OpenBadges: Technical Specification

Document Control

Abstract	Technical specification for OpenBadges feature				
Author	Yuliya Bozhko	**Version**	v.0.1	**Date**	08-10-2012

Introduction

This document describes and details the proposed technical implementation for the OpenBadges functionality for Moodle.

Contents

Deliverables

`OpenBadges` will be a part of Moodle core. It should be implemented with least possible changes to the existing features. `OpenBadges` feature will provide functionality for managing, displaying and awarding badges. Block `openbadges`, when added to a page, will display to users their earned badges.

Database schema

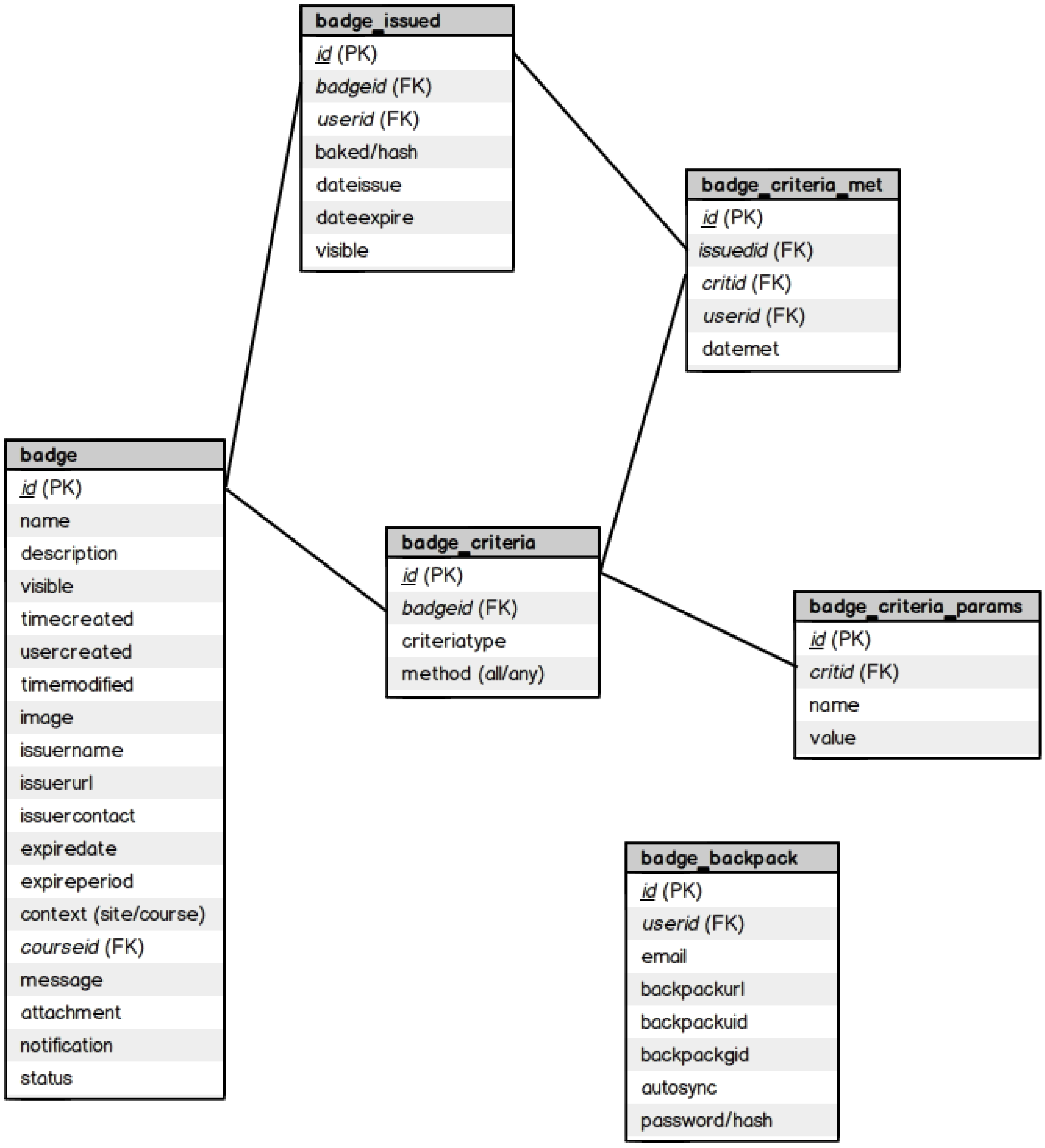

Table `mdl_badge` stores information about badges:

Name	Type	Decription	Default	Required	Indexed
id	Integer	Primary key	N/A	Yes	No
name	Char (255)	Name of a badge	N/A	Yes	No
description	Text	Badge description	NULL	No	No
image	Integer	Badge image file	0	Yes	No
visible	Integer	Badge is visible to users. 0 = invisible; 1 = visible	0	Yes	No
timecreated	Integer	Unix timestamp	0	Yes	No
timemodified	Integer	Unix timestamp	0	Yes	No
modifierid	Integer	ID of the user who performed last modifications (mdl_user.id FK)	N/A	Yes	No
issuername	Char (255)	Name of the issuer	N/A	Yes	No
issuerurl	Char (255)	Issuer URL	N/A	Yes	No
issuercontact	Char (255)	Email address of the issuer	NULL	No	No
expiredate	Integer	Unix timestamp	NULL	No	No
expireperiod	Integer	Period (number of days) from issue date when badge is valid	NULL	No	No
context	Integer	Badge context: 1 = site; 2 = course	N/A	Yes	Yes
courseid	Integer	ID of related course (mdl_course.id FK)	N/A	Yes	No
message	Text	Message when badge is awarded		Yes	No
attachment	Integer	Attach baked badge for download: 1 = include an attachment; 0 = no attachment	1	Yes	No
notification	Integer	Notify badge creator when badge is awarded: 1 = send notification; 0 = no notification	1	Yes	No
status	Integer	Badge status as described in Constants: 0 = inactive; 1 = active; 2 = active+locked; 3 = inactive+locked; 4 = archived	0	Yes	No

Table `mdl_badge_criteria` defines criteria for issuing badges:

Name	Type	Decription	Default	Required	Indexed
id	Integer	Primary key	N/A	Yes	No
badgeid	Integer	Badge ID (mdl_badge.id FK)	N/A	Yes	Yes
criteriatype	Integer	Badge criteria types as described in Constants: 0 = overall; 1 = manual; 2 = course; 3 = module; 4 = profile; 5 = social	N/A	Yes	Yes
method	Integer	Aggregation method: 1 = all; 2 = any	1	Yes	No

Should we consider an index on unique pairs of [badgeid + criteriatype].

Table `mdl_badge_criteria_param` defines parameters for badges criteria:

Name	**Type**	**Decription**	**Default**	**Required**	**Indexed**
id	Integer	Primary key	N/A	Yes	No
critid	Integer	Criteria ID (mdl_badge_criteria.id FK)	N/A	Yes	Yes
name	Char (255)	Badge criteria parameter name	N/A	Yes	No
value	Char (255)	Badge criteria parameter value	N/A	Yes	No

Table `mdl_badge_backpack` defines settings for connecting external backpack:

Name	**Type**	**Decription**	**Default**	**Required**	**Indexed**
id	Integer	Primary key	N/A	Yes	No
userid	Integer	ID of the user (mdl_user.id FK)	N/A	Yes	Yes
email	Char (255)	User email in the backpack	N/A	Yes	No
backpackurl	Char (255)	Backpack service URL address	N/A	Yes	No
backpackuid	Integer	User id in the backpack	N/A	Yes	No
backpackgid	Integer	Badges group to display from the backpack	N/A	Yes	No
autosync	Integer	Automatic pushing badges to backpack: 1 = yes; 0 = no	0	Yes	No
password	Char (255)	Hashed password for automatic connection to backpack	N/A	Yes	No

This table assumes that all backpacks are structured as defined in Mozilla OpenBadges Infrastructure documentations.

Table `mdl_badge_issued` stores information on issued badges:

Name	**Type**	**Decription**	**Default**	**Required**	**Indexed**
id	Integer	Primary key	N/A	Yes	No
badgeid	Integer	Criteria ID (mdl_badge_criteria.id FK)	N/A	Yes	Yes
userid	Integer	ID of the user (mdl_user.id FK)	N/A	Yes	Yes
baked	Char (255)	Unique hash to be used for public URLs	N/A	Yes	No
dateissue	Integer	Unix timestamp of the date when the badge was issued	N/A	Yes	No
dateexpire	Integer	Unix timestamp of the date when the badge expires	NULL	No	No
visible	Integer	Badge visible in user profile: 1 = visible; 0 = hidden	0	Yes	No

Unique baked hash can be calculated using `sha1(rand().userid.badgeid.time())`. Maybe it can be used instead instead of ID?

Table `mdl_badge_criteria_met` defines criteria that were met for an issued badge:

Name	**Type**	**Decription**	**Default**	**Required**	**Indexed**
id	Integer	Primary key	N/A	Yes	No
issuedid	Integer	Issued badge ID (mdl_badge_issued.id FK)	N/A	Yes	Yes
critid	Integer	Criteria IDs for met criteria (mdl_badge_criteria.id FK)	N/A	Yes	No
userid	Integer	ID of the user (mdl_user.id FK)	N/A	Yes	No
datemet	Integer	Unix timestamp of the date when criteria were met	N/A	Yes	No

Administrative permissions

Capabilities that will be defined for managing badges in `/db/access.php`.

