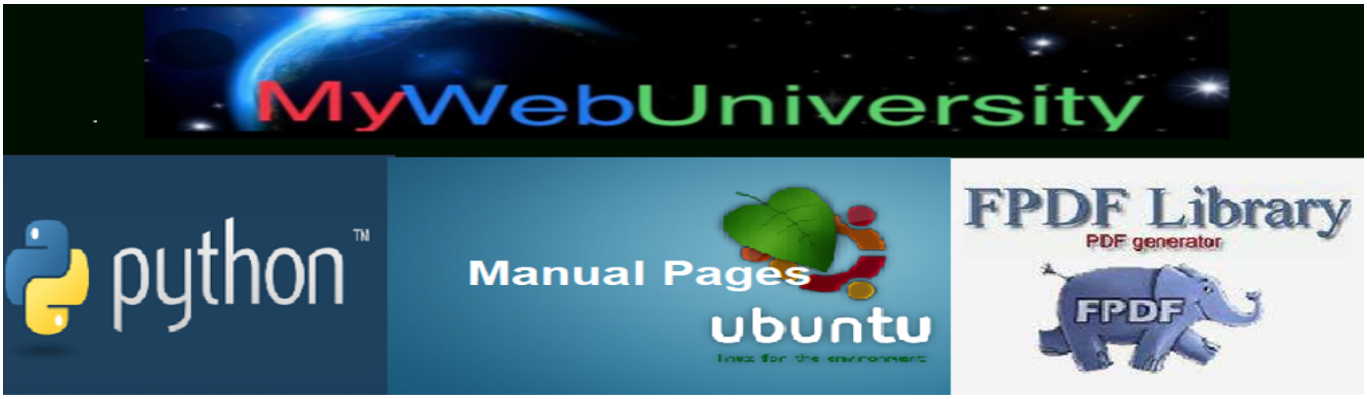

Rocky Enterprise Linux 9.2 Manual Pages on command 'Apache2::SiteControl::UserFactory.3pm'

$ man Apache2::SiteControl::UserFactory.3pm

Apache2::SiteControl::UserFactoUserpContributed Perl DocumeApache2::SiteControl::UserFactory(3pm)

NAME

Apache2::SiteControl::UserFactory - User factory/persistence

DESCRIPTION

This class is responsible for creating user objects (see Apache2::SiteControl::User) and managing the interfacing of those objects with a persistent session store. The default implementation uses Apache::Session::File to store the various attributes of the user to disk.

If you want to do your own user management, then you should leave the User class alone, and subclass only this factory. The following methods are required:

makeUser($$)

This method is called with the Apache Request object, username, password, and all other credential_# fields from the login form. It must create and return an instance of Apache2::SiteControl::User (using new...See User), and store that information (along with the session key stored in cookie format in the request) in some sort of permanent storage. This method is called in response to a login, so it should invalidate any existing session for the given user name (so that a user can be logged in only once). This method must return the key to use as the browser session key, or undef if it could not create the user.

findUser($$)

This method is passed the apache request and the session key (which you defined in makeUser). This method is called every time a "logged in" user makes a request. In other words the user objects are not persistent in memory (each request gets a new

"copy" of the state). This method uses the session key (which was stored in a browser cookie) to figure out what user to restore. The implementation is required to look up the user by the session key, recreate a Apache2::SiteControl::User object and return it. It must restore all user attributes that have been saved via saveAttribute (below).

invalidate($$)

This method is passed the apache request object and a previously created user object. It should delete the user object from permanent store so that future request to find that user fails unless makeUser has been called to recreate it. The session ID (which you made up in makeUser) is available from $user->{sessionid}.

saveAttribute($$$)

This method is automatically called whenever a user has a new attribute value. The incoming arguments are the apache request, the user object, and the name of the attribute to save (you can read it with $user->getAttribute($name)). This method must save the attribute in a such a way that later calls to findUser will be able to restore the attribute to the user object that is created. The session id you created for this user (in makeUser) is available in $user->{sessionid}.

Apache Config Directives

The following is a list of configuration variables that can be set with apache's PerlSetVar to configure the behavior of this class:

SiteControlDebug (default 0):

Debug mode

SiteControlLocks (default /tmp):

Where the locks are stored

SiteControlSessions (default /tmp):

Where the session data is stored

SiteControlUsermap (default none):

Where the usernames are mapped to session files. Required if you want multiple session detection. If unset a single userid can be used to log in multiple times simultaneously.

SiteControlUserFactory (default: Apache2::SiteControl::UserFactory)

An implementation like this module.

UserObjectSaveOtherCredentials (default: 0)

Indicates that other form data from the login screen (credential_2, credential_3, etc.)

should be saved in the session data. The keys will be credential_2, etc. name of the user factory to use when making user objects. These are useful if your web application has other login choices (i.e. service, database, etc.) that you need to know about at login.

UserObjectSavePassword (default 0)

Indicates that the password should be saved in the local session data, so that it is available to other parts of the web app (and not just the auth system). This might be necessary if you are logging the user in and out of services on the back end (like in webmail and database apps).

UserObjectPasswordCipher (default CAST5)

The CBC cipher used for encrypting the user passwords in the session files (See Crypt::CBC for info on allowed ciphers...this value is passed directly to Crypt::CBC->new). If you are saving user passwords, they will be encrypted when stored in the apache session files. This gives a little bit of added security, and makes the apache config the only sensitive file (since that is where you configure the key itself) instead of every random session file that is laying around on disk.

There is a global variable in this package called $encryption_key, which will be used if this variable is not set. The suggested method is to set the encryption key during server startup using a random value (i.e. from /dev/random), so that all server forks will inherit the value.

UserObjectPasswordKey

The key to use for encryption of the passwords in the session files. See UserObjectPasswordCipher above.

SEE ALSO

Apache2::SiteControl::User, Apache::SiteControl::PermissionManager, Apache2::SiteControl::Rule, Apache::SiteControl

AUTHOR

This module was written by Tony Kay, <tkay@uoregon.edu>.

COPYRIGHT AND LICENSE