roomsXML API Specifications

01-August-2014

roomsXML Solutions Limited
1133/5 F C Road, Opposite Police Ground, Pune 411 016, India
Tel: +91 20 2566 2024
Fax: +91 20 2566 2021
E-mail: techsupport@roomsxml.com
Web: www.roomsxml.com

Table of contents

1. Introduction

roomsXML is an accommodation distribution system designed for travel companies. The point of difference with roomsXML is in its proprietary software capability to flawlessly map and de-duplicate the inventory that comes from many destination specific suppliers, direct contracting and dynamically from large hotel chains. Please check www.roomsXML.com for more details.

This document is specification for roomsXML 's Web Service / API.

2. Before you start

2.1. Information you will need

Post URL – The URL to which requests should be posted to.

Test environment:

http://www.roomsxmldemo.com/RXLStagingServices/ASMX/XmlService.asmx

Live environment:

http://www.roomsxml.com/RXLServices/ASMX/XmlService.asmx

Credentials:

Login details for an agency user – these have 3 parts: an organisation identifier, a login name, and a password.

Org – {Agency Id}

User – {User Name}

Password – {Password}

Version – The latest version number of the interface. Please check with roomsXML for the latest value of this. The current version is 1.25

2.2. How to send XML request

2.2.1. XML

The simplest method of sending a request is to send the XML directly in the body of an HTTP POST. The raw HTTP data for such a request looks like:

```
POST [The path to the Web service] HTTP/1.1
Content-Type: text/xml
Content-Length: [Length of the content]
Host: [Web service hostname]
<AvailabilitySearch>
<Authority>
<Org>[Your login organisation]</Org>
<User>[Your login username]</User>
<Password>[Your password]</Password>
<Currency>[Your currency]</Currency>
<Version>[Requiredversion]</Version>
```

```
</Authority>
</AvailabilitySearch>
```

In practice though, most development platforms provide a simple mechanism to send such a request without the need to generate the headers yourself.

Sample code (C#):

The following code sends an XML request to the URL specified in xiTargetURL. The content of the request is specified in xiRequestContent. The return value of the function is the XML content returned by the server.

```
public string SendRequest(string xiRequestContent,
string xiTargetURL)
{
WebRequest lRequest=WebRequest.Create(xiTargetURL);
lRequest.Timeout=System.Threading.Timeout.Infinite;
lRequest.Method="POST";
lRequest.ContentLength=xiRequestContent.Length;
lRequest.ContentType="text/xml";
((HttpWebRequest)lRequest).KeepAlive=false;
Stream lStream=lRequest.GetRequestStream();
byte[]
lBytes=Encoding.ASCII.GetBytes(xiRequestContent);
lStream.Write(lBytes, 0, lBytes.Length);
lStream.Close();
WebResponse lResponse=lRequest.GetResponse();
StreamReader
lReader=newStreamReader(lResponse.GetResponseStream());
returnlReader.ReadToEnd();
}
```

Note the above code is a minimal implementation and omits, for example, error handling. It is intended for illustrative purposes only!

2.2.2. SOAP

Sending the request via SOAP results in similar, but the request content is enclosed in a SOAP envelope.

```
POST [The path to the Web service] HTTP/1.1
Host: roomsXML
Content-Type: text/xml
Content-Length: [Length of the content]
SOAPAction: [The URL of the request type]
<?xml version="1.0" encoding="utf-8"?>
<soap:Envelopexmlns:xsi="http://www.w3.org/2001/XMLSche
ma-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
<soap:Body>
<AvailabilitySearchxmlns="http://www.reservwire.com/nam
espace/WebServices/Xml">
<xiRequest>
<Authority>
<Org>[Your login organisation]</Org>
<User>[Your login username]</User>
<Password>[Your password]</Password>
```

```
<Currency>[Your currency]</Currency>
<Version>[Requiredversion]</Version>
</Authority>
</xiRequest>
</AvailabilitySearch>
</soap:Body>
</soap:Envelope>
```

In general there is no reason to prefer the SOAP version over the plain XML version, unless your development platform provides a framework to simplify working with SOAP data (which many do).

2.2.3. WSDL

The roomsXML interface provides a definition of the interface in WSDL format – this describes the interface, and many programming environments will automatically generate a set of helper classes from the WSDL definition.

The WSDL definition will be located at:

www.roomsxmldemo.com/RXLStagingServices/XmlService.asmx?WSDL

Using WSDL in Microsoft Visual Studio

From within the Visual Studio IDE, you can use the WSDL definition as follows:

• Select ‘Add Web Reference’ from the Project menu.
• Enter the URL as described above in the dialog.
• Click ‘Go’ to download the WSDL file.
• Choose a name, and click ‘Add Reference’.

Having imported the WSDL file in this way, the IDE will generate a set of helper classes which encapsulate the interface.

C# example using WSDL:

To send an ‘AvailabilitySearch’ request with a web reference imported as described above, with the name ‘roomsXMLWebService’, the code would look something like:

```
Using roomsXMLWebService;
AvailabilitySearch lRequest =
         new AvailabilitySearch ();
// TODO: Set search properties on lRequest
```

3. Web Service Operations

The roomsXML Web Service Interface is implemented as a dual XML and SOAP interface using Microsoft .NET XML web services. The interface is exposed via:

http://www.roomsxmldemo.com/RXLStagingServices/ASMX/XmlService.asmx

If you navigate to this URL in a web browser, the .NET web services framework will display descriptions of the interfaces and their methods and offer SOAP schema downloads to simplify generating client applications. XSD schemas for the XML interface can also be obtained directly from the system at the same URL.

The rest of this chapter will be written to describe the syntax of the XML interface. The SOAP interface uses data structures equivalent to the complex elements of the XML interface. These can be constructed from the WSDL schema: http://www.roomsxmldemo.com/RXLStagingServices/ASMX/XmlService.asmx?WSDL

3.1. HTTP compression

The roomsXML Web Service Interface supports HTTP 1.1 compression. Clients must take advantage of compression by ensuring that requests to the interface specify the HTTP header:

```
Accept-Encoding: gzip,deflate
```

If this is specified on any request then the corresponding response will be returned in a compressed format.

