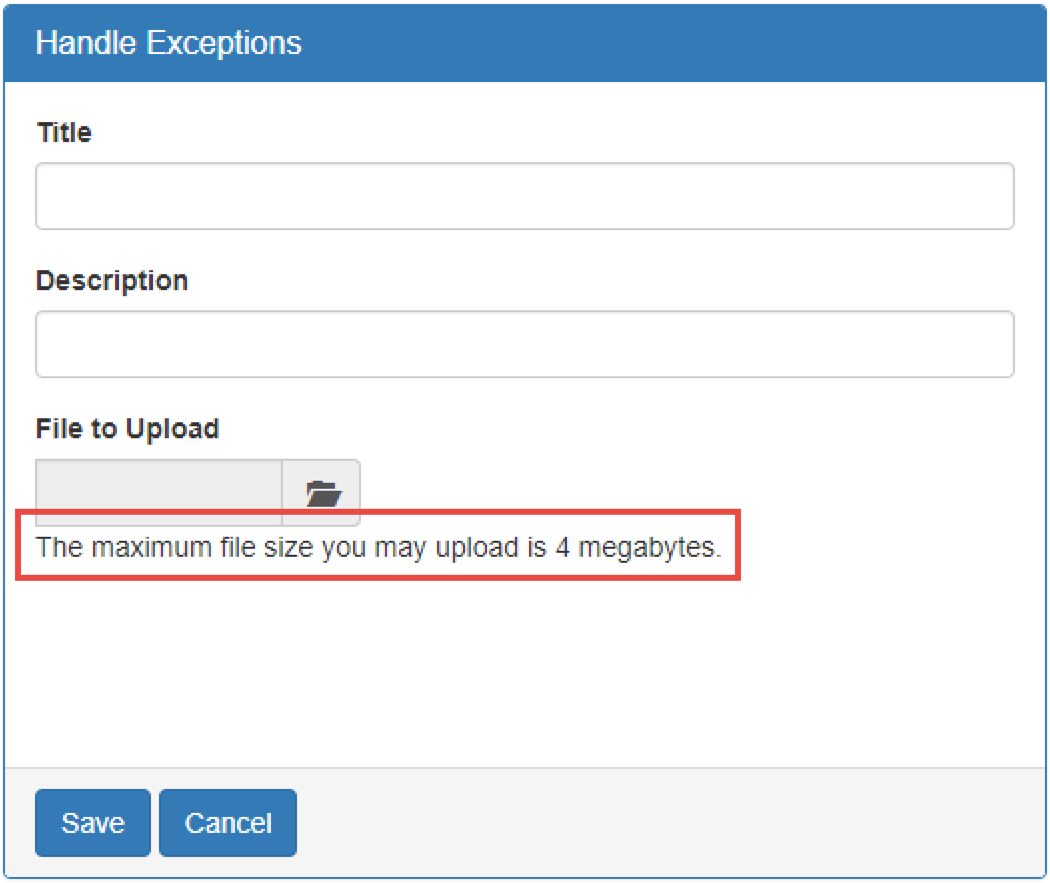
Uploading Files with MVC - Part 6

In this blog series, you have assumed that everything has gone correctly when uploading files. However, if you attempt to upload a file that is too large, you receive an error from your web server. In this blog post you are going to learn how to get the maximum size of file allowed, display that value on the page, handle an error when the file is too large, and modify the maximum size of file allowed. If you have not done so already, please download the sample from Part 5 so you can follow along with this blog post.

Display Maximum File Upload Size

ASP.NET has a limit of 4mb on the size of file that may be uploaded. You may change this value, but this is the default size. It is a good idea to let your user know the maximum size of file they are allowed to upload before they attempt the upload as shown in Figure 1.



Handle Exceptions

Title

Description

File to Upload

The maximum file size you may upload is 4 megabytes.

Figure 1: Display the maximum file size the user is allowed to upload.