Viewing and earning badges	
openbadges:manageownbadges	Allows to view and manage own earned badges
openbadges:viewotherbadges	Allows to view public badges in other users' profiles
openbadges:viewbadges	Allows to view badges without earning them
openbadges:earnbadge	Be able to earn badge (Student by default)
Managing badges	
openbadges:createbadge	Allows to create/duplicate badges
openbadges:deletebadge	Allows to delete badges
openbadges:configuredetails	Allows to set up/edit badge details
openbadges:configurecriteria	Allows to set up/edit criteria of earning a badge
openbadges:configuremessages	Allows to configure badge messages
openbadges:awardbadge	Allows to award badge to a user
openbadges:viewawarded	Allows to view users who earned a specific badge without being able to award a badge (Awarded tab on badge details page)
Administrative settings	
openbadges:manageglobalsettings	Allows general administration of global settings

Constants

Badge status constants (`status` column in `badge` table):

`BADGE_STATUS_ARCHIVED` – Deleted/archived badge can no longer be earned and is not displayed in the list of badges.

`BADGE_STATUS_ACTIVE` – Active badge means that this badge can we earned, but it has not been awarded yet. Can be deactivated for the purpose of changing its criteria.

`BADGE_STATUS_INACTIVE` – Inactive badge means that this badge cannot be earned and has not been awarded yet. Its award criteria can be changed.

`BADGE_STATUS_ACTIVE_LOCKED` – Active badge means that it can be earned and has already been awarded to users. Its criteria cannot be changed now.

`BADGE_STATUS_INACTIVE_LOCKED` – Inactive badge can no longer be earned, but it has been awarded in the past and therefore its criteria cannot be changed.

Badge award criteria type constants (`criteriatype` column in `badge_criteria` table):

`BADGE_CRITERIA_TYPE_OVERALL` – Overall badge award criteria

`BADGE_CRITERIA_TYPE_MANUAL` – Manual award by some specific roles

`BADGE_CRITERIA_TYPE_COURSE` – Course/courseset completion criteria type

`BADGE_CRITERIA_TYPE_MODULE` – Activity completion criteria type

`BADGE_CRITERIA_TYPE_PROFILE` – Completing profile fields criteria type (need to consider how to handle custom fields e.g. deletion)

`BADGE_CRITERIA_TYPE_SOCIAL` – Social activity participation criteria type (posting to forum, writing blogs, etc.)

Badge award criteria aggregation constants (`method` column in `badge_criteria` table):

`BADGE_CRITERIA_AGGREGATION_ALL` – Complete all criteria for a badge

`BADGE_CRITERIA_AGGREGATION_ANY` – Complete any criteria for a badge

Implementation parts

Badge baking service – library that will write metadata into PNG tEXt chunks (See `http://www.libpng.org/pub/png/spec/1.2/PNG-Chunks.html` for more information). Currently, Mozilla OB project is using Node framework with Metapng library which would be difficult to reuse in Moodle. They are also trying to refine their code to separate baking service in its own libraries, but it might take a while and will still be using Node.js. It might make sense to write our own baking service library independent from both Mozilla and Moodle.

Issued badge – public part that will use issued badge hash for the purposes of: generating assertions (public at all times URL that will return valid JSON); generating criteria records (for a public page that will show overall badge criteria); generating evidence records (for a public page that will show criteria met for earning a specific instance of badge).

External backpack interaction – pushing badges to external backpacks. This part can be tricky and is still unclear because there is no decision in Mozilla about standard backpack structure. Currently considering implementation for pushing badges to Mozilla backpack service only.

Badge management – Library of global functions for badges associated with viewing, creating, copying, editing, activating or deactivating, deleting badges.

Badge earning/award criteria – Libraries for managing badge award criteria. Will contain classes and functions to be used when tracking badges criteria completion. We can do something similar to course completion.

Preliminary structure

```
/moodle
 ├─ admin
 │   └─ settings
 │       └─ badges.php ........................... settings for badges administration
 ├─ badges
 │   ├─ criteria
 │   │   └─ ............................................... badge criteria classes
 │   ├─ templates
 │   │   └─ ............................................ design templates related
 │   ├─ index.php
 │   ├─ assertion.php................................public URL, returns JSON
 │   ├─ badge.php............public URL, shows issued badge criteria and evidence
 │   ├─ backpacklib.php
 │   ├─ bakerlib.php
 │   ├─ cron.php
 │   ├─ edit.php
 │   ├─ edit_form.php
 │   ├─ edit_criteria.php
 │   ├─ edit_criteria_form.php
 │   ├─ edit_message.php
 │   ├─ edit_message_form.php
 │   ├─ award.php
 │   ├─ award_form.php
 │   ├─ module.js ............................................ main JS library
 │   ├─ mybadges.php .................. page for user badges, maybe move to /user
 │   └─ view.php ......................... page to show available badges to users
 ├─ pix
 │   └─ ......................................... add pictures related to badges
 ├─ lib
 │   ├─ db
 │   │   ├─ events.php..................................... add event handlers
 │   │   ├─ install.xml .............................. addition to db schema
 │   │   └─ access.php .............................. add badge capabilities
 │   └─ badgeslib.php ............................................ main library
 └─ lang
     └─ en
         └─ badges.php
```