```java
package ballmerpeak.turtlenet.client;

import ballmerpeak.turtlenet.shared.CommentDetails;
import ballmerpeak.turtlenet.shared.PostDetails;
import ballmerpeak.turtlenet.shared.Message;
import ballmerpeak.turtlenet.shared.Conversation;

import com.google.gwt.core.client.*;
import com.google.gwt.event.dom.client.*;
import com.google.gwt.user.client.ui.*;
import com.google.gwt.event.logical.shared.*;
import com.google.gwt.user.client.Window;
import com.google.gwt.user.client.rpc.AsyncCallback;
import com.google.gwt.dom.client.Style.FontWeight;
import com.google.gwt.dom.client.DivElement;
import com.google.gwt.dom.client.Document;
import com.google.gwt.user.client.Timer;
import com.google.gwt.user.client.Window;
import java.util.Date;

public class frontend implements EntryPoint, ClickListener {

    // Create remote service proxy to talk to the server-side Turtlenet service
    private final TurtlenetAsync turtlenet = GWT.create(Turtlenet.class);
    //private final TurtlenetAsync msgfactory = GWT.create(MessageFactory.class);
    public void onModuleLoad() {
        // Remove loading indicator from frontend.html
        DivElement loadingIndicator = DivElement.as(Document.get().getElementById("loading"));
        loadingIndicator.setInnerHTML("");

        /* Add handler for window closing */
        Window.addCloseHandler(new CloseHandler<Window>() {
                public void onClose(CloseEvent<Window> event) {
                    turtlenet.stopTN(new AsyncCallback<String>() {
                    public void onFailure(Throwable caught) {
                        //pretend nothing happened
                    }
                    public void onSuccess(String result) {
                        //bask in success
                    }
                });
            }
        });

        // Call method to load the initial login page
        login();
    }

    private String location = new String("");
    private String refreshID = new String("");

    // LOUISTODO May need to remove ' = new FlexTable()'
    private FlexTable loginPanel = new FlexTable();
    private void login() {
        location = "login";
        refreshID = "";
        RootPanel.get().clear();
        loginPanel = new FlexTable();
        loginPanel.clear();
        RootPanel.get().add(loginPanel);

        // Create login panel widgets
        final Button loginButton = new Button("Login");
        loginButton.addClickListener(this);
        final PasswordTextBox passwordInput = new PasswordTextBox();
        final Label passwordLabel = new Label();

        turtlenet.isFirstTime(new AsyncCallback<String>() {
                public void onFailure(Throwable caught) {
                    System.out.println("turtlenet.isFirstTime failed: " + caught);
                }
                public void onSuccess(String result) {
                    if(result.equals("true")) { //GWT can only return objects
                        passwordLabel.setText("Please choose a password:");
                        final PasswordTextBox passwordConfirmInput = new PasswordTextBox();
                        final Label passwordConfirmLabel = new Label("");
                        passwordConfirmLabel.setText("Confirm your password:");
                        final TextBox usernameInput = new TextBox();
                        final Label usernameLabel = new Label("");
                        usernameLabel.setText("Please choose a username:");

                        // Add widgets to login panel
                        loginPanel.setWidget(1, 1, usernameLabel);
                        loginPanel.setWidget(2, 1, usernameInput);
                        loginPanel.setWidget(3, 1, passwordLabel);
                        loginPanel.setWidget(4, 1, passwordInput);
                        loginPanel.setWidget(5, 1, passwordConfirmLabel);
                        loginPanel.setWidget(6, 1, passwordConfirmInput);
                        loginPanel.setWidget(7, 1, loginButton);

                        // Add click handler for button
                        loginButton.addClickHandler(new ClickHandler() {
                        public void onClick(ClickEvent event) {
                            passwordLabel.setText("Please choose a password:");
                            passwordLabel.getElement().getStyle().setProperty("color", "#000000");
                            passwordConfirmLabel.setText("Confirm your password");
                            passwordConfirmLabel.getElement().getStyle().setProperty("color", "#000000");
                            usernameLabel.setText("Please choose a username:");
                            usernameLabel.getElement().getStyle().setProperty("color", "#000000");

                            if(usernameInput.getText().equals("")) {
                                usernameLabel.setText("Must enter a username");
                                usernameLabel.getElement().getStyle().setProperty("color", "#FFFFFF");
                            } else if(passwordInput.getText().equals("")) {
                                passwordLabel.setText("Must enter a password");
                                passwordLabel.getElement().getStyle().setProperty("color", "#FFFFFF");
                            } else if(passwordConfirmInput.getText().equals("")) {
                                passwordConfirmLabel.setText("Must confirm password");
                                passwordConfirmLabel.getElement().getStyle().setProperty("color", "#FFFFFF");
                            } else if(passwordInput.getText().equals(passwordConfirmInput.getText())) {
                                turtlenet.register(usernameInput.getText(), passwordInput.getText(), new
AsyncCallback<String>() {
                                    public void onFailure(Throwable caught) {
                                        System.out.println("turtlenet.register failed: " + caught);
                                    }
                                    public void onSuccess(String result) {
                                        if (result.equals("success")) {
                                            turtlenet.getMyKey(new AsyncCallback<String>() {
                                                public void onFailure(Throwable caught) {
                                                    System.out.println("turtlenet.getMyKey failed: " + caught);
                                                }
                                                public void onSuccess(String result) {
                                                    wall(result, false);
                                                }
                                            });
                                        } else if (result.equals("taken")) {
                                            usernameLabel.setText("Username already taken. Try again:");
                                            usernameLabel.getElement().getStyle().setProperty("color", "#FFFFFF");
                                        } else {
                                            System.out.println("turtlenet.register onSucess String result did not equal
success or taken");
                                        }
                                    }
                                });
                            } else {
                                passwordLabel.setText("Passwords do not match. Try again:");
                                passwordLabel.getElement().getStyle().setProperty("color", "#FFFFFF");
                                passwordConfirmInput.setText("");
                                passwordInput.setText("");
                            }
                        }
                        });

                    } else {
                        passwordLabel.setText("Please enter your password:");

                        // Add widgets to login panel
                        loginPanel.setWidget(1, 1, passwordLabel);
                        loginPanel.setWidget(2, 1, passwordInput);
                        loginPanel.setWidget(3, 1, loginButton);

                        // Add click handler for button
                        loginButton.addClickHandler(new ClickHandler() {
                        public void onClick(ClickEvent event) {
                            passwordLabel.setText("Please enter your password:");

                            turtlenet.startTN(passwordInput.getText(), new AsyncCallback<String>() {
                                public void onFailure(Throwable caught) {
                                    System.out.println("turtlenet.startTN failed: " + caught);
                                }
                                public void onSuccess(String result) {
                                    if (result.equals("success")) {
                                        turtlenet.getMyKey(new AsyncCallback<String>() {
                                            public void onFailure(Throwable caught) {
                                                System.out.println("turtlenet.getMyKey failed: " + caught);
                                            }
                                            public void onSuccess(String result) {
                                                wall(result, false);
                                            }
                                        });
                                    } else if (result.equals("failure")) {
                                        passwordLabel.setText("Password incorrect. Try again: ");
                                    } else {
                                        System.out.println("turtlenet.startTN onSuccess String does not equal success or
failure");
                                        passwordLabel.setText("INVALID RESPONSE FROM TNClient");
                                    }
                                }
                            });
                        }
                    });
                }
            }
        });

        // Add style name for CSS
```

```java
        loginPanel.addStyleName("gwt-login");
    }

    // Used to track the most recent wall post to be displayed
    Long wallLastTimeStamp = 0L;
    Long conversationLastTimeStamp = 0L;
    Long commentsLastTimeStamp = 0L;

    // When the login button is clicked we start a repeating timer that refreshes
    // the page every 5 seconds.
