Quasar

Developer API Documentation

Revision 1 - August 2018

About the API

The API allows Quasar users to access their data via a RESTful web servce with a JSON response format. If users would like to quickly export their data for building custom reports, or integrate Quasar call scores with a third-party or in-house tool, the API provides real-time access to call scores, agents, and teams.

About this Document

The intended audience for this document is software and web developers experienced with RESTful web services. The reader should already be able to initiate HTTP requests and process JSON responses.

Getting Started

The API endpoint is at:

https://callscores.com/lib

The desired resource is communicated to the server by the client via the URL query string. The value of the 'action' key determines the resource returned, and additional information required for certain resources is encoded in additional key/value pairs. For instance, the query string to retrieve data for all of the agents associated with the specified user is:

?action=getAgents

and the query string to retrieve data on a specific agent is:

?action=getAgent&id=000

with '000' being the associated ID number for the desired agent. The full listing of available resources and the required additional key/value pairs is included later in this document.

In accordance with the stateless nature of RESTful architecture, no client authentication information is stored between requests, each request MUST include a POST request body containing an 'id' key with the user's numeric ID and a 'token' key with the user's API token. If the user is unsure of their id or token, they can retrieve it at:

https://callscores.com/api

An example of a well-formed API request would be:

```
POST /lib/?action=getAgents HTTP/1.1
Host: callscores.com
Content-Length: 79
Content-Type: application/x-www-form-urlencoded; charset=UTF-8
id=000&token=UsEr%T0K3N%Str1NG
```

Example code for generating a suitable request with PHP or jQuery is included later in this document.

Resource Actions

A listing of all the available resources including parameters and example return values.

getAgent

Description: Retrieves information for the specified agent numeric ID
Additional Parameters:

id - the desired agent's numeric ID (required)

Example Query String: ?action=getAgent&id=12
Return Value:

```
{
    "agent_name": "Peter",
    "agent_id": 12,
    "team_name": "Impossible",
    "team_id": 2,
    "status": "active"
}
```

Notes: 'agent_name' is the text description of the agent, 'agent_id' is the agent's numeric ID, 'team_name' is the text description of the team the agent is currently associated with, 'team_id' is the corresponding numeric ID, and 'status' is either 'active' or 'inactive'. Agents with 'inactive' status have been deleted by the user, but are eligible to be restored, and are included for the purposes of associating historic score data with the appropriate agent. See 'getAgents' for additional information.

Resource Actions

A listing of all the available resources including parameters and example return values.

getAgents

Description: Retrieves the list of all agents associated with the specified user
Additional Parameters: None
Example Query String: ?action=getAgents
Return Value:

```
{
    “1”: {
        “agent_name”: “William”,
        “agent_id”: 1,
        “team_name”: null,
        “team_id”: null,
        “status”: “inactive”
    },
   “10”: {
        “agent_name”: “David”,
        “agent_id”: 10,
        “team_name”: “Bad Wolf”,
        “team_id”: 1,
        “status”: “active”
    },
    “12”: {
        “agent_name”: “Peter”,
        “agent_id”: 12,
        “team_name”: “Impossible”,
        “team_id”: 2,
        “status”: “active”
    }
}
```

Notes: The JSON object is indexed by the agent’s numeric ID; since agents are generally infrequently added or changed, the preferred behavior is for the client to locally cache the result of getAgents for the duration of the session and retrieve details as needed from the local cache rather than querying the getAgent resource repeatedly.

Resource Actions

A listing of all the available resources including parameters and example return values.

getCriterion

Description: Retrieves details of the specified scoring criterion.
Additional Parameters:

id - the numeric ID of the desired criterion (required)

Example Query String: ?action=getCriterion&id=868
Return Value:

```
{
    “criterion_id”: 868,
    “status”: “active”,
    “weight”: 10,
    “description”: “Answer phrase”,
    “options”: [
        “Not complete or poor”,
        null,
        null,
        “Partially complete or adequate”,
        “Complete”,
        “Complete, with genuine interest and positive attitude”
    ],
    “values”: [
        0,
        1,
        2,
        3,
        4,
        10
    ]
}
```

