Return Tag Service

Legal and Copyright Notices

Payment

You must remit payment in accordance with the FedEx Service Guide, tariff, service agreement or other terms or instructions provided to you by FedEx from time to time. You may not withhold payment on any shipments because of equipment failure or for the failure of FedEx to repair or replace any equipment.

Inaccurate Invoices

If you generate an inaccurate invoice, FedEx® may bill or refund to you the difference according to the *FedEx Service Guide*, tariff service agreement or other terms or instructions provided to you by FedEx from time to time. A request for refund on a FedEx shipment must be made in accordance with the applicable Service Guide or terms or instructions provided by FedEx from time to time. A shipment given to FedEx with incorrect information is not eligible for refund under any FedEx money-back guarantee. FedEx may suspend any applicable money-back guarantee in the event of equipment failure or if it becomes inoperative.

Confidential and Proprietary

The information contained in this guide is confidential and proprietary to FedEx Corporate Services, Inc. and its affiliates (collectively "FedEx"). No part of this guide may be distributed or disclosed in any form to any third party without written permission of FedEx. This guide is provided to you and its use is subject to the terms and conditions of the FedEx Automation Agreement. The information in this document may be changed at any time without notice. Any conflict between this guide, the FedEx Automation Agreement and the *FedEx Service Guide* shall be governed by the FedEx Automation Agreement and the *FedEx Service Guide*, in that order.

Disclaimer

All Improper Transaction scenarios are for example only. They do not reflect all error condition scenarios.

Contents

Tables

About This Guide

This guide describes how to integrate with FedEx Web Services.

It is written for the application developer who uses web services to design and deploy applications enabled by FedEx. It describes how to get started with application development and how to use the Application Programming Interface (API). It also describes each available service in addition to the business logic that drives each FedEx process.

Document Organization

Each web service provides access to FedEx features. The service description includes service details and a full schema listing to facilitate application development.

Resources

- FedEx Services At-a-Glance: **fedex.com/us/services**
- *FedEx Service Guide* available at **fedex.com/us/service-guide**
- Microsoft Web Services: **msdn.microsoft.com/en-us/library/ms950421.aspx**
- O'Reilly XML.com: **www.xml.com**
- Secure Socket Layer Certificates: **fedex.com/us/developer/downloads/dev_cert.zip**
- Web Services organization home page: **www.web-services.org**

Support

- Contact FedEx Web Services technical support at **websupport@fedex.com**.
- For technical support, call 1.877.339.2774 and state "API" at the voice prompt.
- Support hours are Monday through Friday, 7:00 a.m. to 9:00 p.m. CST, and Saturday, 9:00 a.m. to 3:00 p.m. CST.
- For FedEx Customer Service, call **1.800.GoFedEx 1.800.463.3339**.
- Customers using a FedEx® Compatible Solutions Program automation solution should contact their software provider for support.

1 Introduction

FedEx Web Services gives you the tools to build custom platform- and interface-independent applications that access FedEx features. You can use FedEx Web Services in a variety of ways to create customized integration solutions for your specific shipping needs. Here are just a few of the ways a company can use web services to streamline operations, improve visibility, and provide more choices to clients:

- Give Customers More Options: Help customers learn about all the available shipping options and rates with Ship Service WSDL, OpenShip WSDL, and Rate Services WSDL. You can also extend this service to your shopping cart and website, allowing customers to access money-saving information firsthand.
- More Convenience: Use the Locations Service WSDL to find the FedEx pickup location nearest your customer. Or, send an email to your customers with a link to this service as part of your standard order-receipt process.
- Offer Global Shipping Options: Create shipping labels for worldwide locations. Improve customer service by offering more shipping options to customers in more countries with the consolidated Ship Service WSDL.
- Reduce Customer Service Costs: Decrease phone traffic from customers checking the status of their shipments and cut customer service costs. FedEx provides online Tracking and Visibility Services that allow you to provide customers with the status of shipments, Signature Proof of Delivery (SPOD), and Shipment Notification in the Ship Request.
- Simplify Processes and Improve Satisfaction: In addition to ExpressTagAvailability, provide a simple way to allow customers to return an order with Email Labels. This service sends an email with the address (URL) of a website where the recipient can log in and print a return label.

Why should developers be interested in web services?

- Interoperability: Any web service can interact with any other web service and can be written in any programming language.
- Ubiquity: Web services communicate using HTTP and XML. Any connected device that supports these technologies can both host and access web services.
- Low Barrier to Entry: The concepts behind web services are easy to understand, and developers can quickly create and deploy them using many toolkits available on the web.
- Industry Support: Major content providers and vendors support the web services movement.

Any application running on any platform can interact with a web service by using the Simple Object Access Protocol (SOAP) and Web Services Description Language (WSDL) standards for message transfer and service discovery. By following the standards, applications can seamlessly communicate with platform services.

1.1 Document Overview

This guide provides instructions for coding the functions you need to develop FedEx supported applications. The following chapters make up this guide:

- Introduction (this chapter):
  - Documentation overview and guidelines, including how to use the Help application and how to print this guide.
  - Overview information about web services, including a high-level description of FedEx Web Services methods.
  - Coding basics.
  - Overview information about testing and certifying your application.

Each chapter covering FedEx Web Services coding includes:

- Service Details: Business rules for using the FedEx service.
- Service Options: Links to additional services that can be added to the basic web service.
- Coding Details: Best practices information, basic request and reply elements, and a link to error messages.
- XML Schema: A link to the layout for the service. This layout provides coding requirements for all elements in the schema.

1.2 Printing All or Part of This Guide

You can print all or part of this guide from the PDF version.

1.2.1 Printing from the PDF Version

From the PDF version you can print the complete document or a page range of the document.

1. Open the PDF file and click the printer icon  or click **File** > **Print**.
2. From the **Print** dialog box, print the complete document, specify a page range, or choose from any of the available print options.

1.3 Web Services, WSDL, and SOAP Overview

This section describes the standard coding technologies used in FedEx Web Services.

1.3.1 Web Services

Web services are a collection of programming technologies, including XML, Web Services Description Language (WSDL), and SOAP, which allow you to build programming solutions for specific messaging and application integration.

Web services are, by definition, platform independent. FedEx Web Services allow developers to build custom applications that are independent of changes to the FedEx interface.

Web Services are consumed by many different applications across many platforms. It is based on the basic principles that govern XML standards, one of which is how Namespaces can be declared and applied.

Namespaces are declared as an attribute of an element. It is not mandatory to declare namespaces only at the root element; rather it could be declared at any element in the XML document. The scope of a declared namespace begins at the element where it is declared and applies to the entire content of that element, unless overridden by another namespace declaration with the same prefix name, the content of an element is the content between the <opening-tag> and </closing-tag> of that element. So essentially, XML namespace declarations are scoped, meaning that the declared prefix (or default namespace) is in force for the element on which the declaration occurs (as well as its descendant elements). A namespace declared as follows:

<v12:RateReply xmlns:v12="http://

is semantically same as

<RateReply xmlns="http://fedex.com/ws/rate/v12">

or even (hypothetically) same as

<foo:RateReply xmlns:foo="http://fedex.com/ws/rate/v12">

1.3.2 WSDL

A SOAP request to, or response from, a service is generated according to the service's WSDL definition. A WSDL document describes a service. It is an XML document that provides information about what the service does, the methods that are available, their parameters, and parameter types. It describes how to communicate with the service in order to generate a request to, or decipher a response from, the service.

The purpose of a WSDL is to completely describe a web service to a client. A WSDL defines where the service is available and what communications protocol is used to talk to the service. It defines everything required to write a program to work with an XML web service. A WSDL document describes a web service using seven major elements. Elements can be abstract or concrete.

Abstract XML elements describe the web service: <types>, <message>, <operation>, <portType>. Concrete XML elements provide connection details: <service>, <port>, <binding>.

1.3.2.1 WSDL Elements

Element	Definition
<definitions>	The root element contains name space definitions.
<portType>	The most important WSDL element. It is a set of all operations that a web service can accept and is a container for <operation> elements. This WSDL element describes a web service, the operations that can be performed, and the messages that are involved, and can be compared to a function library (or a module or a class) in a traditional programming language.
<types>	Defines variable types used in the web service (both the parameters passed to a function and the type of the value passed back via the response). The data types are described by XML schema. This element contains user-defined data types (in the form of XML schema). For maximum platform neutrality, WSDL uses XML schema syntax to define data types.
<message>	Defines the data elements of an operation. Each message can consist of one or more parts that can be compared to the parameters of a function call in a traditional programming language.
<operation>	Child of the <binding> element that defines each operation that the port exposes. This element allows only three messages: Message - Definition Input Message - Data web services receive Output Message - Data web services send Fault Message - Error messages from web services
<service>	Contains a <port> child element that describes the URL where the service is located. This is the location of the ultimate web service.
<binding>	Defines the message format and protocol details for each port. The binding element has two attributes: the name attribute and the type attribute. This element specifies how the client and the web service should send messages to one another.

Note: For more information about the WSDL standard, refer to the World Wide Web Consortium (W3C) Website at w3.org/TR/wsdl.

1.3.3 SOAP

- Is a simple XML-based protocol that allows applications to exchange information over HTTP.
- Is built on open standards supported by numerous development tools on various platforms.
- Is a request interface object in your application programming language.
- Provides a way to communicate between applications running on different operating systems, with different technologies and programming languages.

