مقترح لحل مشكلة تحميل الملفات غير المتزامن في ال ASP.NET دون الحاجة الى استخدام تقنية الاجاكس

سلام شاكر كتاب

قسم علوم الحاسبات، كلية التربية – ابن الهيثم، جامعة بغداد

الخلاصة

إن محمل الملفات (File Upload) هو احد ال controls الموجودة ضمن شريط الأدوات في ال VisualStudio.NET ولكن لا يمكن استخدام هذا ال control مع تقنية ال Ajax (فيكاد يكون من المستحيل في الوقت الحاضر ان تهتم بتصميم تطبيقات الوب او تطويرها من دون ان تكون مستخدما لتقنية الاجاكس) لان File Upload لا يدعم هذه التقنيه حيث يكون التنفيذ للايعاز بشكل غير مكتمل أو تظهر عبارة object is not set على الرغم من الحاجة الماسة لاستخدام هذا ال control مع الاجاكس للاستفادة من نقل البيانات غير المتزامن الذي يوفره الاجاكس ، وذلك لان File Upload control لا يعمل إلا عند حدوث عملية Postback من ال client الى ال server والفكرة الاساسيه لفلسفة عمل الاجاكس هي عدم القيام بعملية Postback الى ال server بل انشاء كائن برمجي (XMLHttpRequest) يتصل بال server للقيام بعملية تحديث جزئية لقسم محدد من الصفحة(Web Site). وبعد البحث الموسع في موقع www.asp.netو موقع www.msdn.microsoft.com تم التوصل الى ان هذا ال control سوف يقوم بدعم تقنية الاجاكس في الاصدار اللاحق VisualStudio.net 2008، الا ان الحل لم يكن حقيقيا لان عملية ال Postback مستمرة ايضا، فاضافوا مايعرف بال Postback Trigger. لذلك اقتضت الحاجة الى اقتراح حل بديل عن استخدام تقنية الاجاكس من خلال بناء Web Services لتقوم بعملية التحميل بشكل غير متزامن و تجنب اعادة تحميل الصفحة بالكامل (Postback operationعند محاولة تحميل الملفات.

A Suggestion To Solve The Problem Of Asynchronous File Uploads In ASP.NET Without The Need To Use Ajax Technology

S. S. Kitab
Department of Computer Science, College of Education - Ibn Al-haitham, Baghdad University,

Abstract

The file upload is one of the controls of the tool bar of the VisualStodio.NET, but it can't be used with Ajax technology (It's almost impossible today to be involved in web application design or development and not be aware of Ajax) because the file upload control don't support Ajax technology that the execution of the instruction is not completed or the (object is not set) message will appear, although the essential need to use this control with Ajax to gain the asynchronous data transfer that supplied by Ajax, because the file upload don't work without the Postback operation from the client to the server, and the main idea of Ajax technology is working without Postback operation to the server. After the deep research in www.asp.net and www.msdn.microsoft.com sites the result was that the solution will be in the next version VisualStodio.NET 2008, but it was not a real solution because the Postback operation is continuous. So this paper suggests alternative solution to Ajax by building Web Services to upload files asynchronously and avoid reload the whole web site (Postback operation) when file uploading process is lemented.

Introduction

As the web grows and bandwidth increases, people demand more from their online experiences. They want more content and graphics, smarter applications, and instantaneous responses [1]. Until recently, web application developers and end users for that matter struggled with slow, unresponsive systems where entire pages had to be reloaded to update a single element. The web's architecture simply was not a convenient platform for demanding applications [2]. Many solutions, like Google Maps, were not even possible until the wide-spread adoption of Ajax occurred. AJAX is not a new programming language, but a technique for creating better, faster, and more interactive web applications [3]. With AJAX, your JavaScript can communicate directly with the server, using the JavaScript XMLHttpRequest object. With this object, your JavaScript can trade data with a web server, without reloading the page, but it not supplied by the file upload control ability [4]. One of the most important things that should be transferred through the web are the files, so we discussed an approach to uploading files to a web server directly through an ASPX page, or through a web service. The method is used to asynchronously upload files rather than using Ajax to make Ajax style file upload approach.

