Headai Ltd
Rautatienpuistokatu 7
FI-28130 Pori, Finland

Headai

HEADAI REST API 1.15 API (partial)

Table of Contents

Headai Ltd
Rautatienpuistokatu 7
FI-28130 Pori, Finland

Headai

INTRODUCTION

Whether you are new to REST API or you have already experienced it, this document will provide all the relevant information required for using Headai REST API. This documentation is prepared for the software developers, researchers, and professionals to test, validate, and conceptualize the APIs' functioning and integration.

The Headai API is organized around REST. Our API has predictable resource-oriented URLs, it accepts form-encoded request bodies and typically returns JSON-encoded responses.

We assume the reader is familiar with working with REST APIs and curl. All API calls can be executed using curl (https://curl.se) to ensure your working infrastructure is functional. The Headai API generally uses HTTP POST requests with a body in JSON format and responses in JSON format.

Client-specific information to be used:

TOKEN (“token”)	The bearer token to be used is specified in the subscription details, which can be found on the Headai-Azure SaaS Subscription Registry site. A link is provided once the subscription is completed.
SERVER:PORT	Specified in the subscription details.
Charset	UTF-8.

Headai Ltd
Rautatienpuistokatu 7
FI-28130 Pori, Finland

Headai

HEADAI PREREQUISITES

Headai Graphmind

Headai Graphmind is a product of Headai Ltd and is based on 100% Headai IPR. It combines Cognitive AI and Proprietary Language Models to make huge-scale textual data interoperable and measurable to support strategic decision-making.

Call throttling and queueing

Each API call is always executed using one core/thread. If your Token has more than 1 core available (pending from the licensing scenario and/or used infrastructure), one can execute another API call simultaneously (under a multiprocessor infrastructure). The number of cores doesn't speed up the execution time of one API call (one calculation per thread). One core/thread can be also called a Calculation Unit.

If multiple API calls are made simultaneously for a token that has 1 core/thread available (or in token-free systems, e.g. private local/cloud infrastructure), the server queues the rest of the tasks until the token has a free core/thread. The core/thread is available immediately after the old calculation is finished.

All tokens have equal performance. There is no token prioritization that would perform faster or slow down the execution of other tokens. This prerequisite comes from the chosen architectural and computational reasons. Available cores are dependent on agreement with the client. By default, the token has one (1) core attached to it.

Datetime format, time and functionality

Datetime is in the format of "yyyy-mm-dd hh:mm:ss". If needed the Datetime is parsed to Date and Time strings accordingly by the system based on the actual type used in parameters (Datetime, Date or Time).
Date can always be used when Datetime is required. No padding is required in case of entering values for queries.

API accepts Datetime values “as-is”. No checks for Leap Year, calendar correctness or Daylight-Saving Time are executed.
If time is referenced in any of the results, it is always in the form of milliseconds.

Case sensitivity

Upper and lowercase matters. Headai REST API is case-sensitive. In a case-sensitive context, uppercase and lowercase letters are treated as distinct and separate entities, and they are not considered equivalent. This means that, for example, 'A' and 'a' are considered different characters.

Naming convention

Parameter	Description
param*	This is mandatory parameter. You must enter a value.
[aa|bb|cc]	Enter one the included choices as a value.
[aa|bb|**cc**]	If empty (parameter=) then default (**bold**) will be used.
Word	Enter one word as a value. Partial word starts from minimum of 3 letters in case of search or autocomplete.
List	Enter,a,list,of,words,separated,by,comma. See “Word”.
Text	Collection of one or more Words. Use URL encoding or URL body. No wildcards. Partial words are accepted. See “Word”.
Boolean	[true|**false**|empty], Empty is always false.
Date	Enter as Datetime, Date or Time.
String	See above “Text”.
Integer	Signed Integer. Use in the range of case-by-case specified (-min - +max) values (inclusive). No padding.
Decimal	Signed float. Use in the range of case-by-case specified (-min - +max) values (inclusive). No padding.
Language	Two-letter country codes as defined in ISO 3166.
Country	Full official “Short name” of the country in lower case letters as defined in ISO 3166.

Empty	Refers to an empty parameter "param=". In certain parameters can mean it's autodetected.
...	Three dots/ellipsis. List is endless. You got the point.

The default type of parameter is "Text", if not otherwise stated.

A brief introduction to Request and Response objects

Headai REST API result-set is not in any particular order. One can use any favorable tools that is capable of manipulating and organizing JSONs in limitless ways.

