

# Using Blackboard Learn 9 Web Services - Part 4

## Using The User Web Service

### Introduction

Blackboard Learn 9.1 provides a User web service, which you can use to get information about one or more Blackboard users and create one or more new users in your Blackboard system. As of June 2011, there is a bug in the User web service that prevents using the service to update the information for a current Blackboard user. This tutorial will review how to use the User web service. You should review the previous tutorials (<http://www.brucephillips.name/blog/index.cfm/2011/6/8/Blackboard-Learn-91-Web-Services-Tutorials-and-Documentation>) in this series prior to reading this part.

### Example Application

You can download the example application from <http://www.brucephillips.name/blackboard/updateaddblackboarduser.zip>. The Eclipse project was created using Eclipse 3.5 and Maven 2. The program demonstrates how to get a Blackboard user's information if you know the username, how to create a new Blackboard user, and how to delete an existing user.

See the ReadMe.txt file for how to build and run the application using either Eclipse (with Maven plugin) or just with Maven.

For Eclipse, you'll need to be using Eclipse 3.5 or higher with the Maven 2 plugin. You can get Eclipse at <http://www.eclipse.org/>. Information about installing the Eclipse Maven 2 plugin is at <http://m2eclipse.sonatype.org/installing-m2eclipse.html>.

After importing the project into your Eclipse workspace, read the ReadMe.txt file to view how to configure the application, run the JUnit tests, and how to run the main class. Before running the program you'll need to set the values for the properties listed in the bbws.properties file. Those values are for the Blackboard installation that you want to use.

If you want to just use Maven to build and run the application see the project's ReadMe.txt file.

### Proxy Tool

To use the User web service, the proxy tool you use for logging in (see part 1 of this tutorial) must be authorized to access the User web service and its methods. The example application includes a class, RegisterProxyToolApp, that you may use to register a new proxy tool with the appropriate authorizations. Remember, after registering a new proxy tool, a Blackboard system administrator must authorize the tool.

### User Web Service Client Class

To make calls upon the User web service you'll need a User web service client class. In part 3 of this series I reviewed how to generate the web service client classes using Axis2 and the WSDL file provided by the web service. The example application includes the generated UserWSSStub class, which is the User web service client class. Recall that the client class includes the methods that enable interaction with the web service and the additional data types the web service requires.

### **User Web Service - Get Blackboard User**

The User web service has a getUser method that you can use to get information about one or more users. You can search on different user attributes such as user id or username to find specific users. In the example application, I want to find a Blackboard user with a specific username. In the example application, if you examine class BlackboardUserServiceImpl method getBlackboardUser you'll find this code:

```
UserFilter userFilter = new UserFilter();

userFilter.setFilterType(6);

userFilter.setName(new String [] {username} );

GetUser getUser = new GetUser();

getUser.setFilter(userFilter);
```

The above code first creates a UserFilter object that I use to specify how I want the User web service to search for the Blackboard user. I set the filter type to 6, as 6 is the value to search by username. You have to read the JavaDoc for the UserFilter class's setFilterType method to find the different integer values you can use to specify how the User service should search.

Note the argument to the setName method is an array of String objects whose values are the Blackboard username for the users I want to find. In the example application I just want to find a single user, so the array has just one element. If you want to find multiple users, you can have multiple usernames in the array used as the argument to setName.

The GetUser object above is used to setup all the criteria, including the filter criteria that we want the User web service's getUser method to use.

After creating the UserWSSStub (the User web service client class) and setting up it's security options, you'll find this code in method getBlackboardUser:

```
GetUserResponse getUserResponse = userWSSStub.getUser(getUser);

UserVO [] blackboardUsers = getUserResponse.get_return() ;
```

Processing the result of calling the getUser method provides an array of UserVO objects. The UserVO class is used to store the values about a specific Blackboard user. You can consult the JavaDoc for UserVO to see all the get methods you can call to get data about a Blackboard user (see [part 2](#) for how

to find the JavaDoc for the Blackboard web services). One thing about the UserVO that perplexed me at first was how to get the user's first and last name. That data, along with the user's email, address, and other information is stored in another object of type UserExtendedInfoVO. So to get the user's first name you would call: `userVO.getExtendedInfo().getGivenName()`.

### **User Web Service - Save New Blackboard User**

The User web service has a `saveUser` operation. You can use this method to save a new user in your Blackboard system. In class `BlackboardUserServiceImpl`, method `saveBlackboardUser` is this code

```
SaveUser saveUser = new SaveUser();

saveUser.addUser(blackboardUser);

SaveUserResponse saveUserResponse = userWSStub.saveUser(saveUser);

String [] userIds = saveUserResponse.get_return() ;
```

The `SaveUser` object is used to specify all the `UserVO` objects that we want to save as new users in Blackboard. You can use the `addUser` method to add each `UserVO` object one at a time or you can use the `setUser` method that takes an array of `UserVO` objects as an argument. The result of calling the User web service's `saveUser` method is a `String` array with each element having the user id value for the user added. If the user id value is null, then that user was not added to the Blackboard system.

### **User Web Service - Delete Blackboard User**

The user web service has a `deleteUser` operation. You can use this method to delete an existing user in your Blackboard system. In class `BlackboardUserServiceImpl`, method `deleteBlackboardUser` is this code

```
DeleteUser deleteUser = new DeleteUser();

deleteUser.setUserId( new String [] { blackboardUserFound.getId() } );

DeleteUserResponse deleteUserResponse = userWSStub.deleteUser(deleteUser);

String [] userIds = deleteUserResponse.get_return() ;
```

The `DeleteUser` object is used to specify which users we want to delete. You provide an array of `Strings`, where each element has the user id value for the user you want to delete. The result of calling the User web service's `deleteUser` method is a `String` array, where each element is the user id value for a user that Blackboard deleted. A null value indicates that no user was deleted.

### **User Web Service - Update Blackboard User**

The bug that was prevented using the User web service to update an existing Blackboard User was fixed as of Blackboard Learn 9.1 Service Pack 6. Prior to SP6, my test of using the User web service to update an existing Blackboard user's email address failed. Working with SP6, the test passes.

The `saveUser` method updates the existing user with the values of the provided `UserVO` object's instance fields. The example application `BlackboardUserServiceImpl` includes an `updateBlackboardUser` method that demonstrates how to use the User web service `saveUser` to update an existing Blackboard user's email address and middle name.

## **Summary**

The User web service provides operations that allow you to get information about one or more users out of your Blackboard system. You can also use it to create new users and to delete existing users. You can also use the User web service to update information for existing users.

Please email any questions to [Bruce@Phillips.name](mailto:Bruce@Phillips.name).