We recommend that this header is specified for all requests as it can offer a significant performance improvement, especially for large availability searches.

3.2. Common parameter structures

There are a number of common parameter structures used in the roomsXML Web Service Interface. For example, all Web Service operations require authority information.

3.2.1. Authority

The authority information, required by all Web Service operations, combines authentication and configuration:

```
<Authority>
<Org>(Agency ID)
<User>(User Name)
<Password>
<Language>(optional)
<Currency>
<DebugMode>(optional)
<TestMode>(optional)
<Timeout>(optional)
<Version>
</Authority>
```

`<Org>`	String	The organisation short code identifying the agent responsible for the request (Agency ID)
`<User>`	String	The user's logon, unique within the given organisation (User Name)
`<Password>`	String	The given user's password
`<Language>` (optional)	Enum	Override the organisation's default language; accepts an ISO language identifier (e.g. en, fr, etc.); (Refer to the roomsXML Integration Portal)
`<Currency>`	Enum	Choose the currency used to return selling prices;

		accepts an ISO-4217 / SWIFT three character currency code (e.g. USD, GBP, EUR); (Refer to the roomsXML Integration Portal) Your account has a configured billing currency. Only the configured currency can be specified here.
`<DebugMode>` (optional)	Boolean	Enable debugging information. When this flag is set some operations will generate a small amount of additional response data in a <DebugInfo> element. This is purely diagnostic information and may safely be ignored. **It is not recommended that integrators use this parameter unless advised by roomsXML.**
`<TestMode>` (optional)	Boolean	Whether the request is a test request or not. Certain operations will return a similar <TestMode> parameter confirming the mode passed in. If true, the request will be treated as a test request. If false (or not entered), bookings and searches will be live.
`<Timeout>`	Integer	A timeout in seconds. For certain operations (currently only availability search) the operation will be aborted after the specified timeout, and any results already processed will be returned. Setting this property may therefore lead to a reduction in the number of results returned for a given search, and setting it too low may lead to no results being returned at all. We recommend consulting roomsXML before using this interface parameter
`<Version>`	String	The current interface version is 1.25

3.2.2. HotelStayDetails

The HotelStayDetails XML structure is used in availability search and booking operations to describe the time period and room requirements for a booking. The distinction between the two uses is that this structure must contain a room type selection and guest names when used for a booking operation.

```
<HotelStayDetails>
<ArrivalDate><Nights>
<Nationality>
<Room>(1 or more)
```

```
<Guests>
<Adult title="Mr" first="Test" last="test"> (0 or more)
<Child age="5" title="Mr" first="Test" last="test"> (0
or more)
</Guests>
<Extras>
</Room>
</HotelStayDetails>
```

The elements here are similar to the booking data structure:

`<ArrivalDate>`	Date	The date of the first night of the stay
`<Nights>`	Integer	The duration of the stay as a number of nights
`<Nationality>`	String	The country code (ISO 3166 format) of the guests' nationality. This is required to ensure that the prices and availability returned are valid for the guests who are travelling. (Refer to the roomsXML Integration Portal)
`<Room>` (one or more)	Complex	Structure containing room requirements and occupant details
`<Guests>`	Array	An array of either <Adult> or <Child>elements representing the guests who will stay in this room; they must also be named for a booking operation
`<Adult>` (zero or more)	Complex	A structure representing one adult guest
`title, first, last` (Attributes) (may be optional)	String	The name of this adult guest; may be left empty or omitted for availability queries
`<Child>` (zero or more)	Complex	A structure representing one child guest
`title, first, last, age` (Attributes) (may be optional)	String	The name of this child guest; may be left empty or omitted for availability queries. The age of this child guest, if known; may be left empty or omitted for availability queries

3.2.3. HotelSearchCriteria

The hotel search criteria XML structure is a collection of optional elements to narrow down the set of hotels returned.

```
<HotelSearchCriteria>
<HotelName>(optional)
<HotelType>(optional)
<MinStars>(optional)
```

```
<MinPrice>(optional)
<MaxPrice>(optional)
<AvailabilityStatus>(optional)
</HotelSearchCriteria>
```

Note: This doesn't include a Region ID or a specific Hotel ID; those are passed as separate parameters to the relevant XML Web Services.

`<HotelName>` (optional)	String	Match a text fragment within the hotel name
`<HotelType>` (optional)	Enum	Return only hotels of this type. (Refer to the roomsXML Integration Portal)
`<MinStars>` (optional)	Integer	Return only hotels with at least this star rating
`<MinPrice>` (optional)	Decimal	Return only hotels whose total cost (i.e. cost for the whole stay, not cost per night) is at least this value
`<MaxPrice>` (optional)	Decimal	Return only hotels whose total cost (i.e. cost for the whole stay, not cost per night) is at most this value
`<AvailabilityStatus>`(optional)	Enum	The default value to use will be "any", because roomsXML provides available hotels only.

3.3. Common outputs

These structures are common to multiple responses. Descriptions of when individual structures are outputted will be given in the relevant sections.

3.3.1. DebugInfo

This is returned in 'AvailabilitySearch' response. This is only returned if the <DebugMode> parameter is set to true in the relevant request.

Syntax:

```
<DebugInfo>
<VisitId>
</DebugInfo>
```

The DebugInfo response element contains:

`<VisitId>`	Integer	The visit ID associated with this request in the roomsXML system.

3.3.2. TestMode

The TestMode parameter is simply a boolean value. If 'true', the request was sent in test mode, if 'false', the request was sent in live mode.

This parameter corresponds to the TestMode parameter in the Authority element and is returned whether the request contains a TestMode element or not.

3.4. Data download

The first step towards integration with roomsXML is to map the roomsXML regions and hotels into your database.

For this, roomsXML provides the regions and hotels data on the roomsXML Integration Portal.

3.4.1. Regions

The data is provided in a spreadsheet named Regions.xlsx. The sheet contains the regions and the country to which it belongs to.