Ajax technology

AJAX is a browser technology independent of web server software. It is a way of developing Web applications that combines [1]:

- XHTML (EXtensible HyperText Markup Language) and CSS (Cascading Style Sheets) standards based presentation.
- Interaction with the page through the DOM (Document Object Model).
- Data interchange with XML (EXtensible Markup Language) and XSLT (EXtensible Stylesheet Language Transformations).
- Asynchronous data retrieval with XMLHttpRequest.
- JavaScript to tie it all together.

The Ajax engine works within the Web browser (through JavaScript and the DOM) to render the Web application and handle any requests that the customer might have of the Web server [1]. The beauty of it is that because the Ajax engine is handling the requests, it can hold most information in the engine itself, while allowing the interaction with the application and the customer to happen asynchronously and independently of any interaction with the server [2].

Asynchronous interaction

This is the key. In standard Web applications, the interaction between the customer and the server is synchronous. This means that one has to happen after the other. If a customer clicks a link, the request is sent to the server, which then sends the results back [2]. With Ajax, the JavaScript that is loaded when the page loads handles most of the basic tasks such as data validation and manipulation, as well as display rendering the Ajax engine handles without a trip to the server. At the same time that it is making display changes for the customer, it is sending data back and forth to the server. But the data transfer is not dependent upon actions of the customer [3].

Web Services

Web services are software system designed to support interoperable machine-to-machine interaction over a network [4]. Web services are frequently just Web APIs that can be accessed over a network, such as the Internet, and executed on a remote system hosting the requested services. This definition encompasses many different systems, but in common usage the term refers to clients and servers that communicate over the HTTP protocol used on the Web [5].

Postback operation

A Postback is an action taken by an interactive webpage, when the entire page and its contents are sent to the server for processing some information and then, the server posts the same page back to the browser [6]. This is done to verify passwords for logging in, process an on-line order form, or other such tasks that a client computer cannot do on its own. This is not to be confused with refresh or back actions taken by the buttons on the browser [7]. Postback in an event that is triggered when action is performed by a control on an asp.net page. For example when you click on a button the data on the page is posted back to the server for processing [8].

Upload Files

The following suggested approach to upload files to a web server:

- By using a web service (To simulate Ajax style) Upload files asynchronously.

Ajax style, this paper and included code describe and illustrate this approach. The example will illustrate an approach to classify files upon upload and direct the file to specific folders based on the type of file uploaded.

- Programming Environment: Visual Studio 2005

- Language: Visual Basic 2005, ASP.NET

Algorithm [*Upload File in ASP.NET without the Need to use Ajax Technology*]

[1] Build a web service.

1.1. Call a library that is used to cut down the file extension from interior file upload.

1.2. Build a method used to save the interior file with its extension completely and into folders used to hold different types of files (e.g., images or text).

1.2.1. Filter file options based upon the file type.

[2] Use a web service (To simulate Ajax style) to upload files asynchronously.

2.1. Import a library in order to implement a process involving the file extension.

2.2. Convert the file to bytes.

2.3. Do not use a postback and implement a wait handle to block the page from loading until after the results are returned from the web service.

2.4. Check complete upload or not.

The Code:

The first project is the web site, and the second one is a web service that would be used to upload files. The solution should contain both of them as shown in figure (1).

1- Web Service

The start contains all the declarations, the extension of the file would be cut down from the interior file by System.IO library.

```
Imports System.Web
Imports System.Web.Services
Imports System.Web.Services.Protocols
Imports System.IO

' F I L E  S T O R A G E  W E B  S E R V I C E

' This service contains one web method:
'  1. SFAT - Saves file to folder with extension
<WebService(Namespace:="http://tempuri.org/")> _
<WebServiceBinding(ConformsTo:=WsiProfiles.BasicProfile1_1)> _
<Global.Microsoft.VisualBasic.CompilerServices.DesignerGenerated()> _
Public Class FileStorage
    Inherits System.Web.Services.WebService
```

