[MS-FSFDP]:
Forms Services Feature Detection Protocol Specification

Revision Summary

Date	Revision History	Revision Class	Comments
04/04/2008	0.1		Initial Availability
06/27/2008	1.0	Major	Revised and edited the technical content
12/12/2008	1.01	Editorial	Revised and edited the technical content
07/13/2009	1.02	Major	Revised and edited the technical content
08/28/2009	1.03	Editorial	Revised and edited the technical content
11/06/2009	1.04	Editorial	Revised and edited the technical content
02/19/2010	2.0	Minor	Updated the technical content
03/31/2010	2.01	Editorial	Revised and edited the technical content
04/30/2010	2.02	Editorial	Revised and edited the technical content
06/07/2010	2.03	Editorial	Revised and edited the technical content
06/29/2010	2.04	Editorial	Changed language and formatting in the technical content.
07/23/2010	2.05	Minor	Clarified the meaning of the technical content.
09/27/2010	2.05	No change	No changes to the meaning, language, or formatting of the technical content.
11/15/2010	2.05	No change	No changes to the meaning, language, or formatting of the technical content.
12/17/2010	2.05	No change	No changes to the meaning, language, or formatting of the technical content.

Table of Contents

1 Introduction

This document specifies the Forms Services Feature Detection Protocol, which enables a protocol client to perform the following three functions:

- **Form Server Detection:** Detect if form server features are present and enabled on the protocol server.
- **Form Server Version Retrieval:** Returns the form server version when the form server features are present and enabled on the protocol server.
- **Rendering URL Construction:** Construct the URL that is required to render a form in a Web browser.

1.1 Glossary

The following terms are defined in [MS-GLOS]:

ASCII
Augmented Backus-Naur Form (ABNF)
authentication
Hypertext Transfer Protocol (HTTP)
Hypertext Transfer Protocol over Secure Sockets Layer (HTTPS)
Unicode
UTF-8

The following terms are defined in [MS-OFCGLOS]:

absolute URI
form
form file
form server
form template (.xsn) file
message body
path component
query component
Request-URI
site
site collection
Status-Code
Status-Line
URI (Uniform Resource Identifier)
URL (Uniform Resource Locator)

The following terms are specific to this document:

rendering URL: The URL that is used to render an InfoPath form in a Web browser if the form cannot be opened by using Microsoft® InfoPath®.

MAY, SHOULD, MUST, SHOULD NOT, MUST NOT: These terms (in all caps) are used as described in [RFC2119]. All statements of optional behavior use either MAY, SHOULD, or SHOULD NOT.

1.2 References

1.2.1 Normative References

We conduct frequent surveys of the normative references to assure their continued availability. If you have any issue with finding a normative reference, please contact dochelp@microsoft.com. We will assist you in finding the relevant information. Please check the archive site, http://msdn2.microsoft.com/en-us/library/E4BD6494-06AD-4aed-9823-445E921C9624, as an additional source.

[RFC1945] Berners-Lee, T., Fielding, R., and Frystyk, H., "Hypertext Transfer Protocol -- HTTP/1.0", RFC 1945, May 1996, http://www.ietf.org/rfc/rfc1945.txt

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997, http://www.ietf.org/rfc/rfc2119.txt

[RFC2616] Fielding, R., Gettys, J., Mogul, J., et al., "Hypertext Transfer Protocol -- HTTP/1.1", RFC 2616, June 1999, http://www.ietf.org/rfc/rfc2616.txt

[RFC2818] Rescorla, E., "HTTP Over TLS", RFC 2818, May 2000, http://www.ietf.org/rfc/rfc2818.txt

[RFC3986] Berners-Lee, T., Fielding, R., and Masinter, L., "Uniform Resource Identifier (URI): Generic Syntax", STD 66, RFC 3986, January 2005, http://www.ietf.org/rfc/rfc3986.txt

[RFC5234] Crocker, D., Ed., and Overell, P., "Augmented BNF for Syntax Specifications: ABNF", STD 68, RFC 5234, January 2008, http://www.ietf.org/rfc/rfc5234.txt

1.2.2 Informative References

[MS-FSDAP] Microsoft Corporation, "Forms Services Design and Activation Web Service Protocol Specification", June 2008.