The file is structured as:

RegionId
RegionName
StateCode
CountryId
CountryName

The Region Id should be used when searching for availability and prices of the regions / destinations.

3.4.2. Hotels

Hotel content is provided in the form of individual XML files for each hotel. Each file is named by the HotelId of that hotel.

Every Hotel element consists of a substantial amount of data:

```
<Hotel>
<Id>
<Name>
<Region>
<Id>
<Name>
<CityId>
</Region>
<Type>
<Address>
<Address1>
```

```
<Address2>(optional)
<Address3>(optional)
<City>
<State>(optional)
<Zip>
<Country>
<Tel>(optional)
<Fax>(optional)
<Email>(optional)
<Url>(optional)
</Address>
<Stars>(optional)
<Rank>(optional)
<GeneralInfo>
<CountRooms>(optional)
<CountFloors>(optional)
<CountSuites>(optional)
<CheckInTime>(optional)
<CheckOutTime>(optional)
<Latitude>(optional)
<Longitude>(optional)
</GeneralInfo>
<Photo>(optional; see notes below)
<Url>
<Width>
<Height>
<Bytes>
<Caption>(optional)
<ThumbnailUrl>(optional)
<ThumbnailWidth>(optional)
<ThumbnailHeight>(optional)
<ThumbnailBytes>(optional)
<PhotoType>(optional)
</Photo>
<Description>(optional; see notes below)
<Language>
<Type>
<Text>
</Description>
<Amenity>(optional; see notes below)
<Code>
<Text>
</Amenity>
<SupplierInfo>(optional; see notes below)
<Supplier>
<SupplierCode>
</SupplierInfo>
<Rating>(optional; see notes below)
<Score>
<System>
<Description>
</Rating>
</Hotel>
```

`<Id>` (basic)	Integer	Numeric ID for the hotel
`<Name>` (basic)	String	The name of the hotel
`<Region>` (basic)	Complex	Contains details of the region that the hotel is in.
`<Id>` (basic)	Integer	Numeric ID for the region
`<Name>` (basic)	String	The name of the region
`<CityId>`	String	The id of the city to which the hotel belongs to. Hotels could exist in the city or in the city districts. But the city id of the hotel is available here.
`<Type>` (basic)	Enum	The type of the hotel
`<Address>`	Complex	Contains the hotel's address and contact information
`<Address1>`	String	The first line of the hotel's street address
`<Address2>` (optional)	String	The second line of the hotel's street address, where appropriate
`<Address3>` (optional)	String	The third line of the hotel's street address, where appropriate
`<City>`	String	The city component of the hotel's address
`<State>` (optional)	String	The state or county component of the hotel's address
`<Zip>`	String	The hotel's zip code or post code
`<Country>`	Enum	The hotel's country, specified as a 2-character ISO 3166 country code (e.g. "US" for "United States", "GB" for "United Kingdom")
`<Tel>` (optional)	String	The hotel's telephone number; this may be exposed to customers on the hotel voucher
`<Fax>` (optional)	String	The hotel's fax number; this will likely not be exposed to customers
`<Email>` (optional)	String	An email address for the hotel; this will likely not be exposed to customers

`<Url>` (optional)	String	An URL for the hotel's own website; this will likely not be exposed to customers
`<Stars>` (optional; basic)	Integer	The hotel's generic star rating
`<Rank>` (optional; basic)	Integer	This is not operational
`<GeneralInfo>`	Complex	Contains miscellaneous information about the hotel
`<CountRooms>` (optional)	Integer	The number of rooms in the hotel
`<CountFloors>` (optional)	Integer	The number of floors the hotel has
`<CountSuites>` (optional)	Integer	The number of suites in the hotel
`<CheckInTime>` (optional)	Time	The hotel's published check-in time
`<CheckOutTime>` (optional)	Time	The hotel's published check-out time
`<Latitude>`(optional)	Double	The latitude of the hotel, stored as the number of degrees North of the equator, taking values from -90.0 to +90.0
`<Longitude>`(optional)	Double	The longitude of the hotel, stored as the number of degrees East of the Prime Meridian through Greenwich, taking values from -180.0 to +180.0
`<Photo>` (optional; zero or more; one only for summary)	Complex	Data pointing to a web-hosted hotel photo
`<Url>`	String	URL of a hotel photo image
`<Height>` `<Width>` `<Bytes>`	Integer	The photo image's height in pixels, width in pixels and file size in bytes
`<Caption>` (optional)	String	Caption for the photo image
`<ThumbnailUrl>` (optional)	String	URL of a thumbnail version of the hotel photo image, where available
`<ThumbnailHeight>` (optional) `<ThumbnailWidth>` (optional)	Integer	The thumbnail photo image's height in pixels, width in pixels and file size in bytes, where available

<ThumbnailBytes> (optional)		
<PhotoType>(optional)	String	Code for the type of photo
<Description> (optional; zero or more; one only for summary)	Complex	Categorised descriptive text about the hotel
<Language>	String	ISO language code for the language of the description text.
<Type>	String	A text enumeration for the description text type (for example, general hotel description or location info)
<Text>	String	Descriptive text in the indicated category; this may be localised
<Amenity> (optional; zero or more;)	Complex	Indicates presence of a particular amenity (E.g. swimming pool, dry cleaning service, etc.) at the hotel
<Code>	Enum	Hotel amenity code
<Text>	String	User-presentable localised hotel amenity description
<Rating> (zero or more; full detail level only)	Complex	A hotel rating for this hotel
<Score>	Integer	This is not operational.
<System>	String	The rating system this rating is applicable for (typically stars or keys)
<Description>	String	String representation of the rating

Linking roomsXML hotels to roomsXML regions:

In our hotel static data files, you will find each hotel has tags <Region><Id> and <CityId>. One of the two Ids will always be present in the regions spreadsheet - Regions.xlsx (which is used to map region codes). This will allow you to link each hotel to the correct region.

