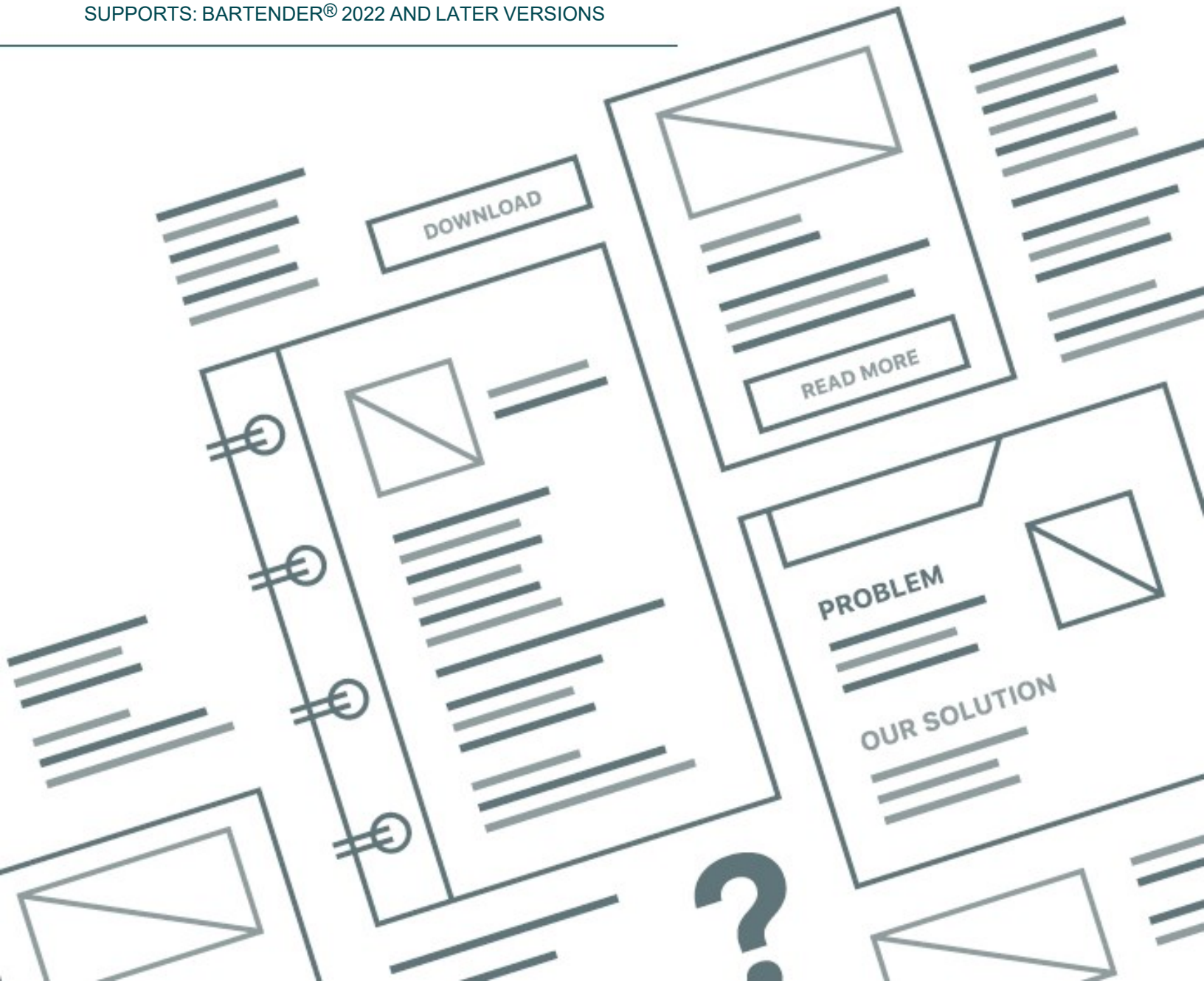
BarTender®
BY SEAGULL SCIENTIFIC

Technical Document
English

Automation with BTXML Script

USING BARTENDER® XML TO AUTOMATE BARTENDER AND INTEGRATE WITH OTHER APPLICATIONS

SUPPORTS: BARTENDER® 2022 AND LATER VERSIONS



Contents

Overview

Extensible Markup Language (XML) is a markup language that defines a set of rules for encoding documents in a format that is both human-readable and machine-readable. BarTender XML (BTXML) script is an XML schema that was created specifically to work with BarTender. You can use BTXML script to control and automate BarTender and print jobs, and you can send BTXML script to your external software applications to run BarTender and print jobs. Third-party programmers can use it to integrate BarTender into their applications.

