HotelsCombined Web Services API

Contents:

Introduction

The HotelsCombined Web Services API ("The API") is an XML-based system that delivers Live Rates and Availability data directly between web servers.

The API allows Affiliates to build customised results pages on their own servers, back-fill their own hotel inventory with HotelsCombined data, or integrate hotel price comparison information directly into other systems (e.g. iPhone applications or map-based widgets).

The API is only enabled for approved Affiliates as per the application procedure at the following URL:

http://affiliates.hotelscombined.com/Tools/WebServices.aspx

Note that Affiliates who simply wish to customise the appearance of the City and Hotel Results pages should use the Private Branding feature instead.

What the API Does

The API returns Live Rates and Availability data, based on one set of search criteria at a time.

When the API is queried, the same search is sent to all 30+ supplier websites. The results are then standardised, matched together, filtered, sorted, grouped into pages and returned through the API.

The returned data includes the room description, total price, tax component, currency, supplier code and a unique key allowing construction of a deep link to the booking page on the supplier's website (a "Supplier Link"). This key expires after a short internal cache period, so it cannot be stored and reused later.

> *Under the standard commission system, the Affiliate generates a Lead and earns commission when an end-user clicks on a Supplier Link. The API is the only feature that allows Affiliates to place Supplier Links on webpages that aren't hosted on the HotelsCombined servers[1].*

The API only allows queries for a single hotel or a whole city[2] at a time. Every search must specify one of these two parameters. (There is currently no option to search multiple cities at once, and there is no option to search for a particular set of map coordinates either.)

The API also allows synchronous and asynchronous queries. This means Affiliates can submit a query and wait for the full set of results, or they can receive incremental results while the query is running, so they can display partial results while the end-user is waiting.

Essentially, the API just allows Affiliates to search all 30+ suppliers' websites at once, so they can display pricing data and Supplier Links on their own webpages using their own linking strategies.

---

[1] The Private Branding feature allows Supplier Links to appear on third-party domain names (Custom URLs), but the actual webpages are still hosted on HotelsCombined web servers.

[2] A "city" may refer to an island, a town, a suburb, a small region, or even a whole metropolitan area; based on what hotels are expected to appear for each particular search term.

What the API Doesn't Do

There are many common misconceptions about what can be achieved with the API. Each of the main ideas is discussed below:

The API does NOT return any static information about hotels or geography.

The API only returns Live Rates and Availability data. All static hotel and geography information is available through the Data Feeds instead. Affiliates must import the Data Feeds into their database and access the information from there.

The API does NOT provide any SEO value at all.

Any webpage that queries the API is not permitted to be indexed by search engines. This is because search engine spiders will generate invalid searches and invalid Leads, which will adversely impact Lead values. Only the Data Feeds provide SEO-friendly content for building webpages.

The API does NOT allow automated searches.

Affiliates are only permitted to query the API when a real Internet user initiates a real search for a single set of search criteria. This means Affiliates are not permitted to generate automated queries or use default search criteria without prior written consent from HotelsCombined.

The API does NOT return information about "special deals".

The API only queries suppliers' websites for one set of search criteria at a time. There is no provision for querying special deals, bonus offers or other promotional hotel rates.

The API results can NOT be stored and reused later.

Every API query is unique, based on the exact search criteria and the specific metrics of the end-user. Affiliates are not permitted to pre-load, cache or otherwise reuse API results to build Supplier Links at another time or for another end-user.

The API can NOT be used for the sole purpose of gathering hotel pricing data.

The API can only be queried when a real Internet user wants to book a hotel room, and the results must be displayed as clickable Supplier Links. Affiliates are not permitted to query the API for the sole purpose of collecting hotel pricing information. Previous results may be kept and displayed in summary form as "minimum rates", but this information is already available in the Data Feeds.

Data Feeds and Private Branding

Affiliates must regularly download all Data Feeds and import them into a local database.

API requests require cities and hotels to be entered as IDs. These proprietary IDs are only available through the Data Feeds. The Data Feeds also include static hotel and geography content, which Affiliates require to build information-rich websites.

Affiliates cannot use the API without first downloading and integrating all Data Feeds.

The Data Feeds are available in the Affiliate Control Panel at the following URL:

http://affiliates.hotelscombined.com/Tools/DataFeedsNew.aspx

Affiliates who use the API need to download and reimport a fresh copy of the Data Feeds around once every 2-4 weeks. This includes adding new data, updating changed data and deleting old data. This data needs to be kept "fresh", to maintain user confidence and to continue displaying the most recent and relevant information to website visitors.

Note that Affiliates who apply for the API can begin integrating the Data Feeds while their application is still being processed.

Affiliates must also create and maintain at least one Private Branding template with Custom URL.

A Custom URL is a domain or subdomain that uses a CNAME Record (a DNS setting) to mask the www.HotelsCombined.com Trademark domain name. The Custom URL also allows tracking to be more professional, as it removes the need for using the Affiliate ID in Supplier Links.

The Private Branding template itself can remain completely blank, because it will only be used for tracking purposes. The only important settings are the template name (used on the Splash Page) and the Custom URL (used for tracking).

Affiliates who require better tracking are permitted to create multiple Private Branding templates. Each template can have its own Custom URL for building Supplier Links, and each will have its own Brand ID for grouping traffic in performance reports.

Overall, the API should not be considered a standalone product. Rather, it relies on and supports the other features offered by the Affiliate Program.

Development and Testing

The API comes with a development and testing environment called "The Sandbox".

The Sandbox development environment is only a simulation of the Live API. Rather than polling all 30+ suppliers with each API request, the Sandbox simply returns fake (dummy) data. This allows developers to generate unlimited trial requests when they are building a website, and simply switch over to the Live API once the website is ready for release.

Affiliates must use the Sandbox for all development and testing.

Accessing the Sandbox is exactly the same as accessing the Live API except for two small differences:

1. Queries are sent to the following URL (note the subdomain):

   http://sandbox.hotelscombined.com/API/Search.svc/pox/

2. The Sandbox uses its own API Key (a Live API Key will not work):

   F7538EB2-2B63-4AE1-8C43-2FAD2D83EACB

When a website uses these settings, all test searches will no longer affect the Affiliate's account. Supplier Links will not function properly, but this ensures invalid Leads aren't generated either.