Get Max Upload Setting

In your controller, add a method to retrieve the *maxRequestLength* attribute from the httpRuntime element. The method should look like the following:

```
private int GetMaxFileUploadSize()
{
  int maxRequestLength = 4096;  // Default size

  HttpRuntimeSection section =
    ConfigurationManager.GetSection("system.web/httpRuntime")
       as HttpRuntimeSection;
  if (section != null) {
    maxRequestLength = section.MaxRequestLength;
  }

  return Convert.ToInt32(maxRequestLength / 1024);
}
```

This method use the ConfigurationManager.GetSection() method to return the <system.web> | <httpRuntime> element from the Web.config file. As this is a known .NET element, you can cast the returned object into an HttpRuntimeSection object. From this HttpRuntimeSection object you can retrieve the MaxRequestLength property and divide that number by 1024 to get the max size in megabytes.

Modify View Model

To display the maximum file upload size on the web page you need to create a property in your view model to hold this value. Add a property to your view model class named *MaxFileUploadSize*.

```
public int MaxFileUploadSize { get; set; }
```

Modify Controller

In the controller where you display your web page, add code to call the GetMaxFileUploadSize() method and place the return value into the *MaxFileUploadSize* property in your view model as shown below:

```
public ActionResult Sample01()
{
  ExceptionsViewModel vm = new ExceptionsViewModel();

  vm.MaxFileUploadSize = GetMaxFileUploadSize();

  return View(vm);
}
```