In some cases, an XML response is created after the script is run. For example, when you use BTXML script to print a document, you receive an XML response that gives you valuable information about the print job, the items that were printed, the printer that was used to complete the job, and the BarTender settings that were used during the print job. You can integrate this response with a custom application.

Most enterprise resource planning (ERP) software packages already have built-in standard functions for generating XML. This is one reason why using XML is so convenient for instructing applications to send commands to BarTender. One potential challenge, however, is that the default XML that is generated by this software might not be compatible with the BTXML format that BarTender uses. As an alternative to generating custom XML from your ERP application, you can generate your XML in its default format and then convert it to BXTML by using Integration Builder.

For more information, refer to the Automation with BarTender XML Script topic in the BarTender help system.

BTXML Script Format

BTXML script uses an XML-style format to communicate with BarTender to process print jobs. BarTender processes the code and then performs the task that the code defines.

All BTXML commands that are sent to BarTender must have the following format.

```
<?xml version="1.0" encoding="utf-8"?>
<XMLScript Version="2.0">
  <Command>
  ...
  </Command>
</XMLScript>
```

For more information, refer to the BarTender XML Script Format topic in the BarTender help system.

Command Tag

The command tag defines the command request. The syntax is as follows.

```
<Command [Name=Command Name] [RepeatCount=Number]>...</Command>
```

In this example, the ellipsis (...) is a placeholder for the tags that the command contains.

For more information about the command tag and supported BTXML commands, refer to the "Reference" section in the Automation with BarTender XML Script book in the BarTender help system.

Methods of Running BTXML Script

You can create BTXML script as a script or as a file and then send it to BarTender by using any of the following methods:

- Integration Builder integration files
- Process Builder process files
- BarTender REST API
- BarTender SDK

Integration Builder

Integration Builder is a BarTender companion application that you can use to integrate BarTender with other software or other sources of data and to automate processes in BarTender. By using Integration Builder, you can create an integration that contains a Print BTXML Script action that you can configure to pass your BTXML script to BarTender to print one or more documents.

BTXML scripts that are run by using a Print BTXML Script action return an XML response.

For more information about how to create and use integration files, refer to the Integration Builder book in the BarTender help system.

Process Builder

Process Builder is a BarTender companion application that you can use to create customizable process files so that you can automate repetitive operations. You can use a Print BTXML Script action to pass your BTXML script to BarTender to print one or more documents.

BTXML scripts that are run by using a Print BTXML Script action return an XML response.

For more information about how to create and use process files, refer to the Process Builder book in the BarTender help system.

BarTender REST API

The BarTender REST API provides REST endpoints that you can use to run a variety of actions in YAML or JSON format or in an existing legacy BTXML script.

The actions that are available in the BarTender REST API include the same actions that you can run in BarTender Designer, Integration Builder, and Process Builder. If you prefer to configure automation by using a graphical user interface, you can use those applications. By using the REST API, however, you can run these actions programmatically.

For more information, refer to Using the BarTender REST API in the BarTender help system.

BTXML scripts that are run by using the BarTender Rest API return an XML response.