Note that the Sandbox development environment is available to all Affiliates. There is no need to apply for the API to be able to use the Sandbox environment. This means eager Affiliates can start developing their websites immediately. It also means that Affiliates who aren't sure about the API can build a "proof of concept" before submitting their application.

Using the API

The API can be accessed using HTTP-GET (POX) or HTTP-POST (SOAP), with the responses always being returned as complete XML. Affiliates can choose which way to access the API, based on the programming environment of their web server.

Accessing through HTTP-GET (POX):

Accessing the API through HTTP-GET is the recommended option, as it is so simple to integrate and later troubleshoot. It is also the most suitable option for PHP programmers, as it requires the least amount of third-party library code.

Essentially, the HTTP-GET option allows Affiliates to construct a URL that represents the user's query. This URL can then be called by the web server, for example using PHP's "cURL" or "file" functions. Affiliates can even test the query by browsing to the URL using their normal Internet browser.

The URL for the API interface is:

http://www.hotelscombined.com/API/Search.svc/pox/

Building a query involves selecting an API method and constructing a parameter list from the user's search criteria, like this:

http://www.hotelscombined.com/API/Search.svc/pox/[Method Name]?[parameter 1]=[value1]&[parameter 2]=[value2]…[parameterN]=[valueN]

For example, the following query describes a "City" search for a one-night stay in Sydney, Australia, with check-in on the 1st of January 2012, and 2 guests in 1 room:

http://www.hotelscombined.com/Api/Search.svc/pox/CitySearch?CityID=1948&Checkin=2012-01-01&Checkout=2010-01-02&Guests=2&Rooms=1

Note that this example query won't actually return results without a valid API Key and the complete set of user data. Also note that the Data Feeds are required to find out that "1948" is the City ID for Sydney, Australia.

Accessing through HTTP-POST (SOAP):

Accessing the API through HTTP-POST using the SOAP protocol is the easiest option for high-level programming languages such as ASPX.NET, where support for SOAP is built-in. Using SOAP can make troubleshooting more difficult, so it is only recommended for advanced developers.

All the information about the API's SOAP interface is available in the following WSDL document:

http://www.hotelscombined.com/api/wsdl/SearchSoap.wsdl

Building a query involves constructing a complete XML document from the user's search criteria, and then submitting it via HTTP-POST to the API. No example is given here, as the WSDL document should provide all the required information.

API Methods

The API offers different methods for requesting Live Rates and Availability data. Each API method returns different results, and is designed for building a certain type of webpage.

Behind these methods is the concept of a "user search". A user search is a single set of search criteria (at the city level) for a single end-user within the duration of the API's internal cache period (approximately 2 hours). Submitting a duplicate API request with different sort, filter or pagination parameters does not initiate a new user search, but changing the city, dates, adults, rooms or any of the user parameters does.

Only the first user search polls all 30+ suppliers, so subsequent API requests for the same user search return results almost instantly. When the Affiliate Agreement discusses the "Search-to-Lead" ratio requirement for API Affiliates, it only refers to user searches and not individual API requests.

HotelSearch

This is the standard method for requesting Live Rates and Availability data for a single hotel only. This method also operates on the idea of "send the query and wait for the results to be returned".

The HotelSearch method is recommended for webpages that only display results for a single hotel. If more than one hotel's results are to be displayed on a single page, the CitySearch method should be used instead.

The concept of a "user search" operates at the city level. This means the HotelSearch method will benefit from the system's internal cache, even when querying a different hotel in the same city. Note that if the CitySearch method is not run first, the initial HotelSearch request for a particular city will take just as long to run, so a "progress bar" or "please wait" page will still be recommended.

The HotelSearch method accepts the "HotelSearch" parameters listed later in this document, and returns results in the "HotelSearch" format as described later in this document.

CitySearch (Standard Method)

This is the standard method for requesting Live Rates and Availability data for all the hotels in a single city at once. It is possible to build an entire website using just this single API method.

The CitySearch method is recommended for webpages that display a list of hotel options to the user. For example, a user may search for a particular city and want to see all hotels that are available and fall within their particular filter criteria (price range, star ratings, etc.).

Using this method is a simple process of sending the query and waiting for the results. Each unique user search will take several seconds to run, so an animated GIF "progress bar" or "please wait" page is recommended to keep users waiting.

The CitySearch method accepts the "CitySearch" parameters listed later in this document, and returns results in the "CitySearch" format as described later in this document.

CitySearch (Asynchronous Methods)

The asynchronous CitySearch methods allow Affiliates to build more interactive "search progress" webpages, by returning partial results before the complete data set is available.

These methods operate on the idea that the search is initiated with an API request, and then whatever results are available at the time are returned by submitting another API request. Therefore, a single search will consist of several API requests.

AsyncCitySearch

This method initiates an asynchronous search for all the hotels in a particular city. It accepts the same parameters as the CitySearch method, and returns a "SearchKey" string for use with the other asynchronous methods.

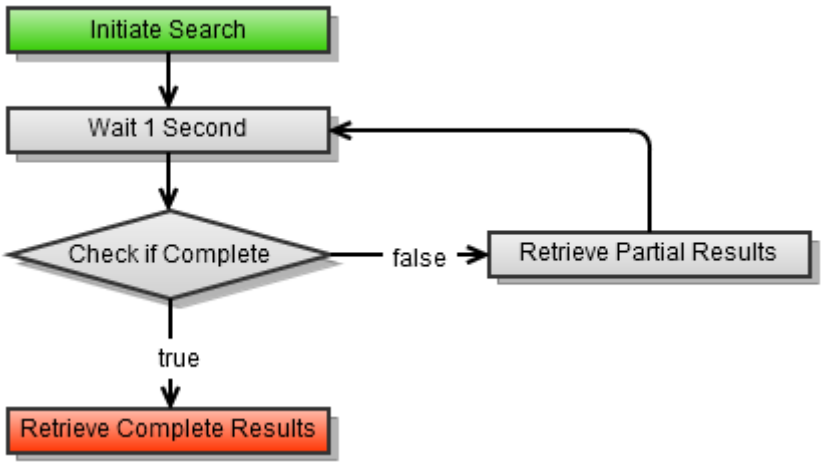
AsyncCitySearchComplete

This method checks whether the asynchronous search is complete yet. It accepts the SearchKey as its only parameter, and returns a Boolean (true/false) result.

