# How to Detect the Web Browser Close Event in ASP.NET MVC (CS\VBASPNETDetectBrowserCloseEvent)

## Introduction

## As we know, HTTP is a stateless protocol, so the browser doesn't keep connecting to the server. When users try to close the browser using alt-f4, browser close(X) and right click on browser and close, all these methods are working fine, but it's not possible to tell the server that the browser is closed.

## The sample demonstrates how to detect the browser close event.

It includes two parts:

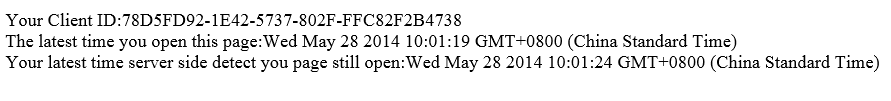
1 Send ajax call every 20 seconds, to tell the server side that the page is still open.

2 Send ajax call when the browser is closed.

## Running the Sample

You can run the sample directly, then type http://localhost:{your port number} /default/ to open the default page.

You will see a page like below:



The third line changes every 20 seconds.

When you close the browser, you can open the xml file in app\_data. You can find the information above is in the xml file.

## Using the Code

The following code snippet will show how to detect the browser close event.

|  |
| --- |
| -Code block start-  --JavaScript code snippet start--  function recordeCloseTime()  {  var targetUrl = "[@Url.Action("RecordCloseTime](mailto:@Url.Action(%22RecordCloseTime)", "Default")" + "?clientId=" + getClientID();  $.ajax({  type: "HEAD",  url: targetUrl,  });  }  --JavaScript code snippet end--  Insert other Programming Language Code Snippet here  -Code block end- |

The following code snippet will show how to create a client id and store in cookie.

|  |
| --- |
| -Code block start-  --JavaScript code snippet start--  function generateGuid() {  var result, i, j;  result = '';  for (j = 0; j < 32; j++) {  if (j == 8 || j == 12 || j == 16 || j == 20)  result = result + '-';  i = Math.floor(Math.random() \* 16).toString(16).toUpperCase();  result = result + i;  }  return result;  }  --JavaScript code snippet end--  Insert other Programming Language Code Snippet here  -Code block end- |

The following code snippet will show how to refresh the refresh time in order to tell server side the page still open.

|  |
| --- |
| -Code block start-  --JavaScript code snippet start--  function getRefreshTime()  {  var targetUrl = "[@Url.Action("GetRefreshTime](mailto:@Url.Action(%22GetRefreshTime)", "Default")" + "?clientId=" + getClientID();  $.ajax({  type: "GET",  url: targetUrl,  cache:false,  success: function (data) {  if (data != "") {  var clientInfo = JSON.parse(data);  $("#lbClientId").text(clientInfo.ClientID);  $("#lbActiveTime").text(new Date(clientInfo.ActiveTime));  $("#lbRefreshTime").text(new Date(clientInfo.RefreshTime));  }  }  });  }  --JavaScript code snippet end--  Insert other Programming Language Code Snippet here  -Code block end- |

This codes will show a simple data source to operate an xml database.