Notes: Like agents, criteria isn’t frequently added or changed, so the preferred behavior is to locally cache the result of getCriteria for the duration of the session and retrieve details as needed from the local cache rather than querying the getCriterion resource repeatedly. However, since the criteria is stored in various category objects when returned by getCriteria, the most useful way to create a local cache may be to iterate over the category objects creating a new array or object of the criteria objects contained therein indexed by numeric criterion ID, for ease in subsequent retrieval by ID. Properties of the criterion object are the criterion’s numeric ‘criterion_id’, ‘status’ as active or inactive (deleted), ‘weight’ within the category, the text ‘description’, and two arrays - ‘options’ and ‘values’. ‘Options’ is an array containing the text description of each possible grading option, ‘values’ is the associated point value. The array indexes are correlated so that, for instance, options[3] corresponds with values[3]. The exception is for criteria in categories of the ‘counter’ type, those have no ‘options’, and ‘values’ is a JSON object with ‘each’ and ‘max’ point values. See ‘getCriteria’ for more information on retrieving categories.

Resource Actions

A listing of all the available resources including parameters and example return values.

getCriteria

Description: Retrieves all the scoring criteria for the specified user along with their categories.
Additional Parameters: None
Example Query String: ?action=getCriteria
Return Value:


```
{
        {
            “category_id”: 194,
            “weight”: 90,
            “type”: “unique”,
            “description”: “Greeting”,
            “criteria”: [
                {
                    “criterion_id”: 868,
                    “status”: “active”,
                    “weight”: 10,
                    “description”: “Answer phrase”,
                    “options”: [
                        “Not complete or poor”,
                        null,
                        null,
                        “Partially complete or adequate”,
                        “Complete”,
                        “Complete, with genuine interest”
                    ],
                    “values”: [
                        0,
                        1,
                        2,
                        3,
                        4,
                        10
                    ]
                }
            ]
       }
}
```


Notes: By far the most complex return object, further examples are included later in this document, this version is simplified with only a single category with a single criterion. The basic structure, however is arrays of JSON objects representing individual criteria inside parent category objects; Category Object->Criteria Array->Criterion Object. The properties of the category object are ‘category_id’ representing the category’s numeric ID, ‘weight’ used in ordering (heavier weights are ordered last), ‘type’ of the category (unique meaning it’s only scored once per call, cloneable meaning it can be scored an unlimited amount of times for each call - e.g. agent holds, or counter whereby each occurence is worth points up to a specified maximum), and the text ‘description’.

Resource Actions

A listing of all the available resources including parameters and example return values.

getScore

Description: Retrieves details of the specified evalution.
Additional Parameters:

id - the numeric ID of the desired evaluation (required)

Example Query String: ?action=getScore&id=1258
Return Value:


```
{
    “score_id”: 1258,
    “total_score”: 100,
    “agent_id”: 12,
    “reviewer_id”: 8,
    “team_id”: 0,
    “time_taken”: 1383771660,
    “time_scored”: 1383772034,
    “client_id”: “7314”,
    “logger_file”: “//LAN/path/to/file.mp3”,
    “comment”: “Lorem ipsum dolor sit amet“,
    “detail”: {
        “214”: [
            “2”
        ],
        “1178”: [
            “5”
        ],
        “4597”: [
            “5”,
            “3”
        ],
        “4358”: [
            “2”
        ]
    },
    “status”: “active”
}
```


Notes: ‘score_id’ is the numeric ID of the evaluation, ‘total_score’ is the number of points scored represented as a percentage of total points possible, ‘agent_id’ is the associated agent’s numeric ID, ‘reviewer_id’ is the numeric ID of the reviewer that scored the call, ‘team_id’ is the numeric ID of the team the agent was associated with when this call was scored (or 0, if the agent wasn’t assigned to a team), ‘time_taken’ and ‘time_scored’ are Unix timestamps representing the date and time the call was taken and the evaluation was completed, respectively, ‘client_id’ is the client ID the user associated this evaluation with, ‘logger_file’ is a file path to the voice recording of the call on the user’s LAN, ‘comment’ is any comments entered, ‘detail’ is the scoring results of each individual criteria (or null if the user is utilizing a deprecated version of the default criteria, ask the user to contact support), and ‘status’ is ‘active’ or ‘inactive’ (deleted). The ‘detail’ object keys are criteria numeric IDs, and the values are arrays of either the index of selected options corresponding with ‘options’ and ‘values’ in getCriteria (a single option for unique criteria, or multiple ones for cloneable criteria, one for each clone generated for this evaluation), or the final tally in the case of counters.

Resource Actions

A listing of all the available resources including parameters and example return values.

getScores

Description: Retrieves details of multiple specified evalutions.