AsyncCityResults

This method retrieves whatever results are available at the time. It accepts the SearchKey as its only parameter, and returns the same result format as the CitySearch method. Note that the results returned by the AsyncCityResults method can only be assumed to be complete if a previous request to the AsyncCitySearchComplete method returns "true".

The correct way of implementing these asynchronous methods is by using the following procedure:



1. Initiate the search with the AsyncCitySearch method.
2. Pause for approximately 1 second.
3. Check the results with the AsyncCitySearchComplete method.
4. If the result is "false", use AsyncCityResults to retrieve partial results. Return to step 2.
5. If the result is "true", use the AsyncCityResults method to retrieve complete results.

Parameters

Each API method has its own set of required and optional parameters which are described below.

Accessing the API through HTTP-GET (POX) requires these parameters to be grouped as "key=value" pairs and appended to the API URL and method name to form a complete URL. As per the example earlier in this document, the following query describes a "City" search for a one-night stay in Sydney, Australia, with check-in on the 1st of January 2012, and 2 guests in 1 room (parameters highlighted):

http://www.hotelscombined.com/Api/Search.svc/pox/CitySearch?CityID=1948&Checkin=2012-01-01&Checkout=2010-01-02&Guests=2&Rooms=1

Accessing the API through HTTP-POST (SOAP) requires these parameters to be built into a complete XML document as per the specifications of the SOAP protocol. No example is given here, because the SOAP protocol should already take care of the exact syntax.

The parameters are grouped into sections: Required (per-Affiliate, per-User and per-Search) and Optional (locality-based, filter fields, sort fields, pagination and search options). The actual order of the parameters in the API request is not important.

The standard result format is an XML document of objects. Where an object type can appear multiple times in a result set, it is described separately in this document.

HotelSearch Parameters

These are the required and optional parameters for the HotelSearch method:

Required Affiliate-Specific Parameter:

ApiKey: (String) HotelsCombined will assign this Key to Affiliate accounts that successfully complete the API application process. The Sandbox development environment has its own API Key; which is completely unrelated to the Live API key, and can be used in the Sandbox by any Affiliate.

Required User-Specific Parameters:

UserID: (String) The Affiliate must identify each end-user with a unique string, to aid the function of the internal cache. The web server's session ID (or a hash of it) is recommended for this parameter, but Affiliates can choose any unique string that persists throughout each user's entire stay on the website.

UserIPAddress: (String) This must be the externally-accessible IP Address of the end-user. This field is used by security systems and to provide country-specific hotel results, so it is prohibited to use local (internal) or dummy IP Addresses in this field.

UserAgent: (String) This must be the complete User Agent string that the user's browser passes to the web server through normal HTTP protocols. This value is required by the internal cache and various security systems, so dummy data is prohibited.

Required Search-Specific Parameters:

HotelID: (Integer) The hotel to be searched must be entered as an ID. The complete list of IDs for each hotel is provided in the Standard and Advanced Data Feeds.

Checkin: (String) The check-in date must be entered in "yyyy-MM-dd" date format.

Checkout: (String) The check-out date must also be entered in "yyyy-MM-dd" date format.

Guests: (Integer) The number of adults + children must be entered as a single number. Future revisions of the API may allow adults and children to be entered separately, but this current version only allows a single value for "guests".

Rooms: (Integer) The number hotel rooms required.

Optional Locality-Based Parameters:

LanguageCode: (String) The two-letter language code of the result set. This mainly affects the room name and description. Language codes are provided in the Appendix. Where a language is not fully supported, or this parameter is not included, results will display in English by default.

DisplayCurrency: (String) The three-letter currency code of the result set. Currency codes are provided in the Appendix. The results will always use the selected currency, even when the supplier doesn't fully support it, so this must be explained to the end-user on the intermediary "splash" page. When this parameter is not provided, the API will determine the most appropriate currency, based on geo-targetting the end-user's IP address.

Optional Search Parameter:

TimeOutInSeconds: (Integer) This parameter specifies how many seconds to wait for each supplier to start returning results. Suppliers that start returning results within this timeout period may take longer to finish sending their results. Any supplier that doesn't start sending results within this timeout period is ignored.

CitySearch Parameters

These are the required and optional parameters for the CitySearch and AsyncCitySearch methods. Most are common to the HotelSearch parameters, but with several important additions for filtering, sorting and paginating the results:

Required Affiliate-Specific Parameter:

ApiKey: (String) HotelsCombined will assign this Key to Affiliate accounts that successfully complete the API application process. The Sandbox development environment has its own API Key; which is completely unrelated to the Live API key, and can be used in the Sandbox by any Affiliate.

Required User-Specific Parameters:

UserID: (String) The Affiliate must identify each end-user with a unique string, to aid the function of the internal cache. The web server's session ID (or a hash of it) is recommended for this parameter, but Affiliates can choose any unique string that persists throughout each user's entire stay on the website.

UserIPAddress: (String) This must be the externally-accessible IP Address of the end-user. This field is used by security systems and to provide country-specific hotel results, so it is prohibited to use local (internal) or dummy IP Addresses in this field.

UserAgent: (String) This must be the complete User Agent string that the user's browser passes to the web server through normal HTTP protocols. This value is required by the internal cache and various security systems, so dummy data is prohibited.

Required Search-Specific Parameters:

CityID: (Integer) The city to be searched must be entered as an ID. The complete list of IDs for each city is provided in the Standard, Advanced and Geography Data Feeds.

Checkin: (String) The check-in date must be entered in "yyyy-MM-dd" date format.

Checkout: (String) The check-out date must also be entered in "yyyy-MM-dd" date format.

Guests: (Integer) The number of adults + children must be entered as a single number. Future revisions of the API may allow adults and children to be entered separately, but this current version only allows a single value for "guests".

Rooms: (Integer) The number hotel rooms required.

Optional Filter Parameters:

AvailableOnly: (Boolean) When this parameter is false, all hotels in the city are returned, regardless of their availability. When this parameter is true, only available hotels are returned. The default state of this parameter is false.

MinPrice: (Integer) Hotels are only returned when their lowest rate is above this value. This filter only uses each hotel's lowest rate, and ignores rooms or suppliers with higher rates. This filter uses the total cost of the stay in the display currency.

MaxPrice: (Integer) Hotels are only returned when their lowest rate is below this value. This filter only uses each hotel's lowest rate, so higher rates for included hotels are still returned. This filter uses the total cost of the stay in the display currency.

