

EUI Portal Tutorial

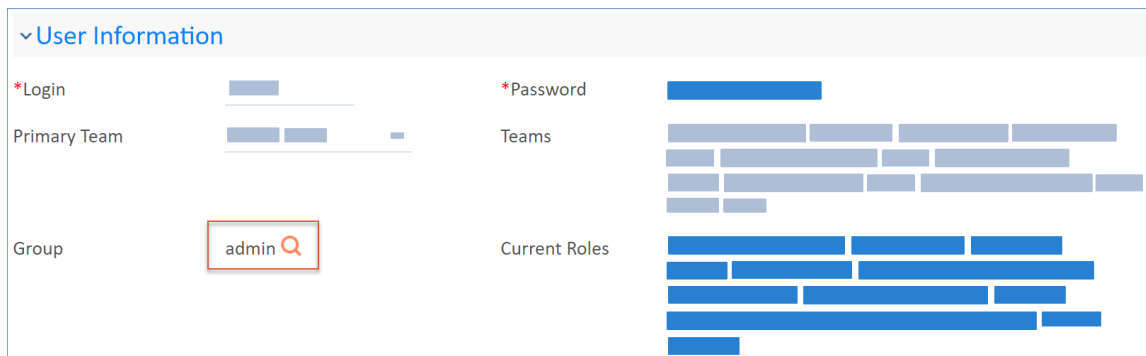
The following pages contain a tutorial aimed at getting you started with creating a portal from scratch. It assumes familiarity with the Agiloft application and user interface. Some basic HTML knowledge is useful, but not absolutely necessary.

The portal you'll create in the tutorial is for illustrative purposes only. The HTML is very basic and designed for simplicity rather than representing advanced web development techniques. However, it is fully functional and should give you the practice you need to carry on with confidence.

Prerequisites

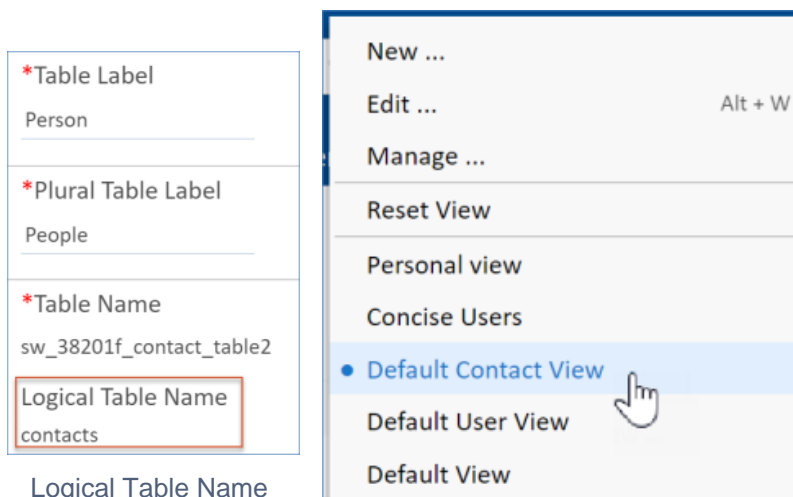
To successfully work through this tutorial, you need access to a knowledgebase (KB) with the following setup:

1. A user who is a member of the admin group.



The screenshot shows the 'User Information' form in Agiloft. The form is divided into two columns. The left column contains fields for 'Login', 'Primary Team', and 'Group'. The right column contains fields for 'Password', 'Teams', and 'Current Roles'. The 'Group' field is highlighted with a red box and contains the text 'admin' with a magnifying glass icon. The 'Teams' field contains a list of team names, and the 'Current Roles' field contains a list of roles.

