Infinite Talent BrassRing

BrassRing Object API

Get Candidate

Version: 1.0.1
Date: October 10, 2023

Table of Contents

INTRODUCTION 2

1.1.1 PURPOSE 2
1.1.2 AUDIENCE 2
1.1.3 PRIVILEGES 2
1.1.4 TERMS USED IN THIS GUIDE 2
1.1.5 HOW DO OBJECT APIS IN THE BRASSRING WORK? 3
1.1.6 USING XML/JSON 4
1.1.7 BRASSRING API LIBRARY – SUPPORTED API CALLS 6
1.1.8 ORDER OF OBJECT DURING REQUESTS AND RESPONSES 6
1.1.9 EXECUTING THE GETCANDIDATE API CALL 6
1.1.10 USE CASE 8
1.1.11 GETCANDIDATE USE CASE HR STATUS WORKFLOW CONFIGURATION 9
1.1.12 CONFIGURING WORKBENCH SUBSCRIPTIONS 14
1.1.13 GETCANDIDATE REQUEST (CANDIDATE OBJECT) 15

2 XML/JSON ELEMENT TABLE FOR BRASSRING API LIBRARY 30

3 ERROR CODES 32

4 LANGUAGE/SITE/LOCALEIDS 34

Introduction

This document provides information about Infinite BrassRing web service Application Programming Interfaces (APIs) for the BrassRing Object Oriented API workflow developed to provide a standardized workflow for the exchange of information between the Infinite BrassRing® (BR) application and client applications.

1.1.1 Purpose

The purpose of this document is to present an overview of the BrassRing GetCandidate Object Oriented API. These APIs can be used to integrate with third-party vendors. This API integration allows BrassRing clients to export their configuration and candidate data into Human Resource Information Systems (HRIS) or other 3rd party systems.

This document provides:

- An overview of BrassRing's GetCandidate API.
- An overview of the supported workflow process, including information exchange with any external, third-party system that can send and receive XML/JSON requests and responses.
- A procedure for consuming the GetCandidate API and xml/json schemas.

1.1.2 Audience

The intended audience for this document is:

- Engineering Services Team, Support Team, System Integrators, Technical Services Group, and Vendors.

1.1.3 Privileges

Power Users, Partner Users, Tech Services, and Super Users can access Workbench Subscription configuration to restrict the objects returned in the API call. Your Support Team can assist you with assigning your privileged users.

1.1.4 Terms Used in this Guide

The following terms are used in this guide. Some terms have been abbreviated for comprehension. Abbreviated terms are identified once.

API	Application Programming Interface
XML	extensible Markup Language
Clients	third-party vendors or client Human Resource Information Systems (HRIS)

What is the BrassRing Object API?

The BrassRing Object-Oriented (OO) API exchanges real-time data between the BrassRing application and clients, vendors, and client Human Resource Information Systems. For the purposes of this document, the term client is intended to be used interchangeably with vendors.

The BrassRing API supports an OO workflow allowing clients to retrieve data objects associated with their BrassRing accounts and configured in BrassRing Workbench. The OO API workflow allows for data categorization of objects and their associated attributes. For example, when the object-oriented API requests the Candidates

data object and no other data subsets are noted, the API response includes all the Candidate data attributes that include email address, candidate full name, status (Active or Inactive) and date of last edit. See GetCandidate Response. (Candidate Objects).

The OO workflow can also be restricted and thereby limited to a subset of the retrieved object data. For example, if a client wants to request a candidate's HR Status alone, the client can restrict the OO response to return only those objects and attributes required. Workbench configuration permits clients to limit (restrict) the scope of the data (objects) returned in the OO response. For more information, please see Configuring Workbench Subscriptions and Restrictions of Objects.

1.1.5 How Do Object APIs in the BrassRing Work?

BrassRing APIs use eXtensible Markup Language (XML/JSON) messages to communicate. XML/JSON messages contain a defined set of elements for each specific API function. At a high level, these XML/JSON elements are the content containers and can contain configuration instructions that indicate, among other things, whether the XML/JSON message is a request or a response. At a more detailed level, the elements in the XML/JSON instruct the application on exactly what values to retrieve and display, and can contain authentication codes, encryption values and other necessary information. Refer to Using XML/JSON for guidelines.

Because the BrassRing API uses object-oriented APIs, an API call can retrieve an "object" and all of its associated object attributes from the BrassRing application. If filtering restrictions are configured in Workbench, the API call returns only a subset of the object's attributes.

For example, when the API call GetCandidate initiates a request for an object named "Req Templates," the API responds and returns the object, Req Templates, and all of that object's attributes. These object attributes are common BrassRing Workbench attributes. For more information on object filters and order rules, refer to Order of Objects during Requests and Responses.

- Restriction of Objects

When the API calls executes, the GetCandidate request can return up to 10,000 objects and their attributes. Clients have the option to restrict the object types and filter the object's attributes returned in the API response. Clients can use Workbench Subscriptions to restrict data objects and the filters to restrict object attributes.

For example, if a client wanted to export only candidates of one requisition or one HR status, they can select these filters in Workbench. If a HR status or requisition id is not sent in the request by client/vendor 3rd party system, the export will fail.

Clients can also determine what fields they want to export by selecting them in the output fields. For example, if they want to export only the candidates' contact details, they can select those fields in the output fields.

For more information, please see Configuring Workbench Subscriptions.

- BrassRing Object Oriented APIs

Each of the BrassRing API calls can request and retrieve up to 10,000 objects including object attributes in one API call. If an API response needs to return more than 10,000 objects (in one API call) to fulfill the API request, the API creates a logical break point between objects at an object attribute level and returns a Globally Unique Identifier (GUID) with the first set of objects. For example, the following is a list of Candidate attributes:

Candidate Attributes

- Resume Key
- Email Address
- Candidate Full Name
- Status (Active or Inactive)
- Date of Last Edit

If a user requests the Candidate object and the 10,000 limit occurs within the Candidate "Type" attribute, the API response creates a logical breakpoint at the Candidate Status attribute level and returns a Globally Unique Identifier (GUID) and all objects and their attributes up to the end of the Global/Member attribute. The subsequent API request containing the GUID identifies where the API response should begin so that the API response begins its response at the "Type" attribute and then returns the remaining Status and Date of Last Edit attributes up to 10,000 items.

1.1.6 Using XML/JSON

XML document are structured documents containing, among other things, elements. Each element contains an instruction that supports the processing of API requests or responses. XML documents and elements must be written in a defined (valid) format. Therefore, when using BrassRing APIs you must adhere to all syntax guidelines regarding the XML document format. If your XML is not valid, the following

- XML Syntax Format Guidelines

The following XML syntax guidelines must be followed when using BrassRing APIs.

- All XML must have closing tags.
- XML markup tags are case sensitive.
- XML elements must be properly nested.
- XML attributes must be quoted.
- XML must use encoding if XML contains special characters ($quot; represents a quotation mark) if you are not using CDATA.
- XML comments must be enclosed with <! Comment -->
- XML element naming conventions:
  - Names cannot start with a number or punctuation character.
  - Names cannot start with the letters xml (or XML, or Xml, etc.)
  - Names cannot contain spaces.

* Start and End XML tags must match in case sensitivity. You cannot have a Start tag that begins with an upper-case letter and an Ending tag that begins with a lower case letter. However, some applications you may use with XML that do have more restrictive case sensitivity. For example, XHTML requires lower case markup tags. Recommended practice is to check naming conventions of other applications you intend to use with XML. Using all lowercase for XML tags is the most universally accepted format.