    public void onClick(Widget sender) {
        Timer refresh = new Timer() {
            public void run() {
                if(location.equals("wall")) {
                    turtlenet.timeMostRecentWallPost(refreshID, new AsyncCallback<Long>() {
                        public void onFailure(Throwable caught) {
                            System.out.println("turtlenet.timeMostRecentWallPost failed: " + caught);
                        }
                        public void onSuccess(Long result) {
                            if(result > wallLastTimeStamp) {
                                System.out.println("Refreshing wall. refreshID: " + refreshID);
                                wall(refreshID, true);
                            }
                        }
                    });
                } else if(location.equals("conversationList")) {
                    System.out.println("Refreshing conversationList");
                    conversationList();
                } else if(location.equals("conversation")) {
                    turtlenet.getConvoLastUpdated(refreshID, new AsyncCallback<Long>() {
                        public void onFailure(Throwable caught) {
                            //TODO Error
                        }
                        public void onSuccess(Long result) {
                            if(result > conversationLastTimeStamp) {
                                System.out.println("Refreshing conversation. refreshID: " + refreshID);
                                conversation(refreshID, true);
                            }
                        }
                    });
                } else {
                    //Do nothing
                }
            }
        };
        refresh.scheduleRepeating(5*1000);
    }

    private void navigation() {
        HorizontalPanel navigationPanel = new HorizontalPanel();
        RootPanel.get().add(navigationPanel);

        // Create navigation links
        Anchor linkMyWall = new Anchor("My Wall");
        linkMyWall.getElement().getStyle().setProperty("paddingLeft" , "100px");
        Anchor linkMyDetails = new Anchor("My Details");
        linkMyDetails.getElement().getStyle().setProperty("paddingLeft" , "100px");
        Anchor linkConversations = new Anchor("Messages");
        linkConversations.getElement().getStyle().setProperty("paddingLeft" , "100px");
        Anchor linkFriends = new Anchor("Friends");
        linkFriends.getElement().getStyle().setProperty("paddingLeft" , "100px");
        Anchor linkLogout = new Anchor("Logout");
        linkLogout.getElement().getStyle().setProperty("paddingLeft" , "100px");

        // Add links to navigation panel
        navigationPanel.add(linkMyWall);
        navigationPanel.add(linkMyDetails);
        navigationPanel.add(linkConversations);
        navigationPanel.add(linkFriends);
        navigationPanel.add(linkLogout);

        // Add style name for CSS
        navigationPanel.addStyleName("gwt-navigation");

        // Add click handlers for anchors
        linkMyWall.addClickHandler(new ClickHandler() {
            public void onClick(ClickEvent event) {
                turtlenet.getMyKey(new AsyncCallback<String>() {
                    public void onFailure(Throwable caught) {
                        System.out.println("turtlenet.getMyKey failed: " + caught);
                    }
                    public void onSuccess(String result) {
                        wall(result, false);
                    }
                });
            }
        });

        // Add click handlers for anchors
        linkMyDetails.addClickHandler(new ClickHandler() {
            public void onClick(ClickEvent event) {
                myDetails();
            }
        });

        linkConversations.addClickHandler(new ClickHandler() {
            public void onClick(ClickEvent event) {
                conversationList();
System.out.println("Wake up, Neo...");
            }
        });

        linkFriends.addClickHandler(new ClickHandler() {
            public void onClick(ClickEvent event) {
                friendsList("All");
            }
        });

        linkLogout.addClickHandler(new ClickHandler() {
            public void onClick(ClickEvent event) {
                turtlenet.stopTN(new AsyncCallback<String>() {
                    public void onFailure(Throwable caught) {
                        System.out.println("turtlenet.stopTN failed: " + caught);
                    }
                    public void onSuccess(String result) {
                        login();
                    }
                });
            }
        });
    }

    String[][] friendsListCategoryMembers = new String[0][0];
    String[][] friendsListCategoryList = new String[0][0];
    private TextBox friendsListPanel_myKeyTextBox;
    private void friendsList(final String currentGroupID) {
        location = "friendsList";
        refreshID = "";

        RootPanel.get().clear();
        navigation();
        final FlexTable friendsListPanel = new FlexTable();
        RootPanel.get().add(friendsListPanel);

        // Column title for anchors linking to messages
        Label friendsNameLabel = new Label("Friend's Name");
        friendsNameLabel.getElement().getStyle().setFontWeight(FontWeight.BOLD);
        friendsNameLabel.getElement().getStyle().setProperty("paddingLeft" , "100px");
        friendsListPanel.setWidget(1, 0, friendsNameLabel);

        // Column title for labels outputing the date a message was recieved
        Label friendsKeyLabel = new Label("Friend's Public Key");
        friendsKeyLabel.getElement().getStyle().setFontWeight(FontWeight.BOLD);
        friendsListPanel.setWidget(1, 1, friendsKeyLabel);

        turtlenet.getCategoryMembers(currentGroupID, new AsyncCallback<String[][]>() {
            int i;
            public void onFailure(Throwable caught) {
                System.out.println("turtlenet.getCategoryMembers failed: " + caught);
            }
            public void onSuccess(String[][] _result) {
                friendsListCategoryMembers = _result;
                for (i = 0; i < friendsListCategoryMembers.length; i++) {
                    //list names/keys
                    Anchor linkFriendsWall = new Anchor(friendsListCategoryMembers[i][0]);
                    linkFriendsWall.getElement().getStyle().setProperty("paddingLeft" , "100px");
                    friendsListPanel.setWidget((i + 2), 0, linkFriendsWall);
                    final String resultString = friendsListCategoryMembers[i][1];
                    TextBox friendKeyBox = new TextBox();
                    friendKeyBox.setText(resultString);
                    friendKeyBox.setVisibleLength(75);
                    friendKeyBox.setReadOnly(true);
                    friendsListPanel.setWidget((i + 2), 1, friendKeyBox);
                    //link names to walls
                    System.out.println("adding link to " + friendsListCategoryMembers[i][0] + "'s wall");
                    final String fkey = friendsListCategoryMembers[i][1];
                    linkFriendsWall.addClickHandler(new ClickHandler() {
                        public void onClick(ClickEvent event) {
                            wall(fkey, false);
                        }
                    });
                }
            }
        });

        int row = friendsListPanel.getRowCount() + 2;

        if(!currentGroupID.equals("All")) {
            Label currentGroupLabel = new Label(currentGroupID);
            friendsListPanel.setWidget((row - 1), 3, currentGroupLabel);
        }

        final ListBox currentGroups = new ListBox();
        currentGroups.setVisibleItemCount(1);
        currentGroups.setWidth("150px");
        currentGroups.addItem("All");
        friendsListPanel.setWidget(3, 3, currentGroups);

        turtlenet.getCategories(new AsyncCallback<String[][]>() {
            int i;
            public void onFailure(Throwable caught) {
                System.out.println("turtlenet.getCategories failed: " + caught);
            }
            int selected;
            public void onSuccess(String[][] _result) {
                friendsListCategoryList = _result;
                for (i = 0; i < friendsListCategoryList.length; i++) {
                    currentGroups.addItem(friendsListCategoryList[i][0]);
                    // Check if the group we've just added is the current group
                    // If it is note the index using selected. We need to add
                    // 1 to selected as "All" always appears first in the list.
