# Contents

**Preface** ................................................................. 5  
  Documentation Conventions ........................................... 5  
  Related Publications .................................................... 6  
  Customer Support ....................................................... 6  
  Help Us to Serve You Better .......................................... 7  
  User Feedback ........................................................... 9  
  Information Builders Consulting and Training .................... 9  

1. Introducing nHTTP Representational State Transfer ............ 11  
   Representational State Transfer (REST) .......................... 11  
   nHTTP REST Support ................................................ 12  
   nHTTP Configuration Overview .................................... 13  

2. Configuring a Process Flow and nHTTP Channel ............... 15  
   Configuring a Process Flow ......................................... 15  
   Configuring an nHTTP Channel .................................... 32  
   Running a REST Style Service ...................................... 36
Preface

This document is written for system integrators and application designers who need to configure and use nHTTP Representational State Transfer (REST) within the iWay Service Manager framework.

Note: This Release 7.0.x content is currently being updated to support iWay Release 8.0.x software. In the meantime, it can serve as a reference for your use of iWay Release 8. If you have any questions, please contact Customer_Success@ibi.com.

How This Manual Is Organized

This manual includes the following chapters:

<table>
<thead>
<tr>
<th>Chapter/Appendix</th>
<th>Contents</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Introducing nHTTP Representational State Transfer</td>
</tr>
<tr>
<td></td>
<td>Provides an overview of nHTTP Representational State Transfer (REST), including key features.</td>
</tr>
<tr>
<td>2</td>
<td>Configuring a Process Flow and nHTTP Channel</td>
</tr>
<tr>
<td></td>
<td>Describes how to configure a process flow and channel for nHTTP Representational State Transfer.</td>
</tr>
</tbody>
</table>

Documentation Conventions

The following table describes the documentation conventions that are used in this manual.

<table>
<thead>
<tr>
<th>Convention</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>THIS TYPEFACE</td>
<td>Denotes syntax that you must enter exactly as shown.</td>
</tr>
<tr>
<td>or</td>
<td></td>
</tr>
<tr>
<td>this typeface</td>
<td></td>
</tr>
<tr>
<td>underscore</td>
<td>Indicates a default setting.</td>
</tr>
<tr>
<td>Key + Key</td>
<td>Indicates keys that you must press simultaneously.</td>
</tr>
<tr>
<td>{ }</td>
<td>Indicates two or three choices. Type one of them, not the braces.</td>
</tr>
</tbody>
</table>
### Convention | Description
--- | ---
| | Separates mutually exclusive choices in syntax. Type one of them, not the symbol.  
... | Indicates that you can enter a parameter multiple times. Type only the parameter, not the ellipsis (...).
|  | Indicates that there are (or could be) intervening or additional commands.

**Related Publications**

Visit our Technical Content Library at [http://documentation.informationbuilders.com](http://documentation.informationbuilders.com). You can also contact the Publications Order Department at (800) 969-4636.

**Customer Support**

Do you have questions about this product?

Join the Focal Point community. Focal Point is our online developer center and more than a message board. It is an interactive network of more than 3,000 developers from almost every profession and industry, collaborating on solutions and sharing tips and techniques. Access Focal Point at [http://forums.informationbuilders.com/eve/forums](http://forums.informationbuilders.com/eve/forums).

You can also access support services electronically, 24 hours a day, with InfoResponse Online. InfoResponse Online is accessible through our website, [http://www.informationbuilders.com](http://www.informationbuilders.com). It connects you to the tracking system and known-problem database at the Information Builders support center. Registered users can open, update, and view the status of cases in the tracking system and read descriptions of reported software issues. New users can register immediately for this service. The technical support section of [http://www.informationbuilders.com](http://www.informationbuilders.com) also provides usage techniques, diagnostic tips, and answers to frequently asked questions.