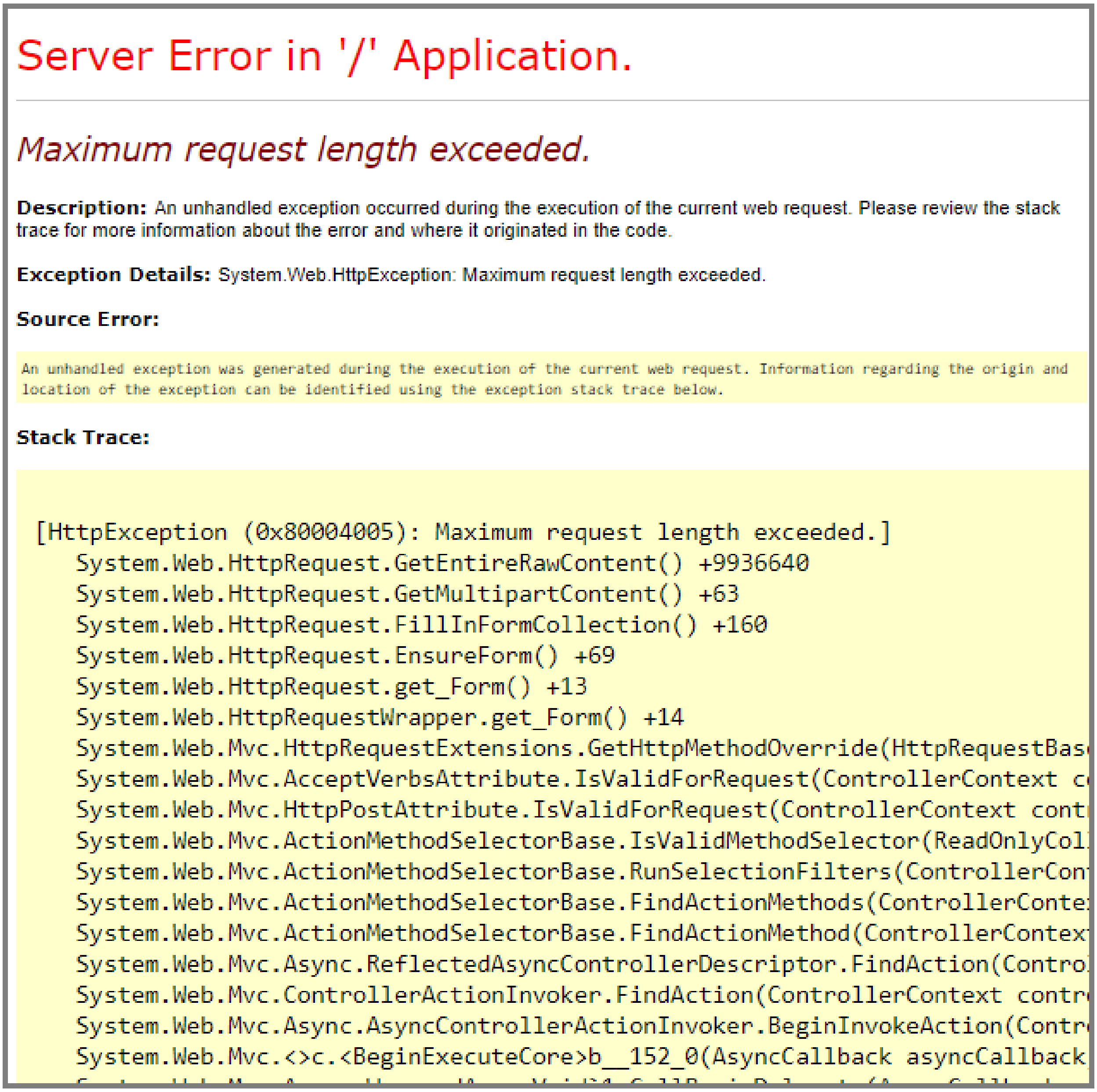
Modify Page

With the property set in the view model, you can now use this value to display the maximum size of file to upload in a span just below the file input control on your page.

```
<span>
  The maximum file size you may upload is
  @Model.MaxFileUploadSize megabytes.
</span>
```

Handle Exceptions

If you don't have error handling in your application, and you try to upload a file that exceeds the maximum file size, you will see an error page from MVC that looks like Figure 2. This is not what you want to show to your users as this reveals too much about your application and looks unprofessional. Instead, you should build your own custom error page.



Server Error in '/' Application.

Maximum request length exceeded.

Description: An unhandled exception occurred during the execution of the current web request. Please review the stack trace for more information about the error and where it originated in the code.

Exception Details: System.Web.HttpException: Maximum request length exceeded.

Source Error:

```
An unhandled exception was generated during the execution of the current web request. Information regarding the origin and
location of the exception can be identified using the exception stack trace below.
```

Stack Trace:

```
[HttpException (0x80004005): Maximum request length exceeded.]
   System.Web.HttpRequest.GetEntireRawContent() +9936640
   System.Web.HttpRequest.GetMultipartContent() +63
   System.Web.HttpRequest.FillInFormCollection() +160
   System.Web.HttpRequest.EnsureForm() +69
   System.Web.HttpRequest.get_Form() +13
   System.Web.HttpRequestWrapper.get_Form() +14
   System.Web.Mvc.HttpRequestExtensions.GetHttpMethodOverride(HttpRequestBas
   System.Web.Mvc.AcceptVerbsAttribute.IsValidForRequest(ControllerContext c
   System.Web.Mvc.HttpPostAttribute.IsValidForRequest(ControllerContext cont
   System.Web.Mvc.ActionMethodSelectorBase.IsValidMethodSelector(ReadOnlyCol
   System.Web.Mvc.ActionMethodSelectorBase.RunSelectionFilters(ControllerCon
   System.Web.Mvc.ActionMethodSelectorBase.FindActionMethods(ControllerConte
   System.Web.Mvc.ActionMethodSelectorBase.FindActionMethod(ControllerContex
   System.Web.Mvc.Async.ReflectedAsyncControllerDescriptor.FindAction(Contro
   System.Web.Mvc.ControllerActionInvoker.FindAction(ControllerContext contr
   System.Web.Mvc.Async.AsyncControllerActionInvoker.BeginInvokeAction(Contr
   System.Web.Mvc.<>c.<BeginExecuteCore>b__152_0(AsyncCallback asyncCallback
```

Figure 2: The page displayed when you have an unhandled exception.