                    if(friendsListCategoryList[i][0].equals(currentGroupID)) {
                        selected = (i + 1);
                    }
                }
                // Use selected to set the selected item in the listbox to the
                // current group
                currentGroups.setSelectedIndex(selected);

                currentGroups.addChangeHandler(new ChangeHandler() {
                    public void onChange(ChangeEvent event) {
                        friendsList(currentGroups.getItemText(currentGroups.getSelectedIndex()));
                    }
                });
            }
        });

        Button newGroup = new Button("Add new category");
        friendsListPanel.setWidget(2, 3, newGroup);
        newGroup.addClickHandler(new ClickHandler() {
            public void onClick(ClickEvent event) {
                newGroup();
            }
        });

        friendsListPanel_myKeyTextBox = new TextBox();
        friendsListPanel_myKeyTextBox.setWidth("480px");
        friendsListPanel_myKeyTextBox.setReadOnly(true);

        turtlenet.getMyKey(new AsyncCallback<String>() {
            public void onFailure(Throwable caught) {
                System.out.println("turtlenet.getMyKey failed: " + caught);
            }
            public void onSuccess(String result) {
                friendsListPanel_myKeyTextBox.setText(result);
            }
        });

        Label myKeyLabel = new Label("My key: ");
        myKeyLabel.getElement().getStyle().setFontWeight(FontWeight.BOLD);
        myKeyLabel.getElement().getStyle().setProperty("paddingLeft" , "100px");
        friendsListPanel.setWidget((row - 1), 0, myKeyLabel);
        friendsListPanel.setWidget((row - 1), 1, friendsListPanel_myKeyTextBox);

        if(currentGroupID.equals("All")) {
            Button addFriend = new Button("Add new friend");
            friendsListPanel.setWidget(1, 3, addFriend);
            addFriend.addClickHandler(new ClickHandler() {
                public void onClick(ClickEvent event) {
                    addFriend();
                }
            });
        } else {
            Button editGroup = new Button("Edit category");
            friendsListPanel.setWidget(1, 3, editGroup);
            editGroup.addClickHandler(new ClickHandler() {
                public void onClick(ClickEvent event) {
                    editGroup(currentGroupID);
                }
            });
        }

        // Add style name for CSS
        friendsListPanel.addStyleName("gwt-friends-list");
    }

    FlexTable conversationListPanel;
    private void conversationList() {
        location = "conversationList";
        refreshID = "";

        //Setup basic page
        RootPanel.get().clear();
        navigation();

        //Create panel to contain widgets
        conversationListPanel = new FlexTable();
        RootPanel.get().add(conversationListPanel);

        turtlenet.getConversations(new AsyncCallback<Conversation[]>() {
            Conversation[] result;
            public void onFailure(Throwable caught) {
                System.out.println("turtlenet.getConversations failed: " + caught);
            }
            public void onSuccess(Conversation[] _result) {
                result = _result;
                for(int j = 0; j < result.length; j++) {
                    System.out.println(result[j]);
                }
                System.out.println("result.length = " + result.length);
                for (int i = 0; i < result.length; i++) {
                    final String conversationID = result[i].signature;
                    // Substrings dont work if we set the end point so its
                    // bigger than our string. If the length is less than 40
                    // we output the full string. If the string is 40 or
                    // about we take the first 40 characters and add ...
                    String linkText = new String("");
                    if ((result[i].firstMessage).length() < 40) {
                        linkText = (result[i].firstMessage);
                    } else {
                        linkText = (result[i].firstMessage).substring(1, 40) + "...";
                    }
                    Anchor linkConversation = new Anchor(linkText);
                    conversationListPanel.setWidget(i, 0, linkConversation);

                    // Add click handlers for anchors
                    linkConversation.addClickHandler(new ClickHandler() {
                        public void onClick(ClickEvent event) {
                            conversation(conversationID, false);
                        }
                    });
                    Label conversationParticipants = new Label(result[i].concatNames());
                    conversationListPanel.setWidget(i, 1, conversationParticipants);
                }
                Button newConversation = new Button("New conversation");
                newConversation.setWidth("400px");
                newConversation.addClickHandler(new ClickHandler() {
                    public void onClick(ClickEvent event) {
                        newConversation();
                    }
                });

                conversationListPanel.setWidget((result.length + 1), 0, newConversation);
            }
        });

        // Add style name for CSS
        conversationListPanel.addStyleName("gwt-conversation-list");
    }

    private void myDetails() {
        location = "myDetails";
        refreshID = "";

        RootPanel.get().clear();
        navigation();
        FlexTable myDetailsPanel = new FlexTable();
        RootPanel.get().add(myDetailsPanel);

        // Create widgets relating to username
        Label usernameLabel = new Label("Username:");
        myDetailsPanel.setWidget(0, 0, usernameLabel);

        final TextBox editUsername = new TextBox();
        editUsername.setWidth("300px");
        turtlenet.getMyUsername(new AsyncCallback<String>() {
            public void onFailure(Throwable caught) {
                System.out.println("turtlenet.getMyUsername failed: " + caught);
            }
            public void onSuccess(String result) {
                editUsername.setText(result);
            }
        });

        myDetailsPanel.setWidget(0, 1, editUsername);

        Button saveUsername = new Button("Save Username");
        myDetailsPanel.setWidget(0, 2, saveUsername);

        final Label editUsernameLabel = new Label();
        editUsernameLabel.setWidth("200px");
        myDetailsPanel.setWidget(0, 3, editUsernameLabel);

        saveUsername.addClickHandler(new ClickHandler() {
            public void onClick(ClickEvent event) {
                turtlenet.claimUsername(editUsername.getText(), new AsyncCallback<String>() {
                     public void onFailure(Throwable caught) {
                         System.out.println("turtlenet.claimUsername failed: " + caught);
                     }
                     public void onSuccess(String result) {
                         if (result.equals("success")) {
                                editUsernameLabel.setText("Username saved");
                            } else if (result.equals("failure")) {
                                editUsernameLabel.setText("Username taken");
                            }
                        }
                    });
            }
        });

        // Create widgets relating to name
        Label nameLabel = new Label("Name:");
        myDetailsPanel.setWidget(1, 0, nameLabel);

        final TextBox editName = new TextBox();
        editName.setWidth("300px");
        turtlenet.getMyPDATA("name", new AsyncCallback<String>() {
            public void onFailure(Throwable caught) {
                System.out.println("turtlenet.getMyPDATA name failed: " + caught);
            }
            public void onSuccess(String result) {
                editName.setText(result);
            }
        });
        myDetailsPanel.setWidget(1, 1, editName);

        Button saveName = new Button("Save Name");
        myDetailsPanel.setWidget(1, 2, saveName);

        final Label editNameLabel = new Label();
        myDetailsPanel.setWidget(1, 3, editNameLabel);

        saveName.addClickHandler(new ClickHandler() {
            public void onClick(ClickEvent event) {
                turtlenet.updatePDATA("name", editName.getText(), new AsyncCallback<String>() {
                        public void onFailure(Throwable caught) {
                            System.out.println("turtlenet.updatePDATA name failed: " + caught);
                        }
                        public void onSuccess(String result) {
                            if (result.equals("success")) {
                                editNameLabel.setText("Name saved");
                            } else if (result.equals("failure")) {
                                // HORRIBLE FIX
                                //editNameLabel.setText("Failed to save name");
                                editNameLabel.setText("Name saved");
                            }
                        }
                    });
            }
        });

        // Create widgets relating to birthday
        Label birthdayLabel = new Label("Birthday:");
        myDetailsPanel.setWidget(2, 0, birthdayLabel);

        final TextBox editBirthday = new TextBox();
        editBirthday.setWidth("300px");
        turtlenet.getMyPDATA("birthday", new AsyncCallback<String>() {
            public void onFailure(Throwable caught) {
               System.out.println("turtlenet.getMyPDATA birthday failed: " + caught);
            }
            public void onSuccess(String result) {
                editBirthday.setText(result);
            }
        });
        myDetailsPanel.setWidget(2, 1, editBirthday);

        Button saveBirthday = new Button("Save Birthday");
        myDetailsPanel.setWidget(2, 2, saveBirthday);

        final Label editBirthdayLabel = new Label();
        myDetailsPanel.setWidget(2, 3, editBirthdayLabel);

        saveBirthday.addClickHandler(new ClickHandler() {
            public void onClick(ClickEvent event) {
                turtlenet.updatePDATA("birthday", editBirthday.getText(), new AsyncCallback<String>() {
                        public void onFailure(Throwable caught) {
                            System.out.println("turtlenet.updatePDATA birthday failed: " + caught);
                        }
                        public void onSuccess(String result) {
                            if (result.equals("success")) {
                                editBirthdayLabel.setText("Birthday saved");
                            } else if (result.equals("failure")) {
                               // HORRIBLE FIX
                               //editBirthdayLabel.setText("Failed to save birthday");
                               editBirthdayLabel.setText("Birthday saved");
                            }
                        }
                    });