2. A table with logical name `contacts`, usually labeled People, with a Default Contact View.



The screenshot shows two parts of the Agiloft interface. On the left is the 'Setup (Table)' General tab, which contains fields for 'Table Label' (Person), 'Plural Table Label' (People), 'Table Name' (sw_38201f_contact_table2), and 'Logical Table Name' (contacts). The 'Logical Table Name' field is highlighted with a red box. On the right is the 'Views' menu, which contains options for 'New ...', 'Edit ...', 'Manage ...', 'Reset View', 'Personal view', 'Concise Users', 'Default Contact View', 'Default User View', and 'Default View'. The 'Default Contact View' option is highlighted with a blue background and a mouse cursor is pointing at it.

Logical Table Name
located in Setup (Table)
General tab

Default Contact View in People table
under Views menu

3. A table with a logical name of `case`, usually labeled Support Cases.

*Table Label Support Case
*Plural Table Label Support Cases
*Table Name sw_98264f_case <div style="border: 1px solid red; padding: 2px;">Logical Table Name case</div>

Logical Table Name in
Setup (Table) General
tab

4. The EUI Templates table is visible on the navigation menu. If it isn't:
- Grant access to your admin user. Go to **Setup > Access > Mange Groups** and edit the admin group, then go to Table tab, edit the EUI Templates table, and set "Allow access to the table?" and "Show table on the Toolbar?" to Yes.
 - Unhide it in the **Setup > Tables** menu by selecting the EUI Templates table and clicking Unhide.
 - Add it to the navigation menu from **User Menu > Preferences > Navigation Menu Setup**.



EUI Templates table in nav bar

From this point forward, we will refer to your Agiloft server host as `{host}` and your knowledgebase as `{kbName}`. Take a look at the URL in your browser after logging in to your KB: the server host is in the first part of the URL.



- For example, for the URL:

`https://al.server.com/gui2/admin/main.jsp;jsessionid=E5 ...`

... the `{host}` is

`https://al.server.com`

- The KB name is displayed in the top-right corner of the screen after logging in. In the example below, the `{kbName}` is Demo.








Creating a Simple Web Page

First, create a dummy user to act as the end user who will access the portal pages you create:

1. Log in as admin and create a new employee with a login of `portalUser` and a password of `test01`. You can enter any Name and Company you want.
2. Set the user to be a member of a group with permission to view all the entries in the People table.

Next, you will create a very simple home page for the new user to access via the portal:

1. Select the EUI Templates table.

EUI Templates			
All EUI Templates ▼			
81 record(s) found, 3 page(s). Click here to count records again.			
<input type="checkbox"/>	Edit	ID	Name
<input type="checkbox"/>		95	public_sourcingevents.html
<input type="checkbox"/>		94	invited_sourcingevents.html
<input type="checkbox"/>		93	new_vendorresponse.html
<input type="checkbox"/>		92	my_vendorresponses.html
<input type="checkbox"/>		91	my_vendorresponses-search.htm

EUI Templates table

2. Create a new EUI template record with the following information:
 - a. In the Name field: `testContacts.htm`
 - b. In the Description field: Test contacts page
 - c. In the Body field, enter the following:

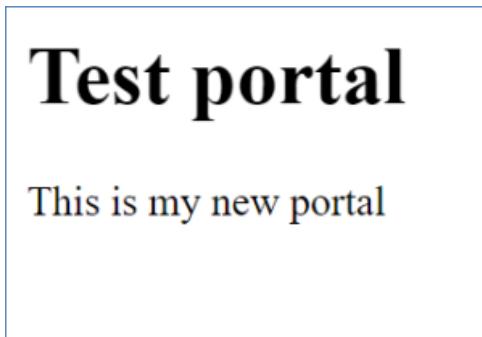
```
<html>
<head><title>Test portal</title></head>
<body>
<h1>Test portal</h1>
<div align="left">This is my new portal</div>
</body>
</html>
```

- d. Click Save when you are finished.

Now, test your page:

1. Open a new browser window and enter the following URL, replacing `{host}` and `{kbName}`:
`{host}/gui2/eui2template/testContacts.htm?kb={kbName}`
`&user=portalUser&passwd=test01`

2. You should see the following:



Test portal page result

Congratulations! Even though the page doesn't look like much, you have accomplished the first step toward creating your custom End User Interface.