[MS-GLOS] Microsoft Corporation, "Windows Protocols Master Glossary", March 2007.

[MS-OFCGLOS] Microsoft Corporation, "Microsoft Office Master Glossary", June 2008.

1.3 Protocol Overview (Synopsis)

This protocol enables a protocol client to communicate with a protocol server over a **Hypertext Transfer Protocol (HTTP)** connection to perform the following supported functions:

- **Form Server Detection:** Using this protocol function, the protocol client can detect if **form server** features are present and enabled on the protocol server. The protocol client sends an HTTP request to the protocol server with a parameter to detect whether form server features are enabled, and the protocol server sends back an HTTP response containing the result.
- **Form Server Version Retrieval:** Using this protocol function, the protocol client can get the form server version after detecting if form server features are present and enabled on the protocol server. The protocol client sends an HTTP request to the protocol server with a parameter to get the form server version, and the protocol server sends back an HTTP response containing the result.
- **Rendering URL Construction:** Using this protocol function, the protocol client can construct the **URL** that is required to render a new or existing **form(2)** in a Web browser. The protocol client sends an HTTP request to the protocol server, and the protocol server sends back an HTTP response containing a URL that can be used to render the form in a Web browser.

1.4 Relationship to Other Protocols

For message transport, this protocol uses the HTTP/1.0 protocol as described in [RFC1945], the HTTP/1.1 protocol as described in [RFC2616], or the **Hypertext Transfer Protocol over Secure Sockets Layer (HTTPS)** protocol as described in [RFC2818].

The following diagram shows the underlying messaging and transport stack used by the protocol:

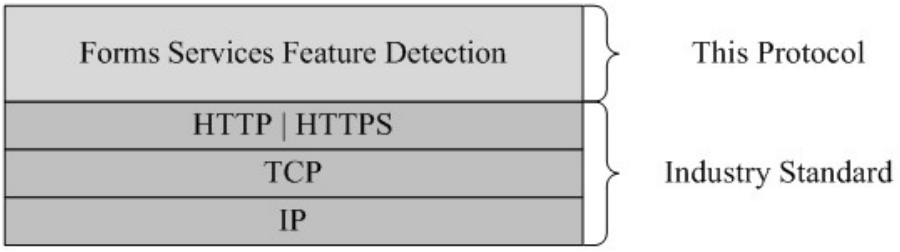

Figure 1: This protocol in relation to other protocols

This protocol is intended to be used as a prerequisite to calling the Forms Services Design and Activation Protocol, as described in [MS-FSDAP].

1.5 Prerequisites/Preconditions

This protocol operates against a **site (2)** that is identified by a URL that is known by protocol clients. The protocol server endpoint is formed by appending "/_layouts/FormServerDetector.aspx" to the URL of the site, for example http://www.contoso.com/Repository/_layouts/FormServerDetector.aspx.

This protocol assumes that **authentication (2)** has been performed by the underlying protocols.

1.6 Applicability Statement

The protocol client can use this protocol in the following scenarios:

- **Form Server Detection:** This function can be used by the protocol client to detect if the protocol server supports form server functionality.
- **Form Server Version Retrieval:** This function can be used by the protocol client to determine if the protocol server supports version-specific functionality.
- **Rendering URL Construction:** This function can be used to construct the required URL for rendering a form in a Web browser if the form cannot be opened by the protocol client.

1.7 Versioning and Capability Negotiation

This document covers versioning issues in the following areas:

- Supported Transports: This protocol uses multiple transports with either HTTP or HTTPS as described in section 2.1, Transport.