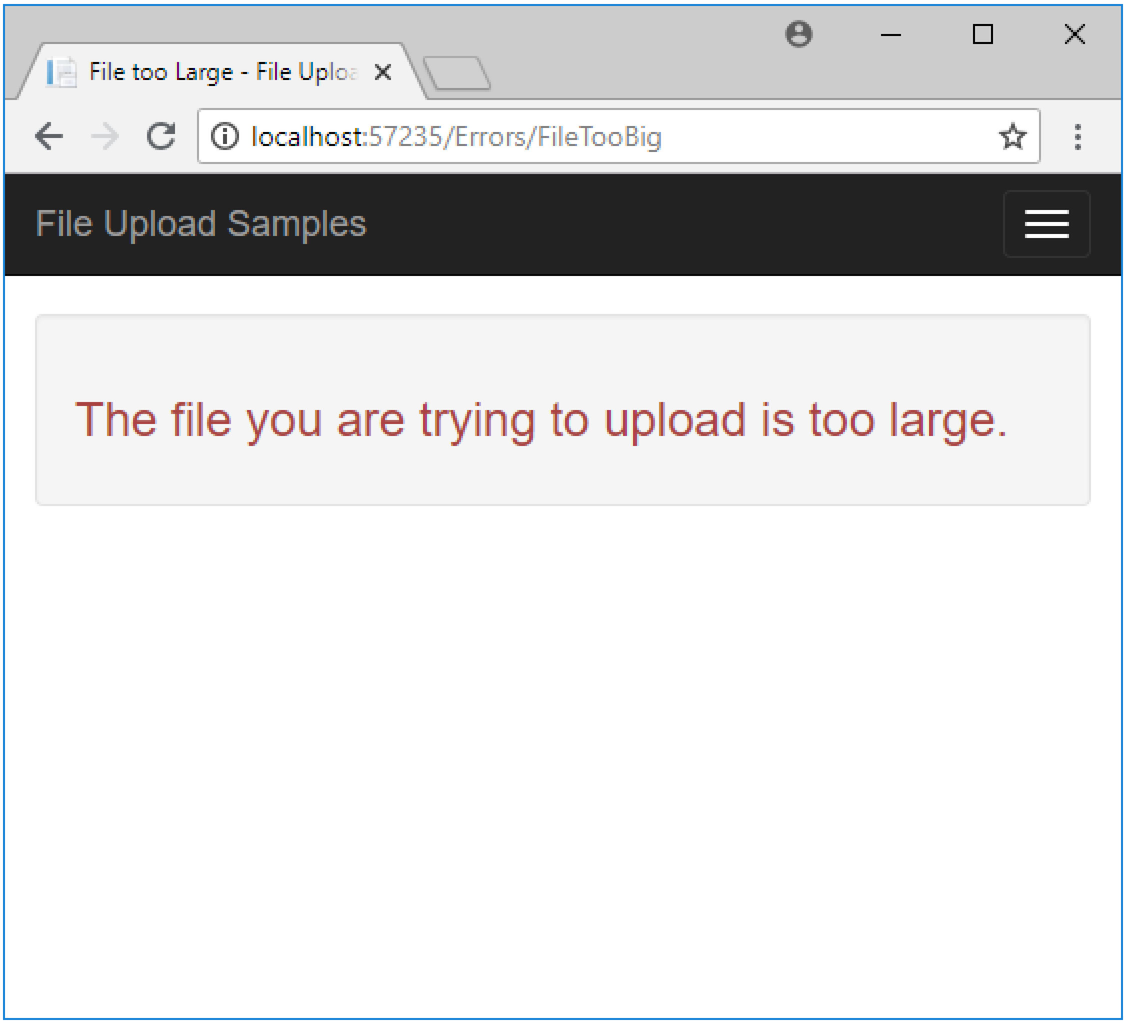
Add Error Page

You probably have a global error handler in your Global.asax file. However, this probably just gives the user a generic page to display that does not inform them what specifically went wrong. It is better to provide a page that tells the user that the file they were attempting to upload is too large.