BarTender SDK

The BarTender .NET software development kit (SDK) interfaces with any .NET language and supports task-based concurrent label printing for high-demand environments such as web servers. It includes the following application programming interfaces (APIs) that support BTXML script:

- **Print Scheduler API**: By using the Print Scheduler API, you can use the **BtXmlAction** class to run BTXML script. To do this, you create the action, set the properties that are required to complete the action, and then call either the **Run** or the **RunSynchronous** method. The **Run** method returns a response immediately and allows the client to do other work while the submitted action is running. The **RunSynchronous** method submits the action to the server and then waits for it to finish before returning the response.
- **Print Server API**: *(This API is superseded by newer technologies and should be used for backward compatibility only.)* By using the Print Server API, you can use instances of the **XMLScriptTask** class to run BTXML script. The XMLScriptTask task is then submitted to the task queue by either the **QueueTaskAndWait** method or the **QueueTask** method. The BTXML script can be used in either file or string mode. In file mode, the path to a BTXML script file is given, and the file is read at print time to automate BarTender. In string mode, the BTXML script itself is placed in the task and then used to perform automation. *This API has been superseded by newer technologies and should be used only for testing and backward compatibility.*
- **Print Engine API**: *(This API is superseded by newer technologies and should be used for backward compatibility only.)* You can run BTXML in the Print Engine API by using the **Engine** class **XMLScript** method. This method passes either a BTXML script string or the path and file name of a file that contains BTXML Script. *This API has been superseded by newer technologies and should be used only for testing and backward compatibility.*

BTXML scripts that are run by using the Print Scheduler, Print Server, and Print Engine APIs return an XML response.

Legacy Methods

The methods that are described in this section are superseded by newer technologies and should be used for backward compatibility only.

ActiveX Automation

ActiveX is a legacy technology and should not be used to develop new products. BarTender exposes the functionality of ActiveX Automation as a set of software objects. By using a programming or scripting language, such as Visual Basic or C#, you can use ActiveX Automation to pass BTXML scripts to BarTender by using the Application object **XMLScript** method.

For more information about how to automate BarTender by using ActiveX Automation and how to use the Application object, refer to the following topics in the BarTender help system:

- Automation with ActiveX
- Using the Application Object

BTXML scripts that are run by using ActiveX Automation return an XML response.

Command-Line Interface

The command-line interface is a legacy technology and should not be used to develop new products. It should be used only for testing or backward compatibility.

You can configure BarTender functions to be performed automatically when BarTender starts by passing BTXML script via the command-line interface. The BTXML script must be passed as a string or a file, and the /XMLScript parameter must be the last argument on the command line.

The following example shows the correct syntax for passing a BTXML script string to BarTender for processing. In this example, "btxml_script" is the BTXML code to run.

```
/XMLScript="btxml_script"
```

The following example shows the correct syntax for passing a BTXML script file to BarTender for processing. In this example, "btxml_file" is the path and file name of the file that contains the BTXML code to run.

```
/XMLScript="btxml_file"
```

For more information about how to pass BTXML script to BarTender by using the command-line interface, refer to the Command Line Parameter Reference topic in the BarTender help system.

BTXML scripts that are run through the command line do not return or save an XML response.

Printing Documents by Using BTXML Script

BTXML script provides a way for users to pass instructions to BarTender that specify how to print a document. By using BTXML script tags, you can create scripts that instruct BarTender to print one simple item or hundreds of complex designs.

Basic Printing

The following example shows a basic BTXML script that opens and prints the Document1.btw document.

```
<?xml version="1.0" encoding="utf-8"?>
<XMLScript Version="2.0">
 <Command Name="Job1">
   <Print>
     <Format>c:\BarTender\Document1.btw</Format>
   </Print>
 </Command>
</XMLScript>
```

Customized Printing

In addition to opening and printing a BarTender document, you can configure various document settings before you print the documents.

The following example shows a BTXML script that opens the Document1.btw document and then sends a print job that contains 100 serialized items to Tray 1 of the HP LaserJet printer.

```
<?xml version="1.0" encoding="utf-8"?>
<XMLScript Version="2.0">
  <Command>
     <Print JobName="Job1">
       <Format>c:\BarTender\Document1.btw</Format>
       <PrintSetup>
          <NumberSerializedLabels>100</NumberSerializedLabels>
          <Printer>HP LaserJet</Printer>
          <PaperTray>Tray 1</PaperTray>
       </PrintSetup>
     </Print>
  </Command>
</XMLScript>
```

Running Multiple Print Jobs

You can send more than one print job by using a single BTXML file or string. To do this, include a `<Command>` tag for each print job that you want to send.