1.8 Vendor-Extensible Fields

None.

1.9 Standards Assignments

None.

2 Messages

The following sections specify the message syntax and the details of how the protocol messages are transported.

2.1 Transport

Protocol servers MUST support HTTP as specified in [RFC1945] or [RFC2616]. Protocol servers SHOULD additionally support HTTPS, as specified in [RFC2818], for securing communication with clients.

2.2 Message Syntax

The following subsections specify the different parts of HTTP request and response messages.

2.2.1 Request Syntax

2.2.1.1 Request HTTP Version

The HTTP Version as specified in [RFC1945], section 3.1 or [RFC2616], section 3.1 MUST be either "HTTP/1.0" or "HTTP/1.1" for the requests that use this protocol.

2.2.1.2 Request HTTP Method

The HTTP Method as specified in [RFC1945], section 8 or [RFC2616], section 9 MUST be GET ([RFC1945], section 8.1 or [RFC2616], section 9.3) for the requests that use this protocol.

2.2.1.3 Request-URI Syntax

2.2.1.3.1 Request-URI Details

The **Request-URI** MUST be a valid **URI** as specified in [RFC3986].

The following **ABNF** [RFC5234] specifies the syntax of the Request-URI.

```
Request-URI  = (path) "?" (query-component)
path         = (base) "/_layouts/FormServerDetector.aspx"
base         = path-absolute
```

The **path component** in the Request-URI MUST end with "/_layouts/FormServerDetector.aspx". The **base** ABNF rule identifies the **site collection** on the server to which the request was made. ABNF for **path-absolute** is specified in [RFC3986], section 3.3. The **query component(1)** in the Request-URI is specified in the following section.

2.2.1.3.2 Query Component Details

The query component of the Request-URI MUST be present. The following subsections specify the syntax of the query component for the three functions that are supported by this protocol.

The protocol server MUST interpret the parameters and values in the query component as case-insensitive.

2.2.1.3.2.1 Request for Form Server Detection

To use the Form Server Detection function, the query component of the Request-URI MUST have the *IsFormServerEnabled* query parameter, and the value of this parameter MUST be "check".

The following ABNF specifies the syntax that the query component and its query parameters MUST adhere to.

```
query-component = (query-parameter)
query-parameter = "IsFormServerEnabled=check"
```

If the protocol client passes any query parameters in addition to the *IsFormServerEnabled* parameter, the protocol server MUST ignore these additional parameters and the request for form server detection MUST take precedence.

2.2.1.3.2.2 Request for Form Server Version Retrieval

The protocol server SHOULD<1> support the Form Server Version Retrieval function. To use this method, the query component of the Request-URI MUST have the *FormServerVersion* parameter, and the value of this parameter MUST be "check".

The following ABNF specifies the syntax that the query component and its query parameters MUST adhere to.

```
query-component = (query-parameter)
query-parameter = "FormServerVersion=check"
```

If the *FormServerVersion* query parameter is not supported by the protocol server, this query parameter MUST be ignored. If no other supported query parameters exist in the query component, the protocol server MUST return **Status-Code** 204, as specified in section 2.2.2.1.1, Success Response.

If the protocol client passes the *IsFormServerEnabled* parameter, the request for form server detection MUST take precedence.

If the *FormServerVersion* parameter is supported by the protocol server and the protocol client passes additional parameters, other than the *IsFormServerEnabled* parameter, the protocol server MUST ignore these additional parameters.

2.2.1.3.2.3 Request for Rendering URL Construction

To use the Rendering URL Construction function, the query component and its query parameters MUST adhere to the syntax that is specified in the following ABNF.

```
query-component        = (xmlLocation-parameter / xsnLocation-parameter)
   ["&" saveLocation-parameter]
xmlLocation-parameter  = "XmlLocation=" (value)
xsnLocation-parameter  = "XsnLocation=" (value)
saveLocation-parameter = "SaveLocation=" (value)
value                  = path-absolute / path-rootless
```

ABNF for **path-absolute** and **path-rootless** rules is specified in [RFC3986], section 3.3.