As an example:

In the roomsXML regions spreadsheet, the region code for New York City is 19793. If you have mapped this region code correctly to New York City in your system, you will be able to link all roomsXML hotels in New York City to your New York City. Below are extracts from the hotel static data file for 2 New York City hotels.

```
<HotelElement>
<Id>963208</Id>
<Name>Blakely</Name>
<Region>
<Id>64379</Id>
<Name>Midtown Manhattan</Name>
<CityId>19793</CityId>
</HotelElement>

<HotelElement>
<Id>58910</Id>
<Name>Affinia Manhattan</Name>
<Region>
<Id>64379</Id>
<Name>Midtown Manhattan</Name>
<CityId>19793</CityId>
</HotelElement>
```

If you see, both these hotels have a link to '19793'. If you have mapped this region code to the correct region (I.e. New York City) in your database, the above hotels will then be correctly linked to New York City on mapping.

Alternatively, for certain regions, you will find that the valid region code is <Region><Id> and not <CityId> in the regions spreadsheet. For example, the code for Nice, France is 20510…

```
<HotelElement>
<Id>39909</Id>
<Name>Ellington</Name>
<Region>
<Id>20510</Id>
<Name>Nice</Name>
<CityId>16466</CityId>

<HotelElement>
<Id>39642</Id>
<Name>Ambassador</Name>
<Region>
<Id>20510</Id>
<Name>Nice</Name>
<CityId>16466</CityId>
```

Basically, if you don't find the <Region><Id> in the regions spreadsheet, use the <CityId>.

3.5. Operations

The roomsXML Web Service interface is implemented as a dual XML and SOAP interface using Microsoft .NET XML web services. The interface is exposed via:

http://www.roomsxmldemo.com/RXLStaging Services/ASMX/XmlService.asmx

If you navigate to this URL in a web browser, the .NET web services framework will display descriptions of the interfaces and their methods and offer SOAP schema downloads to simplify generating client applications. XSD schemas for the XML interface can also be obtained directly from the system at the same URL.

3.5.1. AvailabilitySearch

The AvailabilitySearch operation determines availability in a given region or specific hotel for the dates and occupancy passed. It returns a list of matching hotels, room options and prices together with a temporary Quote Id for each hotel room. The Quote Id should then be passed to the BookingCreate operation to book a room.

Request syntax:

```
<AvailabilitySearch>
<Authority>
<RegionId>or<HotelId>
<HotelStayDetails>
<HotelSearchCriteria>(optional)
<DetailLevel>(optional)
<CustomDetailLevel>(optional)
<MaxResultsPerHotel>(optional, ignored)
<MaxHotels>(optional)
<SortOrder>(optional, ignored)
<MaxSearchTime>(optional, ignored)
</AvailabilitySearch>
```

The AvailabilitySearch element contains:

`<Authority>`	Complex	Authentication information, as described earlier.
`<RegionId>` (optional)	Integer	The ID number of the region to search for availability. This should not be used with a <HotelId> element. Region Ids may be obtained from the roomsXML Integration Portal.
`<HotelId>` (optional)	Integer	The ID number of an individual hotel to search for availability. This should not be used with a <RegionId> element.
`<HotelStayDetails>`	Complex	Structured information about the requested hotel. It need not contain guest names.
`<HotelSearchCriteria>` (optional)	Complex	Structured extra requirements to filter the list of hotels returned, as described earlier.

		You cannot use HotelSearchCriteria filter when also specifying a HotelId.
`<DetailLevel>` (optional)	Enum	Amount of data to return for each hotel with availability levels: basic, summary, full and custom as defined in section 3.9.6 If specified this should always be 'basic'.
`<CustomDetailLevel>`(optional)	String	A comma separated list of enum values specifying the fields to be returned. This field is used if the value of <DetailLevel> is 'custom'. Note that cancellation fees can be requested, but (just like the price) it is not finalised until a CreateBooking Request is sent. (You must send a "prepare" level BookingCreate in order to confirm the cancellation details and prices before you make a booking)
`<MaxResultsPerHotel>`(optional, currently ignored)	Integer	Maximum number of results to return for each hotel. This is included for future use only and is not presently supported.
`<MaxHotels>` (optional)	Integer	Maximum number of hotels to return results for
`<SortOrder>` (optional, currently ignored)	Enum	Order in which to sort the results. Useful when specifying a value for MaxHotels. This is included for future use only and is not presently supported.
`<MaxSearchTime>` (optional, currently ignored)	Integer	Maximum number of seconds to search before aborting and returning all results found so far. This parameter is not used,

		although the <Timeout> parameter on the authority element may be used in a similar way.

Result syntax:

```
<AvailabilitySearchResult>
<Currency>USD</Currency>
<HotelAvailability> (0 or more)
<Hotel>
<Result id=""> (1 or more)
<Room>(1 or more)
<RoomType>
<MealType> (optional; see notes)
<MealTypeCode>
<Night> (optional; see notes)
<Id>
<Price>
</Night>
<Guests>
<Adult id="" title="" first="" last=""> (0 or more)
<Price>
</Adult>
<Child id="" age="" title="" first="" last=""> (0 or
more)
<Price>
</Child>
</Guests>
<Price>
<Confirmation>
</Room>
</Result>
</HotelAvailability>
</AvailabilitySearchResult>
```

The AvailabilitySearchResult element contains an array of zero or more HotelAvailability elements corresponding to hotels in that region which match the search criteria and also have availability for the given nights.