Most of the result objects are self-explanatory. Some will need more details about the technology. This is a list of the most complex result objects to be understood:

Object name	Typical use-case
Ontology	Language models are a fundamental part of many AI applications that process natural language. Their performance continuously improves with advancements in machine learning and neural networks. Headai develops cutting-edge technology for qualitative analysis, classification, and sentiment analysis of text. These models learn patterns and relationships within large text datasets—such as news articles, scientific papers, books, and online posts—and apply that knowledge to various language-related tasks.
Dataset	Pre-trained datasets consist of Headai's proprietary models that have been trained and refined using large volumes of text data. These datasets are carefully curated to ensure diversity, objectivity, and the removal of errors or anomalies that could negatively impact learning. Headai's datasets are fine-tuned for accuracy and reliability in specific use cases, such as workforce analysis or country-wide economic assessments, to enhance performance and result quality.
Value	An integer representing a numeric value assigned to an object or the count of objects

	within a given interest. In some cases, this also correlates with **Relevancy**.
Relevancy	An integer calculated based on complex algorithms and recommendations from multiple sources. In a typical use case, relevancy is determined by factors such as the number of influencing objects and their respective **Weight** values. In some cases, neighboring objects impact the relevancy score. A higher relevancy number generally indicates a stronger match or recommendation. Relevancy is always a sum of multiple factors and intricate algorithms designed for accuracy in each use case. Typical range: - 1, very weak connection - several thousand,strong connection, highly relevant to the case Relevancy can be affected by factors such as **geolocation**. For example, 100 open job positions in a small city might be highly significant, whereas in a larger city, 5000 open positions might be required for the same level of relevancy. Relevancy values are only comparable within the same use case.
Weight	An integer (1-5) indicating the strength of an object's impact on relevancy: 1=light impact (may become significant when combined with many objects) 5=strong impact (may lose meaning if associated with too few objects) In keyword analysis, words with a weight of **3 or higher** tend to have distinct significance.
Node	A node represents an entity or concept within a knowledge graph. It can be a real-world object, an abstract concept, a person, a location, or any other type of structured information.
Edges	Edges define the relationships between nodes in a knowledge graph. They describe how entities are connected. Depending on the use case, the strength of a connection can be expressed through **Value, Relevancy, or Weight.**

Score	A numeric value (1-n) representing the significance of identified keywords and the meaningfulness of the text.

Response Status error Codes

Headai REST API uses the following response status codes, as defined in the RFC-2616 (https://www.ietf.org/rfc/rfc2616.txt):

Status Code	Description
200	OK - The request has succeeded. The client can read the result of the request in the body and the headers of the response.
201	Created - The request has been fulfilled and resulted in a new resource being created.
202	Accepted - The request has been accepted for processing, but the processing has not been completed.
204	No Content - The request has succeeded but returns no message body.
	Not Modified. See Conditional requests.
400	Bad Request - The request could not be understood by the server due to malformed syntax. The message body will contain more information; see Response Schema.
401	Unauthorized - The request requires user authentication or, if the request included authorization credentials, authorization has been refused for those credentials.
403	Forbidden - The server understood the request, but is refusing to fulfill it.
404	Not Found - The requested resource could not be found. This error can be due to a temporary or permanent condition.
429	Too Many Requests - Rate limiting has been applied. See Token throttling and queueing.
500	Internal Server Error. This should never happen. If you are unlucky enough and receive this error, please report it to us.
502	Bad Gateway - The server was acting as a gateway or proxy and received an invalid response from the upstream server.

503	Service Unavailable - The server is currently unable to handle the request due to a temporary condition which will be alleviated after some delay. You can choose to resend the request again.

BEFORE YOU BEGIN, TEST YOUR ACCESS

To make sure you have fully functional access to Headai endpoints, please find the correct SERVER and PORT specified in your subscription details. Then, use e.g. curl to type:

REQUEST EXAMPLE (Replace SERVER:PORT with your own info)

```
% curl -X POST http://headaitest.northeurope.azurecontainer.io:8080/TextToKeywords
```

RESPONSE EXAMPLE

```
{
  "message": "Invalid or missing API-key.",
  "status": 401
}
```

If you received the expected response (which is natural since you didn't include your API key) with Status **401,** you're all set to proceed and start building your integration.