As specified in the preceding ABNF, the query component supports the three parameters—*XmlLocation*, *XsnLocation*, and *SaveLocation*. The following table specifies the meaning of these parameters.

Parameter	Description
XmlLocation	The path to a **form file** on the protocol server. MUST be an **ASCII** string that specifies the location of the form file that needs to be rendered on the protocol server. MUST follow the format as specified in [RFC3986].
XsnLocation	The path to a **form template (.xsn) file** on the protocol server. MUST be an ASCII string that specifies the location of the form template, which can be used to generate a form file to be rendered on the protocol server. MUST follow the format as specified in [RFC3986].
SaveLocation	The path to a folder on the protocol server in the same site collection as FormServerDetector.aspx. MUST be an ASCII string that specifies the location where the form file can be saved, if needed. MUST follow the format as specified in [RFC3986].

The values of the parameters in the query component MUST NOT contain any un-escaped characters that are listed as "reserved" in [RFC3986], section 2.2.

To construct the **rendering URL** for an existing form, the *XmlLocation* parameter MUST be specified. To construct the rendering URL for a new form, the *XsnLocation* parameter MUST be specified. In both cases, the *SaveLocation* parameter is optional.

Any other combination of the supported parameters MUST be treated as invalid input, and in such a case, the protocol server MUST return Status-Code 204, as specified in section 2.2.2.1.1, Success Response. Any additional parameters, other than those specified in Table 1 in this section, MUST be ignored by the protocol server.

The protocol server MUST NOT require that the parameters appear in a particular order.

2.2.1.4 Request Headers Syntax

The following request header is relevant to this protocol:

- Accept - [RFC1945], section D.2.1 or [RFC2616], section 14.1: The protocol client SHOULD specify this header with the value "*/*". The protocol server MAY<2> ignore the value of this header.

2.2.2 Response Syntax

2.2.2.1 Response Status-Line

The response **Status-Line** MUST be valid according to [RFC1945], section 6.1 or [RFC2616], section 6.1.

2.2.2.1.1 Success Response

The protocol server MUST return HTTP Status-Code "200" (HTTP OK) to indicate a success response as specified in section 3.2.5, Message Processing Events and Sequencing Rules. The response body MUST contain detailed results as specified in section 2.2.2.3, Response Body Syntax.

For success responses other than those specified for Status-Code "200", the protocol server MUST return HTTP Status-Code "204" (NO CONTENT) [RFC1945], section 9.2 or [RFC2616], section 10.2.5, as specified in section 3.2.5, Message Processing Events and Sequencing Rules.

2.2.2.1.2 Failure Response

The protocol server MUST return an HTTP 4xx or 5xx Status-Code as specified in [RFC1945], section 6.1.1 or [RFC2616], section 6.1.1 to indicate that the request failed.

The protocol server SHOULD return the HTTP Status-Code "401" to indicate that the protocol client can retry the request using a different authentication protocol or properties, but MAY<3> return a different code for this condition.

2.2.2.2 Response Headers

The following response headers are relevant to this protocol:

- **Content-Length:** as specified in [RFC1945], section 10.4 or [RFC2616], section 14.13.
- **Content-Type:** as specified in [RFC1945], section 10.5 or [RFC2616], section 14.17. MUST be present and MUST be set to "text/html; charset=utf-8" for Status-Code 200.

2.2.2.3 Response Body Syntax

The response body returned from the protocol server for the functions that are supported by this protocol is specified in the following subsections. Failure responses for all functions SHOULD return a message body describing the failure reason but instead MAY<4> return a response with the Content-Length header set to 0 and with no **message body**.