            }
        });

        // Create widgets relating to gender
        Label genderLabel = new Label("Gender:");
        myDetailsPanel.setWidget(3, 0, genderLabel);

        final TextBox editGender = new TextBox();
        editGender.setWidth("300px");
        turtlenet.getMyPDATA("gender", new AsyncCallback<String>() {
            public void onFailure(Throwable caught) {
                System.out.println("turtlenet.getMyPDATA gender failed: " + caught);
            }
            public void onSuccess(String result) {
                editGender.setText(result);
            }
        });
        myDetailsPanel.setWidget(3, 1, editGender);

        Button saveGender = new Button("Save Gender");
        myDetailsPanel.setWidget(3, 2, saveGender);

        final Label editGenderLabel = new Label();
        myDetailsPanel.setWidget(3, 3, editGenderLabel);

        saveGender.addClickHandler(new ClickHandler() {
            public void onClick(ClickEvent event) {
                turtlenet.updatePDATA("gender", editGender.getText(), new AsyncCallback<String>() {
                     public void onFailure(Throwable caught) {
                         System.out.println("turtlenet.updatePDATA gender failed: " + caught);
                     }
                     public void onSuccess(String result) {
                         if (result.equals("success")) {
                             editGenderLabel.setText("Gender saved");
                         } else if (result.equals("failure")) {
                            // HORRIBLE FIX
                            //editGenderLabel.setText("Failed to save gender");
                            editGenderLabel.setText("Gender saved");
                         }
                     }
                 });
            }
        });

        // Create widgets relating to email
        final Label emailLabel = new Label("Email:");
        myDetailsPanel.setWidget(4, 0, emailLabel);

        final TextBox editEmail = new TextBox();
        editEmail.setWidth("300px");
        turtlenet.getMyPDATA("email", new AsyncCallback<String>() {
            public void onFailure(Throwable caught) {
                System.out.println("turtlenet.getMyPDATA email failed: " + caught);
            }
            public void onSuccess(String result) {
                editEmail.setText(result);
            }
        });
        myDetailsPanel.setWidget(4, 1, editEmail);

        Button saveEmail = new Button("Save Email");
        myDetailsPanel.setWidget(4, 2, saveEmail);

        final Label editEmailLabel = new Label();
        myDetailsPanel.setWidget(4, 3, editEmailLabel);

        saveEmail.addClickHandler(new ClickHandler() {
            public void onClick(ClickEvent event) {
                turtlenet.updatePDATA("email", editEmail.getText(), new AsyncCallback<String>() {
                     public void onFailure(Throwable caught) {
                         System.out.println("turtlenet.updatePDATA email failed: " + caught);
                     }
                     public void onSuccess(String result) {
                         if (result.equals("success")) {
                             editEmailLabel.setText("Email saved");
                         } else if (result.equals("failure")) {
                            // HORRIBLE FIX
                            //editEmailLabel.setText("Failed to save email");
                            editEmailLabel.setText("Email saved");
                         }
                     }
                 });
            }
        });

        Button revoke = new Button("Revoke Key");
        myDetailsPanel.setWidget(5, 1, revoke);
        revoke.getElement().getStyle().setProperty("color", "#FF0000");
        revoke.setWidth("310px");

        final Label editkeyRevokeLabel = new Label();
        myDetailsPanel.setWidget(5, 3, editkeyRevokeLabel);

        revoke.addClickHandler(new ClickHandler() {
            public void onClick(ClickEvent event) {
                turtlenet.revokeMyKey(new AsyncCallback<String>() {
                     public void onFailure(Throwable caught) {
                         System.out.println("turtlenet.revokeMyKey failed: " + caught);
```

```
                        }
                        public void onSuccess(String result) {
                            //if (result.equals("success")) {
                                editEmailLabel.setText("Key revoked");
                                login();
                            //} else if (result.equals("failure")) {
                                //editEmailLabel.setText("Failed to revoke key");
                            //}
                        }
                    });
                }
            });

            myDetailsPermissions();

            // Add style name for CSS
            myDetailsPanel.addStyleName("gwt-my-details");
        }

        private void myDetailsPermissions() {
            location = "myDetailsPermissions";
            refreshID = "";

            // Add panel to contain widgets
            final FlexTable myDetailsPermissionsPanel = new FlexTable();
            RootPanel.get().add(myDetailsPermissionsPanel);

            Label keyRevokeLabel = new Label("If you revoke your key your account will be deleted!");
            keyRevokeLabel.getElement().getStyle().setProperty("color", "#FF0000");
            myDetailsPermissionsPanel.setWidget(0, 0, keyRevokeLabel);

            Label myDetailsPermissionsLabel = new Label("Select which groups can view your details:");
            myDetailsPermissionsLabel.getElement().getStyle().setFontWeight(FontWeight.BOLD);
            myDetailsPermissionsPanel.setWidget(1, 0, myDetailsPermissionsLabel);

            turtlenet.getCategories(new AsyncCallback<String[][]>() {
                String[][] result;
                int i;
                public void onFailure(Throwable caught) {
                    System.out.println("turtlenet.getCategories failed: " + caught);
                }
                public void onSuccess(String[][] _result) {
                    result = _result;
                    for (i = 0; i < result.length; i++) {
                        final CheckBox groupCheckBox = new CheckBox(result[i][0]);
                        groupCheckBox.setValue(result[i][1].equals("true"));
                        myDetailsPermissionsPanel.setWidget((i + 1), 0, groupCheckBox);

                        groupCheckBox.addClickHandler(new ClickHandler() {
                            public void onClick(ClickEvent event) {
                                turtlenet.updatePDATApermission(groupCheckBox.getText(), groupCheckBox.getValue(), new
    AsyncCallback<String>() {
                                    public void onFailure(Throwable caught) {
                                        System.out.println("updatePDATApermission failed: " + caught);
                                    }
                                    public void onSuccess(String result) {
                                        //success
                                    }
                                });
                            }
                        });
                    }
                }
            });
            myDetailsPermissionsPanel.addStyleName("gwt-my-details-permissions");
        }

        private void friendsDetails(final String friendsDetailsKey, FlowPanel wallPanel, Button userDetails) {
            userDetails.addClickHandler(new ClickHandler() {
                public void onClick(ClickEvent event) {
                    wall(friendsDetailsKey, false);
                }
            });

            userDetails.setText("Reload page");
            userDetails.getElement().getStyle().setProperty("color", "#61B329");

            location = "friendsDetails";
            refreshID = "";

            // Create main panel
            final FlexTable friendsDetailsPanel = new FlexTable();
            wallPanel.insert(friendsDetailsPanel, 1);
            friendsDetailsPanel.clear();

            // Create widgets
            Label friendsDetailsUsernameTitle = new Label("Username:");
            friendsDetailsPanel.setWidget(0, 0, friendsDetailsUsernameTitle);

            Label friendsDetailsNameTitle = new Label("Name:");
            friendsDetailsPanel.setWidget(1, 0, friendsDetailsNameTitle);

            Label friendsDetailsBirthdayTitle = new Label("Birthday:");
```

```java
        friendsDetailsPanel.setWidget(2, 0, friendsDetailsBirthdayTitle);

        Label friendsDetailsGenderTitle = new Label("Gender:");
        friendsDetailsPanel.setWidget(3, 0, friendsDetailsGenderTitle);

        Label friendsDetailsEmailTitle = new Label("Email:");
        friendsDetailsPanel.setWidget(4, 0, friendsDetailsEmailTitle);

        Label friendsDetailsKeyTitle = new Label("Public Key:");
        friendsDetailsPanel.setWidget(5, 0, friendsDetailsKeyTitle);

        turtlenet.getUsername(friendsDetailsKey, new AsyncCallback<String>() {
            public void onFailure(Throwable caught) {
                System.out.println("turtlenet.getUsername failed: " + caught);
            }
            public void onSuccess(String result) {
                Label friendsDetailsUsernameLabel = new Label(result);
                friendsDetailsPanel.setWidget(0, 1, friendsDetailsUsernameLabel);
            }
        });

        turtlenet.getPDATA("name", friendsDetailsKey, new AsyncCallback<String>() {
            public void onFailure(Throwable caught) {
                System.out.println("turtlenet.getPDATA name failed: " + caught);
            }
            public void onSuccess(String result) {
                Label friendsDetailsNameLabel = new Label(result);
                friendsDetailsPanel.setWidget(1, 1, friendsDetailsNameLabel);
            }
        });

        turtlenet.getPDATA("birthday", friendsDetailsKey, new AsyncCallback<String>() {
            public void onFailure(Throwable caught) {
                System.out.println("turtlenet.getPDATA birthday failed: " + caught);
            }
            public void onSuccess(String result) {
                Label friendsDetailsBirthdayLabel = new Label(result);
                friendsDetailsPanel.setWidget(2, 1, friendsDetailsBirthdayLabel);
            }
        });

        turtlenet.getPDATA("gender", friendsDetailsKey, new AsyncCallback<String>() {
            public void onFailure(Throwable caught) {
                System.out.println("turtlenet.getPDATA gender failed: " + caught);
            }
            public void onSuccess(String result) {
                Label friendsDetailsGenderLabel = new Label(result);
                friendsDetailsPanel.setWidget(3, 1, friendsDetailsGenderLabel);
            }
        });

        turtlenet.getPDATA("email", friendsDetailsKey, new AsyncCallback<String>() {
            public void onFailure(Throwable caught) {
                System.out.println("turtlenet.getPDATA email failed: " + caught);