Call Information Builders Customer Support Services (CSS) at (800) 736-6130 or (212) 736-6130. Customer Support Consultants are available Monday through Friday between 8:00 a.m. and 8:00 p.m. EST to address all your questions. Information Builders consultants can also give you general guidance regarding product capabilities. Please be ready to provide your six-digit site code number (xxxx.xx) when you call.
To learn about the full range of available support services, ask your Information Builders representative about InfoResponse Online, or call (800) 969-INFO.

**Help Us to Serve You Better**

To help our consultants answer your questions effectively, be prepared to provide specifications and sample files and to answer questions about errors and problems.

The following table lists the environment information our consultants require.

<table>
<thead>
<tr>
<th>Platform</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Operating System</td>
<td></td>
</tr>
<tr>
<td>OS Version</td>
<td></td>
</tr>
<tr>
<td>JVM Vendor</td>
<td></td>
</tr>
<tr>
<td>JVM Version</td>
<td></td>
</tr>
</tbody>
</table>

The following table lists the deployment information our consultants require.

| Adapter Deployment | For example, JCA, Business Services Provider, iWay Service Manager |
| Container | For example, WebSphere |
| Version |  |
| Enterprise Information System (EIS) - if any |  |
| EIS Release Level |  |
| EIS Service Pack |  |
| EIS Platform |  |

The following table lists iWay-related information needed by our consultants.

| iWay Adapter |  |
The following table lists additional questions to help us serve you better.

<table>
<thead>
<tr>
<th>Request/Question</th>
<th>Error/Problem Details or Information</th>
</tr>
</thead>
<tbody>
<tr>
<td>Did the problem arise through a service or event?</td>
<td></td>
</tr>
<tr>
<td>Provide usage scenarios or summarize the application that produces the problem.</td>
<td></td>
</tr>
<tr>
<td>When did the problem start?</td>
<td></td>
</tr>
<tr>
<td>Can you reproduce this problem consistently?</td>
<td></td>
</tr>
<tr>
<td>Describe the problem.</td>
<td></td>
</tr>
<tr>
<td>Describe the steps to reproduce the problem.</td>
<td></td>
</tr>
<tr>
<td>Specify the error message(s).</td>
<td></td>
</tr>
<tr>
<td>Any change in the application environment: software configuration, EIS/database configuration, application, and so forth?</td>
<td></td>
</tr>
<tr>
<td>Under what circumstance does the problem not occur?</td>
<td></td>
</tr>
</tbody>
</table>

The following is a list of error/problem files that might be applicable.

- Input documents (XML instance, XML schema, non-XML documents)
- Transformation files
Error screen shots
Error output files
Trace files
Service Manager package to reproduce problem
Custom functions and services in use
Diagnostic Zip
Transaction log

For information on tracing, see the iWay Service Manager User's Guide.

User Feedback

In an effort to produce effective documentation, the Technical Content Management staff welcomes your opinions regarding this document. You can contact us through our website, http://documentation.informationbuilders.com/connections.asp.

Thank you, in advance, for your comments.

Information Builders Consulting and Training

Interested in training? Information Builders Education Department offers a wide variety of training courses for this and other Information Builders products.

For information on course descriptions, locations, and dates, or to register for classes, visit our website (http://education.informationbuilders.com) or call (800) 969-INFO to speak to an Education Representative.
Introducing nHTTP Representational State Transfer

This section provides an overview of nHTTP Representational State Transfer (REST), including key features.

In this chapter:

- Representational State Transfer (REST)
- nHTTP REST Support
- nHTTP Configuration Overview

Representational State Transfer (REST)

Representational State Transfer (REST) is a type of software architecture that is designed for distributed hypermedia systems, such as the web. The REST architectural style can be summarized as four main verbs: GET, POST, PUT, and DELETE. These verbs pertain to the HTTP 1.1 specification and the nouns which are the resources available on the network via a URI. Please note that REST is not a standard itself and implementation for its support varies. However, REST does use standards such as HTTP, URL, Resource Representations (XML/HTML/GIF/JPEG/etc.) and Mime-Types (text/xml, text/html, image/gif/etc.). REST is based on a client-server architecture using the pull-based interaction style. It is stateless and as a result, each request from a client to a server must contain all the information necessary to understand the request. It cannot take advantage of any stored context on the server. The basis for REST is the use of nouns that represent resources identified with the URI and a set of verbs, which define actions for the nouns.
nHTTP REST Support