The file storage class (“FS” for short) presents the save file as type method (“SFAT” for short) that used to save the complete interior file with the extension, these files could be variety kinds such pictures, text, web pages. Etc.

```
<WebMethod()> _
Public Function SFAT(ByVal sFN As String, _
ByVal bFBA As Byte()) As Integer
    ' Saves a file as a specific type with an extension
    ' Capture the file type through its extension
    Dim sFT As String
    sFT = System.IO.Path.GetExtension(sFN)
    sFT = sFT.ToLower()
    ' filter file options based upon the file type
    Select Case sFT
        Case ".doc"
          ' Get the file name and set another path to the storage folder
          Dim sF As String = _
          System.IO.Path.GetFileNameWithoutExtension(sFN)
          sF = System.Web.Hosting.HostingEnvironment.MapPath _
          ("~/StoredText/" & sF & sFT)
          'write the file out to the storage location
          Try
             Dim stream As New FileStream(sF,
             FileMode.OpenOrCreate)
             stream.Write(bFBA, 0, bFBA.Length)
             stream.Close()
             Return 0
          Catch ex As Exception
             Return 1
          End Try
```

The same code will continue for the following extensions:

```
        Case
".docx",".xml",".rtf",".lic",".txt",".html",".htm",".bmp",".jpg",".gif",".png",".tif",".ico",".wmf".

        Case Else
          ' we don't know what kind of file it is, so we put in into
          ' this folder, then Get the file name and set a new path to
          ' the local storage folder
          Dim sF As String = _
          System.IO.Path.GetFileNameWithoutExtension(sFN)
          sF = System.Web.Hosting.HostingEnvirnment.MapPath _
          ("~/StorageStuff/" & sF & sFT)
          'write the file out to the storage location
          Try
             Dim stream As New FileStream(sF,
             FileMode.OpenOrCreate)
             stream.Write(bFBA, 0, bFBA.Length)
```

```
            stream.Close()
            Return 0
        Catch ex As Exception
```


```
            Return 1
        End Try
    End Select
End Function
```

The extension of the interior file is cut down and the case body uses it. The case categorizes the files according to the extension and the file kind (pictures, documents, or unknown file); the case will put the file into related folder. The name of the file and its information create a string that could be receipted by the method which eliminates the extension from the file then decide a new path. The unknown file would be sent to a rubbish folder called storage stuff.

According to a try block, the method sends back (returns) zero if the file was written successfully and sends back one if not. That zeros and ones used by another method to specify the success or failure of the operation.

NOTE: A fail message will return if you try to upload a file that uploaded previously.

2- Web Site.

As shown in figure (2), the web site will lead the user to upload the files asynchronously to web service with its extension.

The page class produces inheritance to the declared object (the solitary one) that represents the proxy described previously. The proxy page doesn't need vast scope but it's not damage if it takes that scope, so it could keep the code inside its own methods.

The initial code part of the site is:

```
Imports System.IO
' D E F A U L T  A S P X  P A G E
'
' This web page offers user file upload option:
' Use a web service (To simulate Ajax style) to upload files asynchronously
Partial Class _Default
    Inherits System.Web.UI.Page
    Private webs As New FileStorage.FileStorage()
```

The code in the main page class is used to implement the upload files asynchronously and keep the extension (To simulate Ajax style).

The code will not use a postback and apply a wait handler to prevent (freeze) the site load while the result comeback from the web service; and the routine will manipulate the results and demonstrates success or failure to the user according to the result that comeback from the web service. If you wish to manipulate the result with another way, you have to execute a postback operation in case that you don't warry about preventing (blocking) the page form loading. A wait handle is also set and the results are only processed after the method returns. The code is as follows:

```
    Protected Sub bUploadWithExAsyn_Click(ByVal sender As Object, _
    ByVal e As System.EventArgs) Handles bUploadWithExAsyn.Click
        ' Asynchronous file upload (keep extension)
```

```
    Try
        ' validate that a file has been selected into the
        ' file upload control
```


```
        If FUp.HasFile Then
            ' convert the file to bytes
            Dim bytes As Byte() = FUp.FileBytes
            ' to catch the return value, set up an IAsyncResult object
              from
            ' the web service call - we are not using a postback here so
            ' the postback argument is set to nothing
            Dim iarst As IAsyncResult
            iarst = webs.BeginSFAT(FUp.FileName.ToString(), _
            bytes, Nothing, Nothing)
            ' Wait for the call to finish
            iarst.AsyncWaitHandle.WaitOne()
            ' the async call is complete, process the results
            Dim rV As String
            rV = webs.EndSFAT(iarst)
            If rV = "0" Then
                MessageBox("The file uploaded")
            Else
                ' this message will appear
                MessageBox("The file is not uploaded")
            End If
        End If
    Catch ex As Exception
        MessageBox(ex.Message.ToString())
    End Try
End Sub
```