Adding More Content

Now that you have a functioning web page, you can start to make it more useful by adding additional features, such as table views and links.

Next, add a view of the Contacts table to this page.

1. While logged in as the admin user, navigate to the EUI Templates table.
2. Edit the `testContacts.htm` template record that you created earlier.
3. Select the text in the Body field and replace it with the code below. Notice this code adds a new `<div>` and the `#ew_table` macro.

```
<html>
<head><title>Test portal</title></head>
<body>
<h1>Test portal</h1>
<div align="left">This is my new portal</div>
<br>
<div align="left"><font size="+2">People</font></div>
#ew_table ("contacts" "Default Contact View" "" "" "" "contacts_frame" "")
</body>
</html>
```

4. Save the template record.

Now, test the page again:

1. Open a new browser window and enter the same URL you used earlier, replacing `{host}` and `{kbName}`:
`{host}/gui2/eui2template/testContacts.htm?kb={kbName}`
`&user=portalUser&passwd=test01`

2. This time you should see something that looks like the following:

Test portal

This is my new portal

People

People

Status: 61 record(s) found, 2 page(s) [Click for details....](#)

Page: [Page 1] [Next](#) [Last](#) Go to page [Go](#)

New Mass Edit Save Changes Cancel Changes Actions

Views Search

<input type="checkbox"/>	Edit	ID <input type="button" value="v"/>	Full Name	Primary Team	Company	Direct Phone <input type="button" value="v"/>
<input type="checkbox"/>	<input type="button" value="v"/>	476	Christopher Lee	Contract Creator Team	Agiloft	123-456-7890
<input type="checkbox"/>	<input type="button" value="v"/>	475	Johnson Jones	Customer Team	IBM	650-344-3232 Ext.125
<input type="checkbox"/>	<input type="button" value="v"/>	474	Sam Smith	Vendor Team	IBM	123-444-5555
<input type="checkbox"/>	<input type="button" value="v"/>	471	Evelyn Wambaugh	Customer Team	AT&T	508-555-1212
<input type="checkbox"/>	<input type="button" value="v"/>	470	Innocent DiSabato	Customer Team	ISC Design Optimizer	451-555-1212

Test portal text above embedded People table

Try out the table. You can view and edit records, apply views, use saved searches—anything that the user's group permissions in Agiloft allows. You have created a fully functional portal into the Contacts table.

How It Works

Before going any further, it is worth taking a step back and understanding how all this works. It will make what follows easier to understand. If you are experienced with HTML and are reading this guide for specific help with the Agiloft setup, you can skip to the [EUI Tips](#) and [EUI Macro Reference](#).

What's in a URL?

A URL is an address. It specifies all the information needed to locate something on the internet. In this case, the URL tells your web browser how to locate the page you are constructing.

Take a look at the URL we used: `{host}/gui2/eui2template/testContacts.htm?kb={kbName}&user=portalUser&passwd=test01`

- `{host}/gui2/eui2template/testContacts.htm` identifies the physical server machine on which the Agiloft software is installed, and specifies the name of the page template to retrieve from the EUI Templates table.
- `?kb={kbName}&user=portalUser&passwd=test01` specifies which Agiloft knowledgebase the page template belongs to, and supplies the user's login credentials to validate and authorize the page request.

When you submit the URL, you might notice it changes as the system logs you in. Once the system verifies the credentials in the URL, it creates a user session on the server. From that point on, the web browser can refer to the session ID instead of resubmitting your credentials. When you stop using Agiloft, or manually log out of the system, the session is closed and the ID is no longer usable. So, if you want to save a URL for future use, save the URL you enter initially, rather than the URL containing the session ID.