The nHTTP listener for iWay Service Manager complies with the HTTP 1.1 specification. The listener implements all the available verbs used in REST-style communication, including the main verbs (GET, POST, PUT, and DELETE). The handling of the verbs can be configured during the nHTTP listener configuration and allows you to indicate if the verb should generate an event document, which will propagate to the process flow or if it should generate an error. The key here is that this ability to create an event document from the HTTP request allows an iSM channel to execute an underlying process flow from a URL request. As the URL request is being served by the nHTTP listener, the event document is generated, the process flow is executed, and the response document is returned via HTTP. This enables any process flow to run from any external application by issuing a URL reference to it. The REST style service support of addressing a resource with a supported set of verbs is available via the nHTTP listener implementation. For more information on a full array of nHTTP supported features, see the *iWay Service Manager User's Guide*.

The following table lists the supported HTTP requests.

<table>
<thead>
<tr>
<th>Request Type</th>
<th>Available Actions</th>
</tr>
</thead>
<tbody>
<tr>
<td>GET</td>
<td>docroot - serve document from docroot.</td>
</tr>
<tr>
<td></td>
<td>event - generate an event document.</td>
</tr>
<tr>
<td></td>
<td>error - return HTTP error.</td>
</tr>
<tr>
<td>POST</td>
<td>flow - non-configurable, request propagated to flow</td>
</tr>
<tr>
<td>HEAD</td>
<td>docroot - serve document from docroot.</td>
</tr>
<tr>
<td></td>
<td>event - generate an event document.</td>
</tr>
<tr>
<td></td>
<td>error - return HTTP error.</td>
</tr>
<tr>
<td>PUT</td>
<td>event - generate an event document.</td>
</tr>
<tr>
<td></td>
<td>error - return HTTP error.</td>
</tr>
<tr>
<td>DELETE</td>
<td>event - generate an event document.</td>
</tr>
<tr>
<td></td>
<td>error - return HTTP error.</td>
</tr>
</tbody>
</table>
### Request Type | Available Actions
--- | ---
**TRACE** | Echoes request as per RFC.
**OPTION** | Returns available options as per RFC.

---

**nHTTP Configuration Overview**

The example that is provided in this documentation demonstrates the configuration of an nHTTP listener and a simple process flow.

**Note:** You must have iWay Service Manager 6.0 or higher installed to configure and use an nHTTP listener.

The nHTTP listener is listening for HTTP requests on port 2222 and is configured to process HTTP GET requests as events. It will generate an event document with all the available information from the HTTP GET request and propagate it to the configured process flow. The process flow will parse the received event document representing the HTTP GET request and will attempt to retrieve the requested file from the directory. If the file is found, it will return the file content as an nHTTP response. If the requested file is not available, an error message will be returned. Please note that even though there is an option to simply serve the documents from the docroot, this example will call a process flow for this purpose. This is done to illustrate a technique to call a process flow as a REST style service call.
For more information on how to configure an nHTTP listener and a simple process flow, see *Configuring a Process Flow and nHTTP Channel* on page 15.
Chapter 2  Configuring a Process Flow and nHTTP Channel

This section describes how to configure a process flow and channel for nHTTP Representational State Transfer.

In this chapter:
- Configuring a Process Flow
- Configuring an nHTTP Channel
- Running a REST Style Service

Configuring a Process Flow

A process flow contains all the business logic and performs the required processing. It parses the received event document to retrieve the parameter for the requested file name. It validates that the requested file exists. If the file exists, the process flow retrieves the content of the file. If the file is not available, the process flow returns a default error message.
Procedure: How to Configure a Process Flow

To configure a process flow:

1. Open iWay Integration Tools (iIT) and create a new Integration Project called REST, as shown in the following image.

2. Expand the REST Integration Project node, right-click the Flows subfolder and select Process Flow from the context menu, as shown in the following image.
3. Create a new process flow called GetFile.pFlow under the new REST Integration Project.
4. Click Finish.
5. Add a Service object to the new process flow.