Conclusions

1- "Web Services" is the umbrella term of group of loosely related Web-based resources and components that may be used by other Web applications over HTTP. These resources could include anything from phone directory data to weather data to sports results.

2- With asynchronous interaction between the customer and the server, Internet applications can be made richer، smaller, faster and more user-friendly.

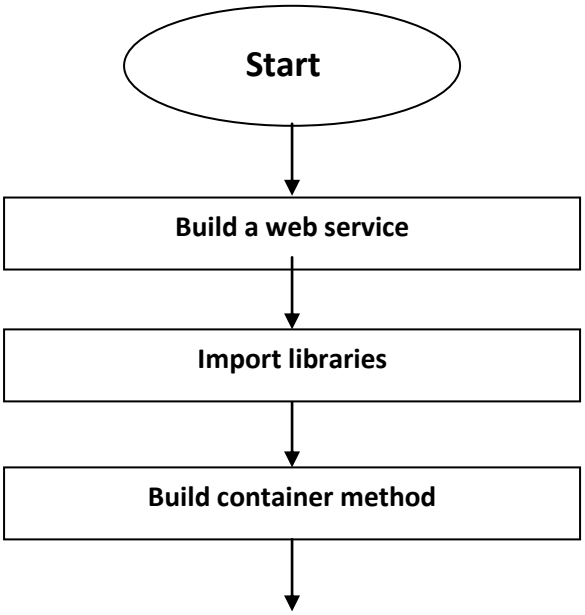
3- The traditional methods to implement file uploading to server involves Postback operation but only the use of asynchronous file upload through web services strategy satisfies the simulation of Ajax technology (without Postback).