Additional Parameters:

agent_id - evaluations for the agent with the specified numeric ID (optional)
client_id - evaluations associated with the client ID specified (optional)
comment - evaluations with a comment containing the specified string (optional)
include_deleted - whether or not to include inactive (deleted) evaluations (boolean, default false)
limit - the number of evaluations to return (default 10, max 10000)
logger_file - evaluations where the voice log filename contains the specified string (optional)
most_recent - in conjunction with limit, whether to fetch the most recent or least recent evaluations (boolean, default true)
reviewer_id - evaluations completed by the the reviewer with the specified numeric ID (optional)
scored_after - evaluations completed after the specified Unix timestamp (optional)
scored_before - evaluations completed before the specified Unix timestamp (optional)
taken_after - evaluations for calls taken after the specified Unix timestamp (optional)
taken_before - evaluations for calls taken before the specified Unix timestamp (optional)
team_id - evaluations for the team with the specified numeric ID (optional)
total_score_greater_than - evaluations with a score higher than the specified value (0-100) (optional)
total_score_less_than - evaluations with a score lower than the specified value (0-100) (optional)

Resource Actions

A listing of all the available resources including parameters and example return values.

getScores

Example Query String: ?action=getScores&total_score_less_than=80&limit=2
Return Value:


```
{
    “1243”: {
        “score_id”: 1243,
        “total_score”: 68,
        “agent_id”: 10,
        “reviewer_id”: 5,
        “team_id”: 102,
        “time_taken”: 1312819440,
        “time_scored”: 1313075966,
        “client_id”: “585”,
        “logger_file”: null,
        “comment”: null,
        “detail”: null,
        “status”: “active”
    },
    “1245”: {
        “score_id”: 1245,
        “total_score”: 72,
        “agent_id”: 11,
        “reviewer_id”: 5,
        “team_id”: 108,
        “time_taken”: 1312821360,
        “time_scored”: 1313076381,
        “client_id”: “864”,
        “logger_file”: null,
        “comment”: null,
        “detail”: null,
        “status”: “active”
    }
}
```


Notes: See ‘getScore’ for more detail on individual properties of the score object. Note that all parameters are optional, if your query string includes no parameters (‘?action=getScores’) by default the 10 most recent active (not deleted) evaluations will be retrieved.

Additional Criteria Examples - Cloneable

This category is notable because of its 'cloneable' type - cloneable criteria can be duplicated and graded multiple times on the same evaluation, for events that can happen multiple times and need to be evaluated separately each time (like agent holds). Also note it contains an 'inactive' criterion - inactive (deleted) criteria are not displayed on the user's scoring form, but are retained to associate with historic evaluation data.


```
...    {
            "category_id": 612,
            "weight": 40,
            "type": "cloneable",
            "description": "Agent Hold",
            "criteria": [
                {
                    "criterion_id": 4599,
                    "status": "active",
                    "weight": 10,
                    "description": "Agent's request to hold",
                    "options": [
                        "Unacceptable",
                        null,
                        null,
                        "Mediocre",
                        null,
                        "Good"
                    ],
                    "values": [
                        0,
                        1,
                        2,
                        3,
                        4,
                        5
                    ]
                },
                {
                    "criterion_id": 4357,
                    "status": "inactive",
                    "weight": 20,
                    "description": "Agent's return from hold was:",
                    "options": [
                        "Poor",
                        null,
                        null,
                        null,
                        null,
                        "Good"
                    ],
                    "values": [
                        0,
                        1,
                        2,
                        3,
                        4,
                        5
                    ]
                }
            ]
        }, ...
```

Additional Criteria Examples - Counter

Counter categories are most often used for criteria like pleases and thank yous, where each use is worth a number of points up to a possible maximum. Because of that, 'counter' category criteria don't have the usual numeric array of options and values - options is an empty array, values has two keys, 'each' and 'max'. For instance, In this example, each 'Please' is worth 2 points up to a maximum of 3.

```
...     {
            "category_id": 613,
            "weight": 70,
            "type": "counter",
            "description": "Courtesy",
            "criteria": [
                {
                    "criterion_id": 4358,
                    "status": "active",
                    "weight": 10,
                    "description": "Please",
                    "options": [],
                    "values": {
                        "each": 2,
                        "max": 3
                    }
                },
                {
                    "criterion_id": 4359,
                    "status": "active",
                    "weight": 20,
                    "description": "Thank You",
                    "options": [],
                    "values": {
                        "each": 2,
                        "max": 2
                    }
                },
                {
                    "criterion_id": 4367,
                    "status": "active",
                    "weight": 30,
                    "description": "Caller's Name",
                    "options": [],
                    "values": {
                        "each": 3,
                        "max": 5
                    }
                }
            ]
        }, ...
```

