



JaxterChat

Developer's Guide

JaxterChat v1.2.8
Published by WebFurbish
Visit <http://www.webfurbish.com>



Table of Contents:

1. System Requirements.....	3
2. Installation.....	3
3. Configuration.....	5
3.1. Selecting a Theme.....	5
3.2. Customizing Appearance.....	6
3.3. Saving Custom Themes.....	6
3.4. Defining Multiple Rooms.....	7
3.5. Integrating with Existing Logins.....	7
3.6. Sending Custom Messages to Rooms from Code Behind.....	7
3.7. Logging Chat Messages.....	8
3.8. Displaying Chat History / Startup Text.....	8
4. Deployment.....	9
5. Troubleshooting.....	10

1. System Requirements

JaxterChat is an ASP.NET control with full designer support for both Visual Studio .NET 2003 and Visual Studio .NET 2005. It has also been tested more recently under Visual Studio 2008 with .NET 3.5. Both Visual Studio and a locally installed copy of JaxterChat are required to develop a web application with this control. No database or special permissions are required from the web server.

JaxterChat has been tested on the latest versions of most modern browsers including Internet Explorer, Mozilla Firefox, Safari and Opera; However, certain browsers will allow extra visual enhancements such as stylized scroll-bars (Internet Explorer, Opera). Performance can vary between browsers with Internet Explorer offering the most optimal speed and visual benefits.

2. Installation

To install JaxterChat, simply extract the downloadable .zip file to your local PC and run Setup.exe. This will install JaxterChat and all support materials to a directory on your local machine (e.g. *C:\Program Files\WebFurbish\JaxterChat*).

1. Launch Visual Studio (2003 or 2005) and open an existing or new website project. (For 2008, use 2005 version)
2. Select a web-form designer view within this project to get a populated view of the web controls toolbox.
3. Add the JaxterChat control to your Visual Studio environment toolbox by right clicking the toolbox panel and selecting '*Choose Items*' for Visual Studio 2005 (Figure 1), or '*Add / Remove Items*' for Visual Studio 2003.

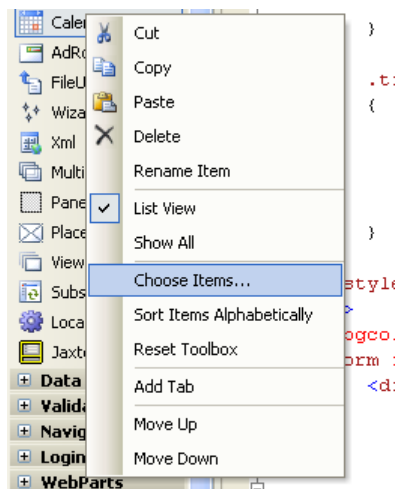


Figure 1: Choose Items from Visual Studio 2005

4. Select the Browse option and navigate to the directory where the JaxterChat component was installed (e.g. *C:\Program Files\WebFurbish\JaxterChat*). Select the 'JaxterChat.dll' file in the root of this directory and click 'Open'. (Figure 2)