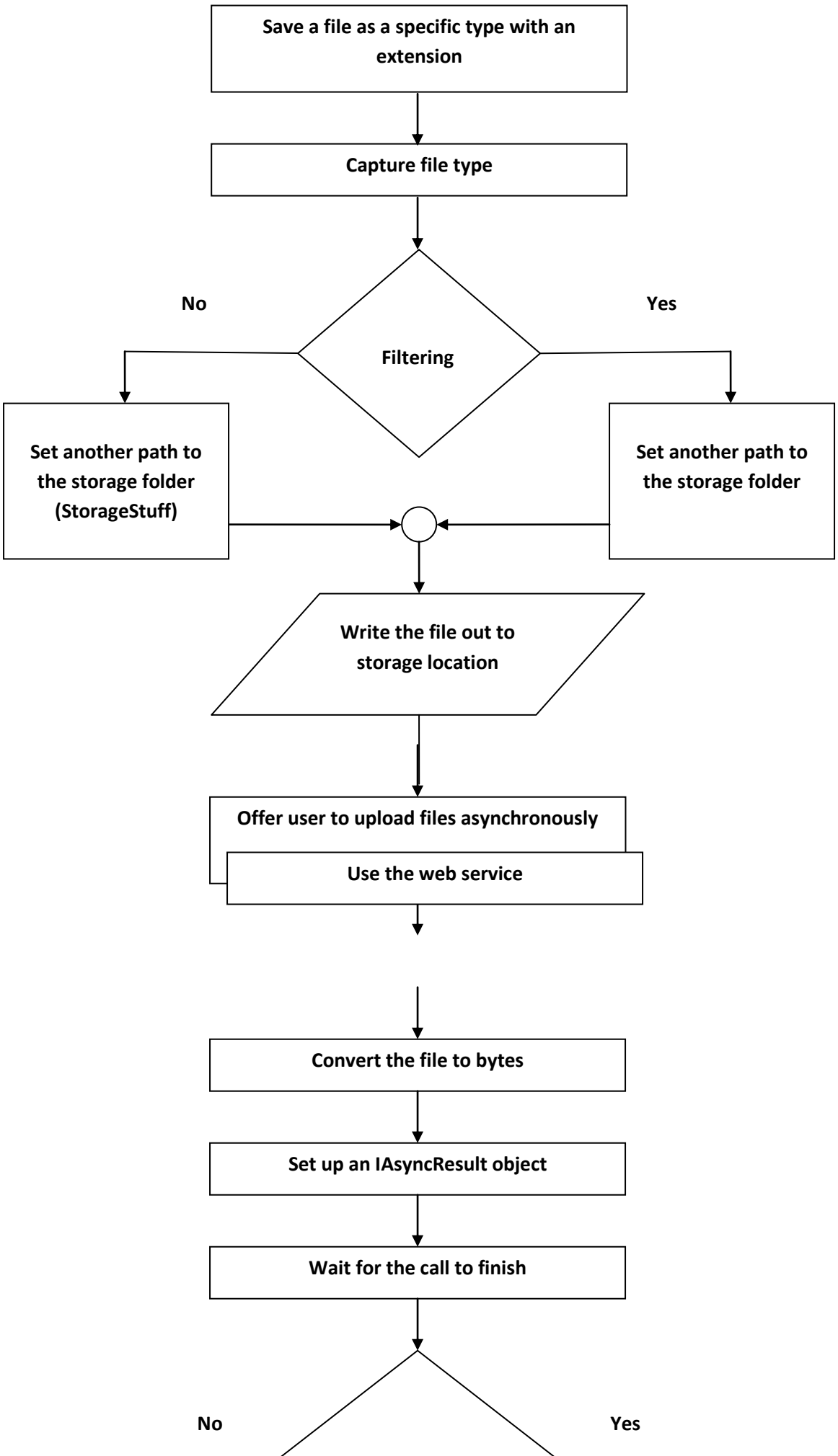
4- The suggested method of uploading files through web services will implement asynchronously the thing that enabled the web page to stay in front of the client who can implement any desired changes to the rest controls in the same web page in asynchronously manner while the uploading file operation incessant.

5- The file should be converted into bytes then uploaded, and the uploading period depends on the size of the file.

References

1. Edmond Woychowsky, (2006). "Ajax: Creating Web Pages with Asynchronous JavaScript and XML", Prentice Hall, Government Sales, August 08,
2. Ryan Asleson; Nathaniel Schutta, (2006), "Beginning Ajax with ASP.NET", Wiley Publishing, Inc.10475 Crosspoint Boulevard,
3. Andy Budd; Cameron Moll; Simon Collison, CSS Mastery: (2006), "Advanced Web Standards Solutions", Prentice Hall, U.S. Corporate & Government Sales,
4. Ingo Rammer; Mario Szpuszta, (2005), (Microsoft), "Advanced .NET Remoting", 1st Edition, Apress, March,.
5. Ralf Westphal, (2004), Christian Weyer, "Programming ASP.NET: Building Web Applications & Services Using C# and VB.NET", Addison Wesley, July.
6. Christian Nagel., (2003), "ASP to ASP.NET Migration Handbook", Wrox, February 18,.
7. Christian Nagel, (2004) "Professional .NET Network Programming", 1st Edition, Wrox, September 30,.
8. Ingo Rammer, (2002), "Advanced .NET Remoting in VB.NET", Apress, July 29,.


Start
Build a web service
Import libraries
Build container method


Save a file as a specific type with an extension
Capture file type
Filtering
No
Yes
Set another path to the storage folder (StorageStuff)
Set another path to the storage folder
Write the file out to storage location
Offer user to upload files asynchronously
Use the web service
Convert the file to bytes
Set up an IAsyncResult object
Wait for the call to finish
No
Yes

File uploaded completely?