The following example shows a BTXML script that sends three print jobs (each of which uses a different `<Command>` tag) to the printer. Each job opens a different document, prints five copies, and then closes the document without saving it.

```
<?xml version="1.0" encoding="utf-8"?>
<XMLScript Version="2.0">
  <Command Name="Job1">
    <Print>
      <Format CloseAtEndOfJob="true">Document1.btw</Format>
```

```
      <PrintSetup>
        <IdenticalCopiesOfLabel>5</IdenticalCopiesOfLabel>
      </PrintSetup>
    </Print>
  </Command>
  <Command Name="Job2">
    <Print>
      <Format CloseAtEndOfJob="true">Document2.btw</Format>
      <PrintSetup>
        <IdenticalCopiesOfLabel>5</IdenticalCopiesOfLabel>
      </PrintSetup>
    </Print>
  </Command>
  <Command Name="Job3">
    <Print>
      <Format CloseAtEndOfJob="true">Document3.btw</Format>
      <PrintSetup>
         <IdenticalCopiesOfLabel>5</IdenticalCopiesOfLabel>
       </PrintSetup>
    </Print>
  </Command>
</XMLScript>
```

Printing Documents by Using the Print Scheduler Service

You can use BTXML script to pass instructions to BarTender through the Print Scheduler service. The Print Scheduler service is a Windows service that manages BarTender print engines and intelligently assigns jobs to them while maximizing performance and maintaining print order. It uses a single pool of print engines so that resources are used efficiently.

The following sample code shows how to submit an XML print job to the Print Scheduler service that will print the "C:\Test\Document1.btw" document to a printer. To build this code, you will need to add a reference to “Seagull.Services.PrintScheduler.dll”.

```
using System;
using Seagull.Services.PrintScheduler;


namespace SDKPrintScheduler_BTXML
{

 class Program
  {

   private static string XmlScript = @"<?xml version='1.0' encoding='utf-8'?>
       <XMLScript Version='2.0' Name='09232006_103601_Job1' ID='123'>
       <Command Name='MyJob'>
       <Print WaitForJobToComplete='false' JobName='Xmltest.btw'
       Timeout='30000' ReturnPrintData='true' ReturnSummary='true'
       ReturnLabelData='true' ReturnChecksum='true'>
       <Format>C:\Test\Document1.btw</Format>
       </Print>
       </Command>
       </XMLScript>";

   static void Main()
   {
       BtXmlAction btXmlAction = new BtXmlAction();
       btXmlAction.CloseDocumentsAfterCompletion = true;
       btXmlAction.XmlString = XmlScript;
       btXmlAction.Run();
       BtXmlResult result = btXmlAction.Result;

       if (result.Exception != null)
          throw result.Exception;

       foreach (ActionMessage message in result.Messages)
       {

          Console.WriteLine(String.Format("{0} {1} {2}", message.Severity,
message.Category, message.Text));
       }
     }
  }
}
```

Automating Database Printing

By using BTXML script, you can automate a document to read data from several sources, including text files, a wide variety of database types, and user input.

Text File

The following example shows a BTXML script that prints the Document1.btw document by using the Data1.csv text data file.

```
<?xml version="1.0" encoding="utf-8"?>
<XMLScript Version="2.0">
  <Command Name="Job1">
    <Print>
      <Format>c:\BarTender\Document1.btw</Format>
      <RecordSet Name="Text File 1" Type="btTextFile">
        <FileName>Data1.csv</FileName>
      </RecordSet>
    </Print>
  </Command>
</XMLScript>
```

Embedded Delimited Text

The following example shows a BTXML script that prints Document1.btw by using embedded comma-delimited text. You can pass comma-separated values (CSV) data, or you can delimit the text by using quotation marks, tabs, or a custom delimiter. Line breaks are used to separate each individual record.

```
<?xml version="1.0" encoding="utf-8"?>
<XMLScript Version="2.0">
  <Command Name="Job1">
    <Print>
      <Format>c:\BarTender\Document1.btw</Format>
      <RecordSet Name="Text File 1" Type="btTextFile">
        <Delimitation>btDelimQuoteAndComma</Delimitation>
        <UseFieldNamesFromFirstRecord>true</UseFieldNamesFromFirstRecord>
        <TextData>
          <![CDATA["FirstName","LastName","City","Zip Code"
          "Amanda","Jones","Bellevue","98008"
          "Jessica","Smith","Kirkland","98293"]]>
        </TextData>
      </RecordSet>
    </Print>
  </Command>
</XMLScript>
```