Putting it all together

When you enter the URL into the web browser, a few things happen:

1. The server looks up the page template in the database and retrieves it.
2. The page template is passed through the Velocity template engine for macro expansion. In the example EUI Template above, `#ew_table(. . .)` is a macro that adds a table to a page. We will explain how that happens in more detail a little later on.
3. Once all the macros have been expanded, the resulting HTML page is sent back to the web browser for display.

Note: The HTML on the page may contain other links (URLs) for page elements, e.g., images. In that case, the web browser will separately request all the links and assemble the results into the finished page.

A Closer Look

Let's take a closer look at the sample portal template to see how the theory works in practice. Here is the page template `testContacts.htm` from earlier, with line numbers added for ease of reference:

```
1 <html>
2 <head><title>Test portal</title></head>
3 <body>
4 <h1>Test portal</h1>
5 <div align="left">This is my new portal</div>
6 <br>
7 <div align="left"><font size="+2">People</font></div>
8 #ew_table ("contacts" "Default Contact View" "" "" "" "contacts_frame" "")
9 </body>
10 </html>
```

- Line 8 is a Velocity macro. Once this template has been retrieved from the database, it is passed through the Velocity template engine for macro expansion, as described earlier. Let's take a look at the HTML page returned once macro expansion has taken place, visible using your browser's HTML or source code inspection tool:

```

<html>
    <head>
        <title>Test portal</title>
        <link rel="stylesheet" type="text/css" href="/ui/js/sweetalert2
/sweetalert2.css">
        <link rel="stylesheet" type="text/css" href="/ui/3dstyles;
page=PfvEV4gfMNmJxAYHx1jhhPdWx8A10025.al?lib=ewDialog">
        <script type="text/javascript" src="/ui/js/jquery/jquery.min.
js"></script>
        <script type="text/javascript" src="/ui/js/sweetalert2
/sweetalert2.min.js"></script>
    </head>
    <body>
        <h1>Test portal</h1>
        <div align="left">This is my new portal</div>
        <br>
        <div align="left">
            <font size="+2">People</font>
        </div>
        <iframe name="contacts_frame" width="100%" height="100%"
border="0" class="" src="/ui/data/search.do;
page=PfvEV4gfMNmJxAYHx1jhhPdWx8A10025.al;en;
CSRF_NONCE=FD7AD661B83FECDD70DC831DCE21BDB0?q_id=29760&
frameID=contacts_frame&_subtype_restricted=3103&view=26933& "><
/iframe>
    </body>
</html>

```

- The macro #ew_table in line 8 of the page template has been replaced and expanded in the finished page with the <iframe> page element. The iframe has a name attribute with the value "contacts_frame" as specified in the macro. You can find out more about the iframe HTML element at <https://www.w3schools.com/html/default.asp>.
- The <iframe> element creates an area on the page and obtains the content for the area by requesting it separately from the server as described above. The value of the src attribute (/gui2/data/search.do.....) gives the browser the location, or URL, of the requested content. The returned HTML is complex; it references CSS stylesheets and JavaScript functions. You can see what it looks like by right-clicking somewhere on the table and choosing View (Page) Source as before.

Useful Additions

What we've done works nicely, but a real portal will have multiple pages, working together to give users access to what they need.

Add Pages

In this section, our aim is to expand the portal you have created to include two pages the user can switch between: one to manage contacts and the other to manage tickets. You will also add the ability to use saved searches to filter the displayed records, a convenient link for the user to logout, and some status information to display the current user's login.

Test portal

This is my new portal

People

[New Person](#)

Show all records

People

Status: 61 record(s) found, 2 page(s) [Click for details....](#)

Page: [Page 1] [Next](#) [Last](#) Go to page [Go](#)

New

Mass Edit

Save Changes

Cancel Changes

Delete

Actions

Views

Search

<input type="checkbox"/>	Edit	ID	Full Name	Primary Team	Company	Direct Phone
<input type="checkbox"/>		476	Christopher Lee	Contract Creator Team	Agiloft	123-456-7890
<input type="checkbox"/>		475	Johnson Jones	Customer Team	IBM	650-344-3232 Ext.125
<input type="checkbox"/>		474	Sam Smith	Vendor Team	IBM	123-444-5555
<input type="checkbox"/>		471	Evelyn Wambaugh	Customer Team	AT&T	508-555-1212
<input type="checkbox"/>		470	Innocent DiSabato	Customer Team	ISC Design Optimizer	451-555-1212

Test portal after completing these changes

Add Records

First, let's add a new link to the page to allow the end user to add a new contact.