- Enables the data to pass through layers of intermediaries and arrive at the ultimate receiver the way it was intended.

Note: You may not need to actually construct the SOAP messages yourself — many development tools available today construct SOAP behind the scenes.

1.3.3.1 SOAP Message

A SOAP message is an XML document that can be a request for a web service from a client or a "reply" from a web service to a client.

- Required <SOAP:Envelope>
- Optional <SOAP:Header>
- Required <SOAP:Body>

1.3.3.1.1 Example: Delete Tag Request (SOAP Message)

```
<SOAP-ENV:Envelope
  xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/"
  xmlns:SOAP-ENC="http://schemas.xmlsoap.org/soap/encoding/"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xmlns:xsd="http://www.w3.org/2001/XMLSchema"
  xmlns="http://fedex.com/ws/ship/v13">
<SOAP-ENV:Body>
<DeleteTagRequest>
<WebAuthenticationDetail>
    <UserCredential>
      <Key>
       User Key
      </Key>
    <Password>
     User Password
    </Password>
  </UserCredential>
</WebAuthenticationDetail>
<Client detail>
   <AccountNumber>xxxxxxxxx</Account number>
   <MeterNumber>xxxxxx</MeterNumber>
</ClientDetail>

<Version>
     <ServiceId>ship</ServiceId>
     <Major>12</Major>
     <Intermediate>0</Intermediate>
     <Minor>0</Minor>
</Version>
<DispatchLocationId>MQYA</DispatchLocationId>
<DispatchDate>2012-06-01</DispatchDate>
```

```
<Payment>
     <PaymentType>shipper</PaymentType>
     <Payor>
         <AccountNumber>xxxxxxxxx</AccountNumber>
         <CountryCode>US</CountryCode>
     </Payor>
</Payment>
<ConfirmationNumber>997037200019454</ConfirmationNumber>
</DeleteTagRequest>
</SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

1.3.4 Non-SOAP Web Services

FedEx offers a non-SOAP web services solution that you can use to send transactions without having to use tools that provide SOAP protocol support for web services. This may be convenient for developers using environments that do not provide support for SOAP. With this interface, XML documents are sent directly to the FedEx servers via the HTTP POST command. FedEx provides a set of specifications and examples to help with the development of this type of communications method.

To use the non-SOAP web service solution, you must have a working knowledge of HTTPS and Secure Socket Layering (SSL) encryption, the ability to provide a secure SSL connection to FedEx and the ability to code to an operation interface using XML.

The interfaces used in the SOAP and non-SOAP web services are defined in WSDL files. The WSDL files contain schemas that define the layout of the operations. The same WSDL file is used for both the SOAP and non-SOAP web service users.

Non-SOAP users are concerned only with the schema definitions and not the other WSDL components that are SOAP-specific. The XML data that is sent via the non-SOAP interface looks almost identical to the data that is sent via the SOAP interface. The only difference is that the data sent via the non-SOAP interface does not contain the wrapping Envelope and Body tags that are specific to SOAP. The following is an example of a TrackRequest using the non-SOAP interface.

1.3.4.1 Example Track Request

```
<q0:TrackRequest>
         <q0:WebAuthenticationDetail>
            <q0:UserCredential>
               <q0:Key>xxxxxxxxxxxxxxxx</q0:Key>
               <q0:Password/>
            </q0:UserCredential>
         </q0:WebAuthenticationDetail>
         <q0:ClientDetail>
            <q0:AccountNumber>xxxxxxxxx</q0:AccountNumber>
            <q0:MeterNumber>xxxxxxxx</q0:MeterNumber>
            <q0:IntegratorId/>
            <q0:Localization>
```

```
               <q0:LanguageCode>EN</q0:LanguageCode>
               <q0:LocaleCode>us</q0:LocaleCode>
            </q0:Localization>
         </q0:ClientDetail>
         <q0:TransactionDetail>
            <q0:CustomerTransactionId>Basic_TrackRequest_q0_Internal</q0:Cust
omerTransactionId>
            <q0:Localization>
               <q0:LanguageCode>EN</q0:LanguageCode>
               <q0:LocaleCode>us</q0:LocaleCode>
            </q0:Localization>
         </q0:TransactionDetail>
         <q0:Version>
            <q0:ServiceId>trck</q0:ServiceId>
            <q0:Major>7</q0:Major>
            <q0:Intermediate>0</q0:Intermediate>
            <q0:Minor>0</q0:Minor>
         </q0:Version>
         <q0:SelectionDetails>
            <q0:CarrierCode>FDXE</q0:CarrierCode>
            <q0:PackageIdentifier>
               <q0:Type>TRACKING_NUMBER_OR_DOORTAG</q0:Type>
               <q0:Value>797843158299</q0:Value>
            </q0:PackageIdentifier>
         </q0:SelectionDetails>
         <q0:ProcessingOptions>INCLUDE_DETAILED_SCANS</q0:ProcessingOptions>
      </q0:TrackRequest>
```

1.3.4.2 Error Handling

Error handling for non-SOAP operations is different from error handling for SOAP operations. The SOAP specification provides an error handling mechanism that is not present for non-SOAP operations. For a SOAP operation, a fault is returned as a SOAP exception. For a non-SOAP request, the contents of the SOAP fault are returned as an XML document. These SOAP fault documents are returned in situations such as schema validation failures or when operation types are unrecognized. In the following example, a SOAP fault document is returned from a schema validation failure in which the AccountNumber element was incorrectly sent as the AccountNumberx element:

```
<soapenv:Fault xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/">
<faultcode>soapenv:Server</faultcode>
<faultstring>5: Schema validation failed for request.</faultstring>
<detail>
<con:fault xmlns:con="http://www.bea.com/wli/sb/context">
<con:errorCode>5</con:errorCode>
<con:reason>Schema validation failed for request.</con:reason>
<con:details>
<con1:ValidationFailureDetail
```

```
xmlns:con1="http://www.bea.com/wli/sb/stages/transform/config">
<con1:message>Expected element 'AccountNumber@http://fedex.com/ws/ship/v8'
instead of 'AccountNumberx@http://fedex.com/ws/ship/v8' here in element
ClientDetail@http://fedex.com/ws/ship/v8</con1:message>
<con1:xmlLocation>
<ship:AccountNumberx
xmlns:ship="http://fedex.com/ws/ship/v8">000000000</ship:AccountNumberx>

</con1:xmlLocation>
<con1:message>Expected element 'AccountNumber@http://fedex.com/ws/ship/v1'
before the end of the content in element
ClientDetail@http://fedex.com/ws/ship/v8</con1:message>
<con1:xmlLocation>
<ship:ClientDetail
xmlns:ship="http://fedex.com/ws/ship/8">
<ship:AccountNumberx>0000000000000000000</ship:AccountNumberx>

