


AudioGo Reporting API

Before calling AudioGo API, you need an authentication key. Please contact AudioGo support to have a key created for you. After you have the key, store it in a secure location.

 **Important:** If you lose your API key, you'll need to request a new one.

Now, you can use your API key for authentication by adding it as a header in your requests:

```
x-api-key: your_api_key_here
```

You should double your retry wait policy after every call so the API can process your requests. This means that the application waits a short time before the first retry and exponentially increases the time between each subsequent retry. For example, it may retry the operation after 2 minutes, 4 minutes, 8 minutes, and so on.

The API treats all time values as UTC, regardless of your agency's time zone. When using the API, please make sure you convert all necessary times to and from UTC.

Base URL: <https://api.adswizz.com/domain>

Rate limits & Quota

All requests sent to the AudioGo API are counted toward a quota. If the quota of requests to the AudioGo API is exceeded, the API returns an error code 429.

Free plan limits & quotas:

- The API requests a steady-state rate limit (average requests per second over an extended period of time): **1 per second**.
- The maximum request rate limit over time, ranging from one to a few seconds (burst rate): **5 per second**.
- The maximum number of requests per month: **10,000**.

Limitations

- A maximum of 3 **splitters/dimensions** can be requested for **synchronous reports**.
- A maximum of 5 **splitters/dimensions** can be requested for **asynchronous reports**.
- The maximum interval available for reporting depends on the number of dimensions used as splitters in the request:
 - for **1-3 splitters/dimensions**, the maximum interval available is the **last 15 months**;
 - for **4-5 splitters/dimensions**, the maximum interval available is the **last 3 months**.

List of Reporting API Endpoints

Name	Method	URL	Description
Create an async report	POST	v8/reports/query	Trigger the generation of an asynchronous report. You will not receive the actual report data in the response; you will need to query Get report status at a later time to check the processing status and download the result.
Get report status	GET	v8/reports/async-query/{internalReportID}	Returns the status of a given report, and if ready, a download URL.
Create a sync report	POST	v8/reports/async-query	Returns a report with the requested metrics grouped by dimensions, in the order you defined in the request.
Get dimensions	GET	v8/reports/dimensions	Get report dimensions. The returned dimensions are environment-related. Alongside the name and display of the dimension, a description is returned that explains its usability.
GET metrics	GET	v8/reports/metrics	Get report metrics. The returned metrics are environment-related. Alongside the name and display of the metric a description is returned that explains the usability of it.
Filter analytics results	POST	v8/reports/filter	This endpoint returns the list of values that can later be used to filter results.

Every report you create with this API is made up of **Dimensions** and **Metrics**. For running custom statistical reports, you will first need to get a list of dimensions and metrics available. Those values will be used afterwards in the **Query** and **Filter** operations.

Dimensions are the attributes for your data. For example, a `country` dimension can indicate the country where the request originated. Another example of a dimension is `os`. It indicates the operating system from which the request originated.

Metrics are quantitative measurements. For example, `impressions` indicate the total number of impressions for a particular ad.

Running a report consists of several steps:

1. You need to have a list of dimensions and metrics available. For this, you need to request the two endpoints: **Dimensions** and **Metrics**. Let's imagine you want to have the number of audio impressions in two cities: New York and Las Vegas, for some specific campaigns that run in those cities only on mobile devices, grouped by advertisers in a specified time interval. You will need to request the list of cities from the system with a filter query like this:

```
None
{
  "id": "geoCity",
  "interval": {
    "from": "2018-10-01T00:00:00Z",
    "to": "2018-10-19T23:59:59Z"
  },
  "limit": 20
}
```

From the response list, we select the two cities we are interested in and request a new filter for the campaign with the two cities as arguments.

2. The second request will look like this:

None

```
{
  "id": "campaignName",
  "filter": {
    "type": "AND",
    "fields": [{
      "id": "geoCity",
      "type": "IN",
      "values": ["new york", "las vegas"]
    }]
  },
  "interval": {
    "from": "2018-10-01T00:00:00Z",
    "to": "2018-10-19T23:59:59Z"
  },
  "limit": 20
}
```

From the response list, we select the campaign we are interested in and request a new filter:

None

```
{
  "id": "deviceTypeLabel",
  "filter": {
    "type": "AND",
    "fields": [{
      "id": "geoCity",
      "type": "IN",
      "values": ["new york", "las vegas"]
    }, {
      "id": "campaignName",
      "type": "IN",
      "values": ["MyAwesomeCampaign(239)",
        "MyProfitableCampaign(240)"]
    }]
  },
  "interval": {
    "from": "2018-10-01T00:00:00Z",
    "to": "2018-10-19T23:59:59Z"
  },
  "limit": 20
}
```

3. With all these results from the previous filters, we can then issue a query request like this:

```
None
{
  "filter": [{
    "id": "geoCity",
    "values": ["new york", "las vegas"]
  },
  {
    "id": "campaignName",
    "values": ["MyAwesomeCampaign(239)",
      "MyProfitableCampaign(240)"]
  },
  {
    "id": "deviceTypeLabel",
    "values": ["mobile"]
  }
],
  "splitters": [{
    "id": "advertiserName",
    "limit": 5
  }],
  "metrics": ["impressions"],
  "interval": {
    "from": "2018-10-01T00:00:00Z",
    "to": "2018-10-19T23:59:59Z"
  },
  "sort": {
    "id": "impressions",
    "dir": "DESC"
  }
}
```

This will return the impressions grouped by advertiser, descending by its number in the specified interval.

Create an async report

Calling this endpoint will trigger the generation of an asynchronous report. You will not receive the actual report data in the response; you will need to query **Get report status** at a later time, to check the processing status and download the result.

In the request body, the filters property defines the criteria for filtering the results. To build the filters array, use the response(s) to the request(s) you made to the **Filter endpoint**.

To specify the dimensions that will be used as criteria for breaking down the results (e.g: by campaign, by city, by zip code), use the splitters property. You can retrieve the list of all available dimensions / splitters, together with their descriptions, by calling the **Dimensions endpoint**.

NOTES:

1. *A maximum of 5 splitters can be requested for asynchronous reports.*
2. *The maximum interval available for reporting depends on the number of dimensions used as splitters used in the request:*
 - *for 1-3 splitters, the maximum interval available is the last 15 months;*
 - *for 4-5 splitters, the maximum interval available is the last 3 months.*

Use the metrics property to request the set of metrics to be included in the report (e.g. Impressions, Clicks). The sort property must reference a single metric, to be used for sorting the results (e.g. sort the results by number of Impressions). To retrieve the list of available metrics, together with their descriptions, see **Metrics endpoint**.

Method	POST
Endpoint	v8/reports/async-query
Example Payload	<pre>None { "queryAsyncRequest": { "interval": { "from": "2025-07-14T14:00:00Z", "to": "2025-07-21T14:00:00Z" }, "metrics": ["objectiveCountableSum"], "filters": [{ "id": "geoCountryName", "values": ["us", "ca"], "exclusion": false }], }, }</pre>

	<pre> "splitters": [{ "id": "__time", "granularity": "HOUR" }], "sort": { "id": "bids", "dir": "ASC" } }, "name": "My report" }</pre>
Sample response	<p>None</p> <pre>{ "internalReportIds": "[clientName-2-666-a4ed1010-5a94-11e9-99b2-06131 ed0ee3a]", "queryRequest": { "queryAsyncRequest": { "interval": { "from": "2025-07-14T14:00:00Z", "to": "2025-07-21T14:00:00Z" }, "metrics": ["objectiveCountableSum"], "filters": [{ "id": "geoCountryName", "values": ["us", "ca"], "exclusion": false }], "splitters": [{ "id": "__time",</pre>

	<pre> "granularity": "HOUR" },], "sort": { "id": "bids", "dir": "ASC" }, }, "name": "My report" }, "meta": "\"meta\": {\"code\": 200}" }</pre>
--	--

Get report status

Calling this endpoint the user can retrieve the report status. If the report is not ready there will be retrieved a status **STARTED** or **FAILURE** if it failed.

If the report is done, then the user will receive the status **SUCCESS** and a **download link**.

Method	GET
Endpoint	v8/reports/async-query/{internalReportID}
Sample response	<pre>None { "name": "My report", "status": "SUCCESS", "downloadLink": "bucket/folder/My report-12as34fg.csv", "generatedAt": "2025-05-08T22:26:58Z" }</pre>

Metrics

Get report metrics. The returned metrics are environment-related. Alongside the metric's name and display, a description that explains its usability is returned.