- 1. Log in as an admin.
- 2. Edit the testContacts .htm page template in the EUI Templates table.
- 3. Replace the contents of the Body field with the code below. This code adds lines 9-12, which create a New Person link on the portal. The URL of the link is supplied by the velocity macro #ew_create_record.

```
<html>
<head><title>Test portal</title></head>
<body>
<h1>Test portal</h1>
```



```
<div align="left">This is my new portal</div>  
<br />  
<div align="left">  
  <font size="+2">People</font>  
  &nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&~  
<a href="#ew_create_record("contacts" "/eui2template/testContacts.htm" "")">  
New Person  
</a>  
</div>  
  
#ew_table ("contacts" "Default Contact View" "" "" "" "contacts_frame" "")  
</body>  
</html>
```

4. Take a look at the parameters for the macro. The first parameter, "contacts", specifies the logical name of the table where the new record should be created. Here, we use the logical name of the People table, which we confirmed at the beginning of this tutorial.

<p>*Table Label</p> <p>Person</p>
<p>*Plural Table Label</p> <p>People</p>
<p>*Table Name</p> <p>sw_38201f_contact_table2</p> <p>Logical Table Name</p> <p>contacts</p>

Logical Table Name
located in Setup (Table)
General tab

5. The second parameter tells Agiloft where to redirect the user when they are finished adding the new record and selects Save or Cancel. In other words, it provides a URL pointing the system back to the contacts page when the user is finished with the record.
6. Save the changes and test the portal interface again. Try adding a new contact using the interface, then check how the contacts page displays after saving.

People [New Contact](#)

New Contact hyperlink
next to People header

Add Saved Searches

Next we'll add the saved search selection drop-down.

1. Edit the `testContacts.htm` template again so that it looks like the following:

```

<html>
<head><title>Test portal</title></head>
<body>
<h1>Test portal</h1>
<div align="left">This is my new portal</div>
<br />
<div align="left">
<font size="+2">People</font>
<a href="#ew_create_record("contacts" "/eui2template/testContacts.htm" "")">
New Person</a>
</div>
<div align="right">#ew_searches_list("contacts" "contacts_frame"
"contacts_select" "" "" "" "" "</div>
#ew_table("contacts" "Default Contact View" "" "" "" "contacts_frame" "")
</body>
</html>

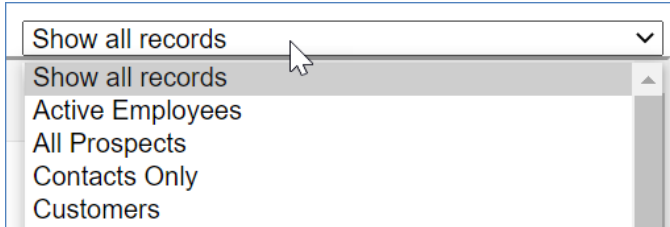
```