            }
            public void onSuccess(String result) {
                Label friendsDetailsEmailLabel = new Label(result);
                friendsDetailsPanel.setWidget(4, 1, friendsDetailsEmailLabel);
            }
        });

        TextBox friendsDetailsKeyBox = new TextBox();
        friendsDetailsKeyBox.setReadOnly(true);
        friendsDetailsKeyBox.setWidth("400px");
        friendsDetailsKeyBox.setText(friendsDetailsKey);
        friendsDetailsPanel.setWidget(5, 1, friendsDetailsKeyBox);

        turtlenet.getMyKey(new AsyncCallback<String>() {
            public void onFailure(Throwable caught) {
                System.out.println("turtlenet.getMyKey failed: " + caught);
            }
            public void onSuccess(String myKey) {
                if(friendsDetailsKey.equals(myKey)) {
                    Button edit = new Button("Edit my details");
                    edit.setWidth("410px");
                    friendsDetailsPanel.setWidget(6, 1, edit);
                    edit.addClickHandler(new ClickHandler () {
                        public void onClick(ClickEvent event) {
                            myDetails();
                        }
                    });
                }
            }
        });

        // Add style name for CSS
        friendsDetailsPanel.addStyleName("gwt-friends-details");
    }

    // Global stuff for wall
    private HorizontalPanel wallControlPanel = new HorizontalPanel();
    private TextArea postText;
    PostDetails[] wallPostDetails;
    int wallCurrentPost;
    private FlowPanel wallPanel = new FlowPanel();
    private Button wallControlPanelUserDetailsButton;
    private Anchor linkToComments;

    private void wall(final String key, final boolean refresh) {
        location = "wall";
        refreshID = key;

        wallPanel = new FlowPanel();
        wallPanel.clear();

        if(!refresh) {
            // Setup basic page
            RootPanel.get().clear();
            navigation();
            RootPanel.get().add(wallPanel);
            // Create a container for controls
            wallControlPanel = new HorizontalPanel();
            wallControlPanel.clear();
            wallControlPanel.addStyleName("gwt-wall-control");
            wallControlPanel.setSpacing(5);
            wallPanel.insert(wallControlPanel,  0);

            wallControlPanelUserDetailsButton = new Button("About");
            wallControlPanelUserDetailsButton.getElement().getStyle().setProperty("color", "#000000");
            wallControlPanel.add(wallControlPanelUserDetailsButton);
            wallControlPanelUserDetailsButton.getElement().getStyle().setProperty("marginRight" , "150px");
            wallControlPanelUserDetailsButton.addClickHandler(new ClickHandler() {
                public void onClick(ClickEvent event) {
                    friendsDetails(key, wallPanel, wallControlPanelUserDetailsButton);
                }
            });

            turtlenet.getMyKey(new AsyncCallback<String>() {
                public void onFailure(Throwable caught) {
                    System.out.println("turtlenet.getMyKey failed: " + caught);
                }
                public void onSuccess(String result) {
                    if(key.equals(result)) {
                        wallControlPanelUserDetailsButton.setText("About Me");
                    } else {
                        turtlenet.getUsername(key, new AsyncCallback<String>() {
                            public void onFailure(Throwable caught) {
                                System.out.println("turtlenet.getUsername failed: " + caught);
                            }
                            public void onSuccess(String result) {
                                wallControlPanelUserDetailsButton.setText("About " + result);
                            }
                        });
                    }
                }
            });

            final Button createPost = new Button("Write a post");
            createPost.getElement().getStyle().setProperty("color" , "#000000");
            wallControlPanel.add(createPost);

            final FlowPanel createPostPanel = new FlowPanel();
            createPostPanel.addStyleName("gwt-create-post");
            postText = new TextArea();
            postText.setCharacterWidth(80);
            postText.setVisibleLines(10);
            createPostPanel.add(postText);

            HorizontalPanel createPostControlPanel = new HorizontalPanel();
            createPostPanel.add(createPostControlPanel);

            final ListBox chooseGroup = new ListBox();
            chooseGroup.setVisibleItemCount(1);
            chooseGroup.setWidth("150px");
            chooseGroup.addItem("All");
            createPostControlPanel.add(chooseGroup);
            createPostControlPanel.setCellWidth(chooseGroup,"217px");


            turtlenet.getCategories(new AsyncCallback<String[][]>() {
                public void onFailure(Throwable caught) {
                    System.out.println("turtlenet.getCategories failed: " + caught);
                }
                public void onSuccess(String result[][]) {
                    for (int i = 0; i < result.length; i++)
                        chooseGroup.addItem(result[i][0]);
                }
            });

            Button cancel = new Button("Cancel");
            createPostControlPanel.add(cancel);
            createPostControlPanel.setCellWidth(cancel,"217px");
            cancel.addClickHandler(new ClickHandler() {
                public void onClick(ClickEvent event) {
                    wallPanel.remove(wallControlPanel);
                    wallPanel.remove(createPostPanel);
                    wall(key, false);
                }
            });

            Button send = new Button("Send");
            send.setWidth("150px");
            createPostControlPanel.add(send);
            send.addClickHandler(new ClickHandler() {
                public void onClick(ClickEvent event) {
                    turtlenet.addPost(key, chooseGroup.getItemText(chooseGroup.getSelectedIndex()), postText.getText(), new
AsyncCallback<String>() {
                        public void onFailure(Throwable caught) {
                            System.out.println("turtlenet.addPost failed: " + caught);
                        }
                        public void onSuccess(String result) {
                            //if (result.equals("success")) {
                                wallPanel.remove(wallControlPanel);
                                wallPanel.remove(createPostPanel);
                                wall(key, false);
                            //} else {
                                //System.out.println("turtlenet.addPost onSuccess String result did not equal success");
                            //}
                        }
                    });
                }
            });

            createPost.addClickHandler(new ClickHandler() {
                public void onClick(ClickEvent event) {
                    location = "createPost";
                    refreshID = "";
                    createPost.setText("Updates paused");
                    createPost.getElement().getStyle().setProperty("color" , "#FF0000");
                    wallPanel.insert(createPostPanel, 1);
                }
            });
        }

        turtlenet.getWallPosts(key, new AsyncCallback<PostDetails[]>() {
            public void onFailure(Throwable caught) {
                System.out.println("turtlenet.getWallPosts failed: " + caught);
            }
            public void onSuccess(PostDetails[] result) {
                wallPostDetails = result;
                for (wallCurrentPost = 0; wallCurrentPost < wallPostDetails.length; wallCurrentPost++) {
                    final PostDetails details = wallPostDetails[wallCurrentPost];

                    if(!refresh || wallPostDetails[wallCurrentPost].timestamp > wallLastTimeStamp) {
                        final FlowPanel postPanel = new FlowPanel();
                        postPanel.clear();
                        wallPanel.insert(postPanel, 1);
                        postPanel.addStyleName("gwt-post-panel");

                        HorizontalPanel postControlPanel = new HorizontalPanel();
                        postPanel.add(postControlPanel);

                        //Name
                        Label postedByLabel = new Label("Posted by: ");
                        postControlPanel.add(postedByLabel);
                        postControlPanel.setCellWidth(postedByLabel,"110");

                        Anchor linkToUser = new Anchor(wallPostDetails[wallCurrentPost].posterUsername);
                        postControlPanel.add(linkToUser);
                        postControlPanel.setCellWidth(linkToUser,"200");
                        linkToUser.addClickHandler(new ClickHandler() {
                            public void onClick(ClickEvent event) {
                                wall(wallPostDetails[wallCurrentPost].posterKey, false);
                            }
                        });

                        //Date
                        wallLastTimeStamp = wallPostDetails[wallCurrentPost].timestamp;
                        Label dateLabel = new Label(new Date(wallPostDetails[wallCurrentPost].timestamp).toString());
                        postControlPanel.add(dateLabel);

                        FlowPanel postContentsPanel = new FlowPanel();
                        postPanel.clear();
                        postPanel.add(postContentsPanel);

                        TextArea postContents = new TextArea();
                        postContents.setCharacterWidth(80);
                        postContents.setVisibleLines(5);
                        postContents.setReadOnly(true);

                        //Text
                        postContents.setText(wallPostDetails[wallCurrentPost].text);
                        postContentsPanel.add(postContents);

                        final HorizontalPanel postContentsFooterPanel = new HorizontalPanel();
                        postContentsFooterPanel.addStyleName("gwt-post-contents-footer");
                        postContentsPanel.add(postContentsFooterPanel);

                        //Like
                        Anchor likePost;

                        if (wallPostDetails[wallCurrentPost].liked) {
                            likePost = new Anchor("Unlike");
                            likePost.addClickHandler(new ClickHandler() {
                                public void onClick(ClickEvent event) {
                                    turtlenet.unlike(details.sig, new AsyncCallback<String>() {
                                        public void onFailure(Throwable caught) {
                                            System.out.println("turtlenet.unlike (post) failed: " + caught);
                                        }
                                        public void onSuccess(String _result) {