- Special Conditions

If transmitting an element field or form field that is a multi-select or a checkbox or contains items in a series, multiple values can be sent in a single node with ~|~ delimiter.

For example, <FormInput fieldid="12584"><![CDATA[chk1~|~chk2]]></FormInput> or

<FormInput fieldid="12584">chk1~|~chk2</FormInput>.

- Encoding Standard

XML version 1.0, encoding UTF-8 is the encoding specification used in BrassRing XML documents.

- CDATA Input

The following CDATA syntax guidelines must be followed when using BrassRing APIs.

- All CDATA input must start with "**<![CDATA[**" and ends with "**]]>**"
- CDATA input items cannot be nested.
- Separate list items within CDATA using commas or ~|~

JSON

JSON (JavaScript Object Notation) is a text-based data exchange format.
It is a collection of key-value pairs where the key must be a string type, and the value can be of any of the following types:

Number, String, Boolean, Array, Object, null

- JSON Syntax Format Guidelines

  The following JSON syntax guidelines must be followed when using BrassRing APIs.

- In the JSON data format, the keys must be enclosed in double quotes.
- The key and value must be separated by a colon (:) symbol.
- There can be multiple key-value pairs. Two key-value pairs must be separated by a comma (,) symbol.
- No comments (// or /* */) are allowed in JSON data.

  For example "FormInput": {"fieldid": "12584", "value": "chk1~|~chk2" }

- Security

The BrassRing API Library uses a token-based authentication system. Users logging into the BrassRing application, must supply their User ID, Password, and Manifest name in the API call XML/JSON request. When the BrassRing application receives this information, it generates an authentication token that encrypts the received information, the GUID, and the Current UTC Datetime and returns the encrypted information in an XML/JSON element named Security. This XML/JSON Security element is then used to authenticate users during the API call process. If users do not submit the Security element, the system reverts to standard Username/Password mode for authentication.

- Certificate based encryption.

The BrassRing Object Oriented API provides certificate-based encryption. When vendors submit API calls to the ATS must contain the User ID, Password, and Manifest name. BrassRing then:

1. Uses the Manifest name & User ID, the Xpath's of the Nodes to be encrypted & signed and reads it from the XML/JSON in the Keymaster table's SecurityXpath column.
2. Once read, the client certificate public key encrypts all the output data in the nodes.
3. The Vendor receives the certificate private key (Infinite Talent's Key) and uses it to digitally sign the nodes. If those nodes do not exist, they are created.

The client can then use the digital signature to decrypt the output data.

1.1.7 BrassRing API Library – Supported API Calls

The GetCandidate API uses the HTTP Post method. During Workbench API configuration, clients can designate which fields the API calls requests and returns.

Technical Details on this approach can be found on the following site: http://msdn.microsoft.com/en-us/library/debx8sh9%28v=vs.110%29.aspx

Each GetCandidate API web service call uses XML (parameters) to initiate requests and receive responses. XML/JSON parameters specify what is being requested and what response is expected.

The supported API call is:

- GetCandidate

During Workbench API configuration, clients can designate which fields the API calls re quests and returns.

1.1.8 Order of Object during Requests and Responses

The business rule for the order of selection of objects is as follows:

- If identifier filters exist, they take the preference as they directly relate to the requested object. The system retrieves these objects first. If there is main object filter as well which was supposed to be applied on the REQUESTED object (and not the child) the result will be filtered further here to get the REQUESTED OBJECTS which falls under both (MAIN AND IDENTIFIER) filters.
- If there is requested child for the object, then then the system retrieves the children for the objects received in the previous step.
- If there is only an identifier filter there for the Requested Object and Main filter needs to be applied to the child object, then the first step is to identify the Requested Object applied the Identifier filter.
- Then, the system retrieves the children of the Requested Object from the previous step and applies the MAIN Filter on them to get the final result.
- If there is no identifier filter on the Requested Object and only Main Filter applied to the Child object, then the system retrieves the children which fall under the Main Filter criteria and then gets the related parent.

1.1.9 Executing the GetCandidate API Call

The general procedure to execute the OO API call requires clients to:

1. Configure Workbench subscriptions to choose the outputable fields and mark required filters.
2. Prepare XML/JSON documents.
3. Access BrassRing portal URL.
4. Execute API call (GetCandidate).

5. Examine results of API Call Response

Configuring Workbench Subscriptions

Clients can configure subscriptions by selecting output fields and required filters in Workbench.

> When a client selects an outputable field for the Candidate Export and the selected field is configured as encrypted at the field level and the Allow Encrypted Fields is unchecked, it means that this field value is transmitted as a NON_ENCRYPTED value.

For additional information about subscriptions and restrictions, see Restriction of Objects.

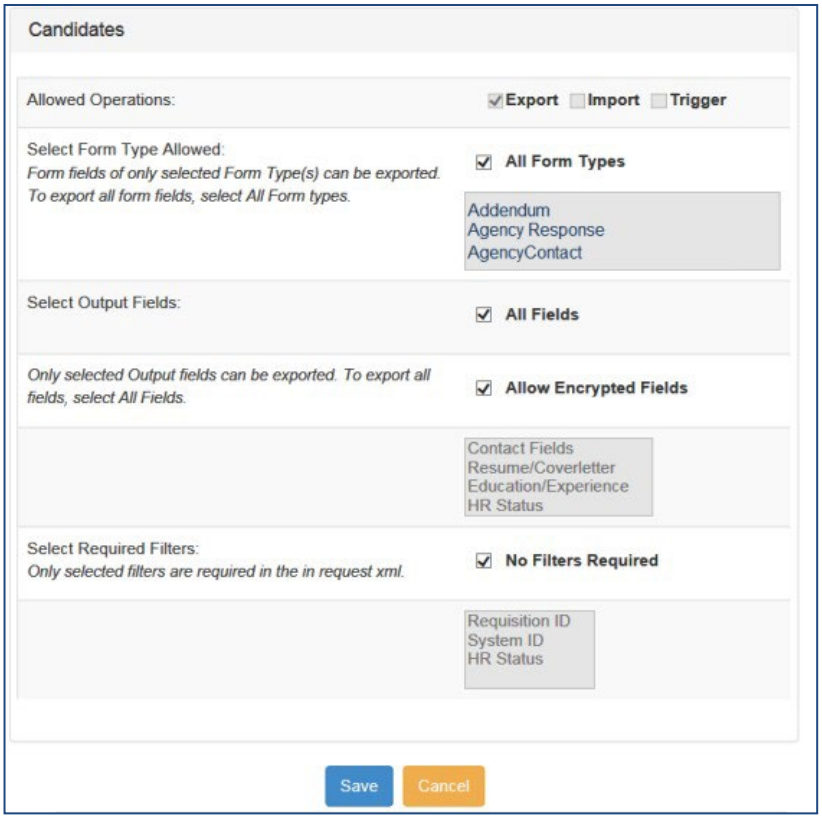
1.1.9.1.1 In Workbench:

While creating/editing a subscription, workbench users with appropriate permissions can see the below settings when the object type “Candidates” is selected.

These are applicable for only Get Candidate API.

1. **Tools > Integrations > Administration.**
2. Select the Subscription Admin Icon for **Object based Import/Export**.