Create an \Errors folder in the \Views folder and add a page named FileTooBig.cshtml. Add the following code into this page. The code creates a page that looks like Figure 3 when the user tries to upload a file that is too large.

```
@{
  ViewBag.Title = "File too Large";
}

<div class="row">
  <div class="col-xs-12">
    <div class="well alert alert-danger">
      <h3>The file you are trying to upload is too large.</h3>
    </div>
  </div>
</div>
```



The file you are trying to upload is too large.

Figure 3: Add your own custom error page to inform the user the file is too large.

Add Error Controller

In the \Controllers folder add a new MVC controller class named ErrorsController.cs. Add the code shown below to display the FileTooBig page.

```
using System.Web.Mvc;

namespace FileUploadSamples.Controllers
{
  public class ErrorsController : Controller
  {
    public ActionResult FileTooBig()
    {
      return View();
    }
  }
}
```

Add Global Exception Handling

You can only catch a file too large error within the global error handler. Open the Global.asax file and add (or modify) the Application_Error event. Call the Server.GetLastError() method to retrieve the last error thrown on the server. Cast it as an HttpException object. Next, check to see if the *WebEventCode* property is a RuntimeErrorPostTooLarge exception. If it is, clear the server error and redirect to the FileTooBig page.

```
protected void Application_Error(object sender, EventArgs e)
{
  // Get last server error
  HttpException err = Server.GetLastError() as HttpException;
  // Is the error because the file posted is too large?
  if (err != null &&
      err.WebEventCode == WebEventCodes.RuntimeErrorPostTooLarge) {
    // Clear server error
    Server.ClearError();
    // Redirect to a "File too Big" page
    ((HttpApplication)sender).Context
       .Response.Redirect("~/Errors/FileTooBig");
  }
}
```

Set Max Upload File Size

If you wish to increase the size of file from the default 4MB, modify this value by adding an <httpRuntime> element within the <system.web> element and specify the *maxRequestLength* in kilobytes. For example, to allow a 20MB file to be uploaded, set the *maxRequestLength* attribute to 20,480 as shown below.

```
<system.web>
  <httpRuntime maxRequestLength="20480" />
</system.web>
```

Besides setting the maximum upload length, you may also need to set the number of seconds before your request times out. As you can imagine, if your user is uploading a large file, the time required to upload will be longer. The default request length is 110 seconds. You might need to increase this value in order to accommodate larger files. Add the *executionTimeout* attribute to the httpRuntime element and set the value to 3600 seconds (5 minutes) for example. The sample below shows both attributes set.

```
 <httpRuntime executionTimeout="3600"
              maxRequestLength="20480" />
```

IIS 7+ Settings

If you are using IIS 7 or later, you also need to set the *maxAllowedContentLength* attribute in the requestLimits element to the same amount of data you are allowing in the *maxRequestLength* attribute. This value is set in bytes, so if you want to set this to 20MB, you need to use the following formula: 20 x 1024 x 1024 = 20,971,520. Below are the elements you need to add within the <system.webServer> element.

```
<system.webServer>
  <security>
    <requestFiltering>
      <requestLimits maxAllowedContentLength="20971520" />
    </requestFiltering>
  </security>
</system.webServer>
```

Summary

In this blog post you learned how to handle file upload exceptions and how to increase the size of the file allowed to upload. You should read the maximum size allowed and display that size to your user on the page where they are attempting to upload a file. You should be specific on your error messages, so add a web page to display a "file too large" error to the user. Handle file too large exceptions in Application_Error event located in the Global.asax file.

Sample Code

You can download the complete sample code at my website. http://www.fairwaytech.com/downloads. Choose "PDSA/Fairway Blog", then "Uploading Files using MVC - Part 6" from the drop-down.