CubiServiceModule

Phase-Implementation

Updated Jun 16, 2012 by agus.suhartono

Cubi Service Module

Cubi core services are under /modules/service. Typically, each service has one XML service metadata file and one php file that defines the service class.

Authentication Service

Authentication service is to check if the given username and password match the existing record in user database.

Public API

- Function: authenticateUser ($username, $password)
  - Input: $username, $password as string
  - Output: boolean

Limitation

- This service currently supports authentication against Cubi user table.

Customization

- Developers can have their own authentication service by creating a new class or extending on the current class
- To use new authentication service, please add define('AUTH_SERVICE', 'youAuthService'); in cubi/bin/app_init.php

User profile service

User profile service is to provide user profile data.

Public API

- Function: InitProfile($userName). This function is usually called when user sign in.
  - Input: $userName as string
  - Output: profile array.
- Function: getProfile($attr=null)
  - Input: $attr. If $attr is given, the function returns of value of requested profile attribute.
  - Output: profile array or profile attribute value.
- Function: SetProfile($profile)
  - Input: $profile array
  - Output: none

User Profile array

The default Cubi profile service returns the following user profile array.

- Id. This is user id
- username. User name
- email. User email address
- status.1 or 0. 1 is active user
- lastlogin. Last login time
- lastlogout. Last logout time
- create_by. User id that created this user
- create_time. Time when this user was created
- update_by. User id that updated this user
- update_time. Time when this user was updated
- profile_keys. Keys are the fields defined in contact.do.ContactDO
- roles. Array that stores role ids .
- roleNames. Array that stores role names.
- roleStartpage. Array that stores role start pages.
- groups. Array that stores group ids.

- groupNames. Array that store group names.

Limitation

- This service currently prepares the profile array from Cubi user and role table.

Customization

- Developers can have their own user profile service by creating a new class or extending on the current class
- To use new profile service, please add define('PROFILE_SERVICE', 'profileService'); in cubi/bin/app_init.php

ACL service

ACL service checks the access right of given user to resource.

Public API

- Function: allowAccess($res_action)
  - Input: $res_action as string. $res_action has syntax as "resource.action". For example, "User.Administer_User". The ACL of each module is defined in mod.xml
  - Output: ALLOW or DENY

Limitation

- This service currently support Cubi role-based access control.

Customization

- Developers can have their own ACL service by creating a new class or extending on the current class
- To use new ACL service, please add define('ACL_SERVICE', 'aclService'); in cubi/bin/app_init.php

Access Service

This service restricts view accessibility.

Configuration

View role mapping can be defined in accessService.xml. Here is one sample that allows admin and member roles to access view1.

```
<access-constraint>
      <view-collection>
          <view name="view1">
              <role name="admin"/>
              <role name="member"/>
          </view>
      </view-collection>
</access-constraint>
```

Note: view name supports regular expression. So "system.**" indicates all view in system module.**

Public API

- Function: allowViewAccess($viewName, $role=null)
  - Input: $viewName as string.
  - Input: $role. If role is not given, the function gets all roles of current user profile and check if anyone of the roles can access the view.
  - Output: true or false

Limitation

- Comparing with ACL service, AccessService only restricts view level access, while ACL service controls the resource action level access. In certain applications, using simple view based access control is good enough.

Customization

- Developers can have their own access service by creating a new class or extending on the current class
- To use new access service, please add define('ACCESS_SERVICE', 'accessService'); in cubi/bin/app_init.php

EventLog service

This service provides a generic API to log user event in eventlog table.

Public API

- Function: Log($eventName,$eventMessage,$eventComment=array())
  - Input: $eventName as string.
  - Input: $eventMessage as string.
  - Input: $eventComment as array.

- Output: void

Customization

- Developers can have their own event log service by creating a new class or extending on the current class
- To use new access service, please add define('EVENTLOG_SERVICE', 'eventlogService'); in cubi/bin/app_init.php

User Email Service

This service can compose email content based on a specified email template and send (or add to email queue) email to given email address.

Configuration

In userEmailService.xml, user can set the email templates. The actual email template files are in /cubi/modules/email/templates/.

```
<Template   Name="WelcomeEmail"
                    Title="Welcome New User"
                    EmailAccount="SystemNotifier"
                    Template="welcome_email.tpl" />

<Template   Name="ForgetPasswordEmail"
                    Title="Reset Your Password"
                    EmailAccount="SystemNotifier"
                    Template="forget_pass_email.tpl" />

<Template   Name="SystemInternalError"
                    Title="System Internal Error"
                    EmailAccount="SystemNotifier"
                    Template="system_internal_error.tpl" />

<Template   Name="CronjobEmail"
                    Title="Execute cron job"
                    EmailAccount="SystemNotifier"
                    Template="execute_cron_job.tpl" />
```

Public API

- Function: UserWelcomeEmail($user_id). This function is to send user welcome email.
  - Input: $user_id as integer.
  - Output: void
- Function: UserResetPassword($token_id). This function is to send reset password email.
  - Input: $token_id as integer. When user requests password reset, Cubi generates a token record in token table. It will be used to validate the url when user clicks the reset link in the email.
  - Output: void
- Function: SystemInternalErrorEmail($recipient, $errMsg). This function is to send system error email to given recipients who are usually IT operation people.
  - Input: $recipient as string. recipients' emails.
  - Input: $errMsg as string.
  - Output: void
- Function: CronJobEmail($recipientEmail, $job_name, $output). This function is to send cron job execution notification to given recipients.
  - Input: $recipientEmail as string. recipients' emails.
  - Input: $job_name as string.
  - Input: $output as string. Output content of the execution of the job.
  - Output: void
- Function: sendEmailNow($email_id). This function is to send email immediately.
  - Input: $email_id as integer. email id of email queue table.
  - Output: void
- Function: sendEmailFromQueue(). This function is to send emails from the email queue.
  - Input: void
  - Output: void

Dependency

- User email service depends on email service to deliver the email out. To configure email service (e.g. SMTP settings), please edit emailService.xml

Customization

- Developers can have their own user email service by creating a new class or extending on the current class
- To use new access service, please add define('USER_EMAIL_SERIVCE', 'userEmailService'); in cubi/bin/app_init.php

Next: CubiEmailModule