Method	GET
Endpoint	v8/reports/metrics
Sample response	<pre> None [{ "id": "fillRate", "display": "Fill Rate", "description": "This metric is not available for Ad, Ad type, Campaign, Campaign type, Advertiser, External references, Order, Audience segments dimensions", "blacklist": ["salesChannelName", "supplyPackSegmentName"], "dependsOn": ["geoCountryName", "geoRegion"] }] </pre>

Filter

This endpoint returns the list of values that can later be used in the Query to filter results. For example, if needed to filter by a single dimension the id of the request is the dimension name, the interval is a valid ISO8601 date format interval of dates. If further filtering of that list of values is wanted e.g. dimensions values that starts with a specific prefix then the filter field must be used with type = "AND" between the array of fields. For this example the array of fields consist of a single element with id equal to that dimension name, type = "SEARCH" and query = "prefix". Other filters can be added in the same way in the array. The response of this endpoint is further used in calling the **Query endpoint**.

Method	POST
--------	------

Endpoint	v8/reports/filter
Sample payload	<div>None</div> <pre>{ "id": "deviceTypeLabel", "filter": { "type": "AND", "fields": [{ "id": "geoCity", "type": "IN", "values": ["dublin", "new york"] }, { "id": "campaignName", "type": "IN", "values": [239] }, { "id": "deviceTypeLabel", "type": "SEARCH", "query": "desktop" }] }, "interval": { "from": "2018-06-01T00:00:00Z", "to": "2018-06-30T23:59:59Z" }, "limit": 20 }</pre>
Sample response	<div>None</div> <pre>{</pre>

```

    "id": "deviceTypeLabel",
    "filter": {
      "type": "AND",
      "fields": [
        {
          "id": "geoCity",
          "type": "IN",
          "values": [
            "dublin",
            "new york"
          ]
        },
        {
          "id": "campaignName",
          "type": "IN",
          "values": [
            239
          ]
        },
        {
          "id": "deviceTypeLabel",
          "type": "SEARCH",
          "query": "desktop"
        }
      ]
    },
    "interval": {
      "from": "2018-06-01T00:00:00Z",
      "to": "2018-06-30T23:59:59Z"
    },
    "limit": 20
  }
}

```

Dimensions endpoint

Get report dimensions. The returned dimensions are environment-related. Alongside the dimension's name and display, a description that explains its usability is returned.

Method	GET
--------	-----

Endpoint	v8/reports/async-query/{internalReportID}
Sample response	<pre> None { "name": "My report", "status": "SUCCESS", "downloadLink": "bucket/folder/My report-12as34fg.csv", "generatedAt": "2025-05-08T22:26:58Z" } </pre>

Query analytics

Calling this endpoint will return the requested **metrics** grouped by **dimensions**, in the order you defined in the request.

In the request body, the `filters` property defines the criteria for filtering the results. To build the filters array, use the response(s) to the request(s) you made to the **Filter endpoint**.

To specify the dimensions that will be used as criteria for breaking down the results (e.g: by Campaign, by city, by zip), use the `splitters` property. You can retrieve the list of all available dimensions (splitters), together with their descriptions, by calling the **Dimensions endpoint**.

NOTE: A maximum of 3 splitters can be requested for a synchronous report.

Use the `metrics` property to request the set of metrics to be included in the report (e.g. Impressions, Clicks). The `sort` property must reference a single metric, to be used for sorting the results (e.g. sort the results by number of Impressions). To retrieve the list of available metrics, together with their descriptions, see the **Metrics endpoint**.

Method	POST
Endpoint	v8/reports/query
Sample payload	<pre> None { "interval": { </pre>

	<pre> "from": "2025-07-14T14:00:00Z", "to": "2025-07-21T14:00:00Z" }, "metrics": ["supplyECPMInUSD", "objectiveCountableSum", "bidResponses", "bids"], "filters": [{ "id": "geoCountryName", "values": ["us", "ca"], "exclusion": false }], "splitters": [{ "id": "__time", "limit": 5, "granularity": "HOUR" }], "sort": { "id": "bids", "dir": "ASC" } }</pre>
Sample response	<pre>None { "name": "My report", "status": "SUCCESS", "downloadLink": "bucket/folder/My report-12as34fg.csv", "generatedAt": "2025-05-08T22:26:58Z" }{</pre>

	<pre> "key": "geoCity", "total": { "bids": 165, "objectiveCountableSum": 90 }, "data": [{}], "attributes": { "granularity": "DAY", "valueType": "TIME" } </pre>
--	---

