirname(__FILE__) . '/mycert.pem'); // REPLACE WITH YOUR CERT
print_r($apns->feedback()); // array of items
```

UC-03

Sending a C2DM Message

```
require_once 'Zend/Mobile/Push/C2dm.php';
require_once 'Zend/Gdata/ClientLogin.php';

try {
    $client = Zend_Gdata_ClientLogin::getHttpClient(
        'my@gmail.com', // REPLACE WITH YOUR GOOGLE ACCOUNT
        'myPassword', // REPLACE WITH YOUR PASSWORD
        Zend_Mobile_Push_C2dm::AUTH_SERVICE_NAME,
        null,
        'myAppName' // REPLACE WITH YOUR APP NAME
    );
} catch (Zend_Gdata_App_CaptchaRequiredException $cre) {
    // manual login is required
    echo 'URL of CAPTCHA image: ' . $cre->getCaptchaUrl() . PHP_EOL;
    echo 'Token ID: ' . $cre->getCaptchaToken() . PHP_EOL;
    exit(1);
} catch (Zend_Gdata_App_AuthException $ae) {
    echo 'Problem authenticating: ' . $ae->exception() . PHP_EOL;
    exit(1);
}

$c2dm = new Zend_Mobile_Push_C2dm();
$c2dm->setLoginToken($client->getClientLoginToken());

$message = new Zend_Mobile_Push_Message_C2dm();
$message->setToken('a-device-token'); // REPLACE WITH A DEVICE TOKEN
$message->setId('testCollapseKey');
$message->setData(array(
    'title' => 'Test Notification',
    'msg' => 'This is a test notification.'
));
$c2dm->send($message);
```

UC-04

Sending a MPNS Raw Message

```
<?php
require_once 'Zend/Mobile/Push/Mpns.php';
require_once 'Zend/Mobile/Push/Message/Mpns/Raw.php';

$mpns = new Zend_Mobile_Push_Mpns();

$message = new Zend_Mobile_Push_Message_Mpns_Raw();
$message->setToken('http://somenotificationuri'); // REPLACE WITH NOTIFICATION URI FROM MPNS
$message->setMessage('<root><value attr="stuff" /></root>');
$mpns->send($message);
```

UC-05

Sending a MPNS Toast Message

```
<?php
require_once 'Zend/Mobile/Push/Mpns.php';
require_once 'Zend/Mobile/Push/Message/Mpns/Toast.php';

$mpns = new Zend_Mobile_Push_Mpns();

$message = new Zend_Mobile_Push_Message_Mpns_Toast();
$message->setToken('http://somenotificationuri'); // REPLACE WITH NOTIFICATION URI FROM MPNS
$message->setTitle('Test Notification');
$message->setMessage('This is a test notification');

// optional attributes for wp7.1+
$message->setParams('?value=12345')'
$mpns->send($message);
```

UC-06

Sending a MPNS Tile Message

```
<?php
require_once 'Zend/Mobile/Push/Mpns.php';
require_once 'Zend/Mobile/Push/Message/Mpns/Toast.php';

$mpns = new Zend_Mobile_Push_Mpns();

$message = new Zend_Mobile_Push_Message_Mpns_Tile();
$message->setToken('http://somenotificationuri'); // REPLACE WITH NOTIFICATION URI FROM MPNS
$message->setBackgroundImage('red.jpg');
$message->setCount(1);
$message->setTitle('Wowzah!');

// other optional attributes for wp7.1+
$message->setTileId('/SomeAction.xaml');
$message->setBackBackgroundImage('blue.jpg');
$message->setBackTitle('Woaaa!');
$message->setBackContent('dude');
$mpns->send($message);
```

9. Class Skeletons

See: https://github.com/mwillbanks/Zend_Mobile

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