`<HotelAvailability>` (zero or more)	Complex	
`<Hotel>`	Complex	Contains hotel details at the requested detail level in the structure described earlier.
`<Result>`	Complex	Contains details of a single search result matching the given search criteria
`Id` (Attribute)	String	Temporary identifier for this search result to be passed to a booking operation

`<Room>` (one or more)	Complex	Contains details of an individual room type included in this search result. A single result may contain multiple rooms if the search merits it.
`<RoomType>`	Complex	A text enumeration (as a <code>, <text> pair) identifying a specific room type within the basic room type category; these codes will vary individually per hotel
`<MealType>`	Enum	A text enumeration identifying the meal type for the room. If not returned the MealType is 'room only'.
`<MealTypeCode>`	String	Code representing the above MealType
`<Night>` (optional)	Complex	Contains price information for an individual night of the booking As of version 1.1 of the interface, this data is not included in any of the default detail levels. This change was made since excluding nightly prices significantly reduces the size of responses and improves performance. To include this data in responses, a custom detail level must be used, including the "nightlyprice" option. Conversely, users may wish to explicitly exclude this data from their responses, again using a custom detail level, in order to improve performance
`<Guests>`	Array	An array of either <Adult> or <Child> elements representing and naming the guests who will stay in this room
`<Adult>` (zero or more)	Complex	A structure representing one adult guest
`id` (attribute)	Integer	A zero-based enumeration for all adults in this room (i.e. 0 for the first adult, 1 for the second, etc.)
`first, last, title` (attributes) (optional)	String	The name of this adult guest

`<Price>`	Complex	The proportion of the price associated with this guest. This is not guaranteed to be accurate, but may be useful as an indication of the relative cost for each guest.
`<Child>` (zero or more)	Complex	A structure representing one child guest
`id` (attribute)	Integer	A zero-based enumeration for all children in this room (i.e. 0 for the first child, 1 for the second, etc.)
`first, last, title, age` (attributes) (optional)	String	The name of this child guest
`<Confirmation>`	Enum	Two-value text enumeration for the confirmation type available for this room: allocation – the room is drawn from allocation and the booking can be confirmed immediately request – the booking cannot be confirmed immediately (Note: roomsXML provides available rooms only)

Searching a region:

Once you've obtained a list of region Ids and / or hotel Ids, you can submit availability requests.

E.g. to search for a room for 2 adults in all hotels in region 1000 on 5th January 2014 for 7 nights:

```
<AvailabilitySearch>
<Authority>
<Org>[Your login organisation]</Org><User>[Your login
username]</User>
<Password>[Your password]</Password>
<Currency>[Your currency]</Currency>
<Version>[Required version]</Version>
</Authority>
<RegionId>1000</RegionId>
<HotelStayDetails>
<ArrivalDate>2014-01-05</ArrivalDate>
<Nights>7</Nights>
<Room>
<Guests>
<Adult/>
```

```
<Adult/>
</Guests>
</Room>
</HotelStayDetails>
</AvailabilitySearch>
```

Note: You don’t need to specify any details of adult guests at this stage. If searching for children, you will need to specify their ages:

E.g. the same search as above, but for 1 adult and a 5-year old child:

```
<AvailabilitySearch>
<Authority>
<Org>[Your login organisation]</Org>
<User>[Your login username]</User>
<Password>[Your password]</Password>
<Currency>[Your currency]</Currency>
<Version>[Required version]</Version>
</Authority>
<RegionId>1000</RegionId>
<HotelStayDetails>
<ArrivalDate>2014-01-05</ArrivalDate>
<Nights>7</Nights>
<Room>
<Guests>
<Adult/>
<Child age=”5”/>
</Guests>
</Room>
</HotelStayDetails>
</AvailabilitySearch>
```

Searching a single hotel:

If you are only interested in a specific hotel, you can specify the hotel Id instead of the region Id.

E.g. the same search as above, but for hotel with Id 2000:

```
<AvailabilitySearch>
<Authority>
<Org>[Your login organisation]</Org>
<User>[Your login username]</User>
<Password>[Your password]</Password>
<Currency>[Your currency]</Currency>
<Version>[Required version]</Version>
</Authority>
<HotelId>2000</HotelId>
<HotelStayDetails>
<ArrivalDate>2014-01-05</ArrivalDate>
<Nights>7</Nights>
<Room>
<Guests>
<Adult/>
<Child age=”5”/>
</Guests>
</Room>
</HotelStayDetails>
</AvailabilitySearch>
```

Note: You don’t need to specify the region Id if you specify the hotel Id. Specifying both will cause an error.

Searching for multiple rooms:

If you wish to search for multiple rooms in the same hotel, you can add extra <Room> elements to your search.

E.g. the same search as above, but with a second room for 3 adults:

```
<AvailabilitySearch>
<Authority>
<Org>[Your login organisation]</Org>
<User>[Your login username]</User>
<Password>[Your password]</Password>
<Currency>[Your currency]</Currency>
<Version>[Required version]</Version>
</Authority>
<HotelId>2000</HotelId>
<HotelStayDetails>
<ArrivalDate>2014-01-05</ArrivalDate>
<Nights>7</Nights>
<Room>
<Guests>
<Adult/>
<Child age=”5”/>
</Guests>
</Room>
<Room>
<Guests>
<Adult/>
<Adult/>
<Adult/>
</Guests>
</Room>
</HotelStayDetails>
</AvailabilitySearch>
```

Note: roomsXML considers multiple rooms booked together as a single booking. A single booking reference is provided for both rooms.

Requesting more detail in searches:

You can control the amount of detail that is returned by most queries by specifiying the <DetailLevel> and <CustomDetailLevel> tags. The basic detail levels are normally basic, summary, full or custom. However, you must use the basic level only. roomsXML does not permit the use of other detail levels for performance reasons.

Note: To use the <CustomDetailLevel>, you must first specify the <DetailLevel> to be custom.

```
<AvailabilitySearch>
<Authority>
<Org>[Your login organisation]</Org>
<User>[Your login username]</User>
<Password>[Your password]</Password>
<Currency>[Your currency]</Currency>
```

```
<Version>[Required version]</Version>
</Authority>
<HotelId>2000</HotelId>
<HotelStayDetails>
<ArrivalDate>2014-01-05</ArrivalDate>
<Nights>7</Nights>
<Room>
<Guests>
<Adult/>
<Child age=”5”/>
</Guests>
</Room>
</HotelStayDetails>
<DetailLevel>custom</DetailLevel>
<CustomDetailLevel>basic</CustomDetailLevel>
</AvailabilitySearch>
```

Filtering your search:

You can filter the search on various criteria. E.g. hotel name, rating or cost. To do this, add a HotelSearchCriteria element to your request