                                            //if (_result.equals("success")) {
                                                wall(key, false);
                                            //} else {
                                                //System.out.println("turtlenet.unlike (post) onSuccess String _result did not equal success");
                                            //}
                                        }
                                    });
                                }
                            });
                        } else {
                            likePost = new Anchor("Like");
                            likePost.addClickHandler(new ClickHandler() {
                                public void onClick(ClickEvent event) {
                                    turtlenet.like(details.sig, new AsyncCallback<String>() {
                                        public void onFailure(Throwable caught) {
                                            System.out.println("turtlenet.like (post) failed: " + caught);
                                        }
                                        public void onSuccess(String _result) {
                                            //if (_result.equals("success")) {
                                                wall(key, false);
                                            //} else {
                                                //System.out.println("turtlenet.like (post) onSuccess String _result did not equal success");
                                            //}
                                        }
                                    });
                                }
                            });
                        }
                        postContentsFooterPanel.add(likePost);
                        final Label stop = new Label("");

                        //Comments
                        int commentCount = wallPostDetails[wallCurrentPost].commentCount;
                        if(commentCount == 0) {
                            linkToComments = new Anchor("Add a comment");
                        } else {
                            linkToComments = new Anchor("Comments(" + Integer.toString(commentCount) + ")");
                        }

                        linkToComments.getElement().getStyle().setProperty("paddingRight" , "100px");
                        postContentsFooterPanel.add(linkToComments);
                        linkToComments.addClickHandler(new ClickHandler() {
                            public void onClick(ClickEvent event) {
                                postContentsFooterPanel.remove(linkToComments);
                                stop.setText("Page auto update paused");
                                stop.getElement().getStyle().setProperty("color" , "#FF0000");
                                comments(details.sig, key, false, postPanel);
                            }
                        });
                        postContentsFooterPanel.add(stop);
                        postContentsFooterPanel.add(likePost);
                        likePost.getElement().getStyle().setProperty("paddingLeft" , "270px");

                    }

                    if(refresh) {
                        // TODO LOUISTODO use this
                        //Window.scrollTo(0, (Window.getScrollTop() + 200));
                    }
                }
            }
        });

        // Add style name for CSS
        wallPanel.addStyleName("gwt-wall");
    }

    // Global stuff for comments
    private int commentCount;
    private TextArea threadReplyContents;
    private String keyOfWallCommentsAreOn = new String("");

    private void comments(final String postID, final String wallKey, final boolean refresh, FlowPanel postPanel) {
        final FlowPanel commentsPanel = new FlowPanel();
        location = "comments";
        refreshID = postID;
        keyOfWallCommentsAreOn = wallKey;

        if(!refresh) {
            commentsPanel.clear();
```

```java

                // Disables the comment anchor for the current post to prevent duplicate
                // comment panels being created.
                linkToComments.addClickHandler(new ClickHandler() {
                    public void onClick(ClickEvent event) {
                        commentsPanel.clear();
                    }
                });

                // Add main panel to page
                postPanel.add(commentsPanel);
                FlexTable commentsReplyThreadPanel = new FlexTable();
                commentsReplyThreadPanel.getElement().getStyle().setProperty("paddingLeft", "60px");
                commentsPanel.add(commentsReplyThreadPanel);

                threadReplyContents = new TextArea();
                threadReplyContents.setCharacterWidth(60);
                threadReplyContents.setVisibleLines(6);
                commentsReplyThreadPanel.setWidget(0, 0, threadReplyContents);

                Button cancel = new Button("Cancel");
                cancel.setWidth("450px");
                commentsReplyThreadPanel.setWidget(1, 0, cancel);
                cancel.addClickHandler(new ClickHandler() {
                    public void onClick(ClickEvent event) {
                        wall(wallKey, false);
                    }
                });

                Button replyToThread;
                if(commentCount == 0) {
                    replyToThread = new Button("Post comment");
                } else {
                    replyToThread = new Button("Reply to thread");
                }
                replyToThread.setWidth("450px");
                commentsReplyThreadPanel.setWidget(2, 0, replyToThread);

                replyToThread.addClickHandler(new ClickHandler() {
                    public void onClick(ClickEvent event) {
                        turtlenet.addComment(postID, threadReplyContents.getText(), new AsyncCallback<String>() {
                            public void onFailure(Throwable caught) {
                                System.out.println("turtlenet.addComment failed: " + caught);
                            }
                            public void onSuccess(String result) {
                                //if (result.equals("success")) {
                                    wall(wallKey, false);
                                //} else {
                                    //System.out.println("turtlenet.addComment onSuccess String result did not equal success");
                                //}
                            }
                        });
                    }
                });
            }

            turtlenet.getComments(postID, new AsyncCallback<CommentDetails[]>() {
                public void onFailure(Throwable caught) {
                    System.out.println("turtlenet.getComments failed: " + caught);
                }
                public void onSuccess(CommentDetails[] result) {
                    commentCount = result.length;
                    for (int i = 0; i < result.length; i++) {
                        if(!refresh || result[i].timestamp > commentsLastTimeStamp) {
                            final CommentDetails details = result[i];
                            // Create panel to contain the main contents of each comment
                            FlowPanel commentsContentsPanel = new FlowPanel();
                            commentsContentsPanel.addStyleName("gwt-comments-contents");
                            commentsPanel.insert(commentsContentsPanel, commentsPanel.getWidgetCount() - 1);

                            final String commentID = result[i].sig;
                            // Create widgets
                            TextArea commentContents = new TextArea();
                            commentContents.setCharacterWidth(60);
                            commentContents.setVisibleLines(3);
                            commentContents.setReadOnly(true);

                            //Text
                            commentContents.setText(result[i].text);
                            commentsContentsPanel.add(commentContents);

                            //Create panel to contain controls for each comment
                            HorizontalPanel commentsControlPanel = new HorizontalPanel();
                            commentsContentsPanel.add(commentsControlPanel);

                            final String postedByKey = result[i].posterKey;

                            Label commentPostedByLabel = new Label("Posted by: ");
                            commentPostedByLabel.getElement().getStyle().setProperty("paddingLeft" , "10px");
                            commentsControlPanel.add(commentPostedByLabel);

                            Anchor postedBy = new Anchor(result[i].posterName);
                            postedBy.getElement().getStyle().setProperty("paddingLeft" , "10px");
                            commentsControlPanel.add(postedBy);

                            postedBy.addClickHandler(new ClickHandler() {
                                public void onClick(ClickEvent event) {
                                    wall(details.posterKey, false);
                                }
                            });

                            Anchor likeComment;

                            if (result[i].liked) {
                                likeComment = new Anchor("Unlike");
                                likeComment.addClickHandler(new ClickHandler() {
                                    public void onClick(ClickEvent event) {
                                        turtlenet.unlike(details.sig, new AsyncCallback<String>() {
                                            public void onFailure(Throwable caught) {
                                                System.out.println("turtlenet.unlike (comment) failed: " + caught);
                                            }
                                            public void onSuccess(String _result) {
                                                //if (_result.equals("success")) {
                                                    wall(wallKey, false);
                                                //} else {
                                                    //System.out.println("turtlenet.unlike (comment) onSuccess String _result did not equal success");
                                                //}
                                            }
                                        });
                                    }
                                });
                            } else {
                                likeComment = new Anchor("Like");
                                likeComment.addClickHandler(new ClickHandler() {
                                    public void onClick(ClickEvent event) {
                                        turtlenet.like(details.sig, new AsyncCallback<String>() {