2.2.2.3.1 Response for Form Server Detection Request

HTTP OK responses MUST return a message body of **UTF-8** encoded text, as specified in the following ABNF:

```
message-body = "<server" %x20 "IsFormServerEnabled" %x20 "=" %x20 "'true'"   %x20 "/>"
```

All white spaces MUST be preserved and any additional white spaces MUST NOT be added.

2.2.2.3.2 Response for Form Server Version Retrieval Request

If the form server supports this protocol function, HTTP OK responses MUST return a message body of UTF-8 encoded text, as specified in the following ABNF:

```
message-body = "<server" %x20 "FormServerVersion" %x20 "=" %x20 "'14'" %x20 "/>"
```

All white spaces MUST be preserved and any additional white spaces MUST NOT be added.

If the form server doesn't support this protocol function, the protocol server MUST NOT return a response body, but instead, it MUST return Status-Code 204, as specified in section 2.2.2.1.1, Success Response.

2.2.2.3.3 Response for Rendering URL Construction Request

Responses with Status-Code "200" (HTTP OK) MUST return a message body as specified in the following ABNF:

```
message-body   = "OpenInFormServer=" (rendering-url)
```

```
rendering-url  = (url-path) "?" (query-component) "&OpenIn=Browser"
url-path       = path-absolute "/_layouts/FormServer.aspx"
```

The **rendering-url** rule in the ABNF refers to the URL that the protocol server MUST return so that the form can be rendered in a Web browser. It MUST be a valid **absolute URI** as specified in [RFC3986], section 4.3. If the form server is enabled on the protocol server, FormServer.aspx MUST exist in the site collection.

The **query-component** rule in the constructed URL MUST contain only the supported parameters that are sent from the protocol client, as specified in section 2.2.1.3.2.3, Request for Rendering URL Construction . The protocol server SHOULD return percent-encoded ([RFC3986], section 2.1) characters for the query parameter values in the query component, but MAY<5> return un-escaped characters when the value of query parameter contains **Unicode** characters. When the *SaveLocation* parameter is present, the protocol server MAY<6> replace the "&" character before the *SaveLocation* parameter with a "?" character in the constructed rendering URL.

ABNF for **path-absolute** is specified in [RFC3986], section 3.3.

The protocol details in section 3.3.5, Message Processing Events and Sequencing Rules specifies how the protocol server builds the rendering URL.

3 Protocol Details

3.1 Common Details

This section specifies details common to both protocol server and protocol client behavior.

Except where specified, protocol clients SHOULD interpret HTTP status codes returned by the protocol server as specified in [RFC1945], section 9 or [RFC2616], section 10 Status Code Definitions.

This protocol allows protocol servers to perform implementation-specific authorization checks and notify protocol clients of authorization faults using HTTP status codes.

3.1.1 Abstract Data Model

This section specifies a conceptual model of possible data organization that an implementation maintains to participate in this protocol. The specified organization is provided to facilitate the explanation of how the protocol behaves. This document does not mandate that implementations adhere to this model as long as their external behavior is consistent with that specified in this document.

The following paragraphs specify the following terms in the context of this protocol.

Base URL: The portion of the Request-URI that matches the **base** rule in the ABNF in section 2.2.1.3.1, Request-URI Details.

Query Parameters: The parameters in the query component of the Request-URI as specified in section 2.2.1.3.2, Query Component Details, and its subsections.

3.1.2 Timers

None.

3.1.3 Initialization

None.

3.1.4 Higher-Layer Triggered Events

None.

3.1.5 Message Processing Events and Sequencing Rules

None.

3.1.6 Timer Events

None.

3.1.7 Other Local Events

None.

3.2 Protocol Client Details

3.2.1 Abstract Data Model

As specified in section 3.1.1, Abstract Data Model.

3.2.2 Timers

None.

3.2.3 Initialization

None.

3.2.4 Higher-Layer Triggered Events

None.

3.2.5 Message Processing Events and Sequencing Rules

The Request-URI that the protocol client sends to the protocol server MUST contain the query component and MUST follow the rules that are specified in section 2.2.1.3, Request-URI Syntax.