E.g. the same search as above, but including only hotels with 4-star rating or higher:

```
<AvailabilitySearch>
<Authority>
<Org>[Your login organisation]</Org>
<User>[Your login username]</User>
<Password>[Your password]</Password>
<Currency>[Your currency]</Currency>
<Version>[Required version]</Version>
</Authority>
<HotelId>2000</HotelId>
<HotelStayDetails>
<ArrivalDate>2014-01-05</ArrivalDate>
<Nights>7</Nights>
<Room>
<Guests>
<Adult/>
<Child age=”5”/>
</Guests>
</Room>
<Room>
<Guests>
<Adult/>
<Adult/>
<Adult/>
</Guests>
</Room>
</HotelStayDetails>
<HotelSearchCriteria>
<MinStars>4</MinStars>
</HotelSearchCriteria>
</AvailabilitySearch>
```

3.5.2. BookingCreate

The booking create operation accepts a Quote ID from an AvailabilitySearch operation and generates a hotel booking. It is also used to perform a dry-run booking which:

• verifies the details of the booking and confirms availability, price and cancellation details.

• returns the hotel's descriptions of the specific room types and meal types selected

• obtains any hotel-specific notes relevant to the stay (E.g. the swimming pool is closed for maintenance)

These should be presented to the end-user prior to final confirmation.

Request syntax:

```
<BookingCreate>
<Authority>
<QuoteId>
<AgentReference>(optional)
<HotelStayDetails>
<HotelBooking>(0 or more)
<QuoteId>
<HotelStayDetails>
</HotelBooking>
<CommitLevel>
</BookingCreate>
```

`<Authority>`	Complex	Authentication information, as described earlier
`<AgentReference>`	String	A custom reference that will be added to the booking and is exposed in Booking elements.
`<HotelBooking>` (0 or more)	Complex	Structure containing a QuoteId and HotelStayDetails structure. Zero or more of these elements can be used. Each of these has information about the items to be added to the specified booking.
`<QuoteId>`	String	The search result Id returned from AvailabilitySearch that corresponds to the hotel to be booked.
`<HotelStayDetails>`	Complex	Structured information about the requested hotel stay, as described earlier; must contain guest names. As the <QuoteId> parameter is specified, only the <Guests> for each room should be included in the <HotelStayDetails> element

`<CommitLevel>`	Enum	Two-valued text enumeration: prepare – perform a dry-run booking, verifying all details such as prices and cancellation fees, as much as possible and obtaining hotel alert information, etc. confirm – go ahead and create the booking

Result syntax:

```
<BookingCreateResult>
<CommitLevel>
<Booking>
<HotelDetails>(0 or 1)
<Alert>(0 or more)
<DateFrom>
<DateTo>
<Text>
</Alert>
</HotelDetails>
</BookingCreateResult>
<BookingCreateResult>
<CommitLevel>
<Booking>
<HotelBooking>
<TotalSellingPrice>
</TotalSellingPrice>
<Status>confirmed</Status>
<Room>
<TotalSellingPrice>
</TotalSellingPrice>
<NightCost>
</NightCost>
<RoomType>
</RoomType>
<Guests>
</Guests>
<Messages>
</Messages>
<Status>confirmed</Status>
<CanxFees>
</CanxFees>
</Room>
<VoucherInfo>
<PayableBy />
<VoucherRef>866642!256 /
PGQSsF2kGMhLtlhcItsyNg==</VoucherRef>
</VoucherInfo>
</HotelBooking>
```

```
</Booking>
</BookingCreateResult>
```

`<CommitLevel>`	Enum	Indicates the commit level with which the booking creation was performed
`<Booking>`	Complex	A booking structure containing as much information as possible. For preparation operations, this will not contain the roomsXML booking reference number, but all other fields will be filled in as appropriate – including the hotel's description of the room type and meal type to be booked. For commit operations, this will now be complete with roomsXML's booking reference number.
`<HotelDetails>`	Complex	Contains extra information about the hotel room booked / to be booked
`<Messages>` (zero or more)	Complex	Special notes provided by the hotel are provided here. These must be displayed to the guests.
`<VoucherInfo>`	Complex	Contains the supplier / hotel reference number that must be displayed on the voucher that is provided to guest.

Error codes at booking:

The meaning of the errors which can be returned in a BookingCreate message is as follows:

7001	The price of the quote has changed since you performed a search. In this case you may perform a new availability search to get a new quote with an accurate price.
7002	You don't have permission to view this quote – this generally means the user who submitted the availability request which created the quote is not the same user who is attempting to book. If you have multiple logins you must use the same one booking a quote as you did when you performed the search which returned it.
7003	The quote is no longer available – quotes are expired after a certain period of time, after which their details are no longer available and they cannot be booked. The interval is generally of 20 minutes.

Making a booking:

To make a booking you should first perform an availability search as detailed above. The result of this will be a list of <HotelAvailability> elements, each of which contains a <Result id=""> element. You need to extract the id="" of the Result you wish to book and use this in the BookingCreate request. As mentioned previously, availability searches aren't guaranteed to return the most accurate data, so you will probably want to check the booking prices, availability and cancellation policy, to confirm that there haven't been any major changes.

This is done by sending a BookingCreate message with the <CommitLevel> set to 'prepare'.

You will get a full Booking Response back, from which you can check the various details before finalising the booking. The booking is finalised by re-sending the Create Request with the <CommitLevel> changed to 'confirm'.

E.g. To dry-run a booking for Quote ID '100-3', for customer 'John Smith':

```
<BookingCreate>
<Authority>
<Org>[Your login organisation]</Org>
<User>[Your login username]</User>
<Password>[Your password]</Password>
<Currency>[Your currency]</Currency>
<Version>[Required version]</Version>
</Authority>
<QuoteId>100-3</QuoteId>
<HotelStayDetails>
<Room>
<Guests>
<Adult title="Mr" first="John" last="Smith" />
</Guests>
</Room>
</HotelStayDetails>
<CommitLevel>prepare</CommitLevel>
</BookingCreate>
```

E.g. to finalise a Booking from Quote ID '100-3', for customer 'John Smith':

```
<BookingCreate>
<Authority>
<Org>[Your login organisation]</Org>
<User>[Your login username]</User>
<Password>[Your password]</Password>
<Currency>[Your currency]</Currency>
<Version>[Required version]</Version>
</Authority>
<QuoteId>100-3</QuoteId>
<HotelStayDetails>
<Room>
<Guests>
<Adult title="Mr" first="John" last="Smith" />
</Guests>
</Room>
</HotelStayDetails>
<CommitLevel>confirm</CommitLevel>
</BookingCreate>
```

Booking query:

The booking query operation accepts any of a large number of optional parameters and returns an array of booking objects that match.