<ship:MeterNumber>0000000</ship:MeterNumber>
</ship:ClientDetail>
</con1:xmlLocation>
</con1:ValidationFailureDetail>
</con:details>
<con:location>
<con:node>Validate</con:node>
<con:pipeline>Validate_request</con:pipeline>
<con:stage>ValidateRequest</con:stage>
<con:path>request-pipeline</con:path>
</con:location>
</con:fault>
</detail>
</soapenv:Fault>
```

Each reply should be checked for the Fault element to indicate failure in processing the message.

> *Note: Normal error processing still applies; this is an additional error check for incorrect syntax in XML documents.*

Keep in mind that if you use either the SOAP or non-SOAP version of FedEx Web Services, labels are returned as Base64 encoded. To print shipping labels, you must decode labels before sending them to your printer.

1.3.4.3 Non-SOAP HTTP POST Example

The following HTTPS POST example is a valid working example, but is not guaranteed to work for all programming languages, applications, and host systems:

POST /xml HTTP/1.0
Referrer: YourCompanyNameGoesHere

Host: ws.fedex.com
Port: 443
Accept: image/gif, image/jpeg, image/pjpeg, text/plain, text/html, */*
Content-Type: text/xml
Content-length: %d
Your FedEx Transaction

Each line is followed by one new line character except Content-length and the FedEx transaction. Two new line characters follow the Content-length line. The FedEx transaction has no extra characters. The Content-length line should have the length of the FedEx transaction in place of the %d variable.

Note: Port 443 must be opened for bi-directional communication on your firewall.

After formatting your non-SOAP transaction and placing it in a HTTP POST request, you will need to open an SSL connection to the FedEx test server and send the request through FedEx by using your SSL connection.

Next, parse the HTTPS response to determine if there were any errors. Examine the HTTP header to determine if any HTTP or Web Server errors were encountered. If you received a 200 status code, parse the reply to determine if there were any processing problems.

1.3.5 Visual Basic Project Error

You may receive an error indicating that an element is not set, even after setting it in the code. When you set a Boolean type element to true, you may also need to set the specified element to true.

1.4 Implementing FedEx Web Services

Before you begin implementing FedEx Web Services, note the following guidelines:

- FedEx Web Services are designed for use by skilled developers who are familiar with the communication standards SOAP and Web Services Description Language (WSDL).
- Unlike traditional client/server models, such as a web server or web page system, web services do not provide the user with a graphical user interface (GUI). Instead, web services share business logic, data, and processes through a programmatic interface across a network.
- To perform a particular FedEx task such as tracking a package, you need to use a class, module, or function that creates your request, sends it to the FedEx platform, and handles the response.
- FedEx Web Services are designed to support any operating system and coding language. Downloadable sample code is available in Java, C#, VB, .Net and PHP languages from the FedEx Developer Resource Center Technical Resources.
- Transactions submitted to FedEx using FedEx Web Services are required to have a minimum of 128-bit encryption to complete the request.

1.5 Understanding the XML Schema

The XML schema defines the messages that you can use to access the FedEx services. You create a request that contains business data and other instructions and you send it to FedEx. FedEx replies with a response that contains the data resulting from the instructions you sent in.

> *Note: The schema diagrams are conveniently linked to help you find information and child values. The XML schema provides a means for defining the structure, content, and semantics of XML documents.*

An XML schema defines:

- Elements and attributes that can appear in a document
- Elements that are child elements
- Order and number of child elements
- Whether an element is empty or can include text
- Data types, default values, and fixed values for elements and attributes

Some important facts about the XML schema:

- Elements that contain sub-elements or carry attributes have complex types.
- Elements that contain numbers (and strings, and dates, etc.), but do not contain any sub-elements, have simple types. Some elements have attributes. Attributes always have simple types.
- Complex types in the instance document, and some of the simple types, are defined in the schema associated with a FedEx Web Service. Other simple types are defined as part of XML schema's repertoire of built-in simple types.
- XML schema built-in simple types are prefixed by "xs:", which is associated with the XML schema namespace through the declaration xmlns:xs="http://www.w3.org/2001// XMLSchema", displayed in the schema element.
- The same prefix, and the same association, are also part of the names of built-in simple types, such as xs:string. This association identifies the elements and simple types as belonging to the vocabulary of the XML schema language, rather than the vocabulary of the schema author.

1.5.1 Guide to the XML Schema

The XML schema for each WSDL provides details about the structure, content, and semantics of the request XML document sent to a FedEx Web Service and the XML document returned by that FedEx Web Service.

The top of each service schema includes:

- Schema location and schema file name that ends in an ".xsd" suffix.
- Alphabetical listing of complex types for the documented service.

- Alphabetical listing of schema simple types for the documented service.
- Input or request data type for the documented service.
- Output or reply data type for the documented service.

The remainder of the service schema contains tables of information about each element, complex type, and simple type.

Each table consists of some or all of the following sections: diagram, namespace, children, type, properties, used by, facets, and source.

1.6 Implementation Process

Planning your integration and organizing your application data to address your shipping needs can sometimes take more time than the actual implementation of the integration. FedEx Web Services conform to industry standards and are compatible with a comprehensive array of developers' tools. This ensures the fastest time-to-market with maximum flexibility to integrate FedEx transactions and information into your applications. FedEx WSDLs are fully interoperable with any product or developer's tool that also conforms to the WS-I Basic Profile. For details, see ws-i.org/Profiles/BasicProfile-1.1-2004-08-24.

To obtain FedEx Web Services and begin integrating with an application, you need to access documentation, sample code, and sample service requests and replies with the WSDLs from the FedEx Developer Resource Center Technical Resources. Also, obtain a test meter number to engage in real-time online testing in the FedEx hosted test environment.

Note: Not all services are available outside the U.S.

1.6.1 Testing

FedEx supplies a complete online operating environment with which to test your applications against live FedEx servers. To execute test interactions, you must first include a test account number, test meter number, authentication key, and password in your code. These credentials are provided to registered developers.

Production credentials can be obtained prior to the certification process. Advanced services are not enabled, but standard services are enabled. Refer to Preproduction Assistance for more information on support from FedEx.

1.6.1.1 Preproduction Assistance

Preproduction assistance is available via the FedEx Web Integrated Solutions Consultation (WISC) team. If you are in the preproduction stages of implementing a FedEx web integrated solution and would like to speak with a FedEx integration consultant who can assist you in understanding FedEx Web Services, contact your FedEx sales executive or technical support at 1.877.339.2774 Monday thru Friday, 7 a.m. to

9 p.m. and Saturday 9 a.m. to 3 p.m. (CST). Both your FedEx sales executive and technical support can request a WISC team member to contact you within 3 business days.

Corporate developers may find that solutions to their needs have already been implemented by a software vendor that is FedEx® Compatible. If improved time-to-market, cost containment, or specialized knowledge is needed, corporate development planners may want to review the available third-party solutions. To see a list of the solutions provided by the FedEx® Compatible providers, go to the Available FedEx® Compatible Solutions page at http://www.fedex.com/us/compatible/.

1.6.2 Certification

Certification is the process of ensuring that your implementation meets a number of requirements for safe, secure, and effective operation of your solution in the FedEx production environment. Certification requirements differ based on whether you are a corporate or commercial developer, and whether you are implementing using the advanced or standard services.

1.6.3 Go To Production

Once an application has passed certification, the developer must replace the test credentials with the production credentials issued by FedEx. The application connection is then directed to the production servers, and the application is live.

1.6.3.1 Requirements for Corporate and Non-Commercial Developers

There are some differences in how support is provided and in the approvals required to go into production that depend on whether you are creating an application for use by your own company or if you are planning to resell your solution to others.

1.6.3.2 Requirements and Resources for Corporate Developers

Corporate developers are typically part of a dedicated development team at a single company. This category also includes third-party developers (consultants) hired by the company to work on its behalf. In all cases, the integration will be used by the company itself and will not be resold or distributed outside of its own footprint. In this situation, FedEx can support the customer directly.

Requirements and Resources for Corporate Developers	
Must be accepted into the FedEx® Compatible Program	No
Self-certification of implementations using standard services	Yes
Self-certification of implementations using advanced services	No
Certification assistance	Yes (WISC team)
FedEx supports the customer directly	Yes

1.6.3.2.1 Requirements for Consultants

Consultants developing on behalf of a corporate customer must ensure that their client provides their account information and a signed End User License Agreement (EULA) to FedEx to obtain a production test meter.

1.6.3.2.2 Requirements and Resources for Commercial Developers

Commercial developers create solutions with the intent of distributing and/or reselling them to their customers. Because they are deployed in a variety of situations, commercial integrations generally require a higher order of "fit and finish." Commercial developers are responsible for supporting their products for their customers. FedEx has a dedicated team of professionals to help developers commercialize their products and to coordinate the three-way interplay between the developer, the end customer, and FedEx.

If you are a commercial developer interested in becoming a FedEx Compatible provider, go to http://www.fedex.com/us/compatible/ for more information about the FedEx Compatible Program.

1.6.3.3 URL Errors

If a VB.NET or C# project still sends transactions to the test server after changing the URL in the WSDLs to print to production, perform the following:

- Make sure permissions are already activated in the production environment.
- Copy the WSDL files to a different folder.
- Follow the directions on changing the new WSDL files to point to production, as described in the FedEx Developer Resource Center in the "Move to Production" topic.
- Remove existing web services references from your project that point to old WSDLs containing the URLs to the test environment.
- Create new web references that point to the modified WSDLs. Use the same names as the old references.
- Compile and test the project. Your new production credentials should work for standard web services, such as rating or tracking without extra permissions. Advanced web services require permissions to be active before they will work. Old test key values will now return an error message.

2 ExpressTagAvailability

Before creating a Process Tag request, you can use the ExpressTagAvailabilityRequest from the ReturnTagService WSDL to check valid access/pickup times and ready times for Express services only. In addition to the standard AuthenticiationDetail and Client detail elements required for all services, the following element is required:

Table 1. ExpressTagAvailabilityRequest Element

Element	Description
ExpressTagAvailabilityRequest	The following elements are required: • OriginAddress • ReadyDateTime. This element must contain the date and time the package will be ready for pickup. • Packaging • Service

In addition to transaction details and error conditions, the ExpressTagAvailabilityReply returns the following information:

Table 2. ExpressTagAvailabilityReply Elements

Element	Description
AccessTime	This is the minimum time window that must be allocated for the FedEx courier to make the pickup. The difference between the Business Close Time (or the local "cutoff time" if it is earlier than the Business Close Time) and the Package Ready Time must be equal to, or exceed, the access time. *Note: Access time requirements vary by postal code.*
ReadyTime	The latest time a FedEx courier can pick up the FedEx ExpressTag® package. *Note: Ready times vary by postal code.*
Availability	Indicates the FedEx ExpressTag service availability at the shipper postal code indicated in the Request. Valid responses are: • NEXT_DAY_AVAILABLE • SAME_DAY_AND_NEXT_DAY_AVAILABLE

Schema ReturnTagService_v1.xsd

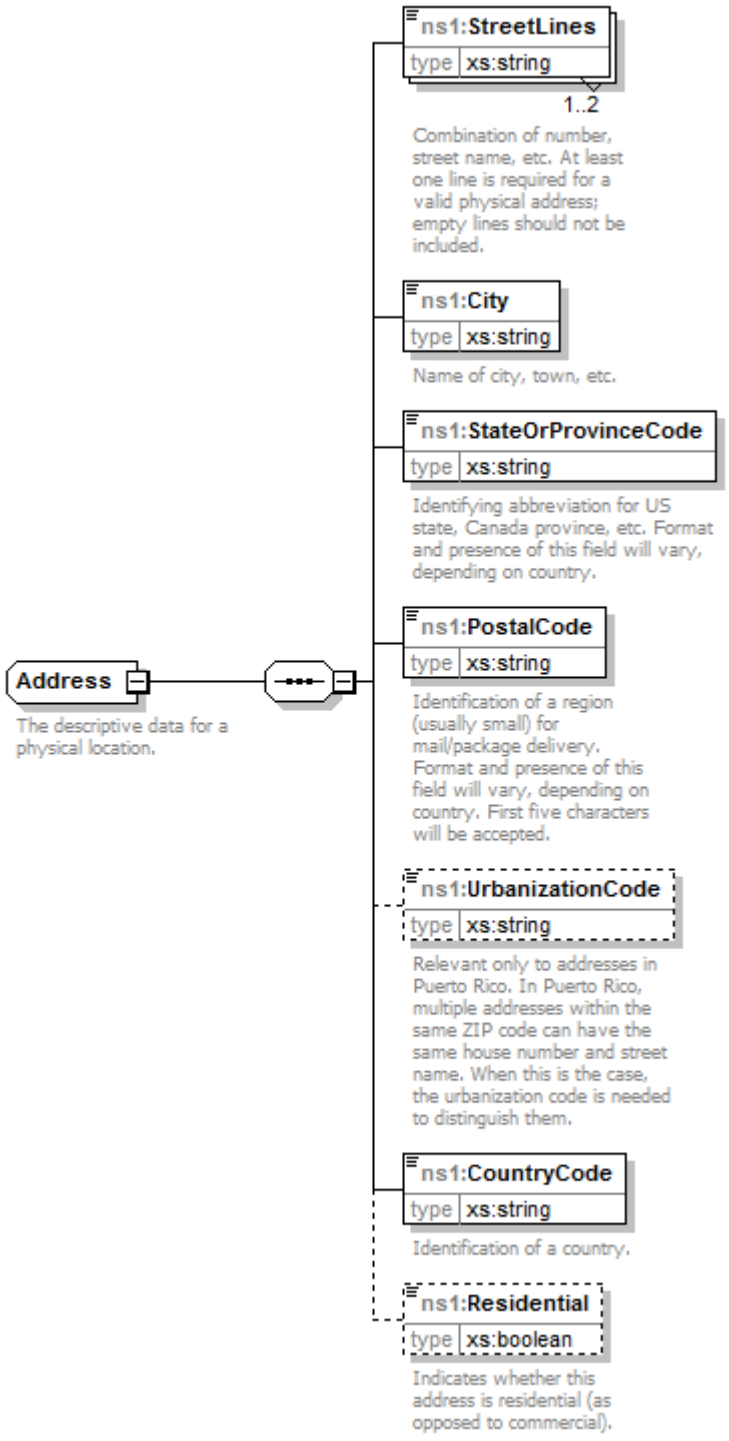
element **ExpressTagAvailabilityReply**

diagram	
namespace	http://fedex.com/ws/returntag/v1
type	**ns:ExpressTagAvailabilityReply**
source	<xs:element name="ExpressTagAvailabilityReply" type="ns:ExpressTagAvailabilityReply"/>

element **ExpressTagAvailabilityRequest**

diagram	
namespace	http://fedex.com/ws/returntag/v1
type	**ns:ExpressTagAvailabilityRequest**
source	<xs:element name="ExpressTagAvailabilityRequest" type="ns:ExpressTagAvailabilityRequest"/>

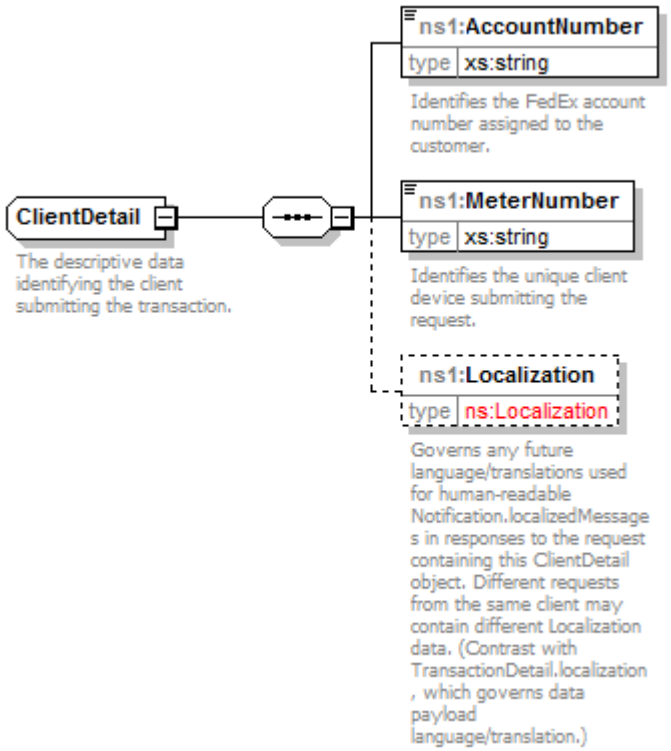
complexType **Address**

diagram	
namespace	http://fedex.com/ws/returntag/v1
children	**ns1:StreetLines ns1:City ns1:StateOrProvinceCode ns1:PostalCode ns1:UrbanizationCode ns1:CountryCode ns1:Residential**
annotation	documentation The descriptive data for a physical location.

source