2. The `#ew_searches_list` macro adds all the HTML and JavaScript needed to add a drop-down list of available saved searches and attach it to the displayed table. Permanently displayed searches are dynamically based on the user's group permissions. When the user selects a saved search, the table re-displays with the filtered results of the chosen search.
3. Take a look at the parameters of the `#ew_searches_list` macro:
 - a. The first `"contacts"` parameter specifies the logical name of the table associated with the saved searches list you want to display.
 - b. The second `"contacts_frame"` parameter tells Agiloft the name of the `iframe` element to use to display the search result. In this example, we've specified the name of the `iframe` created earlier by the `#ew_table` macro. As a result, the table refreshes with the saved search results when an entry in the saved search drop-down is selected.
 - c. The third `"contacts_select"` parameter gives a name for the HTML `<select>` element that is created by this macro. Setting a name can be useful if you want to refer to the element later, perhaps to change its attributes. We won't do that in this tutorial, but it's a good habit to build.
 - d. The fourth, fifth, and sixth parameters let you apply a CSS class to the `<select>` list, indicate request parameters that should be passed to the table when the search is displayed, and specify which default search to use.



The parameters for each macro are described in [Macros Reference](#).

4. Save changes and test your portal again. Locate the new drop down list on the right hand side of the screen. Try choosing a saved search and observe that the Contacts table refreshes with the search results.



Search drop-down

Make an Included Template

Now you are almost ready to add a Tickets page to the portal, but first we will make an `included` template to contain the content that is common to both pages.

1. Create a new record in the EUI Templates table. Name it `testHeader.htm`, and type or copy-and-paste the following into the Body field:

```
<html>
<head><title>Test portal</title>
</head>
<body>
<h1>Test portal</h1>
<div align="left">This is my new portal</div>
<div align="right">
You are logged in as <b>#ew_user()</b>.
<a href="#ew_logout('http://www.agiloft.com')">Click to logout</a>
</div>
<br>
```

Notice the two velocity macros, `#ew_user` and `#ew_logout`, used in the code. These expand to the login of the current user, and a logout link, respectively. The macro `#ew_logout` takes a URL parameter: this is the URL the user is redirected to after they have logged out.

2. Save the new `testHeader.htm` template, then open the `testContacts.htm` template for editing.
3. We are going to remove some of the code from the `testContacts.htm` template. We obtain the content by including our new header template instead.

Here are the first few lines of the `testContacts.htm` template, with line numbers added for convenience:

```
1 <html>
2 <head><title>Test portal</title></head>
3 <body>
4 <h1>Test portal</h1>
5 <div align="left">This is my new portal</div>
6 <br>
7 <div align="left">
8 <font size="+2">Contacts</font>
9 ...
```

4. Delete lines 1-6 and replace them with the `#ew_include` macro. The revised template looks like this:

```
#ew_include("testHeader.htm")
<div align="left">
<font size="+2">Contacts</font>
...
```

5. Save the changes and test out the `testContacts.htm` portal.

We're nearly there. Next you will quickly create a Tickets page, and add a few navigation links to the included header template.

1. Create a new EUI template. Name it `testTickets.htm`, and copy the following into the Body field:

[illegible]

This is really just a straight copy of the `testContacts.htm` template with a few changes to reference tickets rather than contacts. It assumes that you have a table in your KB with a logical name of `case`.

2. Save the new template, and test it to make sure it works. You can do this by viewing the `testContacts.htm` page in your browser, and then editing the URL, substituting `testContacts.htm` with `testTickets.htm`.

Your resulting URL will be in the form:

```
{host}/gui2/eui2template/gui2/testTickets.htm?kb={kbName}
&user=portalUser&passwd=test01
```

3. The final step is to add links to the header template that allow the user to navigate between the tickets and contacts pages. Open the `testHeader.htm` template for editing, and add the two lines highlighted in blue so that it looks like the following:

```
<html>
<head><title>Test portal</title></head>
<body>
<h1>Test portal</h1>
<div align="left">This is my new portal</div>
<div align="right">
You are logged in as <b>#ew user()</b>.
```

```
<a href="#ew_logout('http://www.agiloft.com')">Click to log out</a>
</div>
<a href="#ew_forward("testContacts.htm")">People</a>
<a href="#ew_forward("testTickets.htm")">Tickets</a>
<br>
```

Notice the #ew_forward macro. This macro expands into a URL to forward the user to the template named by the parameter.