Request:

```
<BookingQuery>
<Authority>
<DetailLevel>
<DataSource>
<QueryParams>
<BookingId>(optional)
<BookingItemId>(optional)
<HotelStayDate>(optional)
<HotelStayDateEndRange>(optional)
<BookingCreatedDate>(optional)
<BookingCreatedDateEndRange>(optional)
<BookingModifiedDate>(optional)
<BookingModifiedDateEndRange>(optional)
<CustomerName>(optional)
<GuestName>(optional)
<HotelName>or<HotelId>(both optional)
<RegionName>or<RegionId>(both optional)
<Supplier>(optional)
<SupplierBookingRef>(optional)
</QueryParams>
</BookingQuery>
```

`<Authority>`	Complex	Authentication information.
`<DetailLevel>`	Enum	Two-valued text enumeration selecting the level of detail to return summary – return all IDs, the arrival date and number of nights stayed and the first guest name full – return all booking information
`<QueryParams>`	Complex	Contains many optional elements which form the query filter parameters
`<BookingId>` (optional)	Integer	If the booking Id is matched then just the matching booking is returned regardless of other parameters.
`<BookingItemId>` (optional)	Integer	If the booking item Id is matched then just the matching booking is returned regardless of other parameters.
`<HotelStayDate>` (optional)	Date	If present, return only bookings that contain a hotel stay which includes this night. If <HotelStayDateEndRange> also present, instead return only bookings which intersect with the date range specified.

`<HotelStayDateEndRange>` (optional)	Date	Marks the end of a booking hotel stay date range. If <HotelStayDate> is also present, return only bookings that contain a hotel stay which intersects with the date range specified by these two tags. Otherwise, this value is ignored
`<BookingCreatedDate>` (optional)	Date	If present, return only bookings that were created on the given date, ignoring the time component. If <BookingCreatedDate EndRange> is also present, instead return only bookings that were created in the date range specified
`<BookingCreatedDate EndRange>` (optional)	Date	Marks the end of a booking creation date range. If <BookingCreatedDate> is also present, return only bookings that were created in the date range specified. Otherwise, this value is ignored
`<BookingModifiedDate>` (optional)	Date	If present, return only bookings that were modified on the given date, ignoring the time component. If <BookingModifiedDateEndRange> is also present, instead return only bookings that were modified in the date range specified
`<BookingModifiedDateEndRange>` (optional)	Date	Marks the end of a booking modified date range. If <BookingModifiedDate> is also present, return only bookings that were modified in the date range specified. Otherwise, this value is ignored
`<GuestName>` (optional)	String	This is a guest name fragment. If present, return only bookings with a guest name that contains this string
`<HotelName>` (optional)	String	This is a hotel name fragment. If present, return only bookings with a hotel name that contains this string. If <HotelId> is specified, this field is ignored
`<HotelId>` (optional)	Integer	This is a numeric hotel Id. If specified, return only bookings which contain a stay at this hotel. This overrides any <HotelName>element specified.
`<RegionName>`	String	This is a region name fragment. If present,

(optional)		return only bookings in a region (or a subordinate of a region) whose name that contains this string. If <RegionId> is specified, this field is ignored
<RegionId> (optional)	Integer	This is a numeric region Id. If specified, return only bookings that contain a stay at a hotel in this region or one of its subordinate regions This overrides any <RegionName> element specified.

Result:

```
<BookingQueryResult>
<Booking>                   (0 or more)
</BookingQueryResult>
```

This is an array of zero or more booking structures:

<Booking>	Complex	A booking structure, which matches the specified filter parameters. This is returned to the level of detail specified in the <DetailLevel> element in the query.

3.5.3. Booking Cancel

BookingCancel accepts a single booking ID. In the same way as BookingCreate, it operates in two modes:

• prepare – dry-run; return full details of the selected booking and calculate the cancellation charge due, but does not cancel the booking

• commit – perform the booking cancellation

Request:

```
<BookingCancel>
<Authority>
<BookingId>
<CommitLevel>
<DetailLevel>(optional)
</BookingCancel>
```

<Authority>	Complex	Authentication information
<BookingId>	Integer	Numeric ID for the booking to cancel
<CommitLevel>	Enum	Two-valued text enumeration:

		prepare – return details of the selected booking and calculate the cancellation charge due, but does not cancel the booking confirm – go ahead and cancel the booking
`<DetailLevel>` (optional)	Enum	Two-valued text enumeration overriding the level of detail to return summary – return all IDs, the arrival date and number of nights stayed and the first guest name; this is the default for <CommitLevel> confirm full – return all booking information; this is the default for <CommitLevel> prepare

Result:

```
<BookingCancelResult>
<CommitLevel>
<Booking>
</BookingCancelResult>
```

`<CommitLevel>`	Enum	Indicates the commit level with which the cancellation was performed
`<Booking>`	Complex	A booking structure, containing details of the booking to cancel. This may be either a short summary or full details depending upon the query mode and result detail override. The booking details, and in particular the booking status fields, will appear the same regardless of the specified CommitLevel. Therefore if the CommitLevel was set to prepare then the booking will appear as if cancelled, although it will not have been modified in reality.

You can generally cancel a booking any time before arrival, but there may be a cancellation fee to pay.

E.g. to perform a dry-run cancellation of roomsXML booking Id 3000:

```
<BookingCancel>
<Authority>
<Org>[Your login organisation]</Org>
```

```
<User>[Your login username]</User>
<Password>[Your password]</Password>
<Currency>[Your currency]</Currency>
<Version>[Required version]</Version>
</Authority>
<BookingId>3000</BookingId>
<CommitLevel>prepare</CommitLevel>
</BookingCancel>
```

Note: The commit level ‘prepare’ – this ensures the booking is not actually cancelled yet.