This Service object will set the required Special Registers (SREGs) that are required by the process flow.

6. Name the new Service object as Set Title Register and select SREG Agent as the service type.
The SREG Agent service type allows SREGs to be used throughout the business logic for data manipulation and processing.

7. Click Next.

The Object Properties pane opens, as shown in the following image.

8. Leave the default values that are provided and click Next.
The User Properties pane opens, as shown in the following image.

9. Click the green plus sign (+) icon to define a new design time property.

10. Enter the following settings for the new property:

- **Name**: file
- **Type**: string
- **Value**: `XPATH(http/url/queryparms/queryparm[contains(@name,'file')] )`

- **Description**: Sets file registers to the value received from the incoming event document parameter.
**Note:** The iFL statement for the value will perform an XPATH into the event document that is received by the nHTTP listener. It will extract the parameter value from this document that holds the value for the parameter named `file`. This parameter from the incoming document will contain the file title that will be retrieved. The following is a sample of the event signal document returned by nHTTP with a marked `file` parameter. In this example, the file SREG will be set to myfile.txt.

The following is a sample URL provided by the user:

http://localhost:2222/get?file=myfile.txt

The following is a sample event document that is generated:

```xml
<xml version="1.0" encoding="ISO-8859-1" ?>
  <http user="auto" type="GET">
    <params>
      <param name="base-port">9000</param>
      <param name="Cookie">AlwaysOnLogin=loggin; rolename=administrator; username=admin; password=1way; ntrreport=false; country=; depot=; custosmbauth=undefined; PSCount=25; actFields1=; actFields10Desc=;</param>
      <param name="minmaxHeightChannel">60; minmaxHeightTrans=35; minmaxHeightPartner=60; toolbarTransCount=0</param>
      <param name="Host">localhost:2222</param>
      <param name="Keep-Alive">true</param>
      <param name="pdm">0</param>
      <param name="version">1.1</param>
      <param name="Accept-Language">en-US</param>
      <param name="Utf-8">true</param>
      <param name="false"/>
      <param name="User-Agent">Mozilla/4.0 (compatible; MSIE 8.0; Windows NT 6.1; WOW64; Trident/4.0; SLCC2; .NET CLR 2.0.50727; .NET CLR 3.5.30729; .NET CLR 3.0.30729; Media Center PC 6.0; .NET4.0C; MDDR; MS-RTC LM 8)</param>
      <param name="Length">127.0.0.1</param>
      <param name="source">hostname unknown</param>
      <param name="Accept-Encoding">gzip, deflate</param>
      <param name="reqtype">GET</param>
    </params>
    <body />
    <url secure="false">
      <host>localhost</host>
      <port>2222</port>
      <path>/get</path>
      <query>file=myfile.txt</query>
    </url>
  </http>
```

11. Connect the Start object and the Set Title Register object using the On Completion relation.
12. Add a File object named Check if File Exists to the process flow, as shown in the following image.

This File object checks to see if the file with the given title exists.

13. Click Next.
14. Select PF File Ops Agent (com.ibi.agents.XDPFFileOpsAgent) from the Type drop-down list.
15. Click Next.
The Object Properties pane opens, as shown in the following image.

16. Provide values for the parameters as listed in the following table.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Operation</td>
<td>exist</td>
</tr>
<tr>
<td>File (from)</td>
<td>c:\temp\sreg(file)</td>
</tr>
<tr>
<td><strong>Parameter</strong></td>
<td><strong>Value</strong></td>
</tr>
<tr>
<td>-----------------------------</td>
<td>-----------</td>
</tr>
<tr>
<td>Out Document</td>
<td>Result</td>
</tr>
<tr>
<td>Action on Failure</td>
<td>input</td>
</tr>
</tbody>
</table>

17. Click *Finish*.

18. Connect the Set Title Register object and the Check if File Exists object using the On Completion relation.

**Note:** Error handling is not being configured in this sample configuration. However, it is recommended to add error handling in production applications.

