



PDF SHARE FORMS

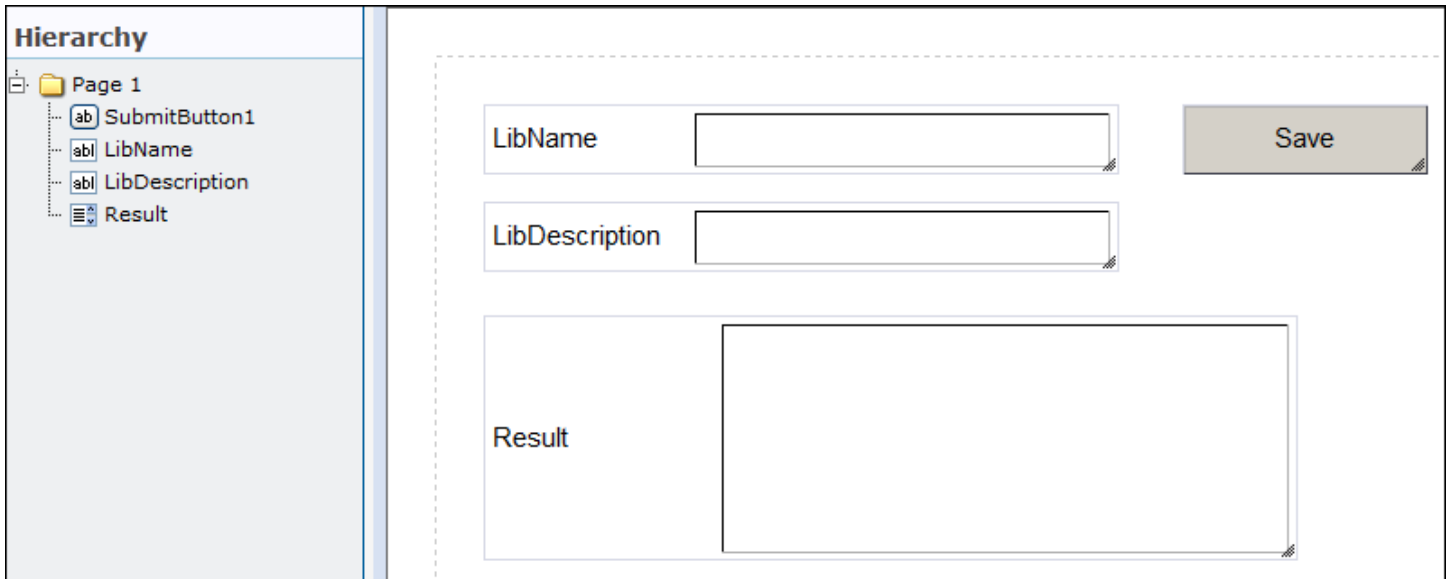
Online, Offline, OnDemand

PDF forms and SharePoint are better together

Invoking SharePoint web services through PDF form

The idea of this guide is to show how to call web services from PDF form. This example is based on SharePoint web services – creating new library using “AddList” operation.

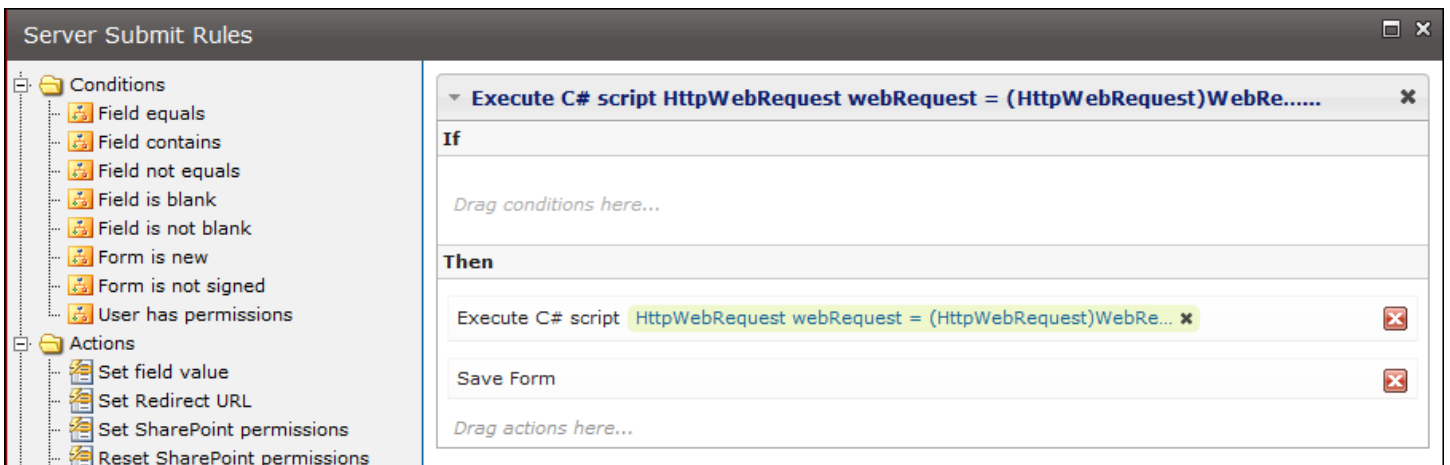
Step 1. Create PDF template with following fields:



The screenshot shows a PDF form design interface. On the left, a 'Hierarchy' pane lists the form's structure: 'Page 1' containing 'SubmitButton1', 'LibName', 'LibDescription', and 'Result'. The main workspace displays a form layout with three fields: a text field for 'LibName', a text field for 'LibDescription', and a larger multiline text field for 'Result'. A 'Save' button is positioned to the right of the 'LibName' field.

Template has submit button, two text field and multiline text field. Text fields “LibName” and “LibDescription” will be used to set name and description for newly created library. Multiline text field “Result” will confirm successful library creation, or display error, if it will occur.

Step 2. Add script action to “Form Submit” event (PDF Form Tools → Developer → Form Submit):



The screenshot shows the 'Server Submit Rules' configuration window. The left pane lists 'Conditions' and 'Actions'. The right pane shows a rule configuration for 'Execute C# script HttpWebRequest webRequest = (HttpWebRequest)WebRe.....'. Under the 'If' section, there is a placeholder 'Drag conditions here...'. Under the 'Then' section, two actions are listed: 'Execute C# script HttpWebRequest webRequest = (HttpWebRequest)WebRe...' and 'Save Form', both with a red 'X' icon to their right. A placeholder 'Drag actions here...' is visible below the actions.

Place "Execute script" action above "Save Form" action and add following code:

This part is responsible for generating header, obtaining required operation and credentials:

```
HttpWebRequest webRequest =
(HttpWebRequest)WebRequest.Create("http://demo.pdfsharepoint.com/demo/pfbvbxke/_vti_bin/Lists.
asmx");
webRequest.Headers.Add("SOAPAction", "http://schemas.microsoft.com/sharepoint/soap/AddList");
webRequest.ContentType = "text/xml; charset=utf-8";
webRequest.Method = "POST";
NetworkCredential nc= new NetworkCredential("accountName", "password", "domain");
webRequest.Credentials = nc;
webRequest.Accept = "text/xml";
```

This part is responsible for getting library name, description and type. Also it will create required library:

```
try
{
  XmlDocument soapEnvelopeXml = new XmlDocument();
  string CreateDocumentListSOAPTemplate =
    @"<?xml version='1.0' encoding='utf-8'?>
<soap:Envelope xmlns:xsi='http://www.w3.org/2001/XMLSchema-instance'
xmlns:xsd='http://www.w3.org/2001/XMLSchema'
xmlns:soap='http://schemas.xmlsoap.org/soap/envelope/'>
<soap:Body>
<AddList xmlns='http://schemas.microsoft.com/sharepoint/soap/'>
<listName>#token_DirectoryListName#</listName>
<description>#token_DirectoryDescription#</description>
<templateID>#token_DirectoryType#</templateID>
</AddList>
</soap:Body>
</soap:Envelope>";
  string listName = data.resolveNode("LibName").value;
  string listDescription = data.resolveNode("LibDescription").value;
  string listType = "101";
  soapEnvelopeXml.LoadXml(CreateDocumentListSOAPTemplate.Replace("#token_DirectoryListName#",
listName).Replace("#token_DirectoryDescription#",
listDescription).Replace("#token_DirectoryType#", listType));
  using (Stream stream = webRequest.GetRequestStream())
  {
    soapEnvelopeXml.Save(stream);
  }
  using (WebResponse response = webRequest.GetResponse())
  {
    using (StreamReader rd = new StreamReader(response.GetResponseStream()))
    {
      string soapResult = rd.ReadToEnd();
    }
  }
  data.resolveNode("Result").value = "Library " + data.resolveNode("LibName").value + " has
been created!";
}
```

This is the last part – it will write error message to the Result multiline text field if some error will occur during creation of library:

```
catch (Exception exception)
{
    //Handle exception
    data.resolveNode("Result").value = exception.ToString();
}
```

This script will use entered credentials to create new library using web services on selected site.