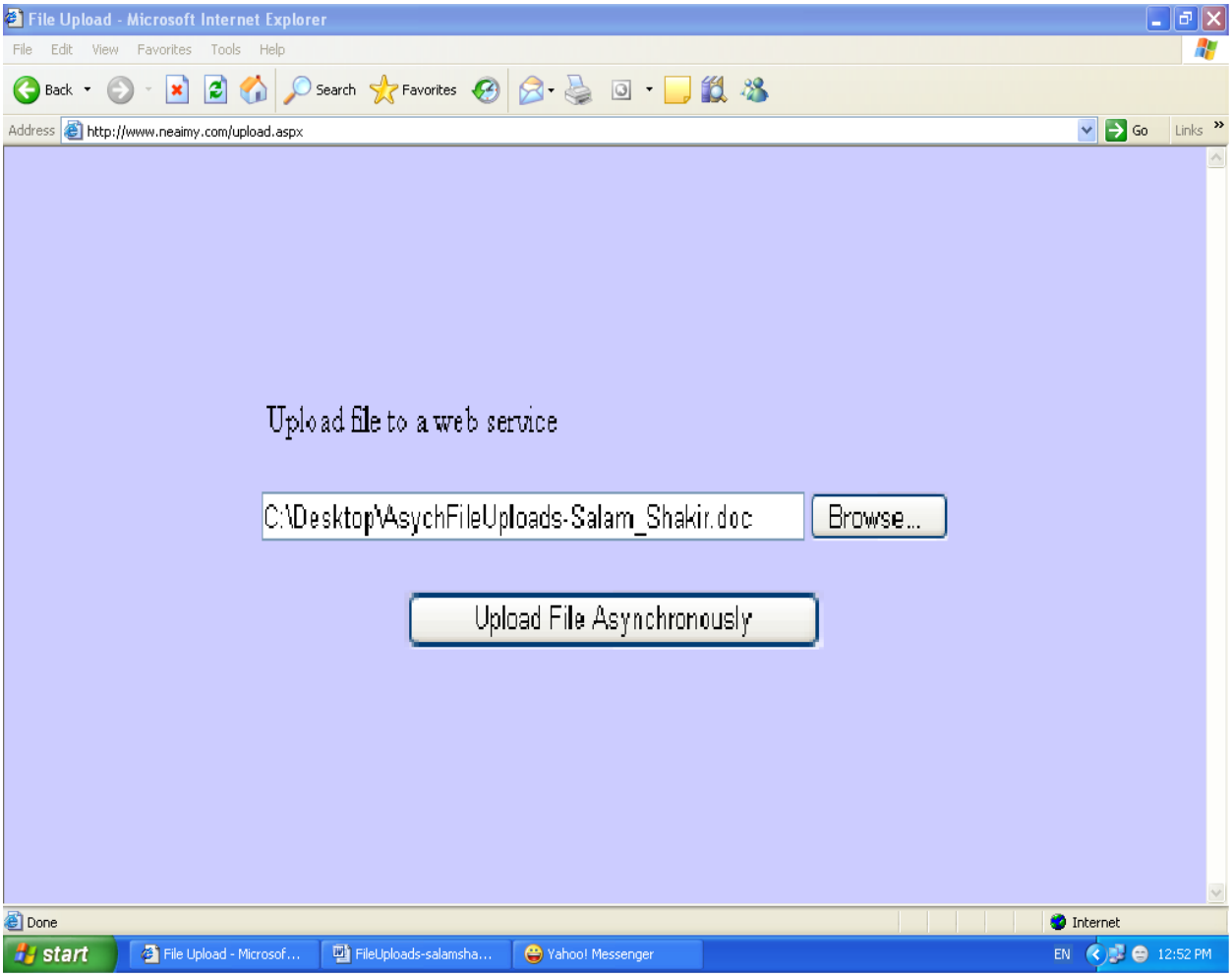


Fig. (2) :The web site for the suggested Asynchronous File Upload in