SQL Statements

By using BTXML script, you can print BarTender documents that are connected to a wide variety of database types. The following example shows a BTXML script that prints database records by using an SQL statement, user ID, and password.

```
<?xml version="1.0" encoding="utf-8"?>
<XMLScript Version="2.0">
  <Command Name="Job1">
    <Print>
```

```
      <Format>c:\BarTender\Document1.btw</Format>
      <RecordSet Name="employees" Type="btOLEDB">
        <UserID>Admin</UserID>
        <Password>seagull</Password>
        <SQLStatement>SELECT [last name] FROM [employees] WHERE [last name]
         = 'Swan'</SQLStatement>
      </RecordSet>
    </Print>
  </Command>
</XMLScript>
```

Query Prompts

You can configure BarTender to prompt the print operator to enter data at print time.

The following example shows a BTXML script that fills the prompt with default data by using the `<DefaultReply>` tag. When the query prompt appears at print time, the print operator can accept the default reply data or delete the default reply and then enter new data.

```
<?xml version="1.0" encoding="utf-8"?>
<XMLScript Version="2.0" Name="09232006_103601_Job1" ID="123">
  <Command Name="Job1">
    <Print>
      <Format>c:\BarTender\Document1.btw</Format>
      <QueryPrompt Name="Product">
        <UserPrompt>Enter Product Name:</UserPrompt>
        <DefaultReply>Exotic Liquids</DefaultReply>
      </QueryPrompt>
    </Print>
  </Command>
</XMLScript>
```

Table-Object Databases

Beginning in BarTender 2021, you can configure automated printing for databases that are connected to table objects. The `<RecordSet>` tag was extended in BarTender 2021 to support object databases.

The following example shows a BTXML script that provides substitute content for a text file database. The database is specified by the **Name** attribute of the `<RecordSet>` tag ("Employees" in this example). This is the name of a database that is associated with the table object, not the name of the table object itself. The **Name** attribute is required.

Table-object database changes are always temporary and are reverted after the print job is finished.

```
<?xml version="1.0"encoding="utf-8"?>
<XMLScript Version="2.0">
  <Command>
    <Print JobName="EmployeeList" SaveAfterPrintCondition="IfSerialized">
      <PrintSetup>
         <IdenticalCopiesOfLabel>1</IdenticalCopiesOfLabel>
      </PrintSetup>
      <Format SaveAtEndOfJob="true">d:\tests\table_employees.btw</Format>
      <RecordSet Type="btTextFile" AddIfNone="true" Name="Employees">
        <Delimitation>btDelimMixedQuoteAndComma</Delimitation>
        <UseFieldNamesFromFirstRecord>true</UseFieldNamesFromFirstRecord>
        <TextData>
```

```
            <![CDATA[ID,LastName,Department,City
1245,Hughes,Engineering,"Seattle, Washington"
2568,Peters,Sales,"Minneapolis, Minnesota"
6582,Becker,HR,"Los Angeles, California"
]]>
        </TextData>
      </RecordSet>
    </Print>
  </Command>
</XMLScript>
```

Using BTXML Script with Integration Builder

You can use BTXML script with Integration Builder to automate your printing processes.

Integration Variables

The following example shows a BTXML script that uses variables to work with a file integration. In this example, the Document1.btw document is placed in the same folder that contains the integration file, and the trigger file is used as the file source of the text database in Document1.btw.

```
<?xml version="1.0" encoding="utf-8"?>
<XMLScript Version="2.0">
 <Command Name="Job1">
   <Print>
     <Format>%IntegrationFileFolder%Document1.btw</Format>
     <RecordSet Type="btTextFile">
         <FileName>%FilePath%</FileName>
     <RecordSet
   </Print>
 </Command>
</XMLScript>
```

XSL Transform

In Integration Builder, the "Transform XML using XSLT" action uses an Extensible Stylesheet Language (XSL) stylesheet to transform another application's output XML from its original XML format into another XML format. The following example shows how an XSL transform is used to convert XML to BTXML.