                                            public void onFailure(Throwable caught) {
                                                System.out.println("turtlenet.like (comment) failed: " + caught);
                                            }
                                            public void onSuccess(String _result) {
                                                //if (_result.equals("success")) {
                                                    wall(wallKey, false);
                                                //} else {
                                                    //System.out.println("turtlenet.like (comment) onSuccess String _result did not equal success");
                                                //}
                                            }
                                        });
                                    }
                                });
                            }

                            likeComment.getElement().getStyle().setProperty("paddingLeft" , "200px");
                            commentsControlPanel.add(likeComment);
                        }
                    }
                }
            });
            commentsPanel.addStyleName("gwt-comments");
        }

        //must be global because it must be referenced from callback
        private TextArea newConvoInput = new TextArea();
        private void newConversation() {
            location = "newConversation";
            refreshID = "";

            // Setup basic page
            RootPanel.get().clear();
            navigation();

            // Create panel to contain widgets
            final FlexTable newConversationPanel = new FlexTable();
            RootPanel.get().add(newConversationPanel);

            final ListBox currentFriends = new ListBox();
            currentFriends.setVisibleItemCount(11);
            currentFriends.setWidth("150px");
            newConversationPanel.setWidget(0, 0, currentFriends);

            newConvoInput.setCharacterWidth(80);
            newConvoInput.setVisibleLines(10);
            newConversationPanel.setWidget(0, 1, newConvoInput);

            final ListBox chooseFriend = new ListBox();
            chooseFriend.setWidth("150px");

            turtlenet.getPeople(new AsyncCallback<String[][]>() {
                String[][] result;
                String[] memberKeys;
                int i;
                public void onFailure(Throwable caught) {
                    System.out.println("turtlenet.getPeople failed: " + caught);
                }
                public void onSuccess(String[][] _result) {
                    result = _result;
                    for (i = 0; i < result.length; i++) {
                        //fill combo box
                        chooseFriend.addItem(result[i][0]);
                        String friendKey = (result[i][1]);
                        chooseFriend.setValue(i, friendKey);
                    }
                    chooseFriend.setVisibleItemCount(1);

                    FlexTable subPanel = new FlexTable();
                    newConversationPanel.setWidget(1, 1, subPanel);
                    subPanel.setWidget(1, 0, new Label("Choose a friend: "));
                    subPanel.setWidget(1, 1, chooseFriend);

                    Button addFriend = new Button("Add to the conversation");
                    subPanel.setWidget(1, 2, addFriend);
                    addFriend.addClickHandler(new ClickHandler() {
                        public void onClick(ClickEvent event) {
                            currentFriends.addItem(chooseFriend.getItemText(chooseFriend.getSelectedIndex()));
                            currentFriends.setValue((currentFriends.getItemCount() - 1), chooseFriend.getValue(chooseFriend.getSelectedIndex()));
                        }
                    });

                    Button send = new Button("Send");
                    newConversationPanel.setWidget(0, 2, send);

                    send.addClickHandler(new ClickHandler() {
                        String[] createChatReturn;
                        public void onClick(ClickEvent event) {
                            memberKeys = new String[currentFriends.getItemCount()+1];
                            for (int i = 0; i < currentFriends.getItemCount(); i++) {
                                memberKeys[i] = currentFriends.getValue(i);
                            }

                            turtlenet.getMyKey(new AsyncCallback<String>() {

                                public void onFailure(Throwable caught) {
                                    System.out.println("turtlenet.getMyKey failed: " + caught);
                                }
                                public void onSuccess(String userkey) {
                                    memberKeys[memberKeys.length-1] = userkey;
                                    turtlenet.createCHAT(memberKeys, new AsyncCallback<String[]>() {
                                        int i;
                                        public void onFailure(Throwable caught) {
                                            System.out.println("createCHAT failed: " + caught);
                                        }
                                        public void onSuccess(String[] _ret) {
                                            createChatReturn = _ret;
                                            if (createChatReturn[0].equals("success")) {
                                                turtlenet.addMessageToCHAT(newConvoInput.getText(), createChatReturn[1], new AsyncCallback<String>() {
                                                    public void onFailure(Throwable caught) {
                                                        System.out.println("turtlenet.addMessageToCHAT failed: " + caught);
                                                    }
                                                    public void onSuccess(String success) {
                                                        //if (success.equals("success")) {
                                                            conversation(createChatReturn[1], false);
                                                        //} else {
                                                            //System.out.println("turtlenet.addMessageToCHAT onSuccess String success did not equal success");
                                                        //}
                                                    }
                                                });
                                            } else {
                                                // THIS IS TEMPORARY!
                                                conversation(createChatReturn[1], false);
                                                //System.out.println("turtlenet.createCHAT onSuccess String createChatReturn[0] did not equal success");
                                            }
                                        }
                                    });
                                }
                            });
                        }
                    });
                }
            });

            // Add style name for CSS
            newConversationPanel.addStyleName("gwt-conversation");
        }

        // Global stuff for conversation
        private String convoPanelSetup_convosig; //needed in inner class
        private TextArea convoPanelSetup_input = new TextArea();
        private FlowPanel conversationPanel;

        private void conversation(final String conversationID, final boolean refresh) {
            location = "conversation";
            refreshID = conversationID;

            conversationPanel = new FlowPanel();
            final ListBox currentFriends = new ListBox();

            if(!refresh) {
                conversationPanel.clear();
                // Set up basic page
                RootPanel.get().clear();
                navigation();
                RootPanel.get().add(conversationPanel);
                HorizontalPanel conversationParticipantsPanel = new HorizontalPanel();
                conversationParticipantsPanel.setSpacing(5);
                conversationPanel.add(conversationParticipantsPanel);
                convoPanelSetup_convosig = conversationID;
                Label participantsLabel = new Label("Participants: ");
                participantsLabel.getElement().getStyle().setProperty("marginRight" , "20px");
                conversationParticipantsPanel.add(participantsLabel);

                currentFriends.setVisibleItemCount(1);
                currentFriends.setWidth("150px");
                conversationParticipantsPanel.add(currentFriends);
            }

            turtlenet.getConversation(convoPanelSetup_convosig, new AsyncCallback<Conversation>() {
                Conversation result;
                int i;
                public void onFailure(Throwable caught) {
                    System.out.println("turtlenet.getConversation failed: " + caught);
                }
                public void onSuccess(Conversation _result) {
                    result = _result;

                    if (!refresh) {
                        for (i = 0; i < result.users.length; i++) {
                            currentFriends.addItem(result.users[i]);
                        }
                    }

                    turtlenet.getConversationMessages(convoPanelSetup_convosig, new AsyncCallback<String[][]>() {
                        String[][] messages;
                        int i;
                        public void onFailure(Throwable caught) {
                            System.out.println("turtlenet.getConversationMessages failed: " + caught);
                        }
                        public void onSuccess(String[][] msgs) {
                            messages = msgs;

                            Button replyToConversation = new Button("Reply");
                            replyToConversation.setWidth("590px");

                            for (int i = 0; i < messages.length; i++) {
                                if(!refresh || Long.parseLong(msgs[i][1]) > conversationLastTimeStamp) {
                                    HorizontalPanel conversationContentsPanel = new HorizontalPanel();
                                    conversationContentsPanel.setSpacing(5);
                                    conversationPanel.add(conversationContentsPanel);
                                    Label postedBy = new Label(messages[i][0]);
                                    postedBy.getElement().getStyle().setProperty("marginRight" , "110px");
                                    postedBy.getElement().getStyle().setFontWeight(FontWeight.BOLD);

                                    // LOUISTODO This might not work
                                    conversationContentsPanel.add(postedBy);
                                    //conversationContentsPanel.insert(postedBy, conversationPanel.getWidgetIndex