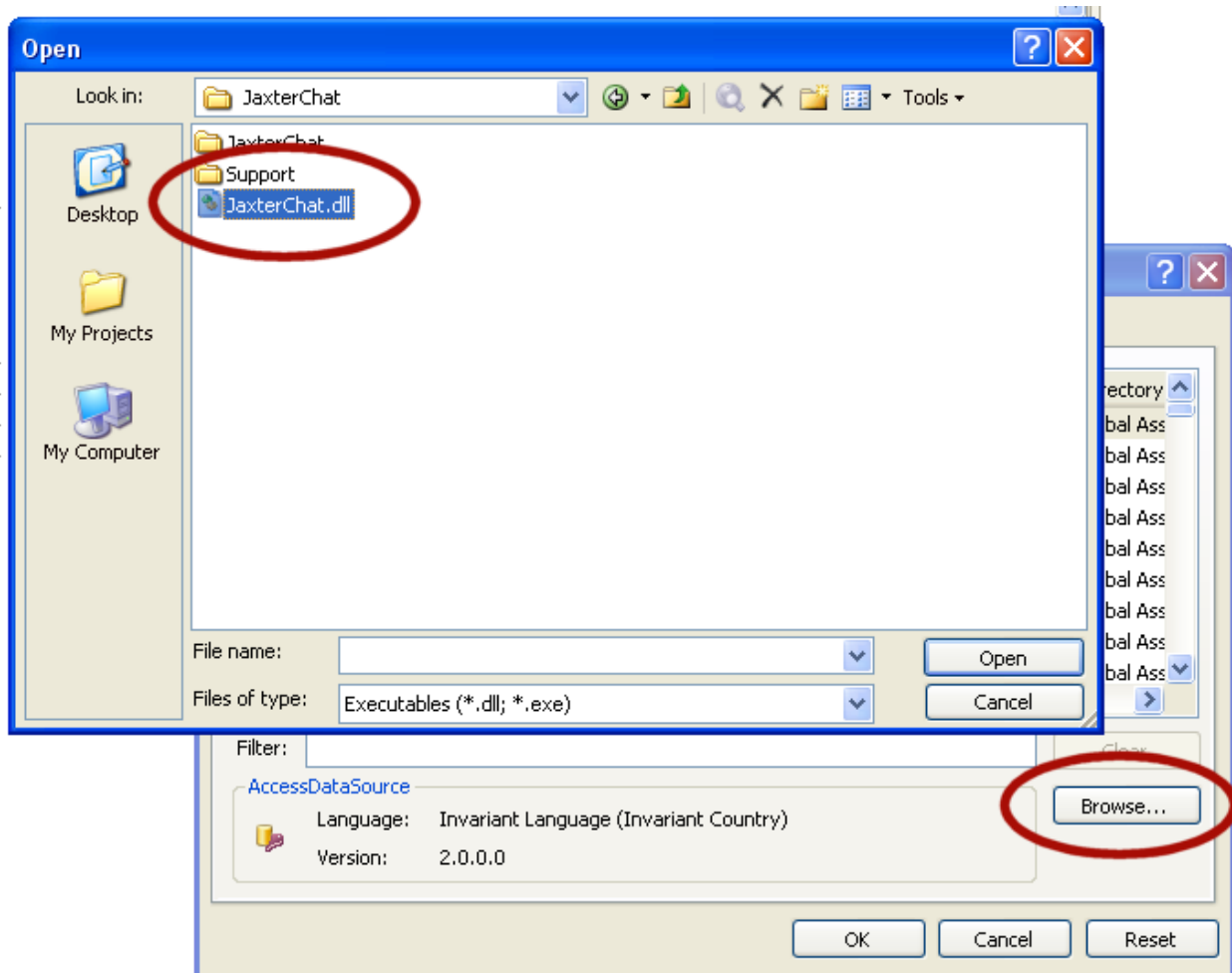


Figure 2: Browse to the JaxterChat.dll control and add it to the toolbox.

5. Drag the JaxterChat icon (Figure 3) onto the WebForm. Dragging the control onto the form will automatically copy the JaxterChat directory from the master install location to your website's root along with any custom theme or room data you may have configured. If the images within the chat do not appear correctly, this could be due to the fact that the target web form is within one or more directories from the root of the website. To correct this, enter the number of folders in the URLPathPrefix as url navigation string (e.g. if your web form is two folders deep from the root of the website, the URLPathPrefix property should be set to be `../..`)