StarRatingFilter: (Pipe-Separated Integers) When this parameter is used, only hotels with a star rating in this list are displayed in the results. An example of this parameter is "StarRatingFilter=3|4|5", which only displays 3, 4 and 5 star hotels. Note that some hotels don't use star ratings, or use half-ratings (e.g. 3.5 stars).

PropertyTypeFilter: (Pipe-Separated Integers) When this parameter is used, only hotels that belong to one of the given property types are displayed in the results. The full list of property types is available in the Appendix. An example of this parameter is "PropertyTypeFilter=0|1", which only displays hotels (ID=0) and Motels (ID=2). Note that many properties are simply listed as "hotels", even when another property type may be more suitable.

FacilityFilter: (Pipe-Separated Integers) When this parameter is used, only hotels with all the listed facilities are displayed in the results. The full list of facilities is available as a Data Feed. An example of this parameter is "FacilityFilter=6|7", which only displays hotels with parking (ID=6) and a swimming pool (ID=7). Note that most hotels rarely provide a complete set of facilities, and rarely update their data when their facilities change, so this filter may often return incorrect results.

ChainIDFilter: (Pipe-Separated Integers) When this parameter is used, only hotels that belong to these hotel chains are displayed in the results. The full list of hotel chains is available as a Data Feed. An example of this parameter is "ChainIDFilter=2|10", which only displays hotels that belong to Best Western (ID=2) or Accor (ID=10).

HotelIDFilter: (Pipe-Separated Integers) When this parameter is used, only those hotels that are not in this list are displayed in the results. The complete list of hotel IDs is provided in the Data Feeds. An example is "HotelIDFilter=1064899|1479839", which excludes the Sydney Hilton (ID=1064899) and the Sydney Travelodge (ID=1479839) from the results list.

HotelNameContains: (String) Only returns hotels that include this string in their name. This parameter helps users find a specific hotel or hotel chain.

LocationID: (Integer) Provides a reference point for filtering or sorting hotels by distance. When this is a positive integer, it refers to a City Location from the Data Feeds. When this is a negative integer, it refers to a City ID, and uses the centre of town. This parameter overrides the Latitude and Longitude parameters.

Latitude: (Decimal) Defines the latitude component of the reference point when the LocationID parameter is excluded. It must be used with the Longitude parameter.

Longitude: (Decimal) Defines the longitude component of the reference point when the LocationID parameter is excluded. It must be used with the Latitude parameter.

Distance: (Integer) Defines the maximum distance a hotel can be from the reference point to be returned in the result set. This parameter uses miles[3], and the reference point is defined by the LocationID parameter, or the Latitude and Longitude parameters.

Optional Sorting Parameters:

SortField: (String) This parameter selects which field to use to sort the result set. Available options are "Popularity", "Distance", "Rating" (star rating), "ConsumerRating", "MinRate" and "Name" (hotel name). The default sort field is "Popularity".

[3] Previous versions of the API used kilometres as the unit of measurement for the Distance parameter.

SortAscending: (Boolean) When this parameter is true, sorting is ascending. When this parameter is false, sorting is descending. The default state of this parameter is true.

Optional Pagination Parameters:

PageIndex: (Integer) This parameter defines which page number is returned. This count is zero-indexed, so a value of zero (0) returns the first page of results, and a value of one (1) returns the second page of results. When this field is excluded, the first page of results is returned by default.

PageSize: (Integer) This parameter defines how many results to display per page. This parameter defaults to 20 when the PageIndex parameter is also used, but it defaults to showing all results when neither parameter is set.

Optional Search Parameters:

TimeOutInSeconds: (Integer) This parameter specifies how many seconds to wait for each supplier to start returning results. Suppliers that start returning results within this timeout period may take longer to finish sending their results. Any supplier that doesn't start sending results within this timeout period is ignored.

Detail: (String) This parameter specifies how much data is returned for each hotel. There are only 3 available options:

"None" - Only returns the cheapest rate from each supplier for each hotel. This is the default value for this parameter.
"SingleCheapest" - Returns the cheapest rate from each supplier for each hotel, including the room type (name).
"All" - Returns all the rates from all the suppliers for all hotels, including room types and room descriptions (where available).

SearchKey Parameter

This is the only required parameter for the AsyncCitySearchComplete and AsyncCitySearch methods. The API Key is not a required parameter, because this is already passed into the API when the AsyncCitySearch method is queried.

SearchKey: (Integer) This is the unique key that is returned by the AsyncCitySearch method. It is the only required parameter.

Result Formats

All API results are returned in XML format, but each method's results adhere to a different format, which are described below.

HotelSearch Result Format

This is the standard XML format for HotelSearch results, with fields described below:

HotelSearchResponse

The HotelSearchResponse object is a wrapper for all results returned from the HotelSearch method. This is the structure of the HotelSearchResponse object:

```
<HotelSearchResponse>
	<Currency></Currency>
	<Rates></Rates>
	<Statistics></Statistics>
</HotelSearchResponse>
```

These are the fields:

Currency: (String) The three-letter currency code of the result set. When the API request doesn't specify a currency, this field shows the currency that the system determines based on the end-user's global location (i.e. the default currency of that country). Currency codes are provided in the Appendix.

Rates: (Rates List) This is a list of "***Rate***" objects, as described below.

Statistics: (Statistics List) This is a list of "***Statistic***" objects, as described below.

Rate

A "Rate" is a single room type for a single provider for the specified set of check-in / check-out dates. This is the structure of an individual Rate object:

```
<Rate>
	<Currency></Currency>
	<Price></Price>
	<Taxes></Taxes>
	<ConvertedPrice></ConvertedPrice>
	<ConvertedTaxes></ConvertedTaxes>
	<Name></Name>
	<Description></Description>
	<Provider></Provider>
	<Key></Key>
</Rate>
```

Note that the order of these fields may change at any time, but the included data fields will not.

These are the fields:

Currency: (String) The three-letter currency code of the "Price" and "Taxes" fields. This is the currency that will be displayed on the supplier's website when the end-user clicks through to the booking page. This will be different to the currency selected by the API request when the selected currency isn't supported by the supplier, or when a subsequent request for the same "User Search" requests a different currency.

Price: (Decimal) The price of the "Hotel Booking" component of the Rate. To determine the final price the end-user needs to pay for the hotel room, this value must be added to the "Taxes and Fees" component (described below).