   The integration type administration screen displays.
3. Select Candidate and then Select the **Edit Settings pencil** icon.
   The Edit Subscription Candidates dialog launches.



4. Scroll down and select the **Export** check box.

   There is no dependency between Form Types and Output Fields. For example, selecting a specific Form Type does not impact selectable output fields.

Select Form Types. You can select the check box for All Form Types or select forms from the list.

5. Select **Output Fields** in Select Output Fields category.

You can select the check box for All Fields or uncheck the All Fields checkbox and manually select each output field using the scroll bars to scroll within the Output Fields selection box.

> Output fields are encrypted by default if the field is configured as encrypted at the field level. To maintain encryption, user must select "Allow encrypted fields".

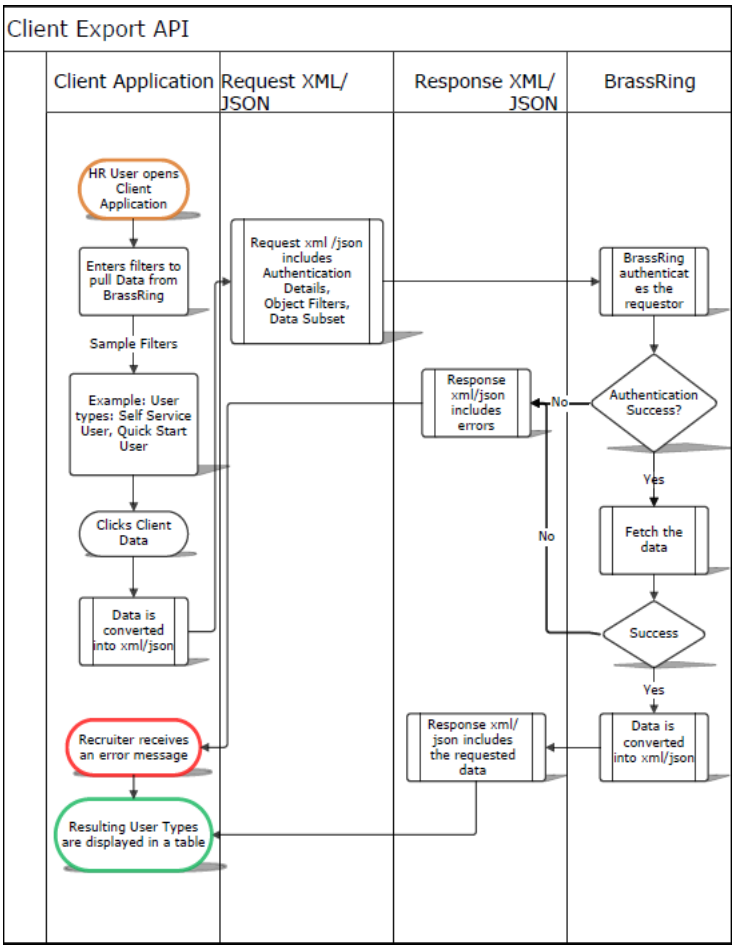
6. Select **Required Filters** in the Required Filters category.

You can select the check box for No Filters Required or uncheck the No Filters Required checkbox and manually select each output field using the scroll bars to scroll within the Output Fields selection box.

7. Select **Save.**

1.1.10 Use Case

This use case workflow diagram demonstrates a typical use case for extracting candidate details.



1.1.11 GetCandidate Use Case HR Status Workflow Configuration

This Use Case demonstrates how a client might configure a workflow to use the GetCandidate API to retrieve candidates with a specific HR status.

This example is just one way you can use subscription filters. For example, you can retrieve the current HR status of a single candidate by providing the candidate ID in the filter or you can retrieve the current HR statuses of all candidate applicants for a specific req by providing the requisition ID in the filter.

The general procedure to execute the OO API call requires clients to:

1. Configure Workbench subscriptions to chose the outputable fields and mark required filters.
2. Prepare XML/JSON documents.
3. Access BrassRing portal URL.
4. Execute API call (GetCandidate).
5. Examine results of API Call Response.

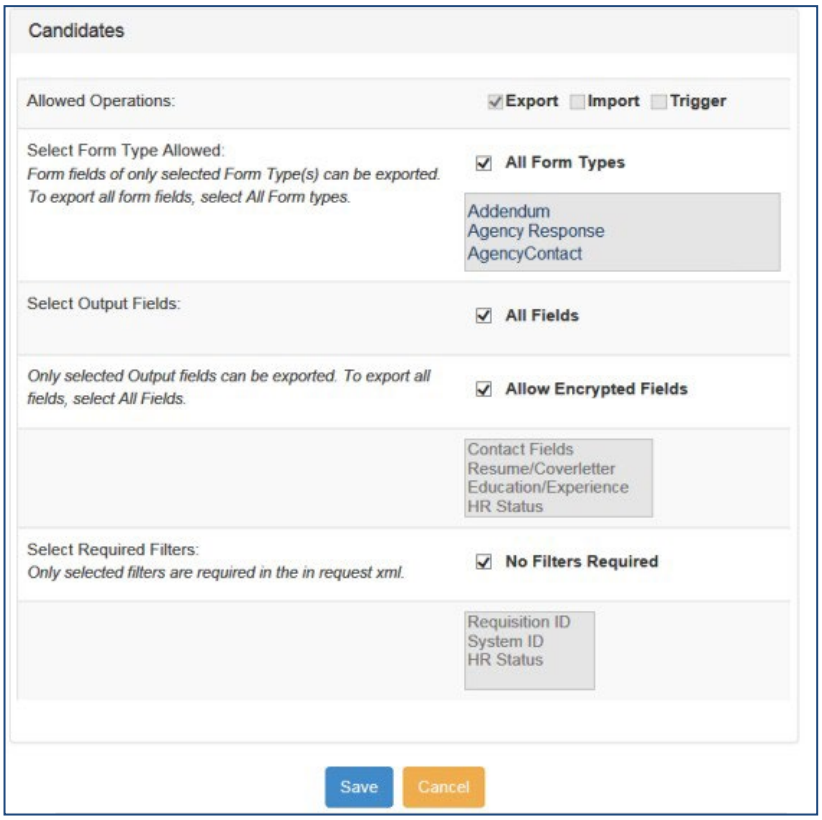
- Use Case – Step One - Configuring HR Status Candidate Export

In this scenario, clients configure the Workbench Subscription and configure the output fields and filters to retrieve candidates that match those criteria. Client logs into Workbench and navigates to:

1. **Tools > Integrations > Administration.**
2. Selects **the Subscription Admin** Icon for Object based Import/Export**.**

   The integration type administration screen displays.
3. Selects Candidate and then Selects the **Subscription Administration** icon.
   The Edit Subscription Candidates dialog launches.

Client scrolls to the Candidate section of the dialog.