    (replyToConversation));
                                    Label messageContents = new Label(messages[i][2]);
                                    conversationContentsPanel.add(messageContents);

                                    conversationLastTimeStamp = Long.parseLong(msgs[i][1]);
                                }
                            }

                            if(!refresh) {
                                conversationPanel.add(replyToConversation);
                                final FlowPanel conversationReplyPanel = new FlowPanel();
                                convoPanelSetup_input.setCharacterWidth(80);
                                convoPanelSetup_input.setVisibleLines(10);
                                conversationReplyPanel.add(convoPanelSetup_input);

                                HorizontalPanel conversationReplyControlsPanel = new HorizontalPanel();
                                conversationReplyPanel.add(conversationReplyControlsPanel);

                                Label stop = new Label("Page auto update paused");
                                stop.getElement().getStyle().setProperty("color" , "#FF0000");
                                stop.getElement().getStyle().setProperty("paddingRight" , "55px");
                                conversationReplyControlsPanel.add(stop);
                                stop.addClickHandler(new ClickHandler() {
                                    public void onClick(ClickEvent event) {
                                        conversation(conversationID, false);
                                    }
                                });

                                Button cancel = new Button("Cancel");
                                conversationReplyControlsPanel.add(cancel);

                                cancel.addClickHandler(new ClickHandler() {
                                    public void onClick(ClickEvent event) {
                                        conversation(conversationID, false);
                                    }
                                });

                                Button send = new Button("Send");
                                conversationReplyControlsPanel.add(send);
                                send.addClickHandler(new ClickHandler() {
                                    public void onClick(ClickEvent event) {
                                        turtlenet.addMessageToCHAT(convoPanelSetup_input.getText(), convoPanelSetup_convosig, new 
AsyncCallback<String>() {
                                            public void onFailure(Throwable caught) {
                                                System.out.println("turtlenet.addMessageToCHAT failed: " + caught);
                                            }
                                            public void onSuccess(String postingSuccess) {
                                                //Reload the conversation after the new message has been added
                                                conversation(convoPanelSetup_convosig, false);
                                            }
                                        });
                                    }
                                });

                                replyToConversation.addClickHandler(new ClickHandler() {
                                    public void onClick(ClickEvent event) {
                                        location = "replyToConversation";
                                        refreshID = "";

                                        conversationPanel.add(conversationReplyPanel);
                                    }
                                });
                            }
                        }
                    });
                }
            });

            // Add style name for CSS
            conversationPanel.addStyleName("gwt-conversation");
        }

        TextBox newGroup_nameInput = new TextBox();
        private void newGroup() {
            location = "newGroup";
            refreshID = "";

            RootPanel.get().clear();
            navigation();
            FlexTable newGroupPanel = new FlexTable();
            RootPanel.get().add(newGroupPanel);

            newGroupPanel.setWidget(0, 0, new Label("Category name: "));
            newGroupPanel.setWidget(0, 1, newGroup_nameInput);

            Button createGroup = new Button("Create category");
            newGroupPanel.setWidget(0, 2, createGroup);

            createGroup.addClickHandler(new ClickHandler() {
                public void onClick(ClickEvent event) {
                    turtlenet.addCategory(newGroup_nameInput.getText(), new AsyncCallback<String>() {
                        public void onFailure(Throwable caught) {
                            System.out.println("turtlenet.addCategory failed: " + caught);
                        }
                        public void onSuccess(String result) {
                            //if (result.equals("success")) {
                                editGroup(newGroup_nameInput.getText());
                            //} else {
                                //System.out.println("turtlenet.addCategory onSuccess String result did not equal success");
                            //}
                        }
                    });
                }
            });

            newGroupPanel.addStyleName("gwt-new-group");
        }

        private void editGroup(final String groupID) {
            location = "editGroup";
            refreshID = "";

            FlexTable editGroupPanel = new FlexTable();
            editGroupPanel.clear();
            RootPanel.get().add(editGroupPanel);

            editGroupPanel.setWidget(1, 0, new Label("Currently in category: "));
            final ListBox currentMembers = new ListBox();
            currentMembers.setVisibleItemCount(10);
            currentMembers.setWidth("150px");
            editGroupPanel.setWidget(1, 1, currentMembers);

            turtlenet.getCategoryMembers(groupID, new AsyncCallback<String[][]>() {
                String[][] result;
                int i;
                public void onFailure(Throwable caught) {
```

```java
                System.out.println("turtlenet.getCategoryMembers failed: " + caught);
            }
            public void onSuccess(String[][] _result) {
                result = _result;
                for (i = 0; i < result.length; i++) {
                    currentMembers.addItem(result[i][0]);
                    currentMembers.setValue(i, result[i][1]); //their key
                }
            }
        });

        Button removeFromGroup = new Button("Remove from group");
        editGroupPanel.setWidget(1, 2, removeFromGroup);
        removeFromGroup.addClickHandler(new ClickHandler() {
            public void onClick(ClickEvent event) {
                turtlenet.removeFromCategory(groupID, currentMembers.getValue(currentMembers.getSelectedIndex()), new
    AsyncCallback<String>() {
                    public void onFailure(Throwable caught) {
                        System.out.println("turtlenet.removeFromCategory failed: " + caught);
                    }
                    public void onSuccess(String result) {
                        friendsList(groupID);
                    }
                });
            }
        });

        editGroupPanel.setWidget(2, 0, new Label("Add a friend: "));
        final ListBox allFriends = new ListBox();
        allFriends.setVisibleItemCount(1);
        allFriends.setWidth("150px");
        editGroupPanel.setWidget(2, 1, allFriends);

        turtlenet.getPeople(new AsyncCallback<String[][]>() {
            String[][] result;
            int i;
            public void onFailure(Throwable caught) {
                System.out.println("turtlenet.getPeople failed: " + caught);
            }
            public void onSuccess(String[][] _result) {
                result = _result;
                for (i = 0; i < result.length; i++) {
                    String friendKey = new String(result[i][1]);
                    allFriends.addItem(result[i][0]);
                    allFriends.setValue(i, friendKey);
                }
            }
        });

        Button addFriend = new Button("Add friend");
        editGroupPanel.setWidget(2, 2, addFriend);
        addFriend.addClickHandler(new ClickHandler() {
            public void onClick(ClickEvent event) {
                turtlenet.addToCategory(groupID, allFriends.getValue(allFriends.getSelectedIndex()), new
    AsyncCallback<String>() {
                    public void onFailure(Throwable caught) {
                        System.out.println("turtlenet.addToCategory failed: " + caught);
                    }
                    public void onSuccess(String result) {
                        //if (result.equals("success")) {
                            friendsList(groupID);
                        //} else {
                            //System.out.println("turtlenet.addToCategory onSuccess String result did not equal success");
                        //}
                    }
                });
            }
        });

        editGroupPanel.addStyleName("gwt-edit-group");
    }

    TextBox addFriend_keyInput = new TextBox();
    private void addFriend() {
        location = "addFriend";
        refreshID = "";

        RootPanel.get().clear();
        navigation();
        FlexTable addFriendPanel = new FlexTable();
        RootPanel.get().add(addFriendPanel);

        addFriendPanel.setWidget(0, 0, new Label("Enter the key of the person you wish to add:"));
        addFriend_keyInput.setVisibleLength(100);
        addFriendPanel.setWidget(1, 0, addFriend_keyInput);

        Button submit = new Button("Add key");
        submit.setWidth("640px");
        addFriendPanel.setWidget(2, 0, submit);
        final Label success = new Label("");
        addFriendPanel.setWidget(3, 0, success);

        submit.addClickHandler(new ClickHandler() {
            public void onClick(ClickEvent event) {
                turtlenet.addKey(addFriend_keyInput.getText(), new AsyncCallback<String>() {
                    public void onFailure(Throwable caught) {
                        success.setText("Key could not be added");
                        System.out.println("turtlenet.addKey failed: " + caught);
                    }
                    public void onSuccess(String result) {
                        if (result.equals("success")) {
                                success.setText("Key has been added");
                        } else {
                            success.setText("Key could not be added");
                            System.out.println("turtlenet.addKey onSucess String result did not equal success");
                        }
                    }
                });
            }
        });
        addFriendPanel.addStyleName("gwt-friend");
    }
}
```