Taxes: (Decimal) The total "Taxes and Fees" component of the Rate. This value is given separately to the "Hotel Booking" component, because some suppliers try to hide this value, and only show customers the "Hotel Booking" component on the reservation form page. Adding the two values together is therefore important to provide a proper price comparison between different suppliers.

ConvertedPrice: (Decimal) This is the price of the "Hotel Booking" component of the Rate, converted into the default currency of the result set (the "Currency" field in the "HotelSearchResponse" object).

ConvertedTaxes: (Decimal) The value of the "Taxes and Fees " component of the Rate, converted into the default currency of the result set (the "Currency" field in the "HotelSearchResponse" object).

Name: (String) This is the name of the room type, as per the supplier's website. Note that each supplier may have different names for the same room types, and each hotel may have different names for similar classes of rooms. There is no standardisation between suppliers or hotels on values in this field. Examples are "Double Room", "Deluxe Ocean View Room", etc.

Description: (String) Additional information about the Rate. This field will often mention whether breakfast is included, and may also mention any special deal that affects the price of the room for this stay. Examples are "Early Bird Bonus - 15% Off", "Buffet Breakfast included", etc.

Provider: (String) The three-letter supplier code of the hotel provider offering this Rate. The complete list of supplier codes and names is available in the Appendix, or at http://www.hotelscombined.com/AboutUs/Providers.aspx

Key: (String) This key is used to construct the deep link into the supplier's booking page (a.k.a. the "Supplier Link"). This key is unique to this Rate and this "User Search", and cannot be cached or offered to other end-users. Instructions for using this Key to construct Supplier Links are provided later in this document.

Statistic

A "Statistic" is an individual piece of summary information that relates directly to the API Request. This is the structure of an individual Statistic object:

```
<Statistic>
	<Group></Group>
	<Name></Name>
	<Value></Value>
</Statistic>
```

These are the fields:

Group: (String) Individual Statistic objects are grouped into categories. Currently, only the "Diagnostics" group is used by the HotelSearch method.

Name: (String) This is the name of the Statistic. Currently, the HotelSearch method offers "TotalServerTimeInSeconds", "ProviderWaitingTimeInSeconds", and "LoginID" only. The "LoginID" is a unique value per API request that is used by the Support Team for troubleshooting purposes (the other values should be self-evident by their names).

Value: (Variable) The value of the Statistic. Depending on the "Name" of the Statistic, this may be a decimal or an integer number, but it may also be a string where required.

CitySearch Result Format

This is the standard XML format for CitySearch results (as returned by the CitySearch method and the AsyncCityResults method), with fields described below:

CitySearchResponse

The CitySearchResponse object is a wrapper for all results returned in the CitySearch Result Format. This is the structure of the CitySearchResponse object:

```
<CitySearchResponse>
	<Currency></Currency>
	<Hotels></Hotels>
	<Statistics></Statistics>
</CitySearchResponse>
```

These are the fields:

Currency: (String) The three-letter currency code of the result set. When the API request doesn't specify a currency, this field shows the currency that the system determines based on the end-user's global location (i.e. the default currency of that country). Currency codes are provided in the Appendix.

Hotels: (Hotels List) This is a list of "***Hotel***" objects, as described below.

Statistics: (Statistics List) This is a list of "***Statistic***" objects, as described below.

Hotel

A "Hotel" contains all Rates for a single hotel. This is the structure of an individual Hotel object:

```
<Hotel>
	<Distance></Distance>
	<ID></ID>
	<Rates></Rates>
</Hotel>
```

These are the fields:

Distance: (Decimal) When a "reference point" is entered in the API request (using the LocationID parameter, or both the Latitude and Longitude parameters), this field provides the distance in miles of the hotel from the reference point.

ID: (Integer) The ID of the hotel, as per the information in the Data Feeds.

Rates: (Rates List) This is a list of "***Rate***" objects, as described below.

Rate

A "Rate" is a single room type for a single provider for the specified set of check-in / check-out dates. This is the structure of an individual Rate object:

```
<Rate>
	<Currency></Currency>
	<Price></Price>
	<Taxes></Taxes>
	<ConvertedPrice></ConvertedPrice>
	<ConvertedTaxes></ConvertedTaxes>
	<Name></Name>
	<Description></Description>
	<Provider></Provider>
	<Key></Key>
</Rate>
```

Note that the order of these fields may change at any time, but the included data fields will not.

These are the fields:

Currency: (String) The three-letter currency code of the "Price" and "Taxes" fields. This is the currency that will be displayed on the supplier's website when the end-user clicks through to the booking page. This will be different to the currency selected by the API request when the selected currency isn't supported by the supplier, or when a subsequent request for the same "User Search" requests a different currency.

Price: (Decimal) The price of the "Hotel Booking" component of the Rate. To determine the final price the end-user needs to pay for the hotel room, this value must be added to the "Taxes and Fees" component (described below).

Taxes: (Decimal) The total "Taxes and Fees" component of the Rate. This value is given separately to the "Hotel Booking" component, because some suppliers try to hide this value, and only show customers the "Hotel Booking" component on the

reservation form page. Adding the two values together is therefore important to provide a proper price comparison between different suppliers.

ConvertedPrice: (Decimal) This is the price of the "Hotel Booking" component of the Rate, converted into the default currency of the result set (the "Currency" field in the "HotelSearchResponse" object).

ConvertedTaxes: (Decimal) The value of the "Taxes and Fees " component of the Rate, converted into the default currency of the result set (the "Currency" field in the "HotelSearchResponse" object).

Name: (String) This is the name of the room type, as per the supplier's website. Note that each supplier may have different names for the same room types, and each hotel may have different names for similar classes of rooms. There is no standardisation between suppliers or hotels on values in this field. Examples are "Double Room", "Deluxe Ocean View Room", etc.

Description: (String) Additional information about the Rate. This field will often mention whether breakfast is included, and may also mention any special deal that affects the price of the room for this stay. Examples are "Early Bird Bonus - 15% Off", "Buffet Breakfast included", etc.

Provider: (String) The three-letter supplier code of the hotel provider offering this Rate. The complete list of supplier codes and names is available in the Appendix, or at http://www.hotelscombined.com/AboutUs/Providers.aspx

Key: (String) This key is used to construct the deep link into the supplier's booking page (a.k.a. the "Supplier Link"). This key is unique to this Rate and this "User Search", and cannot be cached or offered to other end-users. Instructions for using this Key to construct Supplier Links are provided later in this document.