4. Scroll down to the Candidate section and selects the **Export** check box.

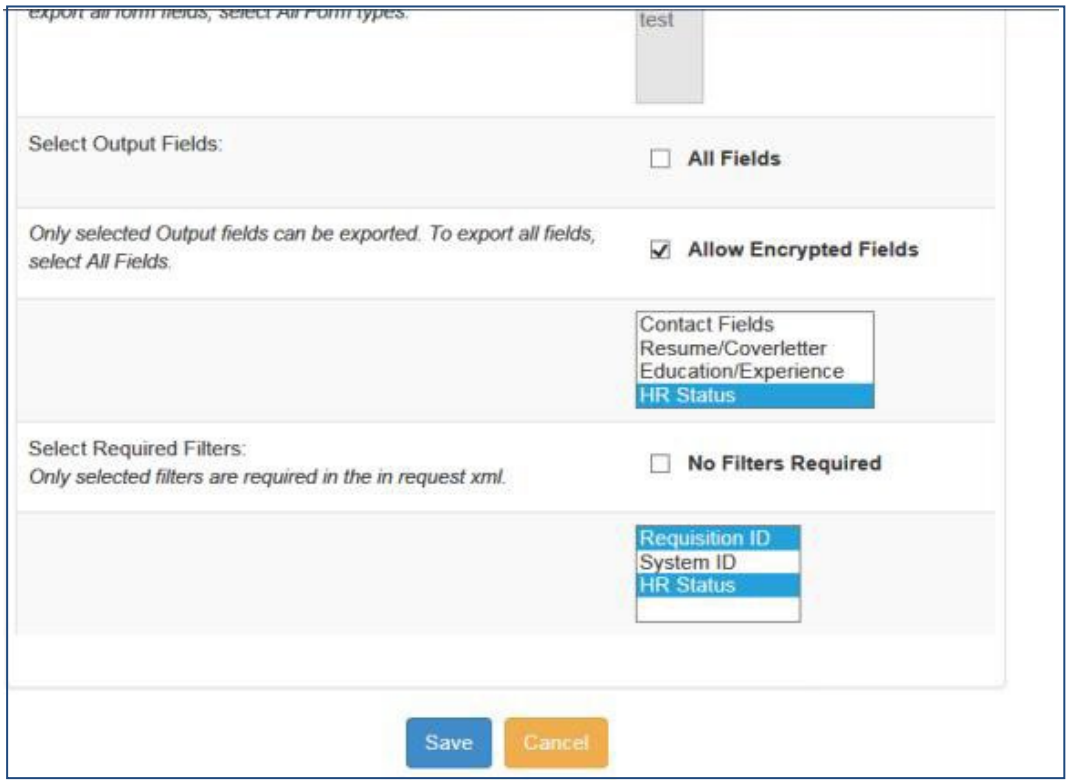
   There is no dependency between Form Types and Output Fields. For example, selecting a specific Form Type does not impact selectable output fields.

   Form Types. You can select the check box for All Form Types or select forms from the list.

5. Selects **Output Fields** in Select Output Fields category.
6. Unchecks the **All Fields** check box, leaves **Allow Encrypted Field** check box checked, and selects **HR Status** in the list box.

Output fields are encrypted by default if the field is configured as encrypted at the field level. To maintain encryption, user must select "Allow encrypted fields".

7. Unchecks the **All Filters** check box and selects **Requisition ID** and **HR Status** in the **Required Filters** list box.



8. Select **Save.**

- Use Case – Step Two

The client must now prepare the XML/JSON documents. In the XML/JSON document as shown, the client specifies the HR status (2) to be retrieved and the requisition ID (789342). The XML/JSON for these queries looks like this.

```
<FILTER TYPE="CANDIDATE_HR_STATUS">
    <VALUE>2</VALUE>
  </FILTER>
  <FILTER TYPE="REQUISITION_ID">
    <VALUE>789342</VALUE>
```

```
"FILTER": [
      {
        "VALUE": "2",
        "TYPE": "CANDIDATE_HR_STATUS"

      },
      {
        "VALUE": "789342",
        "TYPE": "REQUISITION_ID"
```

```
  }
 ]
```

This means the API call will request and return the GetCandidate Object and all candidates that match the filters and both the specified Req ID (789342) AND the HR status (2).

```
<?xml version="1.0" encoding="utf-8" ?>
<GET_DATA>
 <AUTHENTICATION>
  <CLIENT_ID>Client ID</CLIENT_ID>
  <INTEGRATION_USER>Integration User ID</INTEGRATION_USER>
  <REQUESTOR>The Requestor</REQUESTOR>
  <USERNAME>Username</USERNAME>
  <PASSWORD>Password</PASSWORD>
  <MANIFEST_NAME>"objectapi”</MANIFEST_NAME>
  <SECURITY_TOKEN />
 </AUTHENTICATION>
 <REQUEST>
  <OBJECT NAME="CANDIDATES">
   <CLIENT_OBJECT_IDENTIFIER>The value of the integration ID</CLIENT_OBJECT_IDENTIFIER>
   <ADDITIONAL_RECORDS_GUID></ADDITIONAL_RECORDS_GUID>
  </OBJECT>
  <MAIN_OBJECT_FILTERS>
<FILTER TYPE="CANDIDATE_HR_STATUS">
     <VALUE>2</VALUE>
   </FILTER>
   <FILTER TYPE="REQUISITION_ID">
    <VALUE>789342</VALUE>
   </FILTER>
  </IDENTIFIER_OBJECT_FILTERS>
  <OUTPUT>
   <DATA_SUBSET INCLUDE="CANDIDATE_HR_STATUS">TRUE </DATA_SUBSET>
  <DATA_SUBSET INCLUDE="CANDIDATE_ REQUISITION_ID >TRUE </DATA_SUBSET>
   </OUTPUT>
 </REQUEST>
</GET_DATA>
```

JSON:


```
{
  "GET_DATA": {
    "AUTHENTICATION": {
      "CLIENT_ID": "Client ID",
      "SECURITY_TOKEN": "",
      "REQUESTOR": "The Requestor",
      "MANIFEST_NAME": "objectapi",
      "INTEGRATION_USER": "Integration User ID",
      "USERNAME":"Username",
      "PASSWORD":"Password"
    },
    "REQUEST": {
      "IDENTIFIER_OBJECT_FILTERS": {},
      "OUTPUT": {
        "DATA_SUBSET": [
          {
            "INCLUDE": "CANDIDATE_HR_STATUS",
            "VALUE": true
          },
          {
            "INCLUDE": "CANDIDATE_ REQUISITION_ID",
            "VALUE": true
          }
        ]
      },
      "MAIN_OBJECT_FILTERS": {
        "FILTER": [
          {
            "TYPE": "CANDIDATE_HR_STATUS",
            "VALUE": 2
          },
          {
            "TYPE": "REQUISITION_ID",
            "VALUE": 2
          }
        ]
      },
      "OBJECT": {
        "ADDITIONAL_RECORDS_GUID": "",
        "CLIENT_OBJECT_IDENTIFIER": "The value of the integration ID",
        "NAME": "CANDIDATES"
      }
    }
  }
}
```

- Use Case – Step Three

Infinite will configure and provide an endpoint URL for the Client. This URL will be used to execute the GetCandidate API call. This URL is a 2-way SSL integration that will require a certificate handshake between Infinite and the Client.

- Use Case – Step Four

The GetCandidate API returns all Candidates that meet the GetCandidate filters AND the criteria specified in the XML/JSON. In this case, the GetCandidate API returns all candidates associated with the Req ID of (789342) AND the HR status (2).

1.1.12 Configuring Workbench Subscriptions

Clients can configure subscriptions by selecting output fields and required filters in Workbench.

When a client selects an outputable field for the Candidate Export and the selected field is configured as encrypted at the field level and the Allow Encrypted Fields is unchecked, it means that this field value is transmitted as a NON_ENCRYPTED value.

For additional information about subscriptions and restrictions, see Restriction of Objects.