```
<xs:complexType name="Address">
  <xs:annotation>
    <xs:documentation>The descriptive data for a physical location. </xs:documentation>
  </xs:annotation>
  <xs:sequence>
    <xs:element name="StreetLines" type="xs:string" maxOccurs="2">
      <xs:annotation>
        <xs:documentation>Combination of number, street name, etc. At least one line is
required for a valid physical address; empty lines should not be
included.</xs:documentation>
      </xs:annotation>
    </xs:element>
    <xs:element name="City" type="xs:string">
      <xs:annotation>
        <xs:documentation>Name of city, town, etc. </xs:documentation>
      </xs:annotation>
    </xs:element>
    <xs:element name="StateOrProvinceCode" type="xs:string">
      <xs:annotation>
        <xs:documentation>Identifying abbreviation for US state, Canada province, etc.
Format and presence of this field will vary, depending on country.</xs:documentation>
      </xs:annotation>
    </xs:element>
    <xs:element name="PostalCode" type="xs:string">
      <xs:annotation>
        <xs:documentation>Identification of a region (usually small) for mail/package delivery.
Format and presence of this field will vary, depending on country. First five characters will
be accepted. </xs:documentation>
      </xs:annotation>
    </xs:element>
    <xs:element name="UrbanizationCode" type="xs:string" minOccurs="0">
      <xs:annotation>
        <xs:documentation>Relevant only to addresses in Puerto Rico. In Puerto Rico,
multiple addresses within the same ZIP code can have the same house number and street
name. When this is the case, the urbanization code is needed to distinguish
them.</xs:documentation>
      </xs:annotation>
    </xs:element>
    <xs:element name="CountryCode" type="xs:string">
      <xs:annotation>
        <xs:documentation>Identification of a country.</xs:documentation>
      </xs:annotation>
    </xs:element>
    <xs:element name="Residential" type="xs:boolean" minOccurs="0">
      <xs:annotation>
        <xs:documentation>Indicates whether this address is residential (as opposed to
commercial).</xs:documentation>
      </xs:annotation>
    </xs:element>
  </xs:sequence>
</xs:complexType>
```

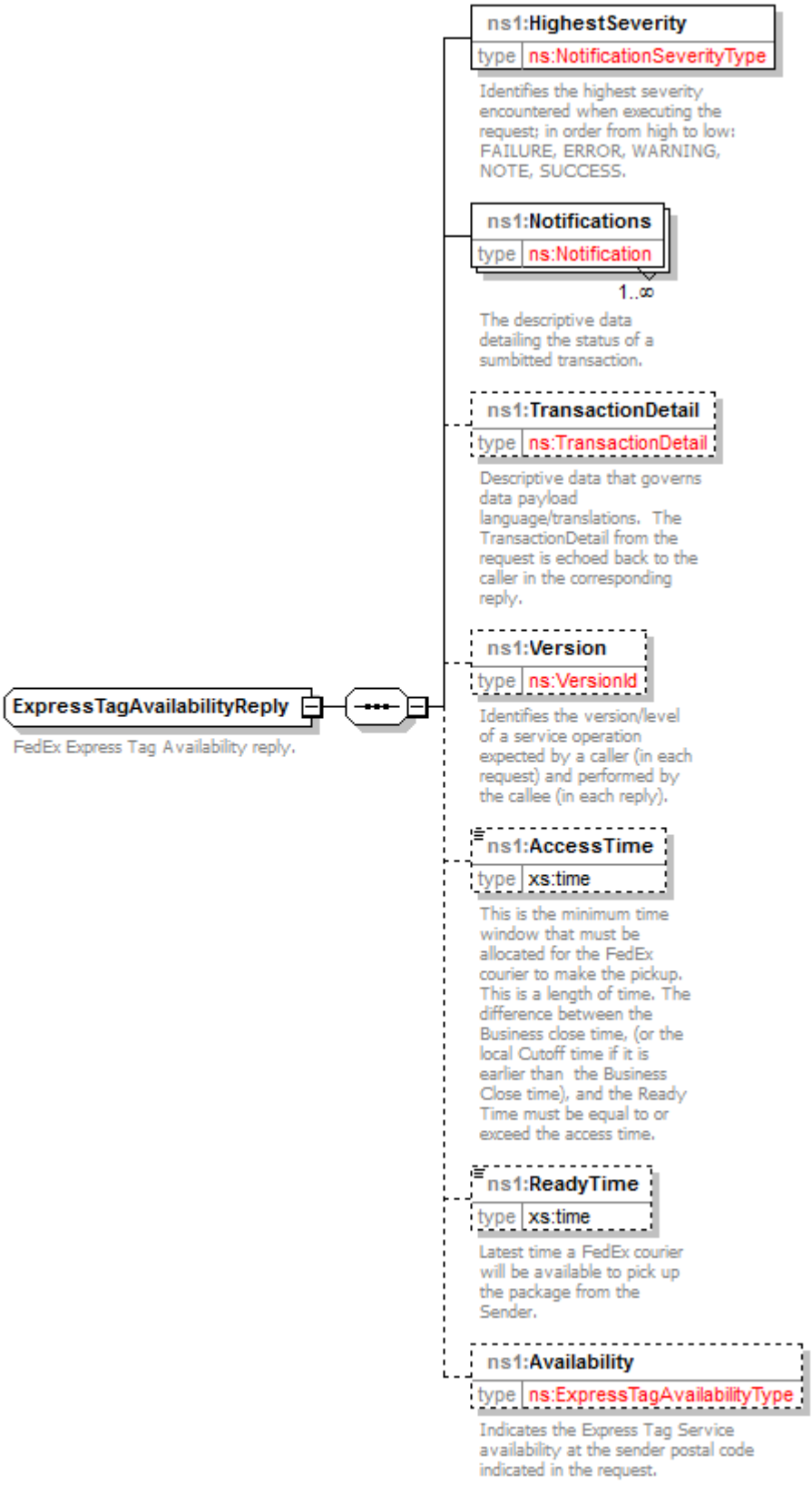
complexType **ClientDetail**

diagram	
namespace	http://fedex.com/ws/returntag/v1
children	**ns1:AccountNumber ns1:MeterNumber ns1:Localization**
annotation	documentation The descriptive data identifying the client submitting the transaction.
source	<xs:complexType name="ClientDetail"> <xs:annotation> <xs:documentation>The descriptive data identifying the client submitting the transaction.</xs:documentation> </xs:annotation> <xs:sequence> <xs:element name="AccountNumber" type="xs:string"> <xs:annotation> <xs:documentation>Identifies the FedEx account number assigned to the customer.</xs:documentation> <xs:appinfo> <xs:MaxLength>12</xs:MaxLength> </xs:appinfo> </xs:annotation> </xs:element> <xs:element name="MeterNumber" type="xs:string"> <xs:annotation> <xs:documentation>Identifies the unique client device submitting the request.</xs:documentation> <xs:appinfo> <xs:MaxLength>10</xs:MaxLength> </xs:appinfo> </xs:annotation>