Suppose that your XML resembles the following.

```
<MyPrintJob Name="Job1">
  <FileToPrint>Document1.btw</FileToPrint>
     <Data>
       <Record>
          <FirstName>Angela</FirstName>
          <LastName>Davis</LastName>
          <City>Bellevue</City>
          <ZipCode>98008</ZipCode>
       </Record>
       <Record>
          <FirstName>John</FirstName>
          <LastName>Lewis</LastName>
          <City>Kirkland</City>
          <ZipCode>98298</ZipCode>
       </Record>
     </Data>
</MyPrintJob>
```

You can use the following XSL to transform this code to BTXML.

```
<?xml version="1.0" encoding="utf-8"?>
<xsl:stylesheet version="1.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
<xsl:output cdata-section-elements="TextData"/>
<xsl:template match="/">
<XMLScript Version="2.0">
  <Command Name="{MyPrintJob/@name}">
    <Print>
```

```
      <Format><xsl:value-of select="//FileToPrint"/></Format>
      <RecordSet Type="btTextFile">
        <TextData>
          <xsl:for-each select="//Record[1]/child::*">
            <xsl:if test="position() != last()">"<xsl:value-of select="normalize-
space(name())"/>",</xsl:if>
            <xsl:if test="position() = last()">"<xsl:value-of select="normalize-
space(name())"/>"<xsl:text>&#xD;</xsl:text></xsl:if>
          </xsl:for-each>
          <xsl:for-each select="//Data/Record">
            <xsl:for-each select="./child::*">
              <xsl:if test="position() != last()">"<xsl:value-of
select="normalize-space(.)"/>",</xsl:if>
              <xsl:if test="position() = last()">"<xsl:value-of select="normalize-
space(.)"/>"<xsl:text>&#xD;</xsl:text></xsl:if>
            </xsl:for-each>
          </xsl:for-each>
      </TextData>
    </RecordSet>
  </Print>
</Command>
</XMLScript>
</xsl:template>
</xsl:stylesheet>
```

The resulting BTXML resembles the following.

```
<XMLScript Version="2.0">
  <Command Name="Job1">
     <Print>
       <Format>Document1.btw</Format>
       <RecordSet Type="btTextFile">
          <TextData><![CDATA["FirstName","LastName","City","ZipCode"
          "Angela","Davis","Bellevue","98008"
          "John","Lewis","Kirkland","98298"
          ]]></TextData>
       </RecordSet>
     </Print>
  </Command>
</XMLScript>
```

Setting Template Object Data

By using BTXML script, you can set the values of named data sources that are linked to a template object in a BarTender document.

The following example shows a BTXML script that opens the Document1.btw document, sets values for the data sources that are named Product and Price, and then prints the document.

```
<?xml version="1.0" encoding="utf-8"?>
<XMLScript Version="2.0">
  <Command Name="Job1">
    <Print>
      <Format>c:\BarTender\Document1.btw</Format>
      <NamedSubString Name="Product">
        <Value>Chai Masala</Value>
      </NamedSubString>
      <NamedSubString Name="Price">
        <Value>$18.00</Value>
      </NamedSubString>
    </Print>
  </Command>
</XMLScript>
```

Exporting a Template to an Image File

You can export a BarTender template to an image file by using the `<ExportPrintPreviewToImage>` tag. When you do this, you can use the image files to create the following items:

- An image that can be stored, transported and reprinted
- A record that you can use to visually validate the quality of the printed item
- A file that can be called in a custom application to preview items before they are printed

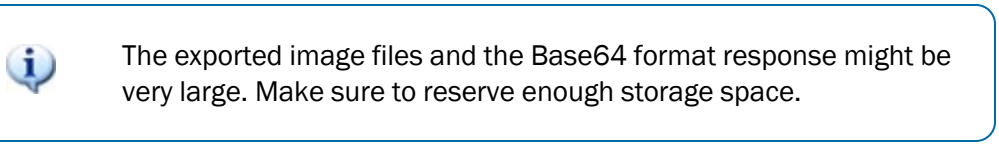
The following example shows the BTXML `<ExportPrintPreviewToImage>` tag. This example creates a 24-bit color *.jpg image file that has a resolution of 300 dots per inch (dpi). Each image includes a border and displays the correct page margins. Because the **ReturnImageInResponse** attribute is set to true, the response that is returned when this command finishes includes a copy of the exported images in Base64 format.