The protocol client MUST interpret the response based on the HTTP Status-Code as follows:

- Status-Code 200 – The request was successful and form server features are enabled on the protocol server. The response MUST be interpreted as specified in the following table.

Protocol Function	Status-Code	Meaning of Status-Code
Form Server Detection	200	Form server features are enabled on the protocol server. The response body contains the text as specified in section 2.2.2.3.1, Response for Form Server Detection Request.
Form Server Version Retrieval	200	Form server features are enabled on the protocol server and the form server version was successfully returned. The response body contains the text as specified in section 2.2.2.3.2, Response for Form Server Version Retrieval Request.
Rendering URL Construction	200	Rendering URL has been successfully constructed. The response body contains the rendering URL in the format as specified in section 2.2.2.3.3, Response for Rendering URL Construction Request.

- Status-Code 204 – The request was successful, and the response MUST be interpreted as specified in the following table. The protocol server MUST NOT return a response body.

Protocol Function	Status-Code	Meaning of Status-Code
Form Server Detection	204	Form server features are not enabled on the protocol server.
Form Server Version Retrieval	204	The *FormServerVersion* query parameter is not supported by the protocol server or form server features are not enabled on the protocol server, and the form server version was not returned.

Protocol Function	Status-Code	Meaning of Status-Code
Rendering URL Construction	204	The protocol server cannot construct the rendering URL, based on the given parameters. Possible reasons can be incorrect syntax or value in the query parameter, or form server features are not enabled on the protocol server.

- Status-Code 4xx/5xx – The request failed. The response body MUST NOT contain the text as specified in section 2.2.2.3.1, Response for Form Server Detection Request, section 2.2.2.3.2, Response for Form Server Version Retrieval Request, and section 2.2.2.3.3, Response for Rendering URL Construction Request, but can include informative text providing details of the failure.

3.2.6 Timer Events

None.

3.2.7 Other Local Events

None.

3.3 Protocol Server Details

3.3.1 Abstract Data Model

As specified in section 3.1.1, Abstract Data Model.

3.3.2 Timers

None.

3.3.3 Initialization

None.

3.3.4 Higher-Layer Triggered Events

None.

3.3.5 Message Processing Events and Sequencing Rules

The protocol server MUST process request messages received from a protocol client as follows:

The protocol server MUST validate that the request syntax matches the syntax as specified in section 2.2.1.3, Request-URI Syntax. If the syntax is not valid, the protocol server MUST return Status-Code 204.

- Before checking any of the other query parameters in the query component, the protocol server MUST look for the presence of the *IsFormServerEnabled* parameter. If the parameter *IsFormServerEnabled* exists:
  - The protocol server MUST process the request as a form server detection request and MUST ignore any extra parameters.

- If the form server is enabled on the protocol server, the protocol server MUST return Status-Code 200 (HTTP OK). For such a response, the protocol server MUST generate a response body that MUST contain the text as specified in section 2.2.2.3.1, Response for Form Server Detection Request.

- If the form server is not enabled on the protocol server, the protocol server MUST return Status-Code "204 NO CONTENT".

- If the parameter *IsFormServerEnabled* does not exist in the query parameters, the protocol server SHOULD<7> look for the presence of the *FormServerVersion* parameter. If the parameter *FormServerVersion* exists and is supported by the protocol server:

  - The protocol server MUST process the request as a form server version retrieval request and MUST ignore any extra parameters.

  - If the form server is enabled on the protocol server, the protocol server MUST return Status-Code 200 (HTTP OK). For such a response, the protocol server MUST generate a response body that MUST contain the text as specified in section 2.2.2.3.2, Response for Form Server Version Retrieval Request.

  - If the form server is not enabled on the protocol server, the protocol server MUST return Status-Code "204 NO CONTENT".