```
    </xs:element>
    <xs:element name="Localization" type="ns:Localization" minOccurs="0">
      <xs:annotation>
        <xs:documentation>Governs any future language/translations used for human-readable Notification.localizedMessages in responses to the request containing this ClientDetail object. Different requests from the same client may contain different Localization data. (Contrast with TransactionDetail.localization, which governs data payload language/translation.)</xs:documentation>
      </xs:annotation>
    </xs:element>
  </xs:sequence>
</xs:complexType>
```

complexType **ExpressTagAvailabilityReply**

diagram


ns1:HighestSeverity
type ns:NotificationSeverityType
Identifies the highest severity encountered when executing the request; in order from high to low: FAILURE, ERROR, WARNING, NOTE, SUCCESS.
ns1:Notifications
type ns:Notification
1..∞
The descriptive data detailing the status of a sumbitted transaction.
ns1:TransactionDetail
type ns:TransactionDetail
Descriptive data that governs data payload language/translations. The TransactionDetail from the request is echoed back to the caller in the corresponding reply.
ns1:Version
type ns:VersionId
Identifies the version/level of a service operation expected by a caller (in each request) and performed by the callee (in each reply).
ns1:AccessTime
type xs:time
This is the minimum time window that must be allocated for the FedEx courier to make the pickup. This is a length of time. The difference between the Business close time, (or the local Cutoff time if it is earlier than the Business Close time), and the Ready Time must be equal to or exceed the access time.
ns1:ReadyTime
type xs:time
Latest time a FedEx courier will be available to pick up the package from the Sender.
ns1:Availability
type ns:ExpressTagAvailabilityType
Indicates the Express Tag Service availability at the sender postal code indicated in the request.
ExpressTagAvailabilityReply
FedEx Express Tag Availability reply.

namespace	http://fedex.com/ws/returntag/v1
children	**ns1:HighestSeverity ns1:Notifications ns1:TransactionDetail ns1:Version ns1:AccessTime ns1:ReadyTime ns1:Availability**
annotation	documentation FedEx Express Tag Availability reply.

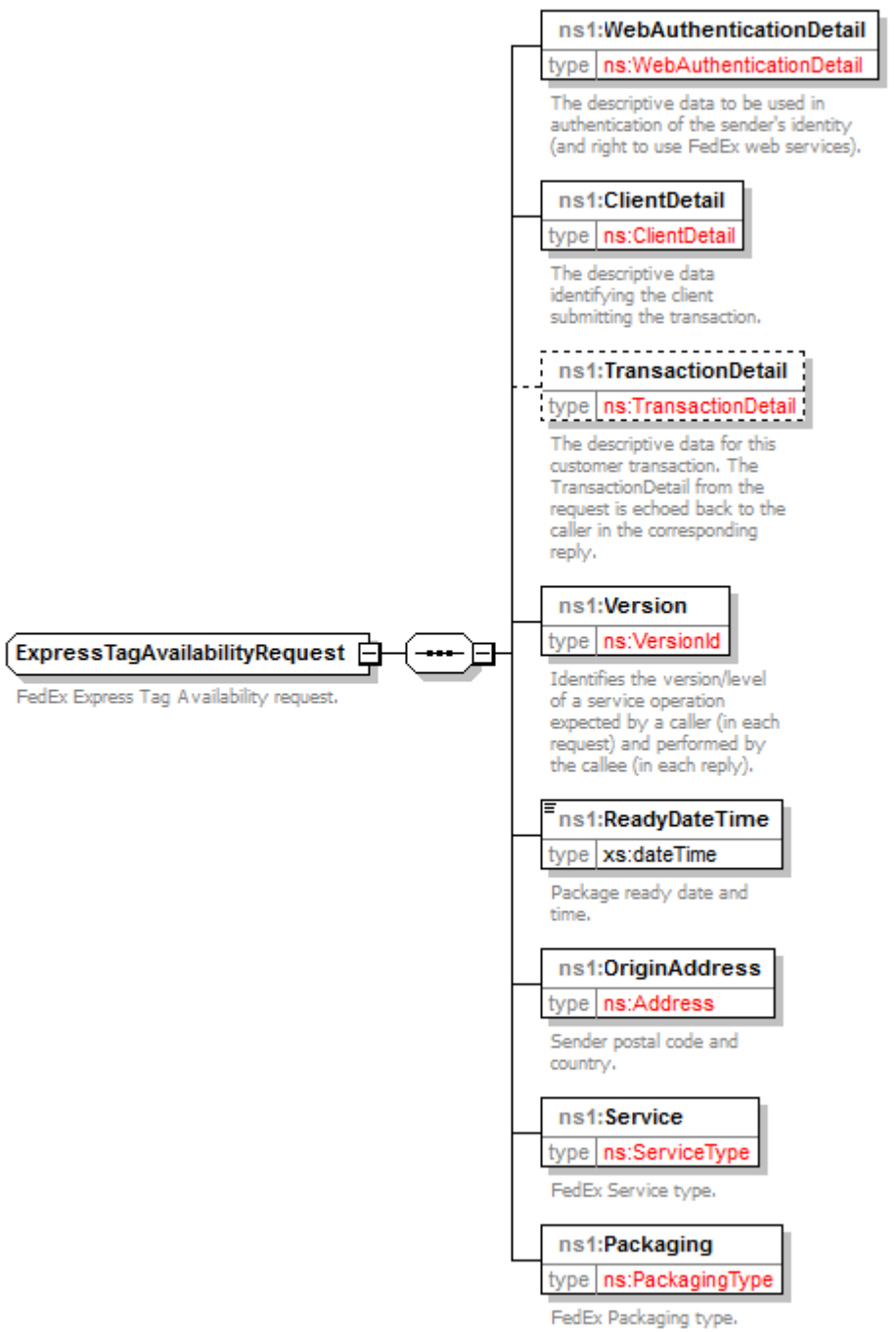
source