Correlating Criteria and Score Detail Objects

This example illustrates how the detail object in a score from getScore or getScores correlates with the criterion object from getCriterion or getCriteria. We can see the array for key '4597' in the score detail object has multiple values, which means the criterion with ID 4597 must be in a cloneable category. If we retrieve the criterion with the numeric ID 4597, we see its description is 'Agent's request to hold'. The two values given in the score detail object for this criteria are 5 and 3, which are array indexes for the 'options' and 'values' arrays in the criterion object - in this case we can see index 3 would correspond with 'Mediocre' and a point value of 5, and index 5 would correspond with 'Good' and a point value of 10. Since the point value at index 5 is '10', we know this criteria is worth a possible 10 points, and since it was applicable twice, we can calculate that for this criterion on this call, this agent (with numeric ID 12) scored 15 of a possible 20 points (10 of 10 for the first request to hold, and 5 of 10 for the second).


```
{
    "score_id": 1258,
    "total_score": 86,
    "agent_id": 12,
    "reviewer_id": 8,
    "team_id": 0,
    "time_taken": 1383771660,
    "time_scored": 1383772034,
    "client_id": "7314",
    "logger_file": "//LAN/path/to/file.mp3",
    "comment": "Lorem ipsum dolor ...",
    "detail": {
        "214": [
            "2"
        ],
        "1178": [
            "5"
        ],
        "4597": [
            "5",
            "3"
        ],
        "4358": [
            "2"
        ]
    },
    "status": "active"
}
```

```
{
    "criterion_id": 4597,
    "status": "active",
    "weight": 10,
    "description": "Agent's request to hold",
    "options": [
        "Unacceptable",
        null,
        null,
        "Mediocre",
        null,
        "Good"
    ],
    "values": [
        0,
        1,
        2,
        5,
        8,
        10
    ]
}
```

Code Examples - PHP

The following example code in PHP 5 will obtain the 5 most recent evaluations with a score of 90% or higher.

```php
<?php
$api_endpoint = 'https://callscores.com/lib'; // Base URL of API Endpoint
$auth = array( // Contains the user's numeric ID and API security token
  'id' => '000',
  'token' => 'U$3R%T0K3N%STR1NG%U$3R%T0K3N%STR1NG%U$3R%T0K3N%STR1NG%'
);
$action = 'getScores'; // The action we'd like to perform
$params = array( // Any additional parameters
  'total_score_greater_than' => '90',
  'limit' => '5'
);

// Generate the URL for our request based on the desired action
// and provided parameters
$request_url = "$api_endpoint/?action=$action&".http_build_query($params);

// Generate the HTTP headers and POST request containing the
// authentication data
$post_data = array(
  'http' => array(
    'header'  => "Content-type: application/x-www-form-urlencoded\r\n",
    'method'  => 'POST',
    'content' => http_build_query($auth)
  )
);

// Send the request to the API server
$context = stream_context_create($post_data);
$json = file_get_contents($request_url, false, $context);

// Convert the response from JSON to a PHP object
$result = json_decode($json);

// $result is now a stdClass object containing the requested data
?>
```

Code Examples - jQuery

The following example code in jQuery 3 will obtain the 5 most recent evaluations with a score of 90% or higher.

```
$(function(){
  var api_endpoint = 'https://callscores.com/lib'; // API Endpoint Base URL
  var auth = { // Contains the user's numeric ID and API security token
    id:'000',
    token:'U$3R%T0K3N%STR1NG%U$3R%T0K3N%STR1NG%U$3R%T0K3N%STR1NG%'
  }
  var action = 'getScores'; // The action we'd like to perform
  var params = { // Any additional parameters
    total_score_greater_than:'90',
    limit:'5'
  }

  // Generate the URL for our request based on the desired action
  //and provided parameters
  var request_url = api_endpoint+'/?action='+action+'&'+$.param(params);

  // Initiate an AJAX request
  $.ajax({
    url: request_url,
    type: 'post',
    contentType: 'application/x-www-form-urlencoded',
    data: $.param(auth),
    success: function(data,status,jQxhr){
      // var data is now an object containing the requested data
      console.dir(data);
    }
  });
});
```