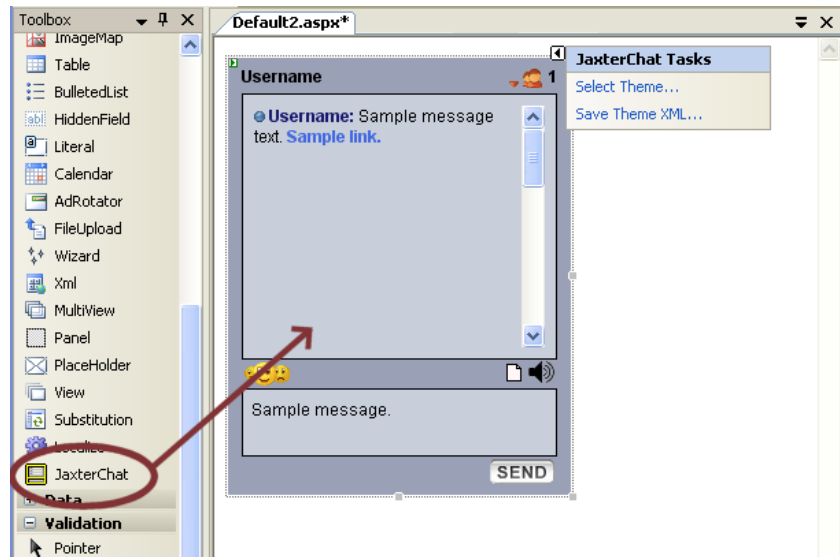


Figure 3: Drag the JaxterChat tool icon onto your web form from the toolbox.

3. Configuration

3.1. Selecting a Theme

JaxterChat comes with several pre-defined themes. You can use these themes as a basis for designing your own custom look and feel or simply choose the most suitable for your website and use it without modification. To select a theme, right click on the JaxterChat control as it appears on your web form and choose 'Select Theme' from the popup menu to display the Theme Browser (Figure 4).

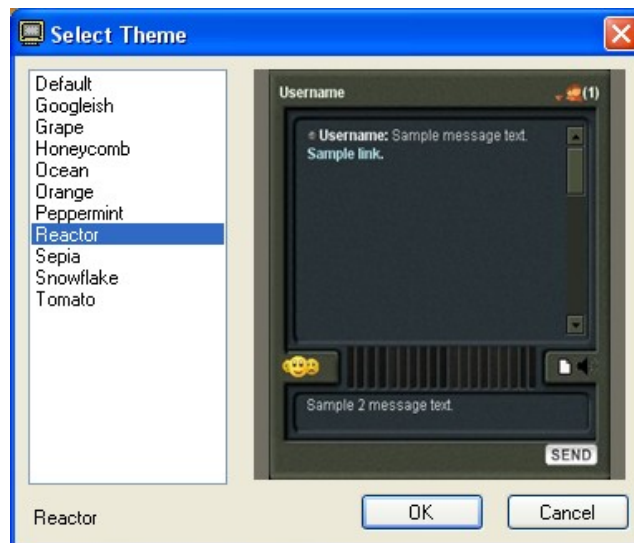


Figure 4: Selecting a theme from the theme browser.



3.4. Defining Multiple Rooms

Rooms are defined in an easily maintained XML file named '*rooms.xml*' which resides in the root of the JaxterChat directory within your website. When visiting the chat, a room is specified via the property *RoomName*. The '*rooms.xml*' file must have one room marked as *default=true* which will take place as the default room name in the case that private rooms are not allowed and the *RoomName* property provided does not comply with one of the rooms listed in '*rooms.xml*'. You may also allow private rooms to be defined by setting the *EnforceRoomsXML* property to false and setting a *RoomName* value other than those defined in '*rooms.xml*'. Rooms created dynamically like this are considered 'private' chat rooms. You may retrieve information about private rooms by providing a 'true' parameter to the function *GetRoomUsage(true)*. This will include private rooms information as well as all public rooms listed in the *rooms.xml* file.

```
public DataSet GetRoomUsage(bool showPrivateRooms)
```

3.5. Integrating with Existing Logins

JaxterChat allows manual setting of a users name set via the *UserName* property. If a user with the same name is already present in the chat room, an auto incrementing number will be used to provide the new user with a unique name inside the chat. This approach will allow integration with almost any existing user system. Simply read the values in whatever form they are provided (usually in *Page_Load* event) and set the properties *UserName* and *RoomName* accordingly.

3.6. Sending Custom Messages to Rooms from Code Behind

In certain circumstances, you may wish to broadcast a message from your application to a room so that all users in the room can see it. This might be used for advertising or news events etc. To do this a public function has been added to the control called *SendCustomMessage*. You can pass raw HTML to the control and it will be output to the chat client for all users within that room to see. To broadcast to all rooms, you must loop through the rooms collection returned by function *GetRoomList()* or *GetPrivateRoomList()* and call *SendCustomMessage()* for each room. The first parameter specifies the html to be sent, the second parameter defines a username to be used (for logging etc) which can be any name you wish, the third parameter defines the target room for the broadcast and the fourth parameter specifies if this message should fire the *MessagePosted* event or not. (for including broadcast messages in your logs, set this value to true) Refer to '*Logging Chat Messages*' in this document for more information.

```
// Example of sending a custom message to a room
```

```
WebFurbish.JaxterChat myChat = new WebFurbish.JaxterChat();  
myChat.SendCustomMessage("news flash 123", "Admin Bot", "MyRoomName", false);
```