- If the parameter *IsFormServerEnabled* does not exist in the query parameters, and either the Form Server Version Retrieval method is not supported or the *FormServerVersion* query parameter is not present:

  - The protocol server MUST verify that the parameter list conforms to one of the supported combinations as specified in section 2.2.1.3.2.3, Request for Rendering URL Construction. If the combination of parameters is not valid, the protocol server MUST return Status-Code "204 NO CONTENT".

  - The protocol server MUST validate that the values for the *XmlLocation* and *XsnLocation* parameters are valid as specified in Table 1 in section 2.2.1.3.2.3, Request for Rendering URL Construction. If the validation fails, the protocol server MUST return Status-Code "204 NO CONTENT".

    The protocol server MUST validate that the protocol client has permission to the resources as specified by the parameter values. If the validation fails, the protocol server SHOULD return Status-Code "401 Unauthorized", but MAY<8> return a different code.

  - If the validation succeeds and form server features are enabled on the protocol server, the protocol server MUST return Status-Code 200 (HTTP OK). For a response with Status-Code 200, the protocol server MUST generate a response body as specified in section 2.2.2.3.3, Response for Rendering URL Construction Request, and as shown in the example in section 4.3.2.1, Response When Form Server Is Enabled.

  - If form server features are not enabled on the protocol server, the protocol server MUST return Status-Code 204 (NO CONTENT).

3.3.6 Timer Events

None.

3.3.7 Other Local Events

None.

4 Protocol Examples

This section illustrates the messages exchanged when a protocol client makes a successful HTTP request to a protocol server using this protocol.

4.1 Form Server Detection

This example shows the client and server interaction during form server detection.

4.1.1 Client Request

An example of a protocol client request to detect whether form server features are enabled on the protocol server is illustrated here:

```
GET /_layouts/FormServerDetector.aspx?IsFormServerEnabled=check HTTP/1.1
Accept: */*
Host: www.contoso.com
```

4.1.2 Server Response

4.1.2.1 Response When Form Server Is Enabled

The following illustrates the response text when form server features are enabled on the protocol server:

```
HTTP/1.1 200 OK
Cache-Control: private
Content-Length: 39
Content-Type: text/html; charset=utf-8
Server: Microsoft-IIS/6.0
X-Powered-By: ASP.NET

<server IsFormServerEnabled = 'true' />
```

4.1.2.2 Response When Form Server Is Not Enabled

The following illustrates the response text when form server features are not enabled on the protocol server:

```
HTTP/1.1 204 No Content
Cache-Control: Private
Content-Length: 0
Server: Microsoft-IIS/6.0
X-Powered-By: ASP.NET
```

4.2 Form Server Version Retrieval

This example shows client and server interaction during the form server version retrieval.

4.2.1 Client Request

An example of a protocol client request to get the version of form server is illustrated here:

```
GET /_layouts/FormServerDetector.aspx?FormServerVersion=check HTTP/1.1
Accept: */*
Host: www.contoso.com
```

4.2.2 Server Response

4.2.2.1 Response When Form Server Is Enabled

The following illustrates the response text when form server features are enabled on the protocol server:

```
HTTP/1.1 200 OK
Cache-Control: private
Content-Length: 35
Content-Type: text/html; charset=utf-8
Server: Microsoft-IIS/6.0
X-Powered-By: ASP.NET

<server FormServerVersion = '14' />
```

4.2.2.2 Response When Form Server Is Not Enabled

The following illustrates the response text when form server features are not enabled on the protocol server:

```
HTTP/1.1 204 No Content
Cache-Control: Private
Content-Length: 0
Server: Microsoft-IIS/6.0
X-Powered-By: ASP.NET
```

4.3 Rendering URL Construction

This example shows client and server interaction during the construction of the rendering URL.