```
<xs:complexType name="ExpressTagAvailabilityReply">
 <xs:annotation>
  <xs:documentation>FedEx Express Tag Availability reply.</xs:documentation>
 </xs:annotation>
 <xs:sequence>
  <xs:element name="HighestSeverity" type="ns:NotificationSeverityType">
   <xs:annotation>
    <xs:documentation>Identifies the highest severity encountered when executing the
request; in order from high to low: FAILURE, ERROR, WARNING, NOTE,
SUCCESS.</xs:documentation>
   </xs:annotation>
  </xs:element>
  <xs:element name="Notifications" type="ns:Notification" maxOccurs="unbounded">
   <xs:annotation>
    <xs:documentation>The descriptive data detailing the status of a sumbitted
transaction.</xs:documentation>
   </xs:annotation>
  </xs:element>
  <xs:element name="TransactionDetail" type="ns:TransactionDetail" minOccurs="0">
   <xs:annotation>
    <xs:documentation>Descriptive data that governs data payload language/translations.
The TransactionDetail from the request is echoed back to the caller in the corresponding
reply.</xs:documentation>
   </xs:annotation>
  </xs:element>
  <xs:element name="Version" type="ns:VersionId" minOccurs="0">
   <xs:annotation>
    <xs:documentation>Identifies the version/level of a service operation expected by a
caller (in each request) and performed by the callee (in each reply).</xs:documentation>
   </xs:annotation>
  </xs:element>
  <xs:element name="AccessTime" type="xs:time" minOccurs="0">
   <xs:annotation>
    <xs:documentation>This is the minimum time window that must be allocated for the
FedEx courier to make the pickup. This is a length of time. The difference between the
Business close time, (or the local Cutoff time if it is earlier than  the Business Close time),
and the Ready Time must be equal to or exceed the access time.</xs:documentation>
   </xs:annotation>
  </xs:element>
  <xs:element name="ReadyTime" type="xs:time" minOccurs="0">
   <xs:annotation>
    <xs:documentation>Latest time a FedEx courier will be available to pick up the
package from the Sender.</xs:documentation>
   </xs:annotation>
  </xs:element>
  <xs:element name="Availability" type="ns:ExpressTagAvailabilityType" minOccurs="0">
   <xs:annotation>
    <xs:documentation>Indicates the Express Tag Service availability at the sender postal
code indicated in the request.</xs:documentation>
```

```
        </xs:annotation>
      </xs:element>
    </xs:sequence>
  </xs:complexType>
```

complexType **ExpressTagAvailabilityRequest**

diagram



namespace http://fedex.com/ws/returntag/v1

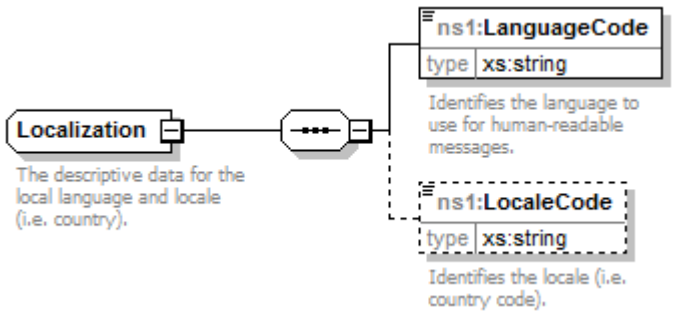
children	**ns1:WebAuthenticationDetail ns1:ClientDetail ns1:TransactionDetail ns1:Version ns1:ReadyDateTime ns1:OriginAddress ns1:Service ns1:Packaging**
annotation	documentation FedEx Express Tag Availability request.
source	

```
<xs:complexType name="ExpressTagAvailabilityRequest">
 <xs:annotation>
  <xs:documentation>FedEx Express Tag Availability request.</xs:documentation>
 </xs:annotation>
 <xs:sequence>
  <xs:element name="WebAuthenticationDetail" type="ns:WebAuthenticationDetail"
minOccurs="1">
   <xs:annotation>
    <xs:documentation>The descriptive data to be used in authentication of the sender's
identity (and right to use FedEx web services).</xs:documentation>
   </xs:annotation>
  </xs:element>
  <xs:element name="ClientDetail" type="ns:ClientDetail">
   <xs:annotation>
    <xs:documentation>The descriptive data identifying the client submitting the
transaction.</xs:documentation>
   </xs:annotation>
  </xs:element>
  <xs:element name="TransactionDetail" type="ns:TransactionDetail" minOccurs="0">
   <xs:annotation>
    <xs:documentation>The descriptive data for this customer transaction. The
TransactionDetail from the request is echoed back to the caller in the corresponding
reply.</xs:documentation>
   </xs:annotation>
  </xs:element>
  <xs:element name="Version" type="ns:VersionId" minOccurs="1">
   <xs:annotation>
    <xs:documentation>Identifies the version/level of a service operation expected by a
caller (in each request) and performed by the callee (in each reply).</xs:documentation>
   </xs:annotation>
  </xs:element>
  <xs:element name="ReadyDateTime" type="xs:dateTime">
   <xs:annotation>
    <xs:documentation>Package ready date and time.</xs:documentation>
   </xs:annotation>
  </xs:element>
  <xs:element name="OriginAddress" type="ns:Address">
   <xs:annotation>
    <xs:documentation>Sender postal code and country.</xs:documentation>
   </xs:annotation>
  </xs:element>
  <xs:element name="Service" type="ns:ServiceType">
   <xs:annotation>
    <xs:documentation>FedEx Service type.</xs:documentation>
   </xs:annotation>
  </xs:element>
  <xs:element name="Packaging" type="ns:PackagingType">
   <xs:annotation>
    <xs:documentation>FedEx Packaging type.</xs:documentation>
   </xs:annotation>
```

```
    </xs:element>
  </xs:sequence>
</xs:complexType>
```

complexType **Localization**

diagram



namespace: http://fedex.com/ws/returntag/v1

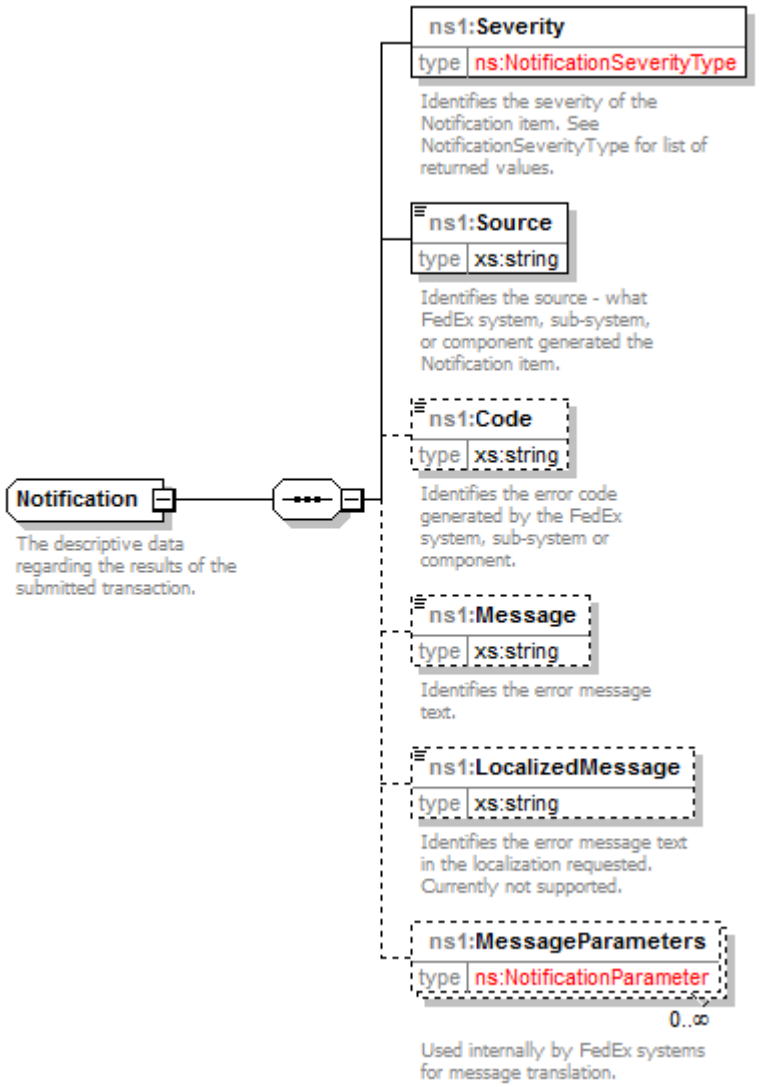
children: **ns1:LanguageCode ns1:LocaleCode**

annotation:
documentation
The descriptive data for the local language and locale (i.e. country).

source

