HOWTO: Remotely Invoke ASP.NET Page Methods (Legacy Web Controls)

Legacy Controls NOTICE

This article references our legacy Web Forms Web Viewing controls (WebImageViewer, WebAnnotationViewer, WebThumbnailViewer). It is preserved for archival purposes, but support strongly recommends using our modern HTML5 web controls: WebDocumentViewer, WebDocumentThumbnailer instead)

INFO: WebDocumentViewer Whitepaper - Getting Started With Web Viewing

Main Article Content

The WebImageViewer control provides the ability to call methods in the owning ASP.NET Page object via client side JavaScript. In addition to being able to send typed information to the Page object, the return value for the remote method is sent back to the calling JavaScript code.

Terminology

The terms listed below are used in the documentation that follows.

Term	Definition
Server side	Code or objects that are invoked on an ASP web server
Client side	Code or objects that are invoked in a user's web browser
Parameter	Value that is passed from one function or method to another
Signature	Combination of parameter types and the return type of a function or method

Preparing a Server Side Method for Remote Invocation

To invoke a method remotely, it must meet the following criteria:

- The method must be a member of a Page object that contains a WebImageViewer
- The method must be public
- The method must be marked with the RemoteInvokable attribute

- Parameters of the method must be one of the following types:
  - int
  - double
  - bool
  - string
- The method must return a type that can be converted to a string. Null or no return value are also acceptable.

An example of a possible method is shown below.

Atalasoft.Imaging.WebControls.RemoteInvokable

```
publicbool WaterMark(int x, int y, string message) { ... }
```

Calling a Method from JavaScript

In order to invoke a method within a server side Page object from JavaScript, the client side code must call the RemoteInvoke() method of the JavaScript object atalaWebImageViewer. The first argument is a string representing the name of the method to invoke. The second argument is an array of values that is passed to the remote method.

An example of a client side remote invocation is provided below.

```
ebImageViewer1.RemoteInvoke("WaterMark", new Array(100, 100, "Preview Only"));
```

Getting the Return Value from a RemoteInvoke

RemoteInvokable() methods can have a return value, as long as they return a type that can be converted to a string. Because the return value is populated asynchronously, the JavaScript WebImageViewer.RemoteInvoked event needs to be handled. An example is shown below.

C#

```
ebImageViewer1.RemoteInvoked = OnRemoteInvoked; function OnRemoteInvoked(){ var success =
WebImageViewer1.getReturnValue(); if (success == true){ alert('WaterMark Succeeded.'); } else
{ alert('WaterMark Failed.') } }
```

Parameter Type Conversion

JavaScript has a limited number of built-in data types that can be readily identified within a client side script. These are number, bool, and string. The JavaScript method RemoteInvoke() bundles up each parameter with information about its data type so that it can be correctly used on the server side. Server side code makes further effort to automatically distinguish between the JavaScript notion of a generic number and the .NET notion of an integer or a floating point number. If a JavaScript number arrives on the server which contains a decimal or an exponent, it will automatically be promoted to a floating-point number. Otherwise, the number is assumed to be an integer.

No attempt is made to interpret the contents of a string.

Method Identification

.NET languages can define functions or methods with the same name but different signatures. These are called overloaded methods. Server side code attempts to find the version of a method that best matches the parameters passed from JavaScript. The match happens in two stages. Server side code first tries to find an exact match where each client-passed parameter type matches the server side parameter type exactly. If there are no matches, server side code then tries to find a method for which numeric parameters can be converted without loss of information.

Example

If the client side includes this remote invocation:

```
ebImageViewer1.RemoteInvoke("Overload", new Array(1, 2));
```

and the server side has the following methods defined:

```
Atalasoft.Imaging.WebControls.RemoteInvokable] publicstring Overload(int a, int b) { ... }
[Atalasoft.Imaging.WebControls.RemoteInvokable] publicstring Overload(double a, double b) {
... }
```

then the RemoteInvoke matches the first method, since it takes two integers as parameters.

If, instead, the client side had the following remote invocation:

```
ebImageViewer1.RemoteInvoke("Overload", new Array(1.0, 2));
```

then the RemoteInvoke matches the second method although it is not a perfect match.

See Also

How to: Work with Remote Events

Original Article:

Q10360 - HOWTO: Remotely Invoke ASP.NET Page Methods (Legacy Web Controls)

Atalasoft Knowledge Base
https://www.atalasoft.com/kb2/KB/50127/HOWTO-Remotely-Invoke-ASPNET-Page-Me...