|  |
| --- |
| -Code block start-  --C# code snippet start--  public class ClientInfoDataSource  {  private static string filePath = HttpContext.Current.Server.MapPath("~/App\_Data/ClientInfos.xml");  private static XDocument clientInfosXDoc;    public ClientInfoDataSource()  {  clientInfosXDoc = XDocument.Load(filePath);  }  /// <summary>  /// Get ClientInfo by ClientId  /// </summary>  /// <param name="clientID"></param>  /// <returns></returns>  public ClientInfo GetClientInfoByClientId(string clientID)  {  var query = from clientInfoXml in clientInfosXDoc.Descendants("ClientID")  where clientInfoXml.Value == clientID  select clientInfoXml.Parent;  return convertToClientInfo(query.FirstOrDefault());  }  /// <summary>  /// Insert ClientInfo message to XML file  /// </summary>  /// <param name="clientInfo"></param>  /// <returns></returns>  public void InsertClientInfo(ClientInfo clientInfo)  {  clientInfosXDoc.Root.Add(convertToClientInfoXElement(clientInfo));  }  /// <summary>  /// Update ActiveTime and RefreshTime  /// </summary>  /// <param name="clientInfo"></param>  public void UpdateClientInfo(ClientInfo clientInfo)  {  var query = from x in clientInfosXDoc.Root.Elements()  where x.Element("ClientID").Value == clientInfo.ClientID  select x;  if (query.Count()>0)  {  query.FirstOrDefault().Element("ActiveTime").Value = clientInfo.ActiveTime.ToString("MM/dd/yyyy HH:mm:ss");  query.FirstOrDefault().Element("RefreshTime").Value = clientInfo.RefreshTime.ToString("MM/dd/yyyy HH:mm:ss");  }  }  /// <summary>  /// Save data source changes  /// </summary>  public void Save()  {  clientInfosXDoc.Save(filePath);  }  /// <summary>  /// Convert XML message to Class  /// </summary>  /// <param name="clientInfoXml"></param>  /// <returns></returns>  private ClientInfo convertToClientInfo(XElement clientInfoXml)  {  if (clientInfoXml!=null)  {  ClientInfo clientInfo = new ClientInfo();  clientInfo.ClientID = clientInfoXml.Element("ClientID").Value;  clientInfo.ActiveTime = DateTime.Parse(clientInfoXml.Element("ActiveTime").Value);  clientInfo.RefreshTime = DateTime.Parse(clientInfoXml.Element("RefreshTime").Value);  return clientInfo;  }  return null;  }  /// <summary>  /// Convert Class to XML message  /// </summary>  /// <param name="clientInfo"></param>  /// <returns></returns>  private XElement convertToClientInfoXElement(ClientInfo clientInfo)  {  if (clientInfo!=null)  {  XElement xDoc = new XElement("ClientInfo",  new XElement("ClientID", clientInfo.ClientID),  new XElement("ActiveTime", clientInfo.ActiveTime.ToString("MM/dd/yyyy HH:mm:ss")),  new XElement("RefreshTime", clientInfo.RefreshTime.ToString("MM/dd/yyyy HH:mm:ss")));  return xDoc;  }  return null;  }  }  --C# code snippet end--  --VB code snippet start--  Public Class ClientInfoDataSource  Private Shared filePath As String = HttpContext.Current.Server.MapPath("~/App\_Data/ClientInfos.xml")  Private Shared clientInfosXDoc As XDocument  Public Sub New()  clientInfosXDoc = XDocument.Load(filePath)  End Sub  ''' <summary>  ''' Get ClientInfo by ClientId  ''' </summary>  ''' <param name="clientID"></param>  ''' <returns></returns>  Public Function GetClientInfoByClientId(clientID As String) As ClientInfo  Dim query = From clientInfoXml In clientInfosXDoc.Descendants("ClientID") Where clientInfoXml.Value = clientID Select clientInfoXml.Parent  Return convertToClientInfo(query.FirstOrDefault())  End Function  ''' <summary>  ''' Insert ClientInfo message to XML file  ''' </summary>  ''' <param name="clientInfo"></param>  Public Sub InsertClientInfo(clientInfo As ClientInfo)  clientInfosXDoc.Root.Add(convertToClientInfoXElement(clientInfo))  End Sub  ''' <summary>  ''' Update ActiveTime and RefreshTime  ''' </summary>  ''' <param name="clientInfo"></param>  Public Sub UpdateClientInfo(clientInfo As ClientInfo)  Dim query = From x In clientInfosXDoc.Root.Elements() Where x.Element("ClientID").Value = clientInfo.ClientID Select x  If query.Count() > 0 Then  query.FirstOrDefault().Element("ActiveTime").Value = clientInfo.ActiveTime.ToString("MM/dd/yyyy HH:mm:ss")  query.FirstOrDefault().Element("RefreshTime").Value = clientInfo.RefreshTime.ToString("MM/dd/yyyy HH:mm:ss")  End If  End Sub  ''' <summary>  ''' Save data source changes  ''' </summary>  Public Sub Save()  clientInfosXDoc.Save(filePath)  End Sub  ''' <summary>  ''' Convert XML message to Class  ''' </summary>  ''' <param name="clientInfoXml"></param>  ''' <returns></returns>  Private Function convertToClientInfo(clientInfoXml As XElement) As ClientInfo  If clientInfoXml IsNot Nothing Then  Dim clientInfo As New ClientInfo()  clientInfo.ClientID = clientInfoXml.Element("ClientID").Value  clientInfo.ActiveTime = DateTime.Parse(clientInfoXml.Element("ActiveTime").Value)  clientInfo.RefreshTime = DateTime.Parse(clientInfoXml.Element("RefreshTime").Value)  Return clientInfo  End If  Return Nothing  End Function  ''' <summary>  ''' Convert Class to XML message  ''' </summary>  ''' <param name="clientInfo"></param>  ''' <returns></returns>  Private Function convertToClientInfoXElement(clientInfo As ClientInfo) As XElement  If clientInfo IsNot Nothing Then  Dim xDoc As New XElement("ClientInfo", New XElement("ClientID", clientInfo.ClientID), New XElement("ActiveTime", clientInfo.ActiveTime.ToString("MM/dd/yyyy HH:mm:ss")), New XElement("RefreshTime", clientInfo.RefreshTime.ToString("MM/dd/yyyy HH:mm:ss")))  Return xDoc  End If  Return Nothing  End Function  End Class  --VB code snippet end--  -Code block end- |