4.3.1 Client Request

An example of a request to obtain the rendering URL of an existing form from the protocol server is illustrated here.

```
GET /_layouts/FormServerDetector.aspx?XmlLocation=/Folder/filename.xml HTTP/1.1
Accept: */*
Host: www.contoso.com
```

4.3.2 Server Response

4.3.2.1 Response When Form Server Is Enabled

The following illustrates the response text of a request as given in section 4.3.1, Client Request, when the parameter is valid and form server features are enabled on the protocol server:

```
HTTP/1.1 200 OK
Cache-Control: private
Content-Length: 102
Content-Type: text/html; charset=utf-8
Server: Microsoft-IIS/6.0
X-Powered-By: ASP.NET

OpenInFormServer=http://www.contoso.com/_layouts/FormServer.aspx?XmlLocation=/Folder/filename
.xml&OpenIn=Browser
```

4.3.2.2 Response When Form Server Is Not Enabled

The following illustrates the response text of a request as given in section 4.3.1, Client Request, when form server features are not enabled on the protocol server:

```
HTTP/1.1 204 No Content
Cache-Control: Private
Content-Length: 0
Server: Microsoft-IIS/6.0
X-Powered-By: ASP.NET
```

5 Security

5.1 Security Considerations for Implementers

This protocol introduces no additional security considerations beyond those applicable to its underlying protocols.

5.2 Index of Security Parameters

None.

6 Appendix A: Product Behavior

The information in this specification is applicable to the following Microsoft products or supplemental software. References to product versions include released service packs:

- Microsoft® Office Forms Server 2007
- Microsoft® Office InfoPath® 2007
- Microsoft® InfoPath® 2010
- Microsoft® Office SharePoint® Server 2007
- Microsoft® SharePoint® Server 2010

Exceptions, if any, are noted below. If a service pack or Quick Fix Engineering (QFE) number appears with the product version, behavior changed in that service pack or QFE. The new behavior also applies to subsequent service packs of the product unless otherwise specified. If a product edition appears with the product version, behavior is different in that product edition.

Unless otherwise specified, any statement of optional behavior in this specification that is prescribed using the terms SHOULD or SHOULD NOT implies product behavior in accordance with the SHOULD or SHOULD NOT prescription. Unless otherwise specified, the term MAY implies that the product does not follow the prescription.

<1> Section 2.2.1.3.2.2: The form server version retrieval function is only supported in SharePoint Server 2010.

<2> Section 2.2.1.4: Office SharePoint Server 2007 and SharePoint Server 2010 will ignore the value of the Accept header if it is not "*/*".

<3> Section 2.2.2.1.2: Office SharePoint Server 2007 and SharePoint Server 2010 will return a "401 Unauthorized" status-code if the client is not authorized to access the path in the Request-URI. If the client is authorized to access the path in the Request-URI but is not authorized to access a resource identified by the query parameters in the Request-URI, SharePoint Server 2010 will return a "204 No Content" status-code and Office SharePoint Server 2007 will return a "302 Found" status-code.

<4> Section 2.2.2.3: Office SharePoint Server 2007 and SharePoint Server 2010 will return a text/html response body for "401" HTTP status-codes.

<5> Section 2.2.2.3.3: Office SharePoint Server 2007 and SharePoint Server 2010 will return un-escaped Unicode characters when the value of query parameter contains Unicode characters.

<6> Section 2.2.2.3.3: Office SharePoint Server 2007 and SharePoint Server 2010 will replace the "&" character before the "SaveLocation" parameter with a "?" character in the constructed rendering URL.

<7> Section 3.3.5: The *FormServerVersion* query parameter is only supported in SharePoint Server 2010.

<8> Section 3.3.5: If the client is authorized to access the path in the Request-URI but is not authorized to access a resource identified by the query parameters in the Request-URI, SharePoint Server 2010 will return a "204 No Content" status-code and Office SharePoint Server 2007 will return a "302 Found" status-code.

7 Change Tracking

No table of changes is available. The document is either new or has had no changes since its last release.

8 Index

T

V