If the print job consists of multiple pages, each page is saved as a unique image file that has a file name that includes the words "Preview Label" and the page number.

```
<?xml version="1.0" encoding="utf-8"?>
<XMLScript Version="2.0" Name="09232006_103601_Job1" ID="123">
  <Command Name="Job1">
    <ExportPrintPreviewToImage ReturnImageInResponse="true">
      <Format>c:\format1.btw</Format>
      <Folder>c:\Images</Folder>
      <FileNameTemplate>Preview_Label%PageNumber%.jpg</FileNameTemplate>
      <ImageFormatType>JPG</ImageFormatType>
      <Colors>btColors24Bit</Colors>
      <DPI>300</DPI>
      <Overwrite>true</Overwrite>
      <IncludeMargins>true</IncludeMargins>
      <IncludeBorder>true</IncludeBorder>
      <BackgroundColor>16777215</BackgroundColor>
    </ExportPrintPreviewToImage>
  </Command>
</XMLScript>
```

Each time that an `<ExportPrintPreviewToImage>` command is passed to BarTender, one or more image files are created in the target folder, and an XML response is generated.

If the **ReturnImageInResponse** attribute is set to false, the command exports a copy of the image that you can view in BarTender History Explorer. If the **ReturnImageInResponse** attribute is set to true, the XML response contains a copy of the image in Base64 format that can be saved and viewed in BarTender History Explorer.



Using the XML Response

In some cases, an XML response is created after the BTXML script runs. An XML response provides you with valuable information about the print job, the printer, the printed items and the BarTender settings that were in effect during the print job. This response can be integrated with a custom application. An XML response is returned when you run BTXML script by using the following methods:

- BarTender SDK
- Integration Builder integration files
- Process Builder process files

You can use the information in the response to do the following:

- Troubleshoot any problems that occurred when the print job was processed.
- Record the status of print jobs.
- Create a record that you can use to validate the content of documents and individual printed items.
- Create a summary of the entire print job.

The default response that is returned contains information about the user who sent the BTXML request, the printer that was used to process the request, the status of the print job and any messages that were created.

The following example shows the response that is returned when no attributes are used or when the **ReturnPrintData** attribute is set to true and the **ReturnSummary** and **ReturnLabelData** attributes are set to false.

```
<?xml version="1.0" encoding="utf-16"?>
<Response Version="2.0" Name="09232006_103601_Job1" ID="123" AppName="BarTender"
AppVersion="9.00" AppVersionId="900" AppVersionMajor="9" AppVersionMinor="00"
AppVersionBuild="2345" AppInstancePid="12345" AppInstanceGUID="{5EFC7975-14BC-
11CF-9B2B-00AA00573819}">
  <User>Administrator</User>
  <Server>MyServer</Server>
  <Command Name="Job1">
    <Print GUID="{C87068F8-4972-41F1-A6E8-724381703764}"
    JobName="MCIJob" ID="1234" JobLastStatus="Sent"
    JobCompleted="true">
      <JobStatus Completed="true">
        <TimeJobStart>02:24:34</TimeJobStart>
        <TimeJobQueued>02:24:34</TimeJobQueued>
        <TimeJobSent>02:24:34</TimeJobSent>
        <LastStatus>Sent</LastStatus>
        <Description>Finished sending print job to printer.</Description>
      </JobStatus>
      <Message ID="1606" GUID="{8A8E8550-C822-4e84-8713-
      212793DFD6E1}" Severity="Information" Category=
      "Miscellaneous" Response="OK">
        <Text>BarTender successfully sent the print job to the spooler.
          Job Name: MyJobName
          BarTender Document: Document1.btw
          Printer: Datamax H-4212 7.1.4 </Text>
      </Message>
    </Print>
```

```
  </Command>
</Response>
```

You can configure the response to include additional information by setting the attributes that are associated with the following commands:

- `<Print>` command attributes: **ReturnPrintData**, **ReturnSummary**, **ReturnLabelData**, and **ReturnChecksum**.
- `<ExportPrintPreviewToImage>` command attribute: **ReturnImageInReponse**.

