Eric Roberts
CS 106A

Handout #51
February 22, 2010

Assignment #5—The FacePamphlet Repository

As you write the FacePamphlet application, you will need to make extensive use of the **`FPRepository`** class, which manages the client side of the repository. As part of the assignment, we want you to learn about this class by reading its documentation, which is available on the assignments area of the course web site. To get you started, however, we've given you the method summary as Figure 1.

In many ways, the FacePamphlet repository acts like a giant **`HashMap`** that allows you to look up keys for users on a central, permanent database. The keys for the map are a formed by writing the user id of the owner and the name of the key separated by a dot. For example, to look up my full name, for example, you need to use the key

`eroberts.name`

Figure 1. Javadoc method summary of the FPRepository class

Constructor Summary

`FPRepository()` Creates a connection to the FacePamphlet repository.
`FPRepository(String id, String password)` Creates a connection to the FacePamphlet repository using the specified name and password.

Method Summary

void	**`acceptFriend(String id)`** Accepts a friend request from the user with the specified id.
void	**`closeConnection()`** Closes the connection to the server.
String	**`convertGImageToString(GImage image)`** Converts a `GImage` object to a string so that it can be stored in the repository.
GImage	**`convertStringToGImage(String str)`** Converts a string to a `GImage`.
String[]	**`getAllUsers()`** Returns an array all user ids currently in the repository.
String	**`getCurrentUserId()`** Returns the id for the current user.
String[]	**`getFriendRequests()`** Returns an array containing the ids of the users who have requested to be your friends.
String	**`getLastServerMessage()`** Gets the last status message returned by the server.
String[]	**`getMyFriends()`** Returns an array containing the ids of the users who are currently your friends.
String	**`getProperty(String key)`** Gets the value associated with `key` on the server.
boolean	**`isMyFriend(String id)`** Returns `true` if the specified id is a friend, and `false` otherwise.
void	**`rejectFriend(String id)`** Rejects a friend request from the user with the specified id.
void	**`requestFriend(String id)`** Extends a friend request to the specified person.
void	**`sendMessage(String msg, String id)`** Sends a message to the wall for the specified id.
void	**`setProperty(String key, String value)`** Sets the value associated with `key` on the server.

although you wouldn't ordinarily write it explicitly. The ids are usually stored in variables, and the names of the keys are defined as constants in the **FPConstants** interface. Thus, if the name of the user whose profile is being visited is stored in the variable **visitId**, you could look up this property by making the following call:

```
String name = repository.getProperty(visitId + "." + NAME_KEY);
```

If the period is missing from a key, the server assumes that it refers to the active user. Thus, you could change your password to *secret* by calling

```
repository.setProperty(PASSWORD_KEY, "secret");
```

All values in the repository map are strings, which means that non-string values need to be converted to strings in order to store them there. The most important type you need to represent in this way is the **GImage** type. Fortunately, the **FPRepository** class exports methods for converting a **GImage** to a string and back again, so all you need to do is read the documentation to figure out what methods you need and how to apply them.