Kigo – Listing API

DEVELOPMENT INTRODUCTION

API revision

1

Document revision

5

Contact

api@kigo.net

1 Revision history

API Rev	Document		Date	Changes
	Rev	Status		
1	1			
1	2			Clarifying the process to get the inventory.
1	3			Clarifying the booking process.
1	4			Clarifying the authorization process.
1	5			Update on the diffs response and usage.

2 Contents

3 Glossary

UNIT	The minimum independent (with its own key) element bookable.
Listing	An advertiser of one or more units that have the same characteristics: location, size, capacity, amenities and pricing.
Content information	The information of any Listing that is less susceptible to change, such the name, description, photos, number of beds or amenities, i.e.
Pricing structure	The information used to calculate how it costs to book any Listing: daily rates, fees, taxes, discounts. Example: Building 1 has 4 units (Units 1 to 4) each one of them with 2 beds and 2 baths. Building 1 has also 4 units (Units 5 to 8) each one of them with 1 bed and 1 bath. Building 2 has 3 units (Units 9 to 11) each one of them with 2 beds and 2 baths. Building 2 has also 2 units (Units 12 to 13) each one of them with 1 bed and 1 bath. Listing 1 contains: Unit 1, Unit 2, Unit 3, Unit 4, Unit 9, Unit 10 and Unit 11. Listing 2 contains: Unit 5, Unit 6, Unit 7, Unit 8, Unit 12 and Unit 13.

4 Preface

The Listing API is a REST-like implementation based on the HTTP protocol, JSON data encoding and Unicode character encoding, over the secure HTTPS transport.

The API consists on a series of methods invoked by making HTTPS requests on the Kigo Rental Agency REST API servers. These methods can pull information from the Kigo system and can also create and cancel reservations.

It uses the OAuth 2.0 protocol for authentication and authorization.

4.1 What you need to integrate your Kigo Solution with an external system using the Kigo Listing API

In order to get the authorization, you'll need:

Token URL: https://auth.kigo.net/connect/token

Client ID: it will be provided by Kigo

Client Secret: it will be provided by Kigo

Scope: cm.api

Once you have the token, you must include it in the Header of all your API calls, "Authorization": "Bearer {access_token}".

Once you have completed the integration, you will need to complete a Certification process:

- A couple of listings will be temporarily shared with your account. They will be unshared once the certification process is completed.
- You will need to publish the information and share with our team the urls so they could start a set of tests, reviewing your API integration.
- You will receive Listings from other Kigo agencies through the Kigo Channel Manager once the Kigo team has certified your integration.

From now on, when this document refers to "your Listings", it means the Listings in your Kigo account shared with your company from another Kigo Agency account through the Kigo Channel Manager.

4.2 Reporting issues

Please, send any API related questions and issues to api@kigo.net.

Be aware that Kigo IT and support departments will neither check, revise, debug nor correct your code. They will only solve high level doubts and questions about what is possible and what isn't using the Listing API and generic best approaches.

4.3 Copyright notice

Kigo exclusively owns the intellectual property in this documentation. You acknowledge that you must not perform any act which infringes the copyright or any other intellectual property rights of Kigo.

5 API version and revision numbers

This is the documentation for the Listing API revision 1.

5.1 API version number

Backward compatibility from the technical point of view is guaranteed for all API revisions within the same API version number.

A change in the API version number implies that the new version is no longer backward compatible with the previous versions.

While we do our best to continue supporting old versions of API, it may be sometimes necessary, due to constant evolution of the application, to end the support for those versions, and ask our customers to upgrade their applications to use the latest version of the API.

5.2 API revision number

The API revision number changes each time the API is updated in such a way that the backward compatibility with previous revisions of the same API version is usually maintained.

New API methods may be added, and existing methods may become deprecated but still supported.

6 Working with the API

We suggest the following order of operations/API calls:

6.1 Getting the catalog

You will need to pull all the information from the Listings (shared with your Kigo Solution) to your own system and store it there in your own database.

You must make regularly calls to the endpoint **/channels/v2/connections** passing the param **options=IncludeAggregatedConnections** in order to get the list of PM accounts that have shared listings with your account. You'll get the array aggregatedChannelConnections in which every element will have a unique id. You need to make calls to the endpoint **/channels/v2/listings** passing each of these ids as **ccId**, so you'll get the list of all the Listings shared with your Kigo account. You should keep this info in your database.

Please note that it's possible that a PM has started the process to connect with your account, but it doesn't share any Listings yet, so it's important that you repeat this process regularly to keep your inventory updated.

6.2 Getting the content information

Once you have the list of Listings Ids (obtained from the first call to of **/listings**) it is time to pull the information of all those Listings calling to the endpoint **/listings/{id}** for each one to import the Content information.

6.3 Getting the pricing structure

The endpoint **/listings/{id}** will also provide the pricing structure for each Listing. Although you can use this information to show the breakdown of the rates, fees, taxes or discounts, **it should not be used to calculate the final price of a booking.**

6.4 Getting the LOS pricing

The endpoint **/listings/{id}/lospricing.csv** will provide the price per day and number of guests of a Listing for the requested period. You can store the information provided in your side to use it later for searching by price, i.e.

6.5 Getting the availability

The endpoint **/listings/{id}/availability** will also provide the calendar for any given Listing within a range of specific dates. The data provided by it combined with some of the information given by **/listings/{id}**, such as the blocked-out nights or the maximum stay allowed, i.e.

6.6 Keeping the information updated

You need to make calls regularly to the endpoint **/listings/diff** to get the list of the Listings that have changed their content, pricing, availability and/or status information since the last time you called.

Then, you should call **/listings/{id}, /listings/{id}/lospricing.csv** and **/listings/{id}/availability** to get the changes and update the information on your side.

If a change of the Listing **status** has been indicated, you need to take it out of the catalog, any attempt to retrieve information regarding that listing will fail.

Note: the required parameter **modifiedAfter** is limited to 7 days in the past. Since you must make the calls with a higher frequency than that, this limitation shouldn't cause any issue.

6.7 Bookings management

In order to create a booking, you need to make a quote first, through the **/quotes** endpoint. The API will provide either a Quote ID or the reason why the booking cannot be accepted. If the booking could be created, you can call the endpoint **/bookings/** and you **must** include the Quote ID in the call.

The reservation will be created according with the data (Listing Id, dates and number of guests) that you pass in the quote call and the API will provide a booking id.

You need to specify how the payment will be collected through the "**paymentCollectionMode**" field:

- **ChannelCollect** if the payment is collected on your side.
- **PMCollect** if the payment is processed on Kigo side.

You could specify the payment plan that will be applied to the booking.

You can use the booking id to verify the booking has being created properly through the endpoint **/bookings/{id}.**

If you need to cancel a booking, you can do it through the endpoint **bookings/{id}/cancel**