3.7. Logging Chat Messages

JaxterChat provides an event handler that is fired for every message that is posted. Simply provide code to store these messages in a text file or database by binding to the *MessagePosted* event for the control. The username, room name, and message can be read via the *JaxterMessageEventArgs* event parameter. Refer to following example.

// Example of logging messages to a text file

```
private void MessagePosted(object o, WebFurbish.JaxterMessageEventArgs e)
{
    string message = e.Message;
    string messageHTML = e.MessageHTML;
    string room = e.RoomName;
    string user = e.UserName;
    StreamWriter sw = new StreamWriter("C:\\dumpit.txt", true);
    sw.WriteLine("Message at:" + DateTime.Now + " Room:" + e.RoomName + " User:" +
        e.UserName + " Message:" + e.Message + " HTML:" + messageHTML);
    sw.WriteLine("");
    sw.Flush();
    sw.Close();
}
```

3.8. Displaying Chat History / Startup Text

JaxterChat provides an event handler *ChatStarted* that is fired once when the chat loads which allows the application to set startup text or a custom managed chat history log. Simply provide code to handle the *ChatStarted* event and set the *e.StartupText* value. This event behaves differently to the startup properties / text settings available in the designer as it works independently of the browsers cache when the user hits the back button. Any non-static startup text should be supplied via this event to prevent browser cache displaying incorrect content after the back-button is pressed. Refer to the following example of the *ChatStarted* event handler and a modified *MessagePosted* example which displays this historic data when a user enters the chat. Be sure to set the *EnableViewState* property to *false* when manually displaying chat history to prevent double-up display of chat messages.

// Sample Code - Display entire chat history upon startup

// NOTE: In a production environment a database would be used to store / read the historic posts.

```
private void JaxterChat1_ChatStarted(object o, WebFurbish.JaxterOnStartEventArgs e)
{
    string room = e.RoomName;
    string user = e.UserName;
    StreamReader sr = new StreamReader("C:\\dumpit.txt");
    string readText = sr.ReadToEnd();
    e.StartupText = readText;
}
```



```
// Modified MessagePosted for example of message history from a text file
private void MessagePosted(object o, WebFurbish.JaxterMessageEventArgs e)
{
    string message = e.Message;
    string messageHTML = e.MessageHTML;
    string room = e.RoomName;
    string user = e.UserName;
    StreamWriter sw = new StreamWriter("C:\\\\dumpit.txt",true);
    sw.WriteLine("<B>" + user + ":\</B> " + messageHTML + "<br>");
    sw.Flush();
    sw.Close();
}
```

4. Deployment

Since JaxterChat does not rely on any particular user rights or database back-end systems on the web server, deployment is as simple as copying the working website to the ASP.NET web server via FTP or equivalent.

5. Troubleshooting

1) Images do not appear on the default theme after dragging onto the form.

This is usually caused by the path to the webform in question containing one or more directories below the website root. For example, if your web forms are stored in a directory called 'WebForms' below the root of your website, the chat component is unable to find the JaxterChat resources directory which was automatically copied to the root of the website when first adding the control to your web form. To correct this, enter the number of folders in the URLPathPrefix as url navigation string (e.g. if your web form is two folders deep from the root of the website, the URLPathPrefix property should be set to be `../..`) (Figure 6)