19. Add a File object named Read File to the process flow, as shown in the following image.

![New File Object](image_url)

This File object reads the requested file if it exists.

20. Click *Next*.
The File Object Type pane opens, as shown in the following image.

21. Select *PF File Read Agent (com.ibi.agents.XDPFFileReadAgent)* from the Type drop-down list.
22. Click Next.
The Object Properties pane opens, as shown in the following image.

23. Provide values for the parameters as listed in the following table.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name of File</td>
<td>c:\temp\sreg(file)</td>
</tr>
<tr>
<td>Response Type</td>
<td>flat</td>
</tr>
</tbody>
</table>

24. Click Finish.

25. Connect the Check if File Exists object and the Read File object using the On Success relation.
26. Add another Service object to the process flow.

This Service object will set a default error message if the requested file is not found in the specified location.

27. Name the new Service object as Default MSG if No File.
28. Select *Constant Agent* as the service type, which sets the current content of the document to the application provided value.

29. Click Next.
The Object Properties pane opens, as shown in the following image.

30. Provide values for the parameters as listed in the following table.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Constant Output To</td>
<td>Error: Requested file not found!</td>
</tr>
<tr>
<td>Emit</td>
<td>This indicates to the user that the file does not exist in the current system.</td>
</tr>
<tr>
<td>Output Format</td>
<td>flat</td>
</tr>
</tbody>
</table>

31. Click Finish.

32. Connect the Check if File Exists object and Default MSG if No File object using the On Custom (not found) relation.

33. Add an End object to the process flow.

34. Connect the Read File object to the End object using the On Success relation.

35. Connect the Default MSG if No File object to the End object using the On Success relation.
The following image shows the completed process flow.

36. Right-click the GetFile.pFlow process flow, select Integration Tools, and then click Publish, as shown in the following image.
The Publish Resource Wizard opens, as shown in the following image.

37. Provide the required connection parameters to iSM and click Finish.

Configuring an nHTTP Channel

The nHTTP channel includes an nHTTP listener running on port 2222 and is used to generate event documents for HTTP GET requests. The channel has a route containing the process flow created in the first section. It will contain a default outlet to allow responses on the requested nHTTP listener.

Procedure: How to Configure an nHTTP Channel

To configure an nHTTP channel:

1. Create an nHTTP Listener by going to the iWay Service Manager Administration Console and navigating to the Listeners section under the Registry. Select HTTP 1.1 [nonblocking] (nhttp), as shown in the following image.
2. Click Next and provide listener configuration. In this example, listening for events occurs on HTTP port 2222. The configuration is port 2222. Under GET Handling, select Event, leaving all other parameters with the default settings.

When the listener receives an HTTP GET request, it generates an event document which will be sent to the route.

3. Click Next and provide a listener name as GetFile.Listener with an optional description. Click Finish.

4. Create a new inlet which will contain the listener. Under the registry go to inlet section and select New. Provide inlet name as GetFile.Inlet with an optional description. Click Finish.
5. Add the **GetFile.Listener** to the **GetFile.Inlet**.

   ![Construct Inlet](image)

   Below are the components currently registered in the inlet. The order of decryptor and parser components may be changed within each component type by checking a component and using the 'Move Up' and 'Move Down' buttons.

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Move</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>GetFile Listener</td>
<td>Listener</td>
<td></td>
<td>nHTTP listener on port 2222, GET handling set to event</td>
</tr>
</tbody>
</table>

6. Create a new route to host the process flow which was created. Under the Registry section, click **routes**, and click **New** to create a new route. Provide a name **GetFile.Route** with an optional description. Click **Finish**.

   ![New Route Definition](image)

   **New Route Definition**

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>GetFile.Route</td>
<td>Route to host the GetFile.pFlow which processes the HTTP GET event document and retrieves the requested file.</td>
</tr>
</tbody>
</table>

7. Add **GetFile.pFlow** to the **GetFile.Route**.

   ![Construct Route](image)

   Below are the components currently registered in the route.