Statistic

A "Statistic" is an individual piece of summary information that relates directly to the API Request. This is the structure of an individual Statistic object:

```
<Statistic>
	<Group></Group>
	<Name></Name>
	<Value></Value>
</Statistic>
```

These are the fields:

Group: (String) Individual Statistic objects are grouped into categories. There are several groups included in the CitySearch Result Format, including "Counts", "Rates", "ChainPrices", "StarPrices", "PropertyTypes", "Availability" and "Diagnostics". The "Diagnostics" field is used for troubleshooting, while all other fields are useful for building page elements like the filter menu, or titles like "From $x for 5 Star Hotels".

Name: (String) This is the name of the Statistic. There are many different statistic names included in the CitySearch Result Format, and more may be added from time to time as required by API-enabled Affiliates. For details on specific statistic names, or to request a particular statistic to be added to this list, simply email the Support Team.

Value: (Variable) The value of the Statistic. Depending on the "Name" of the Statistic, this may be a decimal or an integer number, but it may also be a string where required.

SearchKey Result Format

The SearchKey is the only data returned by the AsyncCitySearch method. It is a long integer that is returned as a string in XML format. This is the structure of the XML document:

```
<string></string>
```

This is the data:

string: (Integer) This value provides an identifier for the result set. It is required by the AsyncCitySearchComplete and AsyncCityResults methods.

SearchComplete Result Format

The SearchComplete is the only data returned by the AsyncCitySearchComplete method. It is a Boolean value that is returned in XML format. This is the structure of the XML document:

```
<boolean></boolean>
```

This is the data:

boolean: (Boolean) This value is only ever "true" or "false". A value of "true" indicates that the API is ready to return complete results when the AsyncCityResults method is next requested. A value of "false" indicates that only partial results are currently available, and the user will have to wait longer for complete results.

Exceptions

All exceptions (errors) are returned in XML format too. The two different types of exceptions are described below.

Standard Exception

When there is an error with any API request, the Standard Exception Result Format is returned. This is the structure of the XML document:

```
<fault>
	<code>
		<value>Sender</value>
		<subcode>
			<value>a:FaultException</value>
		</subcode>
	</code>
	<reason>
		<text xml:lang="en-US"></text>
	</reason>
</fault>
```

In most cases, the only field that changes is the `<text>` field in the `<reason>` section.

This is the data:

text: (String) This is a short description of the error, which helps narrow down the problem to a particular part of the API query. Following are the common error messages:

"InvalidLogin" - The API Key is invalid. Only the Sandbox API Key will work on the Sandbox, and only the Affiliate's Live API Key will work on the Live API.

"HotelDoesNotExist" - The "HotelID" parameter does not refer to a valid hotel when running a hotel-based search.

"CityDoesNotExist" - The "CityID" parameter does not refer to a valid city when running a city-based search.

"InvalidSearchID" - The "SearchKey" parameter does not refer to a valid asynchronous search.

"InvalidParameter" - One of the parameters is invalid or uses the wrong format. This message may also indicate that a date field is missing.

"CompressionRequired" - The Affiliate's web server needs to be configured to allow compressed HTTP requests and responses. This is a server setting that is beyond the scope of this document to provide instructions for.

"ServerError" - There is an error with the server. The Development Team is automatically notified about such issues.

Note that invalid data provided in the correct format will often return an empty data set, rather than an actual error message. For example, when the check-in / check-out dates are in the past, or when the user-specific parameters don't refer to a real end-user.

End-User Ban

When an individual end-user initiates too many User Searches within any 24 hour period, they are banned from running any more searches for the next 24-48 hours. When this happens, only the Statistics list is returned in the result set. Also, the following "***Statistic***" object is added:

```
<Statistic>
      <Group>Abuse</Group>
      <Name>IPBanned</Name>
      <Value>true</Value>
</Statistic>
```

The user is identified by the 3 user-specific parameters in the API request; not just the IP Address. Affiliates who experience this ban in error should therefore check their user-specific parameters.

Affiliates who experience this ban during development or testing should be reminded to use the Sandbox development environment for all API requests during development or testing. Only during the very last stages of development should the website or application switch to the Live API.

Constructing Supplier Links

Supplier Links must be carefully constructed as per the instructions below, to allow proper tracking and to ensure adherence to the Affiliate Agreement.

Supplier Links are constructed using the "Key" values that are returned in the "Rate" objects of the HotelSearch and CitySearch result formats. Each Key is unique to a particular hotel room rate from a particular supplier for a particular User Search. This means that Key values cannot be cached, and they cannot be reused for other searches or other users.

A new API request must be run for each webpage that displays Supplier Links.

Supplier Links use the ProviderRedirect.aspx file on the HotelsCombined web servers to facilitate the redirect to the supplier's website. This file must be accessed using a **Private Branding Custom URL**. Using a HotelsCombined domain name instead constitutes a breach of the Affiliate Agreement, as it produces an effect similar to "cookie stuffing".

Affiliates must create a Private Branding template for this Custom URL. The template itself can remain completely blank, as long as the Custom URL is properly associated with the Brand ID. Affiliates may want to use multiple Custom URLs to aid in tracking, so each additional Custom URL will need its own Private Branding template.

The URL for the ProviderRedirect.aspx file is:

http://<PrivateBrandingCustomURL>/ProviderRedirect.aspx?<ParametersList>

The Parameters for this file are as follows:

Key: (String) The "Key" value that is returned in the "Rate" object of the HotelSearch and CitySearch result formats. This is a required parameter, as it specifies exactly which supplier booking page to redirect the end-user to.

Splash: (Boolean) This refers to the redirect page (or "Splash Page") that the end-user sees during the transition to the supplier's booking page. When this parameter is "true", the default Private Branding Splash Page is displayed for 10 seconds upon redirect. When this parameter is "false", no splash page is used, which allows the Affiliate to build their own. By default, this parameter is set to "true".

Label: (String) This is the freeform, 15-character tracking parameter that all Affiliate Links are permitted to use. This parameter is optional, but highly recommended to allow much more useful reporting on booking revenue performance over time.

Note that the Splash Page is important, because it advises end-users that the next page is on a completely different website, and points out when the display / payment currency will be different. Affiliates who decide to remove the Splash Page using the "Splash=false" parameter are advised to replace it with their own redirect page, so this important information can still be displayed to users.