```
<xs:complexType name="Localization">
  <xs:annotation>
    <xs:documentation>The descriptive data for the local language and locale (i.e.
country).</xs:documentation>
  </xs:annotation>
  <xs:sequence>
    <xs:element name="LanguageCode" type="xs:string">
      <xs:annotation>
        <xs:documentation>Identifies the language to use for human-readable
messages.</xs:documentation>
        <xs:appinfo>
          <xs:MaxLength>2</xs:MaxLength>
        </xs:appinfo>
      </xs:annotation>
    </xs:element>
    <xs:element name="LocaleCode" type="xs:string" minOccurs="0">
      <xs:annotation>
        <xs:documentation>Identifies the locale (i.e.  country code).</xs:documentation>
        <xs:appinfo>
          <xs:MaxLength>2</xs:MaxLength>
        </xs:appinfo>
      </xs:annotation>
    </xs:element>
  </xs:sequence>
</xs:complexType>
```

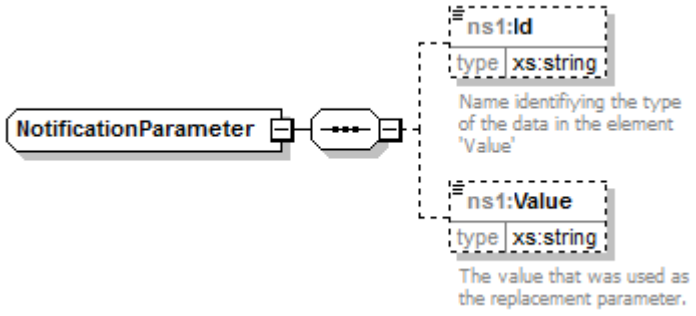
complexType **Notification**

diagram	
namespace	http://fedex.com/ws/returntag/v1
children	**ns1:Severity ns1:Source ns1:Code ns1:Message ns1:LocalizedMessage ns1:MessageParameters**
annotation	documentation The descriptive data regarding the results of the submitted transaction.
source	<xs:complexType name="Notification"> <xs:annotation> <xs:documentation>The descriptive data regarding the results of the submitted transaction.</xs:documentation> </xs:annotation> <xs:sequence> <xs:element name="Severity" type="ns:NotificationSeverityType"> <xs:annotation> <xs:documentation>Identifies the severity of the Notification item. See NotificationSeverityType for list of returned values.</xs:documentation> </xs:annotation> </xs:element>

```
    <xs:element name="Source" type="xs:string">
      <xs:annotation>
        <xs:documentation>Identifies the source - what FedEx system, sub-system, or
component generated the Notification item.</xs:documentation>
      </xs:annotation>
    </xs:element>
    <xs:element name="Code" type="xs:string" minOccurs="0">
      <xs:annotation>
        <xs:documentation>Identifies the error code generated by the FedEx system, sub-
system or component.</xs:documentation>
        <xs:appinfo>
          <xs:MaxLength>8</xs:MaxLength>
        </xs:appinfo>
      </xs:annotation>
    </xs:element>
    <xs:element name="Message" type="xs:string" minOccurs="0">
      <xs:annotation>
        <xs:documentation>Identifies the error message text.</xs:documentation>
        <xs:appinfo>
          <xs:MaxLength>255</xs:MaxLength>
        </xs:appinfo>
      </xs:annotation>
    </xs:element>
    <xs:element name="LocalizedMessage" type="xs:string" minOccurs="0">
      <xs:annotation>
        <xs:documentation>Identifies the error message text in the localization requested.
Currently not supported.</xs:documentation>
      </xs:annotation>
    </xs:element>
    <xs:element name="MessageParameters" type="ns:NotificationParameter"
minOccurs="0" maxOccurs="unbounded">
      <xs:annotation>
        <xs:documentation>Used internally by FedEx systems for message
translation.</xs:documentation>
      </xs:annotation>
    </xs:element>
  </xs:sequence>
</xs:complexType>
```

complexType **NotificationParameter**

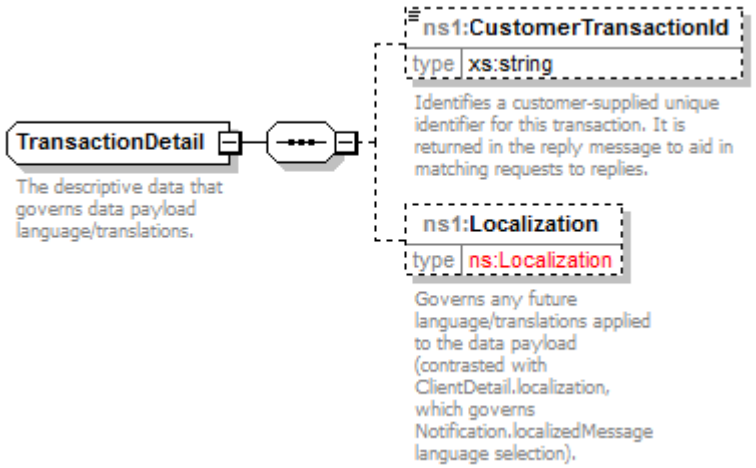
diagram



namespace http://fedex.com/ws/returntag/v1

children	**ns1:Id ns1:Value**
source	<xs:complexType name="NotificationParameter"> <xs:sequence> <xs:element name="Id" type="xs:string" minOccurs="0"> <xs:annotation> <xs:documentation>Name identifiying the type of the data in the element 'Value'</xs:documentation> </xs:annotation> </xs:element> <xs:element name="Value" type="xs:string" minOccurs="0"> <xs:annotation> <xs:documentation>The value that was used as the replacement parameter.</xs:documentation> </xs:annotation> </xs:element> </xs:sequence> </xs:complexType>

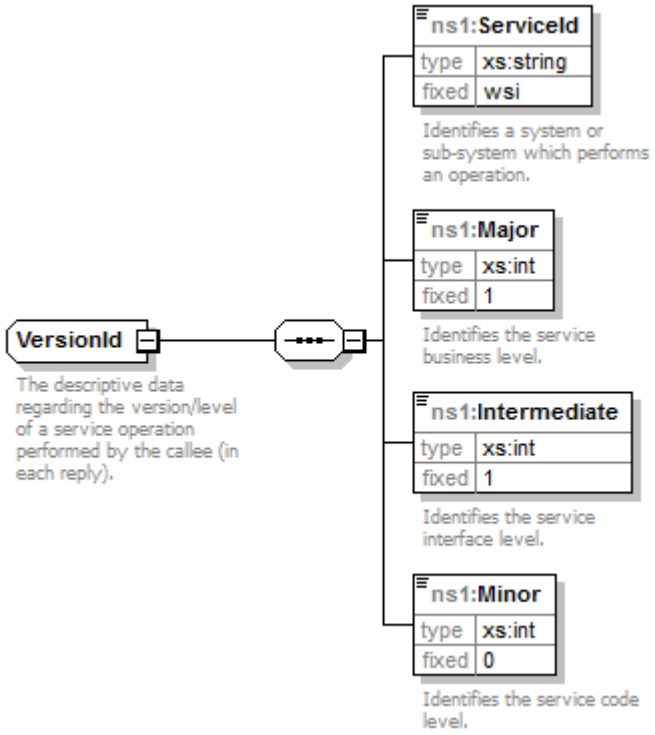
complexType **TransactionDetail**

diagram	
namespace	http://fedex.com/ws/returntag/v1
children	**ns1:CustomerTransactionId ns1:Localization**
annotation	documentation The descriptive data that governs data payload language/translations.
source	<xs:complexType name="TransactionDetail"> <xs:annotation> <xs:documentation>The descriptive data that governs data payload language/translations.</xs:documentation> </xs:annotation> <xs:sequence> <xs:element name="CustomerTransactionId" type="xs:string" minOccurs="0"> <xs:annotation> <xs:documentation>Identifies a customer-supplied unique identifier for this transaction. It is returned in the reply message to aid in matching requests to replies.</xs:documentation> <xs:appinfo>

```
            <xs:MaxLength>40</xs:MaxLength>
          </xs:appinfo>
        </xs:annotation>
      </xs:element>
      <xs:element name="Localization" type="ns:Localization" minOccurs="0">
        <xs:annotation>
          <xs:documentation>Governs any future language/translations applied to the data
payload (contrasted with ClientDetail.localization, which governs
Notification.localizedMessage language selection).</xs:documentation>
        </xs:annotation>
      </xs:element>
    </xs:sequence>
  </xs:complexType>
```

complexType **VersionId**

diagram



namespace	http://fedex.com/ws/returntag/v1
children	**ns1:ServiceId ns1:Major ns1:Intermediate ns1:Minor**
annotation	documentation The descriptive data regarding the version/level of a service operation performed by the callee (in each reply).

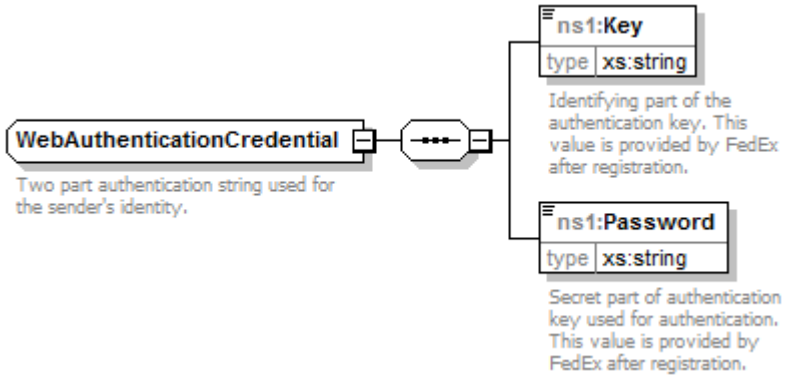
source

```
<xs:complexType name="VersionId">
  <xs:annotation>
    <xs:documentation>The descriptive data regarding the version/level of a service
operation performed by the callee (in each reply).</xs:documentation>
  </xs:annotation>
  <xs:sequence>
    <xs:element name="ServiceId" type="xs:string" fixed="wsi" minOccurs="1">
      <xs:annotation>
```