4. Save the changes to the template and test out the portal. You should now have two pages, with links labeled Contacts and Tickets. The user can now switch between the two pages and can view, search and add new contacts and tickets. There is also text displaying the user's login name, and a link to logout.

Add Style

For our last exercise in this tutorial we demonstrate the use of the #ew_url macro to link in a .css stylesheet. You can see <http://www.w3schools.com> to find out more about stylesheets.

1. Create a new template and name it testStyles.css. Type or copy-and-paste the following into the Body field, and don't forget to click Save when done:

```
body {font-family: arial;}
a:link {color: blue}
a:visited{color: blue}
a:hover {color: teal}
```

This simple stylesheet will change the font on your page to Arial. It also sets the color of links to blue, or to teal when you move the mouse over the link.

2. Now open the testHeader.htm template for editing, and add line 3 so that it looks like the following:

```
1. <html>
2. <head><title>Test portal</title></head>
3. <link rel="stylesheet" type="text/css" href="#ew_url("eui2template
   /testStyles.css")"/>
4. <body>
5. <h1>Test portal</h1>
6. <div align="left">This is my new portal</div>
7. <div align="right">
8. You are logged in as <b>#ew_user()</b>.
9. <a href="#ew_logout('http://www.Agiloft.com')">Click to log out</a>
10. </div>
11. <a href="#ew_forward("testContacts.htm")">People</a>
12. <a href="#ew_forward("testTickets.htm")">Tickets</a>
13. <br>
```

3. Click Save, then login to see the changes to your portal's appearance.

Why did we use the #ew_url macro?

The 'testStyles.css' stylesheet that you created is stored as a record inside the EUI Templates table. When the browser receives the portal page HTML, it follows the URL in the <href> attribute of the <link> element to request the contents of the stylesheet.

The #ew_url macro turns the relative URL, passed as the parameter, into a valid Agiloft URL complete with user session information. Just like a URL for a page template, it must contain all the information needed to prove the request is coming from a valid Agiloft user, such as the jsessionId described earlier.

Here is an example of what the HTML looks like after macro expansion has been performed:

```
<link rel="stylesheet" type="text/css" href="/gui2/eui2template/testStyles.css;jsessionId=83BB5B689B95B0A352942DAEAB0247"/>
```

Wrap Up

That's it! You now have the basic knowledge you need to carry on and create a useful interface for your users.

We've covered:

- How to create portal page templates in the EUI Templates table, and how to access them through a browser.
- What's contained in the URLs that your browser uses to access page templates.
- Use of the #ew_table, #ew_searches_list and #ew_create_record macros to add record management functionality
- Use of the #ew_user and #ew_logout macros to display login information and provide a logout link
- Use of the #ew_forward macro to provide links that forward the user to different page templates
- Use of the #ew_include macro to pull common content into multiple pages
- Use of the #ew_url macro to create a valid Agiloft URL including user session and authentication information

A good way to learn more is to look through the default EUI template pages that are included with the default knowledgebase. With the knowledge you've gained already you should be able to understand how the templates work.

Because templates use the Velocity engine for Agiloft macros, any other Velocity constructions are also available for your use. Please see Velocity's documentation for details here: <http://velocity.apache.org/engine/releases/velocity-1.4/user-guide.html>.

Notes

1. If no login details are included in the URL when a user accesses Agiloft, the system will try to login with user=eui2, password=qwerty. If you would like to take advantage of this automatic login, ensure that a user created with those login credentials exists, and is granted the appropriate permissions
2. If users belong to different groups, and their end-user interfaces allow them access to different areas of your Agiloft application, you will need to either create a separate portal interface for each group or use conditional #if macros to determine what they see. All users logging into one particular EUI template will see the same HTML, regardless of the permissions granted to their individual groups. When users begin to interact with the Agiloft application through the portal controls, such as attempting to add a new record, detailed group and user permissions will come into play. The user's group permissions may restrict or permit access to tables, fields, and records.