1.1.12.1.1 In Workbench:

While creating/editing a subscription, workbench users with appropriate permissions can see the below settings when the object type “Candidates” is selected.

These are applicable only for Get Candidate API.

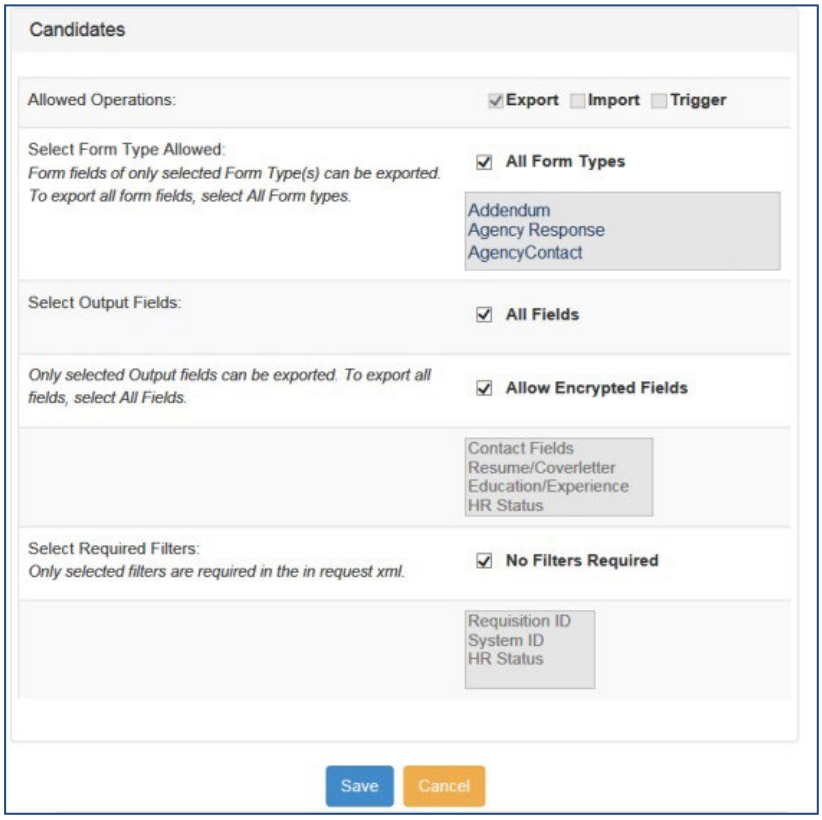
1.Tools > Integrations > Administration.

2. Select the Subscription Admin Icon for **Object based Import/Export**.

   The integration type administration screen displays.

3. Select Candidate and then Select the **Edit Settings pencil** icon.

4. The Edit Subscription Candidates dialog launches.



5. Scroll down and select the **Export** check box.

   There is no dependency between Form Types and Output Fields. For example, selecting a specific Form Type does not impact selectable output fields.

Select Form Types. You can select the check box for All Form Types or select forms from the list.

6. Select **Output Fields** in Select Output Fields category.

You can select the check box for All Fields or uncheck the All Fields checkbox and manually select each output field using the scroll bars to scroll within the Output Fields selection box.

> Output fields are encrypted by default if the field is configured as encrypted at the field level. To maintain encryption, user must not select "Allow encrypted fields".

7. Select **Required Filters** in the Required Filters category.
   You can select the check box for No Filters Required or uncheck the No Filters Required checkbox and manually select each output field using the scroll bars to scroll within the Output Fields selection box.
8. Select **Save.**

1.1.13 GetCandidate Request (Candidate Object)

In the GetCandidate Request, the request can be crafted to return a list of candidates, or a single candidate with varying degrees of data for each candidate depending on the filter criteria used.

1.1.13.1.1 *Request XML* API call: GetCandidate

```
<?xml version="1.0" encoding="utf-8" ?>
<GET_DATA>
 <AUTHENTICATION>
  <CLIENT_ID>Client ID</CLIENT_ID>
  <INTEGRATION_USER>Integration User ID</INTEGRATION_USER>
  <REQUESTOR>The Requestor</REQUESTOR>
  <USERNAME>Username</USERNAME>
  <PASSWORD>Password</PASSWORD>
  <MANIFEST_NAME>"objectapi”</MANIFEST_NAME>
  <SECURITY_TOKEN />
 </AUTHENTICATION>
 <REQUEST>
  <OBJECT NAME="CANDIDATES">
   <CLIENT_OBJECT_IDENTIFIER>The value of the integration ID</CLIENT_OBJECT_IDENTIFIER>
   <ADDITIONAL_RECORDS_GUID></ADDITIONAL_RECORDS_GUID>
  </OBJECT>
  <MAIN_OBJECT_FILTERS>
   <FILTER TYPE="DATETIME_FROM" FORMAT="YYYY-MM-DDThh:mm:ssTZD">Date following the format</FILTER>
<FILTER TYPE="DATETIME_TO" FORMAT="YYYY-MM-DDThh:mm:ssTZD">Date following the format</FILTER>
   <FILTER TYPE="INCLUDE_ACTIVE"> To include the element must be present with the value of “TRUE” </FILTER>
   <FILTER TYPE="INCLUDE_INACTIVE">To include the element must be present with the value of “TRUE”</FILTER>
  </MAIN_OBJECT_FILTERS>
  <IDENTIFIER_OBJECT_FILTERS>
   <FILTER TYPE="SYSTEM_ID">
    <VALUE>Resume Key of a candidate</VALUE>
   </FILTER>
   <FILTER TYPE="CONFIGURED_ID">
    <VALUE>Email Address of a candidate</VALUE>
   </FILTER>
```

```
<FILTER TYPE="CANDIDATE_HR_STATUS">
    <VALUE>HR Status</VALUE>
   </FILTER>
   <FILTER TYPE="REQUISITION_ID">
    <VALUE>Requisition ID</VALUE>
   </FILTER>
   <FILTER TYPE="Candidate Reference_ID">
    <VALUE>Candidate Reference ID</VALUE>
   </FILTER>
   <FILTER TYPE="Language">
    <VALUE>Language Name</VALUE>
   </FILTER>
  </IDENTIFIER_OBJECT_FILTERS>
  <OUTPUT>
   <DATA_SUBSET INCLUDE="ADDITIONAL_DETAILS">TRUE or FALSE, returns full candidate profile</DATA_SUBSET>
   <DATA_SUBSET INCLUDE="CANDIDATE_FORMS_DATA">TRUE or FALSE, returns candidate forms data</DATA_SUBSET>
<DATA_SUBSET INCLUDE="CANDIDATE_HR_STATUS">TRUE of FALSE, returns candidate related HR statuses</DATA_SUBSET>
            <DATA_SUBSET INCLUDE="CANDIDATE_RESUME_AND_COVERLETTER">TRUE or FALSE, returns resume and cover letters
of candidates</DATA_SUBSET>
   <DATA_SUBSET INCLUDE="CANDIDATE_EDUCATION_EXPERIENCE">TRUE or FALSE, returns candidate education and
work experience</DATA_SUBSET>
   <DATA_SUBSET INCLUDE="CANDIDATE_ATTACHMENTS">TRUE or FALSE, returns all candidate attachements besides cover letters
and resumes</DATA_SUBSET>
  </OUTPUT>
 </REQUEST>
</GET_DATA
      >
```