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Conditions</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>GetFile.pFlow</td>
<td>Process</td>
<td></td>
<td>pFlow for retrieving files based on the nHTTP request</td>
</tr>
</tbody>
</table>

8. Create a new channel by selecting **Channels** under the Registry. Provide Channel name as **GetFile.Channel** with optional description. Click **Finish**.

   ![New Channel Definition](image)

   **New Channel Definition**

<table>
<thead>
<tr>
<th>Name *</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>GetFile.Channel</td>
<td>Channel with nHTTP (port 2222) listener serving GET requests based on the underlying pFlow. The pFlow will retrieve the requested file and send it back as HTTP response.</td>
</tr>
</tbody>
</table>
9. Add the GetFile.Inlet, GetFile.Route and a default outlet to the GetFile.Channel. The default outlet will indicate that the response has to be returned back on the HTTP listener’s response. Set the GetFile.Route as a default route by clicking the '-' sign under the conditions column. This will indicate that the route is a default one and no condition exists to execute.

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Conditions</th>
<th>Move</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>GetFile.Inlet</td>
<td>Inlet</td>
<td></td>
<td></td>
<td>Inlet for the nHTTP GetFile.Listener</td>
</tr>
<tr>
<td>GetFile.Route</td>
<td>Route</td>
<td></td>
<td></td>
<td>Route to host the GetFile.pFlow which processes the HTTP GET event document and retrieves the requested file.</td>
</tr>
<tr>
<td>default outlet</td>
<td>Outlet</td>
<td></td>
<td></td>
<td>The default outlet defines an empty outlet. An outlet that does not contain an emitter is considered a default outlet whose emitter is defined by the channel’s inlet listener.</td>
</tr>
</tbody>
</table>

10. Once the channel is configured, click Build to build the channel. Ensure that the channel builds without any errors or warnings.

```
GetFile.Channel
Build result for channel
Message level    Message
Info            Start
Info            Validating Channel
Info            Channel is valid
Info            Validating Inlet
Info            Inlet is valid
Info            Validating Routes
Info            Routes are valid
Info            Validating Outlets
Info            Outlets are valid
Info            Build Successful
Info            End
Info            Channel archive c:\6.0.0_xfc.19297\etc\repository\manager\car\GetFile.Channe\GetFile.Channe.1.\GetFile.Channe.car has been created/updated
```
11. Now the channel can be deployed under the Deployments section of iSM Console. Click Deploy to deploy a new channel. Under the available channels, select GetFile.Channel and click Deploy.

12. Now you can start your channel under the channels page by clicking its status or by selecting a channel and clicking Start.

Running a REST Style Service

After you have configured a process flow and a nHTTP channel, you are ready to execute your REST style service.

Procedure: How to Run a REST Style Service

To run a REST style service:

1. Create a simple text file under the c:\temp\ directory (or if you configured for another directory, use the one you configured in the process flow). Name the file myfile.txt. This will be a file that we’ll try to retrieve using the HTTP REST-Style service call.

2. Open a web browser and issue a REST Style service call request by requesting the following URL:

   http://localhost:2222/get?file=myfile.txt
The content of the file will be returned for display, as shown in the following image.

The created process flow also contains logic to generate an error for a file that does not exist. For example, if you provide a request URL, such as http://localhost:2222/get?file=error.txt, then the following error message will be returned in the browser indicating that the file is not found.
Feedback

Customer success is our top priority. Connect with us today!

Information Builders Technical Content Management team is comprised of many talented individuals who work together to design and deliver quality technical documentation products. Your feedback supports our ongoing efforts!

You can also preview new innovations to get an early look at new content products and services. Your participation helps us create great experiences for every customer.

To send us feedback or make a connection, contact Sarah Buccellato, Technical Editor, Technical Content Management at Sarah_Buccellato@ibi.com.

To request permission to repurpose copyrighted material, please contact Frances Gambino, Vice President, Technical Content Management at Frances_Gambino@ibi.com.