When a user clicks on the completed Supplier Link, a "Lead" is generated and commission is earned. Testing or demonstrating this link will generate invalid Leads, so must be avoided where possible. Supplier Links built using the Sandbox are "safe", in that they won't redirect to suppliers' websites, and won't generate invalid Leads either.

Best Practices

This list of Best Practices offers guidelines for developing webpages and other systems with the API in a way that adheres to the Affiliate Agreement and also offers the best end-user experience.

Only one API query is required per page load.

The API methods were selected so that only one query is required for each webpage. A single query refers to a single set of search criteria, and may include many API method calls to retrieve the data (such as in the case of asynchronous methods). If a webpage requires more than one API query per page load, most likely the webpage is not properly structured or is using the wrong API methods.

Only one city is to be searched per page load.

Affiliates should not try to search "nearby cities" by running multiple User Searches per page load. This activity will directly impact the account's Search-to-Lead ratio by generating invalid searches, and may result in a breach of the Affiliate Agreement. Affiliates specialising in map-based interfaces should contact the Support Team for instructions on working around this issue.

Static data pages should be used when the user doesn't select check-in / check-out dates.

When the end-user doesn't select check-in / check-out dates, it is not permitted to use default dates to query the API. The best solution is to build "static" webpages using content from the Data Feeds. This content includes "Minimum Nightly Rates" for every available hotel, so that each webpage can still be built to display indicative prices to the end-user.

Supplier Links must use Custom URLs.

All Supplier Links (using the ProviderRedirect.aspx file) must use Private Branding Custom URLs to mask the HotelsCombined domain name. Failure to do so may result in a situation that closely resembles "cookie stuffing", which is prohibited by the Affiliate Agreement.

Non-aggressive linking strategies are best.

Supplier Links should be clearly presented as links to external websites, and should only appear in rational places. Affiliates who aggressively or deceptively try to generate as many Leads as possible will suffer reduced Lead values, because this often results in poor Lead-to-Booking conversion rates. A much more profitable strategy is to keep visitors on site longer, to market other products to them.

Results should be shown for multiple suppliers.

Linking only to the cheapest rate results in a poor user experience and reduced Lead values. Therefore, to show that prices are compared between multiple websites, Supplier Links must be presented for more than one supplier per hotel.

Display a progress bar while the API is working.

Affiliates who use the HotelSearch or CitySearch methods (instead of the asynchronous methods) are advised to display some sort of "progress bar" or "incrementally populated supplier list" to keep the user waiting while the results are fetched. This may be a GIF/PNG animation, a Flash video file or a JavaScript function, as long as it occupies the user while waiting for the results.

Report back-filling to HotelsCombined.

Affiliates who only use the API to back-fill their own hotel inventory must advise HotelsCombined. This ensures that duplicate suppliers are properly removed from the feed, to avoid exceeding each supplier's "Look-to-Book Ratio" limits.

All development and testing must use the Sandbox development environment.

The Affiliate Agreement states that Affiliates must maintain a reasonable Search-to-Lead ratio. Queries sent to the Sandbox do not contribute to this ratio. This means the Sandbox is a useful way to run test searches while developing a website. The Sandbox is described earlier in this document.

Appendix

Language Codes

LanguageCode	LanguageName	DisplayName
AR	Arabic	العربية
BG	Bulgarian	Български
CN	Traditional Chinese	繁體中文
CS	Simplified Chinese	简体中文
CZ	Czech	čeština
DA	Danish	Dansk
DE	German	Deutsch
EL	Greek	Ελληνικά
EN	English	English
ES	Spanish	Español
FR	French	Français
HE	Hebrew	עברית
HR	Croatian	Croatian
IS	Icelandic	Íslenska
IT	Italian	Italiano
JA	Japanese	日本語
KO	Korean	한국어
NL	Dutch	Nederlands
NO	Norwegian	Norsk
PB	Brazilian	Português Brasileiro
PL	Polish	Polski
PT	Portuguese	Português
RO	Romanian	Română
RU	Russian	Русский
SV	Swedish	Svenska
TR	Turkish	Türkçe

Currency Codes

CurrencyCode	CurrencyName
ADF	Andorran Franc
ADP	Andorran Peseta
AED	United Arab Emirates Dirhams
AFA	Afghanistan Afghani
AFN	Afghanistan Afghani
ALL	Albanian Lek
AMD	Armenian Dram
ANG	NL Antillian Guilder
AOA	Angolan Kwanza
AON	Angolan New Kwanza
ARS	Argentine Peso
ATS	Austrian Schilling
AUD	Australian Dollars
AWG	Aruban Florin
AZM	Azerbaijan Manat
AZN	Azerbaijan New Manat
BAM	Bosnian Mark
BBD	Barbados Dollar
BDT	Bangladeshi Taka
BEF	Belgian Franc
BGN	Bulgarian Lev
BHD	Bahrain Dinar
BIF	Burundi Franc
BMD	Bermudian Dollar
BND	Brunei Dollars
BOB	Bolivian Boliviano
BRL	Brazil reais
BSD	Bahamian Dollar
BTN	Bhutan Ngultrum
BWP	Botswana Pula
BYR	Belarusian Ruble
BZD	Belize Dollar
CAD	Canadian Dollars
CDF	Congolese Franc
CHF	Swiss Francs
CLP	Chile Pesos
CNY	Chinese yuan
COP	Colombian Peso
CRC	Costa Rican Colon
CSD	Serbian Dinar
CUC	Cuban Convertible Peso
CUP	Cuba Pesos

CurrencyCode	CurrencyName
CVE	Cape Verde Escudo
CYP	Cyprus Pound
CZK	Czech koruny
DEM	Deutsche Marks
DJF	Djibouti Franc
DKK	Danish Kroner
DOP	Dominican R. Peso
DZD	Algerian Dinar
ECS	Ecuador Sucre
EEK	Estonian Kroon
EGP	Egyptian Pound
ESP	Spanish Peseta
ETB	Ethiopian Birr
EUR	Euros
FIM	Finish Markkaa
FJD	Fijian Dollar
FKP	Falkland Islands Pound
FRF	French Francs
GBP	British Pounds
GEL	Georgian Lari
GHC	Ghanaian Cedi
GHS	Ghanaian New Cedi
GIP	Gibraltar Pound
GMD	Gambian Dalasi
GNF	Guinea Franc
GRD	Greek Drachma
GTQ	Guatemalan Quetzal
GYD	Guyanese Dollar
HKD	Hong Kong Dollars
HNL	Honduran Lempira
HRK	Croatian Kuna
HTG	Haitian Gourde
HUF	Hungarian Forint
IDR	Indonesian Rupiah
IEP	Irish Punt
ILS	Israeli New Shekel
INR	Indian rupees
IQD	Iraqi Dinar
IRR	Iranian Rial
ISK	Iceland Krona
ITL	Italian Lira
JMD	Jamaican Dollar