It is best practice to execute all calls in curl. You are free to use your favorite tool, but you need to ensure you understand the following prerequisites concerning URL encoding and charsets:

1) Most of the integrations to different programming languages will need explicit URL encoding to happen for successful execution.
2) Almost any snippet of natural language copied from different sources can and will include special letters, characters, or special meaning markings which will require URL encoding.
3) In this documentation, we rely on the reader to understand URL encoding and have an adequate understanding of his/her current charset (in the operating system and in the application) and how it differs from the default (UTF-8).
4) It is a typical practice to use JSON-body in the POST URL to enter special characters and markings. This is typically the safest of transmitting special characters between client and server.

Headai Ltd
Rautatienpuistokatu 7
FI-28130 Pori, Finland

Headai

HEADAI REST API 1.15 API (PARTIAL)

1) TextToKeywords

POST **TextToKeywords**

Extract contextually strong keywords and their semantic weights.

It's a semantic algorithm-based function that extracts the most impactful words from a given text and maps their relationships to others. By analyzing semantic weight and contextual relevance, it generates a structured list of key terms, helping to uncover hidden patterns and meanings within the text.

Params	Type, Default values, Description
language*	[fi|en], Defines language to be used in analysis.
threads	[**1**-8], number of threads used to run request. The more used the more faster execution time will be. Typical factor is the number itself. 4 is four times faster.
Type	[**json**|html], Defines the type of an output.
text*	[text], text to be sent to analysis.

REQUEST EXAMPLE

```
POST http://headaitest.northeurope.azurecontainer.io:8080/TextToKeywords
Authorization: API-key TOKEN
Content-Type: application/json

{
  "language": "en",
  "threads": 1,
  "type": "json",
  "text": "Student understands the basic principles of data and software business and the special characteristics of software industry. He/she can critically analyze how it is possible to monetize with data and software. He/she can analyze the feasibility of software business models. Student can apply theoretical knowledge and understanding of the data and software business characteristics to create a solid lean canvas model for a software start-up."
}
```

You will get the following response in case of success:

RESPONSE EXAMPLE


```
{
  "skills": [
    {
      "alternative_concepts": [],
      "concept": "software",
      "displayname": "software",
      "count": 4,
      "weight": 1,
      "relevancy": 4,
      "language": "en",
      "relations": []
    },
    {
      "alternative_concepts": [],
      "concept": "business_to_business",
      "displayname": "business_to_business",
      "count": 6,
      "weight": 1,
      "relevancy": 6,
      "language": "en",
      "relations": []
    },
    ...
  ],
  "original": "Student understands the basic principles of data and
software business and the special characteristics of software industry.
He/she can critically analyze how it is possible to monetize with data and
software. He/she can analyze the feasibility of software business models.
Student can apply theoretical knowledge and understanding of the data anad
software business characteristics to create a solid lean canvas model for
a software start-up.",
  "indicators": {
    "processing_time": "",
    "word_count": 68,
    "information_quality_indicator": "",
    "keyword_count": 30,
    "information_density": "",
    "total_words": 68,
    "information_quality": "",
    "unique_keyword_count": 17,
    "cumulative_weight": 0,
    "butterfly_effect": "",
    "knowledge_gravity": ""
  }
}
```

RESPONSE OBJECTS (those which aren't self-explainable and not part of prerequisites)	
Name	**Description**
Concept	A Concept is a core term or category that captures the essential meaning within a given piece of text. It represents a central idea around which related words and entities are structured.
Alternative_concepts	A higher-level semantic unit that links multiple words, synonyms, or phrases with similar meanings.
Relations	Connections between data points, concepts, or entities in Headai's AI models. Relations help structure information by linking nodes meaningfully, enabling accurate insights and recommendations.
Information_quality_indicator	A metric assessing the accuracy and reliability of data used in Headai's AI models. This indicator evaluates factors such as source credibility, consistency, and contextual relevance.
Information_density	A measure of how much meaningful information is present in a dataset. Higher density means a dataset contains a rich set of relevant, non-redundant data, improving AI efficiency.
information_quality	A broad measure of data integrity that includes accuracy, completeness, consistency, and relevance. High-quality data ensures more reliable AI outputs and minimizes bias.
cumulative_weight	The combined weight of multiple factors influencing an AI-driven outcome. This is used when multiple objects or keywords contribute to a decision, determining their collective impact.
butterfly_effect	A concept describing how small changes in input data can lead to significant, sometimes unpredictable shifts in AI-generated results. This highlights the importance of data consistency.
knowledge_gravity	A measure of how strongly a concept attracts related information within Headai's AI ecosystem. High knowledge gravity indicates that a concept is a central and highly connected part of the knowledge graph.

[end of the document]