If there are currently cancellation fees applicable, then the response from this will look like (with some detail omitted for clarity):

```
<BookingCancelResult>
<CommitLevel>prepare</CommitLevel>
<Booking>
<Id>3000</Id>
…
<HotelBooking>
…
<Room>
…
<Status>confirmed</Status>
<CanxFees>
…
</CanxFees>
</Room>
</HotelBooking>
<Charge>
…
<TotalSellingPrice>
<Currency>GBP</Currency>
<Amount>100</Amount>
<Estimated>false</Estimated>
<Converted>false</Converted>
</TotalSellingPrice>
</Booking>
</BookingCancelResult>
```

The "Status: confirmed" indicates that the booking hasn't yet been cancelled, whilst the CanxFees Tag restates the policy governing cancellations and the "Charge tag" explicitly indicates that cancelling the booking at this point would incur a £100 charge.

If there are no charges then the status will still be "confirmed" and the Cancellation Policy will still be stated, but no Charge tag will be present.

To accept any appropriate charges and proceed to cancel the booking, you should then send exactly the same request, but with a CommitLevel of ‘confirm’:

```
<BookingCancel>
<Authority>
<Org>[Your login organisation]</Org>
<User>[Your login username]</User>
<Password>[Your password]</Password>
<Currency>[Your currency]</Currency>
```

```
<Version>[Required version]</Version>
</Authority>
<BookingId>3000</BookingId>
<CommitLevel>confirm</CommitLevel>
</BookingCancel>
```

Note: You are not required to send a ‘prepare’ message before the ‘confirm’ message.

3.6. Money

All money values are returned as a structure specifying the currency as a text enumeration and the amount as a decimal. The text enumeration for the currency is the ISO-4217 / SWIFT code

```
<Price curr=”USD” amt=”99.50” />
```

represents $99.50 exactly.

3.7. Dates and times

3.7.1. Dates

Dates that do not include time information should be formatted as “yyyy-mm-dd”.

E.g. 24th April 2014 would be represented as “2014-04-24”

3.7.2. Dates with local times

Dates that contain local time information should be formatted as “yyyy-mm-ddTHH:mm:ss.fffffff”.

E.g. 3 PM on 24th April 2014 would be represented as “2014-04-24T15:00:00.0000000”

3.7.3. Dates with global times

Dates that contain global time information should be formatted as “yyyy-mm-ddTHH:mm:ss.fffffffzzz

E.g. 3 PM on 24th April 2014 in France would be represented as “2014-04-24T15:00:00.0000000+01:00

3.8. Data enumeration strings

There are places in the interface where it is necessary to pass across values from a fixed set (such as basic room type: single, double, twin, etc.) Wherever possible these are passed across the XML interface as human-readable text. However, these are likely not suitable for presentation to users.

3.9. Deployment-independent enumerations

3.9.1. Item status

confirmed	The booking is confirmed

onrequest	The booking has been requested, but the hotel has yet to confirm availability
cancelled	The booking is cancelled
Quoted	This is a quote only. Booking has not yet been attempted

3.9.2. Booking status

confirmed, cancelled, quoted as above, plus:

allonrequest	All items on the booking are on request
someonrequest	The booking contains both confirmed and on request items

3.9.3. Availability confirmation

allocation	The quote can be booked immediately from allocation
Request	The quote cannot be confirmed immediately, but will be put into on request status until the hotel can confirm availability

3.9.4. Availability status

allocation	Only return quotes which can be booked from allocation
Request	Only return quotes which can be booked on request
Any	Return all quotes

3.9.5. Commit level

Prepare	Simulates the operation, without making any permanent changes to the booking
Commit	Perform the operation and permanently commit the result

3.9.6. Detail level

Basic	Returns minimal detail – e.g. prices only
Summary	Returns a minimum amount of additional data suitable for e.g. a list of search results – a brief description, single photo etc.
Full	Returns all available information

Custom	Return the information specified in the CustomDetailLevel parameter

4. Appendix – error codes

The following error codes may currently be returned by the system. This list may change in future though, and should not be relied upon.

Where possible, the system will return helpful human-readable error messages in addition to the error codes below to indicate the nature of the problem.

Error Number	**Explanation**
1001	The operation completed but there was an error generating the response. Resubmitting a request in this case may result in a duplicate booking
1002	The operation completed with some non-fatal errors. Please contact roomsXML to confirm the status of the booking
2001	The request was in an invalid format
2002	The request was missing a required parameter / element
2003	The request contained a parameter / element with an illegal value
2004	The operation requested is not available to this user type
3001	The feature requested is not supported on this version
4001	General system error
4002	General user error. The error message should give sufficient details of the problem to enable it to be corrected
4003	Could not access requested booking. You may not have permission to view this booking Id
4004	Operation failed
4005	Operation previously performed. Most likely you have attempted to submit the same request twice
4006	The details you have provided when attempting to book do not match those on the quote. For example – the pax counts differ. The booking cannot be made in this situation
4007	This means roomsXML has disabled the XML interface, and generally indicates the system is undergoing maintenance or is temporarily offline. Please check with roomsXML for further information
4008	Timeout: the request exceeded the time-limit for generating a response. If the request was a search, you might like to retry it with

	more restrictive parameters.
5001	The payment type is unrecognised or not supported on this version
6002	You attempted to make a booking with duplicate extras
6004	The extra specified for the booking is invalid
6006	The upgrade specified for the booking is invalid
6007	The price for the upgrade could not be calculated
7001	The price of a quote changed when booking.
7002	Could not access requested quote. Either you do not have permission, or the quote ID was incorrect.
7003	The specified quote is no longer available.

5. Appendix – explicit omissions

The following features are explicitly not supported in the current version.

• The MaxResultsPerHotel, SortOrder, and MaxSearchTime parameters on the AvailabiltySearch request are not supported. If specified they are ignored.