List of AudioGo API Reporting Dimensions and Metrics

Dimension	Description	ID
Ad	The name of the ad being reported on.	adName
Advertiser	Name of the advertiser running the campaign.	advertiserName
Advertiser ID	System-generated unique identifier for the advertiser.	advertiserId
Age	Represents age segments based on targeting or inferred demographics.	requestVariables_aw_0_awz.a_name
Agency	Name of the media agency managing the campaign.	agencyName
Agency ID	System-generated unique identifier for the agency.	agencyId
Behavioural Audience (Targeted)	Indicates which behavioral segments were targeted by the campaign.	usedSegments_aw_0_a wz.segments / usedSegments_aw_0_a wz.segments_name
Campaign	The name of the campaign.	campaignName
Campaign ID	Unique identifier for the campaign.	campaignId

Device Type	Type of device where the ad was served.	deviceTypeLabel
Gender	Represents gender segments based on targeting or inferred demographics.	requestVariables_aw_0_awz.g_name
Genre	Indicates the genre targeted (from publisher metadata).	requestVariables_aw_0_azn.pgenre
Genres	Indicates targeted genres via behavioral/interest segments.	usedSegments_aw_0_azn.pgenre
Geo - City	The name of the targeted or reporting city.	geoCity
Geo - Country	The name of the targeted country.	geoCountryName
Geo - Postal Codes	Postal/ZIP codes of targeted areas.	geoPostal
Geo - Region	The name of the region within a country.	geoRegionName
Geo - Region Code	Abbreviated or coded version of region name.	geoRegion
Geo - Sub-Region	More granular division within a region.	geoSubRegionName
Marketplace Publishers	Publisher name from whom inventory was resold, dependent on transparency settings.	requestVariables_aw_0_pub.name
Network Publisher	Indicates the network publisher (used exclusively with “Impressions”).	requestVariables_aw_0_azn.pname
Player Name	Name of the player rendering the ad.	playerName
Podcast - Show Name (Insights)	Name of the podcast show where impressions were delivered.	requestVariables_aw_0_cnt.internal_series_id
Podcast Contextual Audiences (Targeted)	Contextual audience segments based on podcast content targeting.	usedSegments_aw_0_pod.con2be_name
Predictive Audiences by Comscore (Targeted)	Audiences derived from Comscore predictive modeling and targeted.	usedSegments_aw_0_pod.pa_comscore_name
Sonar - Age (Insights)	Demographic age groups inferred by Sonar technology.	requestVariables_aw_0_awz.a
Sonar - Gender (Insights)	Inferred gender segments derived via Sonar.	requestVariables_aw_0_awz.g

Time	Time intervals for reporting (e.g. hour, day, month).	__time
------	---	--------

Metric	Description	ID
Audio Impressions	The number of times a listener heard an audio ad.	demandAudioImp
Audio Spend	Total cost associated with audio ad delivery.	totalSpend
CTR	Click-through rate for Companion Banner or Second Screen ads.	ctr
Clicks	Total number of user interactions (clicks) on Companion or Second Screen ads.	clicks
Companion Banner Clicks	Number of clicks specifically on companion banners.	clickDisplay
Companion Banner Impressions	Number of impressions where a companion banner was served. Not compatible with Ad Duration.	companionSum
Impressions In Objective	Number of impressions counted towards the campaign objective.	objectiveCountableSum
LTR	Listen Through Rate — % of audio ads heard to completion.	ltr
Media eCPM	Effective Cost Per Mille paid to the publisher.	demandECPM
Reach Audio Unique Users	Number of distinct users exposed to audio impressions.	demandAudioReachUniqueUsers
Retargeting Clicks	Clicks from users on retargeting/second screen elements.	clickSecondScreen
Retargeting Impressions	Number of ad impressions served on retargeting-enabled second screens.	secondScreenImp
Retargeting Spend	Total spend associated with retargeting ads.	totalSecondScreenSpend
Total Agency Margin	Total fee collected by the agency, not invoiced but tracked.	totalAgencyMarginSum
Total Conversions	Sum of all conversion events.	attributionConversions
Total Data Spend	Total cost for using audience data.	segmentsPriceSum

Total Reach - Unique Users	Overall number of distinct listeners exposed to the campaign.	listenerIdHLL
Total Spend	Grand total spent, including media, data, platform, and attribution costs.	demandTotalSpend
Unique Conversions	Count of unique users who converted.	attributionConversionReachUniqueUsers