Please pay attention to **red** fragments in the code:

1. `HttpRequest webRequest = (HttpRequest)WebRequest.Create("http://demo.pdfsharepoint.com/demo/pfbvbxke/_vti_bin/Lists.asmx");` - this line of code contains site url + link to necessary web service. Use site url where you want changes to occur
2. `NetworkCredential nc= new NetworkCredential("accountName","password","domain");` - here you need to write account, its password and domain. This account will be used to access required service action.

Note: you can preview list of available web service actions here:

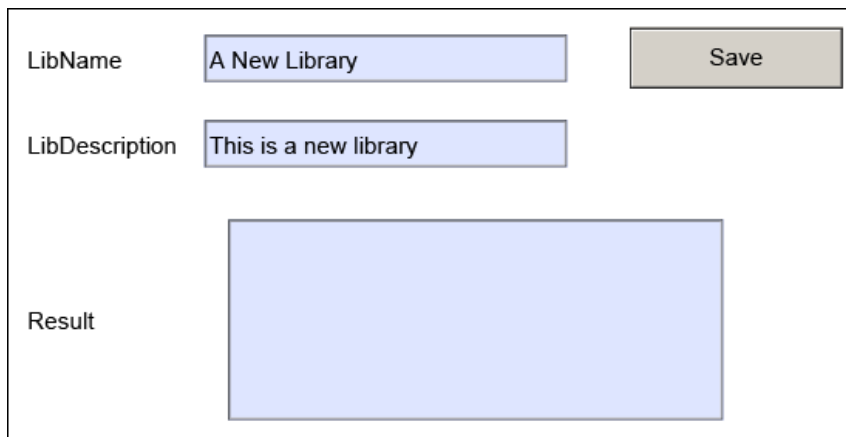
<http://blog.crsw.com/sharepoint/index-of-sharepoint-web-services/>

Full text of this script can be found in **Appendix 1** of this guide.

Step 3. Save template and deploy to library.






Step 4. Runtime

Step 4.1. Create new form:



In this example “A New Library” was selected as name and “This is a new library” as description for newly created library.

Step 4.2. Save this form and check “All Site Content”:

		Items	Last Modified
Document Libraries			
	A New Library	This is a new library	0 6 minutes ago
	Claim Form Samples		0 7 months ago
	Customized Reports	This Document library has the templates to create Web Analytics custom reports for this site collection	9 6 months ago
	Excel Chart Sample		0 7 months ago
	External Forms		2 6 months ago

As the result, new library was created.

Appendix 1

```

HttpRequest webRequest =
(HttpWebRequest)WebRequest.Create("http://demo.pdfsharepoint.com/demo/pfbvbxke/_vti_bin/Lists.asmx");
webRequest.Headers.Add("SOAPAction", "http://schemas.microsoft.com/sharepoint/soap/AddList");
webRequest.ContentType = "text/xml;charset=utf-8";
webRequest.Method = "POST";
NetworkCredential nc= new NetworkCredential("accountName","password","domain");
webRequest.Credentials = nc;
webRequest.Accept = "text/xml";

try
{
    XmlDocument soapEnvelopeXml = new XmlDocument();
    string CreateDocumentListSOAPTemplate =
        @"<?xml version='1.0' encoding='utf-8'?>
<soap:Envelope xmlns:xsi='http://www.w3.org/2001/XMLSchema-instance'
xmlns:xsd='http://www.w3.org/2001/XMLSchema' xmlns:soap='http://schemas.xmlsoap.org/soap/envelope/'>
<soap:Body>
<AddList xmlns='http://schemas.microsoft.com/sharepoint/soap/'>
<listName>#token_DirectoryListName#</listName>
<description>#token_DirectoryDescription#</description>
<templateID>#token_DirectoryType#</templateID>
</AddList>
</soap:Body>
</soap:Envelope>";
    string listName = data.resolveNode("LibName").value;
    string listDescription = data.resolveNode("LibDescription").value;
    string listType = "101";
    soapEnvelopeXml.LoadXml(CreateDocumentListSOAPTemplate.Replace("#token_DirectoryListName#",
listName).Replace("#token_DirectoryDescription#", listDescription).Replace("#token_DirectoryType#", listType));
    using (Stream stream = webRequest.GetRequestStream())
    {
        soapEnvelopeXml.Save(stream);
    }
    using (WebResponse response = webRequest.GetResponse())
    {
        using (StreamReader rd = new StreamReader(response.GetResponseStream()))

```



```
{
    string soapResult = rd.ReadToEnd();
}
}
data.resolveNode("Result").value = "Library " + data.resolveNode("LibName").value + " has been created!";
}
catch (Exception exception)
{
    //Handle exception
    data.resolveNode("Result").value = exception.ToString();
}
```