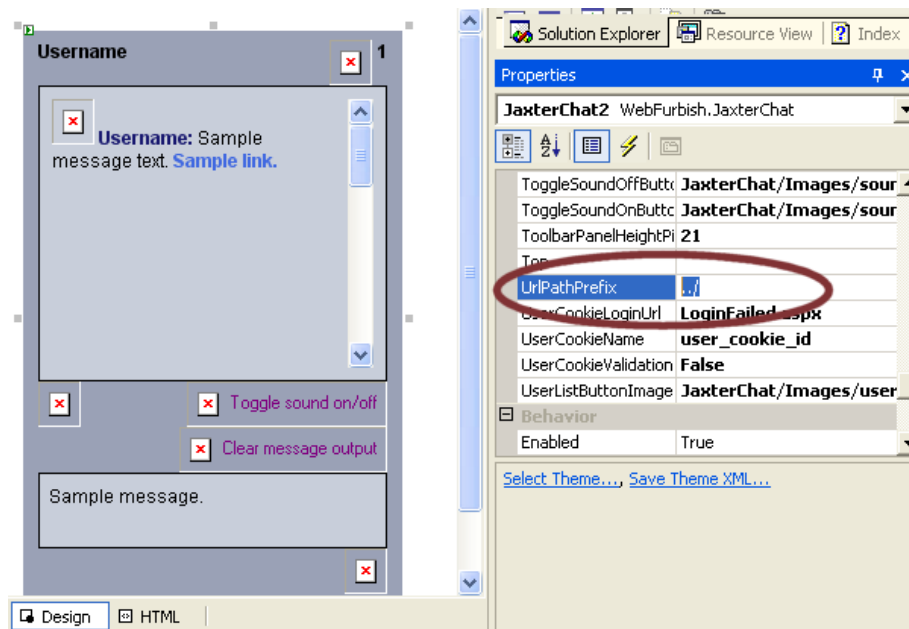


Figure 6: Set the URLPathPrefix if images don't appear in the chat designer.

2) I have updated the rooms.xml in the directory where I originally installed JaxterChat but my website is not finding these changes?

The directory containing JaxterChat where the control was originally installed to is to be considered the 'master' location. When you drag a JaxterChat control onto a web form for the first time in a web project, this master folder is copied to the root of your website automatically. If you had already dragged the control onto the web form before changing the rooms.xml file, then you will need to change the copy of this file for the website to reflect these changes.



- 3) I have created a new theme, saved the settings.xml in the master directory and also created a screenshot.jpg file for the theme browser. However, when I right click the control and choose 'Select Theme' it does not appear in the list.**

This problem is similar to that outlined in troubleshooting item 2. In the case where you had already dragged a control onto the web form before saving the theme to the master directory located in your Program Files install location, your website is now referring to a copy of this master JaxterChat directory which was made before your changes were applied. You will need to copy this newly created theme directory to the JaxterChat/Themes directory in your website as well as the master install location.

- 4) When I drag the JaxterChat control onto a complex web form layout that has a variety of divs and tables, the chat is not completely visible and scrollbars appear at the top and sides of the control within Visual Studio.**

In certain cases, Visual Studio does not complete the layout within the designer until you rebuild the web project. After rebuilding, the control should occupy the correct amount of space and force surrounding containers to accommodate it. This is a bug only within the designer view itself and will not be reflected when running the website.

- 5) In Visual Studio 2003 I see the an error icon in place of the control in the designer view.**

Close and open Visual Studio and the control should render properly. This usually occurs when upgrading from earlier versions of the JaxterChat control and is typically only found in Visual Studio 2003.

- 6) Instead of the control appearing in the designer, I get a custom error stating 'Missing Files'**

When the control is first dragged into a solution, the designer will attempt to copy the JaxterChat resources directory from the installation location to the root of your website. If an error occurred during install or the directory has been moved since installation, the designer may be unable to locate / copy this directory automatically. To resolve this, you should locate the installation directory of the JaxterChat control and copy the JaxterChat directory located inside the installed directory path (e.g. copy *c:\Program Files\WebFurbish\JaxterChat\JaxterChat* to the root of your website). Be careful to copy the JaxterChat directory that is located inside the installation directory and not the installation directory itself which may also be named JaxterChat by the installer.



- 7) I am using Visual Web Developer 2005 (Free version of Visual Studio). When I drag the control onto the web form and run the application, the chat does not work and I get a Javascript error 'WebFurbish is undefined'**

Unfortunately, The drag and drop functionality does not work correctly with this developer version of Visual Studio. To get the chat working, you must manually copy the JaxterChat resources directory into the root of your website found at the following location.

(C:\ProgramFiles\WebFurbish\JaxterChat\Support\JaxterChatSample_vs2005\JaxterChat)