JSON


```
        {
  "GET_DATA": {
    "AUTHENTICATION": {
      "CLIENT_ID": "Client ID",
      "SECURITY_TOKEN": "",
      "REQUESTOR": "The Requestor",
      "MANIFEST_NAME": "objectapi",
      "INTEGRATION_USER": "Integration User ID",
      "USERNAME":"Username",
      "PASSWORD":"Password"
    },
    "REQUEST": {
      "IDENTIFIER_OBJECT_FILTERS": {
        "FILTER": [
            {
              "VALUE": "Resume Key of a candidate",
              "TYPE": "SYSTEM_ID"

            },
            {
              "VALUE": "Email Address of a candidate",
              "TYPE": "CONFIGURED_ID"

            },
            {
              "VALUE": "HR Status",
              "TYPE": "CANDIDATE_HR_STATUS"

            },
            {
              "VALUE": "Requisition ID",
              "TYPE": "REQUISITION_ID"

            },
            {
              "VALUE": "Candidate Reference ID",
              "TYPE": "Candidate Reference_ID"

            },
            {
              "VALUE": "Language Name",
              "TYPE": "Language"

            }
```

```
    ]
  },
  "OUTPUT": {
    "DATA_SUBSET": [
      {
        "INCLUDE": "ADDITIONAL_DETAILS",
        "VALUE": TRUE or FALSE, returns full candidate profile
      },
      {
        "INCLUDE": "CANDIDATE_FORMS_DATA",
        "VALUE": TRUE or FALSE, returns candidate forms data
      },
      {
        "INCLUDE": "CANDIDATE_HR_STATUS",
        "VALUE": TRUE of FALSE, returns candidate related HR statuses
      },
      {
        "INCLUDE": "CANDIDATE_RESUME_AND_COVERLETTER",
        "VALUE": TRUE or FALSE, returns resume and cover letters of candidates
      },
      {
        "INCLUDE": "CANDIDATE_EDUCATION_EXPERIENCE",
        "VALUE": TRUE or FALSE, returns candidate education and work experience
      },
      {
        "INCLUDE": "CANDIDATE_ATTACHMENTS",
        "VALUE": TRUE or FALSE, returns all candidate attachements besides cover letters and resumes
      }
    ]
  },
  "MAIN_OBJECT_FILTERS": {
    "FILTER": [
      {
        "TYPE": "DATETIME_FROM",
        "FORMAT": "YYYY-MM-DDThh:mm:ssTZD"
        "VALUE": "Date following the format"
      },
      {
        "TYPE": "DATETIME_TO",
        "FORMAT": "YYYY-MM-DDThh:mm:ssTZD"
        "VALUE": "Date following the format"
      },
      {
        "TYPE": "INCLUDE_ACTIVE",
        "VALUE": "To include the element must be present with the value of "TRUE""
      },
```

```
            {
              "TYPE": "INCLUDE_INACTIVE",
              "VALUE": "To include the element must be present with the value of "TRUE""
            }
          ]
        },
        "OBJECT": {
          "ADDITIONAL_RECORDS_GUID": "",
          "CLIENT_OBJECT_IDENTIFIER": "The value of the integration ID",
          "NAME": "CANDIDATES"
        }
      }
    }
}
```

- GetCandidate Response (Candidate Object)

The response data set varies based on the requested object filters. Returned fields also vary based on the output data subsets in the request.

- If the no additional data subsets are selected for each candidate returned will be:
    - Resume Key
    - Email Address
    - Candidate Full Name
    - Status (Active or Inactive)
    - Date of Last Edit
- If the ADDITIONAL_DETAILS data set is selected the following additional fields will show up for each candidate.
    - Address
    - City
    - Zipcode
    - State
    - Country
    - Home Phone
    - Cell Phone
    - Work Phone
    - Fax
    - Homepage
    - BRUID
    - Candidate Type
- If the CANDIDATE_HR_STATUS data set is selected the following additional fields will show up for each candidate and all of their associated job applications.
    - Job Requisition Number
    - HR Status
- If the CANDIDATE_RESUME_AND_COVERLETTER data set is selected the following additional fields will show up for each candidate and all of their associated job applications.
    - Resume Text
    - Cover Letter Text
- If the CANDIDATE_FORMS_DATA data set is selected the following additional fields will show up for each candidate form.
    - Candidate Form ID
    - Candidate Form Name
    - Talent Gateway ID
    - Status (Active or Inactive)
    - Form Type

- Requisition ID
- Questions
- Answers

- If the CANDIDATE_EDUCATION_EXPERIENCE data set is selected the following additional fields will show up for each candidate.
  - For each Education item
    - School Name
    - Degree
    - GPA
    - Major
    - Most Recent Indicator
    - End Date
  - For each Work Experience Item
    - Employer Name
    - Job Description
    - Start Date
    - End Date
    - Most recent Indicator