CurrencyCode	CurrencyName
JOD	Jordan Dinar
JPY	Japanese Yen
KES	Kenyan Shilling
KGS	Kyrgyzstanian Som
KHR	Cambodian Riel
KMF	Comoros Franc
KPW	North Korean Won
KRW	South Korean Won
KWD	Kuwait Dinars
KYD	Cayman Islands Dollar
KZT	Kazakhstan Tenge
LAK	Lao Kip
LBP	Lebanese Pound
LKR	Sri Lanka Rupee
LRD	Liberian Dollar
LSL	Lesotho Loti
LTL	Lithuanian Litas
LUF	Luxembourg Franc
LVL	Latvian Lats
LYD	Libyan Dinar
MAD	Moroccon Dirham
MDL	Moldovan Leu
MGA	Malagasy Ariary
MGF	Malagasy Ariary
MKD	Macedonian Denar
MMK	Myanmar Kyat
MNT	Mongolian Tugrik
MOP	Macau Pataca
MRO	Mauritanian Ouguiya
MTL	Maltese Lira
MUR	Mauritius Rupee
MVR	Maldive Rufiyaa
MWK	Malawi Kwacha
MXN	Mexican pesos
MYR	Malaysian Ringgits
MZM	Mozambique Metical
MZN	Mozambique New Metical
NAD	Namibia Dollar
NGN	Nigerian Naira
NIO	Nicaraguan Cordoba Oro
NLG	Netherlands Guilders
NOK	Norwegian Kroner
NPR	Nepalese Rupee

CurrencyCode	CurrencyName
NZD	New Zealand Dollars
OMR	Omani Rial
PAB	Panamanian Balboa
PEN	Peruvian Nuevo Sol
PGK	Papua New Guinea Kina
PHP	Philippine Pesos
PKR	Pakistan Rupee
PLN	Polish Zloty
PTE	Portuguese Escudo
PYG	Paraguay Guarani
QAR	Qatari Rial
ROL	Romanian Lei
RON	Romanian New Lei
RSD	Serbian Dinar
RUB	Russian Rouble
RWF	Rwandan Franc
SAR	Saudi riyal
SBD	Solomon Islands Dollar
SCR	Seychelles Rupee
SDD	Sudanese Dinar
SDG	Sudanese Pound
SDP	Sudanese Pound
SEK	Swedish Kronas
SGD	Singapore Dollars
SHP	St. Helena Pound
SIT	Slovenian Tolar
SKK	Slovak Koruna
SLL	Sierra Leone Leone
SOS	Somali Shilling
SRD	Suriname Dollar
SRG	Suriname Guilder
STD	Sao Tome/Principe Dobra
SVC	El Salvador Colon
SYP	Syrian Pound
SZL	Swaziland Lilangeni
THB	Thai baht
TJS	Tajikistani Somoni
TMM	Turkmenistan Manat
TMT	Turkmenistan New Manat
TND	Tunisia Dinars
TOP	Tonga Pa'anga
TRL	Turkish Lira
TRY	Turkish New Lira

CurrencyCode	CurrencyName
TTD	Trinidad/Tobago Dollar
TWD	Taiwan dollars
TZS	Tanzanian Shilling
UAH	Ukraine Hryvnia
UGX	Uganda Shilling
USD	U.S. Dollars
UYP	Uruguayan Peso
UYU	Uruguayan Peso
UZS	Uzbekistan Som
VEB	Venezuelan Bolivar
VEF	Venezuelan Bolivar Fuerte
VND	Vietnamese Dong
VUV	Vanuatu Vatu
WST	Samoan Tala

CurrencyCode	CurrencyName
XAF	CFA Franc BEAC
XAG	Silver (oz.)
XAU	Gold (oz.)
XCD	East Caribbean Dollar
XEU	ECU
XOF	CFA Franc BCEAO
XPD	Palladium (oz.)
XPF	CFP Franc
XPT	Platinum (oz.)
YER	Yemeni Rial
YUN	Yugoslav Dinar
ZAR	South African Rand
ZMK	Zambian Kwacha
ZWD	Zimbabwe Dollar

Supplier Codes

Note that this list changes as suppliers are added or removed from the list of compared hotel rates. The following URL shows the currently active suppliers:

http://www.hotelscombined.com/AboutUs/Providers.aspx

ProviderCode	ProviderName
ACC	Accorhotels.com
AGD	Agoda.com
BKS	Booking.com
BWN	BestWestern.com
CAR	Carlson.com
CHH	ChoiceHotels.com
EBK	Ebookers.com
ECT	EasyClickTravel.com
ELG	eLong.com
ETB	EasyToBook.com
EVI	BookDirectRooms.com
EXP	Expedia.com
FSB	FastBooking.com
GAR	getaroom.com
GTA	OctopusTravel.com
GTS	dhr.com
H4U	hotels4U.com
HCM	Hotel Website
HCT	HotelsChart.com
HLT	Hilton.com
HOA	Hotelopia.com
HPL	HotelPlanner.com

ProviderCode	ProviderName
HRS	hrs.com
HTC	HotelClub.com
IAN	Hotels.com
IHG	InterContinental.com
LMN	lastminute.com
LRM	LateRooms.com
MAR	Marriott.com
NIN	NeedItNow.com
OCT	OctopusTravel.co.uk
OTL	Otel.com
PLN	Priceline.com
QBD	QuickBeds.com
RMF	RoamFree.com
RST	ReserveTravel.com
RTG	RatesToGo.com
SKH	Skoosh.com
TCK	TravelCLICK.net
TCY	Travelocity.com
TGU	TravelGuru.com
VNR	Venere.com
ZUJ	zuji.com

Property Types

TypeID	PropertyType
0	Hotel
1	Motel
2	Inn
3	Bed & Breakfast
4	Vacation Rental
5	Hostel
6	Retreat
7	Resort
8	Other
9	Apartment