```
        <xs:documentation>Identifies a system or sub-system which performs an
operation.</xs:documentation>
      </xs:annotation>
    </xs:element>
    <xs:element name="Major" type="xs:int" fixed="1" minOccurs="1">
      <xs:annotation>
        <xs:documentation>Identifies the service business level.</xs:documentation>
      </xs:annotation>
    </xs:element>
    <xs:element name="Intermediate" type="xs:int" fixed="1" minOccurs="1">
      <xs:annotation>
        <xs:documentation>Identifies the service interface level.</xs:documentation>
      </xs:annotation>
    </xs:element>
    <xs:element name="Minor" type="xs:int" fixed="0" minOccurs="1">
      <xs:annotation>
        <xs:documentation>Identifies the service code level.</xs:documentation>
      </xs:annotation>
    </xs:element>
  </xs:sequence>
</xs:complexType>
```

complexType **WebAuthenticationCredential**

diagram



namespace: http://fedex.com/ws/returntag/v1

children: **ns1:Key ns1:Password**

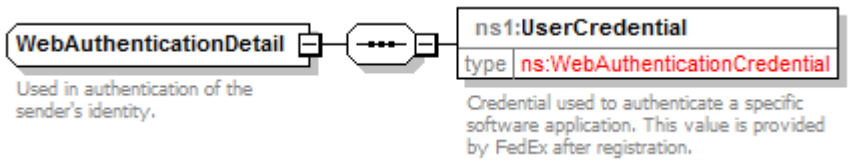
annotation: documentation
Two part authentication string used for the sender's identity.

source

```
<xs:complexType name="WebAuthenticationCredential">
  <xs:annotation>
    <xs:documentation>Two part authentication string used for the sender's
identity.</xs:documentation>
  </xs:annotation>
  <xs:sequence>
    <xs:element name="Key" type="xs:string" minOccurs="1">
      <xs:annotation>
        <xs:documentation>Identifying part of the authentication key. This value is provided
by FedEx after registration.</xs:documentation>
        <xs:appinfo>
```

```
          <xs:MaxLength>16</xs:MaxLength>
        </xs:appinfo>
      </xs:annotation>
    </xs:element>
    <xs:element name="Password" type="xs:string" minOccurs="1">
      <xs:annotation>
        <xs:documentation>Secret part of authentication key used for authentication. This
value is provided by FedEx after registration.</xs:documentation>
        <xs:appinfo>
          <xs:MaxLength>25</xs:MaxLength>
        </xs:appinfo>
      </xs:annotation>
    </xs:element>
  </xs:sequence>
</xs:complexType>
```

complexType **WebAuthenticationDetail**

diagram	
namespace	http://fedex.com/ws/returntag/v1
children	**ns1:UserCredential**
annotation	documentation Used in authentication of the sender's identity.
source	<xs:complexType name="WebAuthenticationDetail"> <xs:annotation> <xs:documentation>Used in authentication of the sender's identity.</xs:documentation> </xs:annotation> <xs:sequence> <xs:element name="UserCredential" type="ns:WebAuthenticationCredential" minOccurs="1"> <xs:annotation> <xs:documentation>Credential used to authenticate a specific software application. This value is provided by FedEx after registration.</xs:documentation> </xs:annotation> </xs:element> </xs:sequence> </xs:complexType>

simpleType **ExpressTagAvailabilityType**

namespace	http://fedex.com/ws/returntag/v1		
type	restriction of **xs:string**		
properties	base xs:string		
facets	Kind enumeration	Value NEXT_DAY_AVAILABLE	Annotation

	enumeration SAME_DAY_AND_NEXT_DAY_AVAILABLE
annotation	documentation Identifies the Express Tag Service availability.
source	<xs:simpleType name="ExpressTagAvailabilityType"> <xs:annotation> <xs:documentation>Identifies the Express Tag Service availability.</xs:documentation> </xs:annotation> <xs:restriction base="xs:string"> <xs:enumeration value="NEXT_DAY_AVAILABLE"/> <xs:enumeration value="SAME_DAY_AND_NEXT_DAY_AVAILABLE"/> </xs:restriction> </xs:simpleType>

simpleType **NotificationSeverityType**

namespace	http://fedex.com/ws/returntag/v1
type	restriction of **xs:string**
properties	base xs:string
facets	Kind Value Annotation enumeration ERROR enumeration FAILURE enumeration NOTE enumeration SUCCESS enumeration WARNING
annotation	documentation Identifies the set of severity values for a Notification.
source	<xs:simpleType name="NotificationSeverityType"> <xs:annotation> <xs:documentation>Identifies the set of severity values for a Notification.</xs:documentation> </xs:annotation> <xs:restriction base="xs:string"> <xs:enumeration value="ERROR"/> <xs:enumeration value="FAILURE"/> <xs:enumeration value="NOTE"/> <xs:enumeration value="SUCCESS"/> <xs:enumeration value="WARNING"/> </xs:restriction> </xs:simpleType>

simpleType **PackagingType**

namespace	http://fedex.com/ws/returntag/v1
type	restriction of **xs:string**
properties	base xs:string
facets	Kind Value Annotation enumeration FEDEX_10KG_BOX enumeration FEDEX_25KG_BOX

	enumeration FEDEX_BOX enumeration FEDEX_ENVELOPE enumeration FEDEX_PAK enumeration FEDEX_TUBE enumeration YOUR_PACKAGING
annotation	documentation Identifies the packaging used by the requestor for the package. See PackagingType for list of valid enumerated values.
source	<xs:simpleType name="PackagingType"> <xs:annotation> <xs:documentation>Identifies the packaging used by the requestor for the package. See PackagingType for list of valid enumerated values.</xs:documentation> </xs:annotation> <xs:restriction base="xs:string"> <xs:enumeration value="FEDEX_10KG_BOX"/> <xs:enumeration value="FEDEX_25KG_BOX"/> <xs:enumeration value="FEDEX_BOX"/> <xs:enumeration value="FEDEX_ENVELOPE"/> <xs:enumeration value="FEDEX_PAK"/> <xs:enumeration value="FEDEX_TUBE"/> <xs:enumeration value="YOUR_PACKAGING"/> </xs:restriction> </xs:simpleType>

simpleType **ServiceType**

namespace	http://fedex.com/ws/returntag/v1
type	restriction of **xs:string**
properties	base xs:string
facets	Kind Value Annotation enumeration EUROPE_FIRST_INTERNATIONAL_PRIORITY enumeration FEDEX_1_DAY_FREIGHT enumeration FEDEX_2_DAY enumeration FEDEX_2_DAY_FREIGHT enumeration FEDEX_3_DAY_FREIGHT enumeration FEDEX_EXPRESS_SAVER enumeration FEDEX_GROUND enumeration FIRST_OVERNIGHT enumeration GROUND_HOME_DELIVERY enumeration INTERNATIONAL_DISTRIBUTION_FREIGHT enumeration INTERNATIONAL_ECONOMY enumeration INTERNATIONAL_ECONOMY_DISTRIBUTION enumeration INTERNATIONAL_ECONOMY_FREIGHT enumeration INTERNATIONAL_FIRST enumeration INTERNATIONAL_PRIORITY enumeration INTERNATIONAL_PRIORITY_DISTRIBUTION enumeration INTERNATIONAL_PRIORITY_FREIGHT enumeration INTERNATIONAL_PRIORITY_PLUS

	enumeration PRIORITY_OVERNIGHT enumeration STANDARD_OVERNIGHT
annotation	documentation Identifies the FedEx service to use in shipping the package. See ServiceType for list of valid enumerated values.
source	<xs:simpleType name="ServiceType"> <xs:annotation> <xs:documentation>Identifies the FedEx service to use in shipping the package. See ServiceType for list of valid enumerated values.</xs:documentation> </xs:annotation> <xs:restriction base="xs:string"> <xs:enumeration value="EUROPE_FIRST_INTERNATIONAL_PRIORITY"/> <xs:enumeration value="FEDEX_1_DAY_FREIGHT"/> <xs:enumeration value="FEDEX_2_DAY"/> <xs:enumeration value="FEDEX_2_DAY_FREIGHT"/> <xs:enumeration value="FEDEX_3_DAY_FREIGHT"/> <xs:enumeration value="FEDEX_EXPRESS_SAVER"/> <xs:enumeration value="FEDEX_GROUND"/> <xs:enumeration value="FIRST_OVERNIGHT"/> <xs:enumeration value="GROUND_HOME_DELIVERY"/> <xs:enumeration value="INTERNATIONAL_DISTRIBUTION_FREIGHT"/> <xs:enumeration value="INTERNATIONAL_ECONOMY"/> <xs:enumeration value="INTERNATIONAL_ECONOMY_DISTRIBUTION"/> <xs:enumeration value="INTERNATIONAL_ECONOMY_FREIGHT"/> <xs:enumeration value="INTERNATIONAL_FIRST"/> <xs:enumeration value="INTERNATIONAL_PRIORITY"/> <xs:enumeration value="INTERNATIONAL_PRIORITY_DISTRIBUTION"/> <xs:enumeration value="INTERNATIONAL_PRIORITY_FREIGHT"/> <xs:enumeration value="INTERNATIONAL_PRIORITY_PLUS"/> <xs:enumeration value="PRIORITY_OVERNIGHT"/> <xs:enumeration value="STANDARD_OVERNIGHT"/> </xs:restriction> </xs:simpleType>