Response XML

```
<?xml version="1.0" encoding="utf-8" ?>
```

```
<RESULTS>
 <CLIENT_ID>11721</CLIENT_ID>
 <REQUESTOR>Test</REQUESTOR>
 <OBJECT NAME="CANDIDATES">
  <CLIENT_OBJECT_IDENTIFIER> The value of the integration ID </CLIENT_OBJECT_IDENTIFIER>
  <ADDITIONAL_RECORDS_GUID></ADDITIONAL_RECORDS_GUID>
  <INSTANCE>
   <SYSTEM_ID>Candidate Resume Key</SYSTEM_ID>
   <CONFIGURED_ID>Candidate Email Address</CONFIGURED_ID>
   <DESCRIPTION>
    <VALUE LANGUAGE="EN">Candidate Full Name (last, first, middle initial)</VALUE>
   </DESCRIPTION>
   <STATUS>Active or Inactive</STATUS>
   <MODIFIED_DATE>Date of Last Edit</MODIFIED_DATE>
   <STANDARD_DETAILS TYPE="ADDRESS">Candidate Street Address</STANDARD_DETAILS>
   <STANDARD_DETAILS TYPE="CITY">Candidate City</STANDARD_DETAILS>
   <STANDARD_DETAILS TYPE="ZIPCODE">Candidate Zip Code</STANDARD_DETAILS>
   <STANDARD_DETAILS TYPE="STATE">Candidate State</STANDARD_DETAILS>
   <STANDARD_DETAILS TYPE="COUNTRY">Candidate Country</STANDARD_DETAILS>
   <STANDARD_DETAILS TYPE="HOMEPHONE">Candidate Home Phone #</STANDARD_DETAILS>
   <STANDARD_DETAILS TYPE="CELLPHONE">Candidate Cell Phone #</STANDARD_DETAILS>
   <STANDARD_DETAILS TYPE="WORKPHONE">Candidate Work Phone #</STANDARD_DETAILS>
   <STANDARD_DETAILS TYPE="FAX">Candidate Fax #</STANDARD_DETAILS>
   <STANDARD_DETAILS TYPE="HOMEPAGE">Candidate Homepage</STANDARD_DETAILS>
   <STANDARD_DETAILS TYPE="BRUID">Candidate BRUID</STANDARD_DETAILS>
   <STANDARD_DETAILS TYPE="CANDIDATE_TYPE">Candidate Type (internal or
   external)</STANDARD_DETAILS>
   <STANDARD_DETAILS TYPE="REQUISITION_STATUS">
    <VALUE TYPE="HR_STATUS">Candidate HR Status related to this job</VALUE>
    <VALUE TYPE="REQUITION_ID">Job Requisition ID</VALUE>
   </STANDARD_DETAILS>
   <STANDARD_DETAILS TYPE="EDUCATION">
    <VALUE TYPE="SCHOOL_NAME">Name of School</VALUE>
    <VALUE TYPE="DEGREE">Type of Degree</VALUE>
    <VALUE TYPE="MAJOR">Major</VALUE>
    <VALUE TYPE="GPA">GPA</VALUE>
    <VALUE TYPE="ENDING_YEAR">Year of Graduation</VALUE>
    <VALUE TYPE="MOST_RECENT">Is this the most recent</VALUE>
   <STANDARD_DETAILS TYPE="EXPERIENCE">
    <VALUE TYPE="EMPLOYER_NAME">employer Name</VALUE>
```

```
<VALUE TYPE="POSITION_DESCRIPTION">Job Description</VALUE>
<VALUE TYPE="STARTING_YEAR">Year Started</VALUE>
<VALUE TYPE="ENDING_YEAR">Year Ended</VALUE>
<VALUE TYPE="MOST_RECENT">Is this the most recent</VALUE>
</STANDARD_DETAILS>
<STANDARD_DETAILS TYPE="RESUME_TEXT">Full Resume Text</STANDARD_DETAILS>
<STANDARD_DETAILS TYPE="COVERLETTER_TEXT"> Full Cover Letter Text</STANDARD_DETAILS>
<OBJECT NAME="CANDIDATE_FORMS_DATA">
<INSTANCE>
<SYSTEM_ID>Candidate Form ID</SYSTEM_ID>
<CONFIGURED_ID>Candidate Form Name</CONFIGURED_ID>
<DESCRIPTION>
<VALUE LANGUAGE="EN">Talent Gateway ID</VALUE>
</DESCRIPTION>
<STATUS>Active or in Inactive</STATUS>
<MODIFIED_DATE>Last Date of Edit</MODIFIED_DATE>
<STANDARD_DETAILS TYPE="FORM_TYPE">Form Type</STANDARD_DETAILS>
<STANDARD_DETAILS TYPE="REQUITION_ID">Requisition ID</STANDARD_DETAILS>
<STANDARD_DETAILS TYPE="CANDIDATE_RESPONSE">
<VALUE TYPE="QUESTION" IDENTIFIER="QUESTION_DBFIELDNAME" PROPERTY="This will be field type. Values are Radio, CheckBox, SingleSelect, MultiSelect, Text, Grid, Auto-Fill, and Query Select">Question Name</VALUE>
<VALUE TYPE="RESPONSE" IDENTIFIER="No">Answer</VALUE>
</STANDARD_DETAILS>
</INSTANCE>
</OBJECT>
<OBJECT NAME="CANDIDATE_ATTACHMENTS">
<INSTANCE>
<SYSTEM_ID>File ID</SYSTEM_ID>
<CONFIGURED_ID>File Name</CONFIGURED_ID>
<DESCRIPTION>
<VALUE LANGUAGE="EN">locale specific file name</VALUE>
</DESCRIPTION>
<STATUS>Active or Inactive</STATUS>
<MODIFIED_DATE>Last date of edit</MODIFIED_DATE>
<STANDARD_DETAILS TYPE="FILE_CONTENT">File Content</STANDARD_DETAILS>
<STANDARD_DETAILS TYPE="ATTACHMENT_CATEGORY">Attachment Category</STANDARD_DETAILS>
</INSTANCE>
</OBJECT>
</RESULTS>
```

Response JSON


```
{
  "RESULTS": {
    "CLIENT_ID": "11721",
    "REQUESTOR": "Test",
    "OBJECT": [
      {
        "CLIENT_OBJECT_IDENTIFIER": "The value of the integration ID",
        "ADDITIONAL_RECORDS_GUID": "",
        "INSTANCE": {
          "SYSTEM_ID": "Candidate Resume Key",
          "CONFIGURED_ID": "Candidate Email Address",
          "DESCRIPTION": {
            "VALUE": {
              "LANGUAGE": "EN",
              "content": "Candidate Full Name (last, first, middle initial)"
            }
          },
          "STATUS": "Active or Inactive",
          "MODIFIED_DATE": "Date of Last Edit",
          "STANDARD_DETAILS": [
            {
              "TYPE": "ADDRESS",
              "content": "Candidate Street Address"
            },
            {
              "TYPE": "CITY",
              "content": "Candidate City"
            },
            {
              "TYPE": "ZIPCODE",
              "content": "Candidate Zip Code"
            },
            {
              "TYPE": "STATE",
              "content": "Candidate State"
            },
            {
              "TYPE": "COUNTRY",
              "content": "Candidate Country"
            },
            {
              "TYPE": "HOMEPHONE",
```

```
        "content": "Candidate Home Phone #"
    },
    {
        "TYPE": "CELLPHONE",
        "content": "Candidate Cell Phone #"
    },
    {
        "TYPE": "WORKPHONE",
        "content": "Candidate Work Phone #"
    },
    {
        "TYPE": "FAX",
        "content": "Candidate Fax #"
    },
    {
        "TYPE": "HOMEPAGE",
        "content": "Candidate Homepage"
    },
    {
        "TYPE": "BRUID",
        "content": "Candidate BRUID"
    },
    {
        "TYPE": "CANDIDATE_TYPE",
        "content": "Candidate Type (internal or external)"
    },
    {
        "VALUE": [
            {
                "TYPE": "HR_STATUS",
                "content": "Candidate HR Status related to this job"
            },
            {
                "TYPE": "REQUITION_ID",
                "content": "Job Requisition ID"
            }
        ],
        "TYPE": "REQUISITION_STATUS"
    },
    {
        "VALUE": [
            {
                "TYPE": "SCHOOL_NAME",
                "content": "Name of School"
            },
```

```
            {
              "TYPE": "DEGREE",
              "content": "Type of Degree"
            },
            {
              "TYPE": "MAJOR",
              "content": "Major"
            },
            {
              "TYPE": "GPA",
              "content": "GPA"
            },
            {
              "TYPE": "ENDING_YEAR",
              "content": "Year of Graduation"
            },
            {
              "TYPE": "MOST_RECENT",
              "content": "Is this the most recent"
            }
          ],
          "TYPE": "EDUCATION"
        },
        {
          "VALUE": [
            {
              "TYPE": "EMPLOYER_NAME",
              "content": "employer Name"
            },
            {
              "TYPE": "POSITION_DESCRIPTION",
              "content": "Job Description"
            },
            {
              "TYPE": "STARTING_YEAR",
              "content": "Year Started"
            },
            {
              "TYPE": "ENDING_YEAR",
              "content": "Year Ended"
            },
            {
              "TYPE": "MOST_RECENT",
              "content": "Is this the most recent"
            }
```

```
                    ],
                    "TYPE": "EXPERIENCE"
                },
                {
                    "TYPE": "RESUME_TEXT",
                    "content": "Full Resume Text"
                },
                {
                    "TYPE": "COVERLETTER_TEXT",
                    "content": "Full Cover Letter Text"
                }
            ]
        },
        "NAME": "CANDIDATES"
    },
    {
        "INSTANCE": {
            "SYSTEM_ID": "Candidate Form ID",
            "CONFIGURED_ID": "Candidate Form Name",
            "DESCRIPTION": {
                "VALUE": {
                    "LANGUAGE": "EN",
                    "content": "Talent Gateway ID"
                }
            },
            "STATUS": "Active or in Inactive",
            "MODIFIED_DATE": "Last Date of Edit",
            "STANDARD_DETAILS": [
                {
                    "TYPE": "FORM_TYPE",
                    "content": "Form Type"
                },
                {
                    "TYPE": "REQUITION_ID",
                    "content": "Requisition ID"
                },
                {
                    "VALUE": [
                        {
                            "TYPE": "QUESTION",
                            "IDENTIFIER": "QUESTION_DBFIELDNAME",
                            "PROPERTY": "This will be field type. Values are Radio, CheckBox, SingleSelect,
MultiSelect, Text, Grid, Auto-Fill, and Query Select",
                            "content": "Question Name"
                        },
```

```
                        {
                            "TYPE": "RESPONSE",
                            "IDENTIFIER": "No",
                            "content": "Answer"
                        }
                    ],
                    "TYPE": "CANDIDATE_RESPONSE"
                }
            ]
        },
        "NAME": "CANDIDATE_FORMS_DATA"
    },
    {
        "INSTANCE": {
            "SYSTEM_ID": "File ID",
            "CONFIGURED_ID": "File Name",
            "DESCRIPTION": {
                "VALUE": {
                    "LANGUAGE": "EN",
                    "content": "locale specific file name"
                }
            },
            "STATUS": "Active or Inactive",
            "MODIFIED_DATE": "Last date of edit",
            "STANDARD_DETAILS": [
                {
                    "TYPE": "FILE_CONTENT",
                    "content": "File Content"
                },
                {
                    "TYPE": "ATTACHMENT_CATEGORY",
                    "content": "Attachment Category"
                }
            ]
        },
        "NAME": "CANDIDATE_ATTACHMENTS"
    }
]
}
}
```