The following sections describe these attributes and how they affect the XML response.

Returning Print Data in the Response

The summary data and template information that the **ReturnPrintData** attribute displays depend on how the **ReturnSummaryData** and **ReturnLabelData** attributes are set. When you set the **ReturnPrintData** attribute to true and do not include any other attribute, the response includes the print job details, a summary of the print job and the individual item information. When you set the **ReturnSummaryData** or **ReturnLabelData** attribute to false, their information is removed from the response.

To view an example of the response that is returned when the **ReturnPrintData** attribute is set to true and no other attributes are set or when the **ReturnSummary**, **ReturnLabelData** and **ReturnChecksum** attributes are included and also set to true, refer to the Example of Using the Response Tag topic in the BarTender help system.

Returning Summary Data in the Response

When you set the **ReturnSummary** attribute of the `<Print>` command to true or do not include the attribute at all, the response that is returned includes the **ReturnPrintData** attribute information and a summary of the entire print job. The summary includes document and printer information, general page and layout information, and information about how BarTender was configured when it processed the print job.

When you set the **ReturnSummary** attribute of the `<Print>` command to false, the response that is returned does not include a detailed summary of the print job.

For more information about the various summary tags, refer to the Summary Response Tag topic in the BarTender help system.

To view an example that displays the response that is returned when the **ReturnSummary** attribute is set to true and the **ReturnLabelData** attribute is set to false, refer to the Response Format Example: Summary Data Only topic in the BarTender help system.

Returning Template Data in the Response

When you set the **ReturnLabelData** attribute to true or do not include the attribute at all, the response that is returned includes the print job information that is specified in the **ReturnPrintData** attribute and a detailed definition that includes page, template, object and data source information. You can use this information to validate and reprint the print job.

When you set the **ReturnLabelData** attribute to false, the response that is returned does not include any template data.

To view an example that displays the response that is returned when the **ReturnLabelData** attribute is set to true but the **ReturnSummary** attribute is set to false, refer to the Response Format Example: Template Data Only topic in the BarTender help system.

Returning Checksum Data in the Response

To return checksum data in the response, you must set the **ReturnChecksum** attribute to true. When you do this, the attribute returns checksums that represent various parts of the print job and objects that are included on the document that is being printed.

History Explorer and Reprint Console use these checksum values to verify the content of the documents that are being reprinted by comparing the checksum values of the new document with the original one. If the checksum values match, you can be sure that the document that you print exactly matches the original. If the checksum values do not match, you know that the new document changed in some way. In this case, you might choose to print the document that has the difference or correct the difference so that the items match again before you print. For more information about the returned checksum values, refer to the FormatChecksum Response Tag topic in the BarTender help system.

To view an example that displays the response that is returned when the **ReturnChecksum** attribute is set to true, refer to the Example of Using the Response Tag topic in the BarTender help system.

Returning an Exported Template in a Response

The **ReturnImageInReponse** attribute of the `<ExportPrintPreviewToImage>` command returns a response that contains an image of the exported template as a string in Base64 format.

A response that contains template data in Base64 format uses a lot of space. Make sure to reserve enough storage space.

Related Documentation

Technical Documents

- *BarTender Integration Methods*

To view and download technical documents, visit:

https://www.seagullscientific.com/resources/white-papers/

User Guides

- *Getting Started with BarTender*
  https://support.seagullscientific.com/hc/categories/200267887

BarTender Help System

- Automation with BarTender XML Script
- BarTender XML Script Format

Other Resources

Please visit the BarTender website at *https://www.seagullscientific.com*.

BarTender® BY SEAGULL SCIENTIFIC