2 XML/JSON Element Table for BrassRing API Library

Request/Response	XML Element Tag/JSON Keys	Required?	Functionality Details
	?xml version=”1.0”/encoding+”utf-8”?		Indicates version of xml and encoding type For more information see: http://www.w3.org/standards/xml/
Request	GET_DATA		The root node for retrieving the data
	CLIENT_ID	Yes	Client_ID identifies the client.
	INTEGRATION_USER	Yes	Any active BrassRing user of the client. If the API is being used by a Partner, the partner must get this from the mutual client. No password is needed for this.
	REQUESTOR	Yes	Ay text that can be used to identify the response for a request. Example Request1, Request2 etc for each request.
	USERNAME		Client(consumer of ths API) Application specific user name. BrassRing will randomly create one if this is not provided.
	PASSWORD		Client(consumer of ths API) Application specific password. BrassRing will randomly create one if this is not provided.
	MANIFEST_NAME		The value must always be “Objectapi”
	SECURITY_TOKEN		Cilents use their Security Token to validate themselves when using the APIs. It is generally returned to Client after a successful login, and is used when Clients make additional API requests.
	OBJECT		Object type from Workbench configuration
	CLIENT_OBJECT_IDENTIFIER		Object specific unique identifier
	ADDITIONAL_RECORDS_GUID		Batch id for more than 10,000 records
	MAIN_OBJECT_FILTERS		Object specific filders. Ex: For candidate object, the filters would be Candidate Name, HR Status, Applied On, etc.
	IDENTIFIER_OBJECT_FILTERS		
	OUTPUT		All the fields that we want to in the result.
	DATA_SUBSET		

Response	RESULT		
	CLIENT_ID		Client_ID identifies the client.
	REQUESTOR		Ay text that can be used to identify the response for a request. Example Request1, Request2 etc for each request.
	OBJECT		Object type from Workbench configuration
	CLIENT_OBJECT_IDENTIFIER		Object specific unique identifier
	ADDITIONAL_RECORDS_GUID		Batch id for more than 10,000 records
	INSTANCE		
	SYSTEM_ID		
	CONFIGURED_ID		
	DESCRIPTION		
	STATUS		
	MODIFIED_DATE		
	STANDARD_DETAILS		

3 Error Codes

Error Code	Error Description
18	AuthenticationToken node cannot be empty.
19	AuthenticationToken node is missing from input.
46	JobIDs node is missing from input.
47	JobIDs node is missing from input.
50	ResumeType node cannot be empty.
51	Invalid ResumeType specified in input.
52	ResumeType node is missing from input.
53	Resume node cannot be empty.
54	Resume node is missing in input.
55	ResumeName node cannot be empty.
56	ResumeName is missing from input.
57	FileExtension node cannot be empty when resumeType input is "Upload"
58	FileExtension node cannot be missing when resumeType input is "Upload"
61	ResumeID node cannot be empty.
62	ResumeID node is missing from input.
70	CoverLetterType node cannot be empty.
71	Invalid CoverLetterType specified in input.
72	CoverLetterType node is missing from input.
73	CoverLetter node cannot be empty.
74	CoverLetter node is mising from input.
75	CoverLetterName node cannot be empty.
76	CoverLetterName node is missing from input.
77	FileExtension node cannot be empty when CoverLetterType input is "Upload"
78	FileExtension node cannot be missing when CoverLetterType input is "Upload"
79	CoverLetterId node cannot be empty.

Error Code	Error Description
80	CoverLetterId node is missing from input.
230	The authentication status is not valid for this request.
430	Resume name is not unique.
431	Maximum number of resumes exceeded.
432	The Resume Id does not exist.
433	File Type is not supported.
434	File size exceeds 3 MB.
435	Cover Letter Name is not unique.
436	Maximum number of Cover Letters allowed exceeded.
437	The passed Resume is already configured as default resume.
438	An internal error has occurred.
439	FileExtension node cannot be empty when resumeType input is "Upload".
440	The Cover Letter Id does not exist.

4 Language/Site/LocaleIDs

This is a current table of Language, Site and Locale IDs.

Language	Language ID	Site Locale ID	ISO State Code (Not US State
Arabic	6	1025	AR
Azerbaijani	9	1068	AZ
Chinese	138	2052	ZH
Chinese (Traditional)	126	1028	TW
Croatian	44	1050	HR
Czech	20	1029	CS
Danish	22	1030	DA
Dutch	82	1043	NL
Dutch-informal	82	11043	NL-IN
English	1	1033	EN
English (Intl)	140	2057	GB
English RTL	144	21033	ER
English UK Government	145	12057	GG
English US Government	143	11033	GV
Finnish	31	1035	FI
French (Canada)	141	3084	FC
French (France)	34	1036	FR
German	23	1031	DE
German-informal	23	11031	DE-IN
Greek	25	1032	EL
Hebrew	42	1037	HE
Hungarian	45	1038	HU

Language	Language ID	Site Locale ID	ISO State Code (Not US State
Hungarian-informal	45	11038	HU-IN
Italian	52	1040	IT
Japanese	54	1041	JA
Korean	61	1042	KO
Norwegian	83	1044	NO
Polish	88	1045	PL
Portuguese (Brazil)	90	1046	PT
Portuguese (Portugal)	142	2070	PP
Romanian	94	1048	RO
Russian	95	1049	RU
Serbian	108	2074	SR
Slovak	102	1051	SK
Slovenian	103	1060	SL
Spanish	27	3082	ES
Spanish-informal	27	13082	ES-IN
Swedish	112	1053	SV
Turkish	123	1055	TR
Welsh	21	1106	CY