

# IBM Resilient



## Incident Response Platform

JSON REFERENCE GUIDE v1.0

---

Licensed Materials – Property of IBM

© Copyright IBM Corp. 2010, 2018. All Rights Reserved.

US Government Users Restricted Rights: Use, duplication or disclosure restricted by GSA ADP Schedule Contract with IBM Corp.

## Resilient Incident Response Platform JSON Reference Guide

Platform Version	Publication	Notes
1.01	June 2018	Updated company name and logos.
1.0	August 2015	Initial release.

Table of Contents

1. Overview ..... 5

2. Basics ..... 5

3. Data Types ..... 6

    3.1. Text ..... 6

    3.2. Numbers ..... 6

    3.3. Dates and Times ..... 6

    3.4. Boolean ..... 6

    3.5. Null ..... 6

    3.6. Lists ..... 6

4. Other values ..... 7

    4.1. Rich Text ..... 7

    4.2. Object Handles ..... 7

    4.3. Structured Values and Custom Fields ..... 8

5. Examples ..... 9

    5.1. Incident Template ..... 9

    5.2. Task (for creation) ..... 9

    5.3. Task (full) ..... 9



# 1. Overview

JavaScript Object Notation (JSON) is the native format for messages in the Resilient REST API and in the Actions Module. This document provides an outline of the JSON structures in the Resilient Systems platform, including incidents, tasks, and other objects, without going into specific details of the programming involved.

The [Customer Success Hub](#) contains the complete reference documentation on the REST API. This reference material includes details of each of the REST methods, their parameters and their return values. The documentation package is updated with each release of the Resilient platform.

Additionally, you can access the Github repositories containing examples that use the REST API and the Actions Module to perform common tasks. These examples are updated and expanded on a regular basis.

## 2. Basics

An incident is represented as a JSON document, with its various properties (fields), thus:

```
{
  "discovered_date": 1434029747498,
  "name": "Phishing emails"
}
```

The order of the fields in a JSON document is not important. However, some values are **lists**, and the order of items in a list is important.

Formatting and whitespace between the fields is not important. Quotation marks can be single-quote or double-quote, but must not be the “curly quotes” that word-processors like to use.

When you receive incident data from the Resilient platform, it is in the form of a JSON document with all the incident's properties. Some of these properties are for internal use and have no meaning in your application; however, you should retain these properties when sending an updated JSON document back to the platform.

When you create a new incident, you need only to specify the fields that are required. In a standard installation without any field customizations, the only two required fields are **name** and **discovered\_date**.

In the REST API documentation, these JSON documents are referred to as Data Transfer Objects (DTO) elements. The Java API includes a set of DTO classes that represent the same data structures. There are a number of these DTOs. An incident might be represented as an `incidentDTO`, or as a `fullIncidentDataDTO` (which includes more fields), or as a `partialIncidentDTO` (which includes fewer fields), depending on the context.

## 3. Data Types

The basic data types include text, numbers, dates and times, lists, and more complex values.

### 3.1. Text

There are two types of text field: plain text, such as the incident name; and text areas.

Text areas can be multi-line and also have rich-text values, as described in the [Rich Text](#) section.

JSON text values are quoted. To include a quote character within JSON text, it must be escaped with a backslash. To include a literal backslash, it too must be escaped with a backslash.

### 3.2. Numbers

Numeric fields in Resilient are integers. They do not support fractional values.

Note that fields such as dates and times, and object references are not numeric fields.

### 3.3. Dates and Times

All dates and times in Resilient are represented with a numeric value. This encodes the number of milliseconds since January 1<sup>st</sup> 1970, UTC.

Depending on how you access the REST APIs, you often need to convert these into a different representation for display, storage or processing. For example, the value 1439993716000 might be represented as "2015-08-19 14:15:16 UTC", "08/19/15 09:15:16 -0400" or "Wednesday".

### 3.4. Boolean

Boolean values are either **true** or **false**.

### 3.5. Null

Null values are **null**.

### 3.6. Lists

A list is a sequence of values. JSON lists are defined with square brackets. A list of numbers would be represented as [1,2,3]. A list of strings would be represented as ["one", "two", "three"].

## 4. Other values

Other types of value include object handles (references), which refer to a value that is defined elsewhere; and structured values, where the value has several components.

### 4.1. Rich Text

Rich text can include a subset of HTML to describe the text formatting. A rich-text value includes HTML tags such as `<div>` and `<em>`.

In versions of Resilient prior to v23, multi-line text areas only supported plain text. These values do not include HTML tags. Using the `text_content_output_format` query string parameter or HTTP header, you can ask the server to convert these into HTML, or convert HTML into plain text depending on your needs. For detail on the conversion rules, see **textContentOutputFormat** in the REST API documentation.

**NOTE:** If you update incident or other data (round-trip), it is best to keep text-area fields in their original format to avoid accidental updates that might cause loss of formatting or unnecessarily notify users of updates.

### 4.2. Object Handles

You will often encounter object handles in the Resilient REST API and Actions Module. The handle is a reference to an object that is defined elsewhere and can be referenced by ID. This allows the text of the object to change without having to go back and update every occurrence in the system. In database terms, object handles correspond to foreign keys.

For example, the `resolution_id` field has several valid values: "Unresolved", "Duplicate", "Not an Issue" and "Resolved". Each value has an ID, for example: 53, 54, 55, 56.

Internally, the incident stores a reference to the value using its ID. This value may appear in the incident JSON as the ID,

```
{ "resolution_id": 53 }
```

or as the string value,

```
{ "resolution_id": "Unresolved" }
```

or as the full object:

```
{ "resolution_id": { "id": 55, "name": "Unresolved" } }
```

Data is returned from the server with the ID values by default. If you wish to receive name values back for object handle fields from the server instead, you can set `handle_format=names` either in an HTTP header or a query string; for example:

```
https://example.mycompany.com/rest/orgs/:orgId/incidents?handle_format=names)
```

For a description of the possible values of the `handle_format` query string parameter/HTTP header, refer to **objectHandleFormat** in the REST API documentation.

When setting object handle values, the server works with either format. Numeric values (not quoted) are interpreted as IDs, and string values (surrounded by quotes) are resolved to the underlying IDs by the server. You can also specify the format by setting `handle_format=ids` in the query string or HTTP header.

## 4.3. Structured Values and Custom Fields

Some values have multiple parts. For example, the full incident DTO produced by the server includes information about the creator, which is represented as a field with a structured value containing the creator's name, id, and so on.

Custom fields in an incident are represented in a similar way, as values within a group "properties".

For example, if you have defined custom fields with API names "source\_types" (a multi-select field), "external\_case\_id" (a text field) and "cmdb\_info" (a hidden text field), the incident JSON might show:

```
{
  "id": 2269,
  "properties": {
    "source_types": [],
    "external_case_id": "INC00020478",
    "cmdb_info": null
  },
  "phase_id": 1005
  /* etc */
}
```



## 5. Examples

The Resilient API includes a Python utility “gadget” that lets you easily get and post JSON using the REST API. It includes several template examples of JSON document structures. Refer to those templates along with the examples below.

### 5.1. Incident Template

This incident template sets the required fields (name, discovered\_date) and other values. The discovered date is set to zero, which is interpreted as undefined. The incident type has multiple values, specified using text; these could also be specified by ID.

```
{
  "name" : "Default name",
  "description" : "Default description",
  "discovered_date": 0,
  "incident_type_ids" : ["Malware", "Phishing"],
  "crimestatus_id": 5,
  "exposure_type_id": "Unknown"
}
```

### 5.2. Task (for creation)

A new Task with name, phase, due date and status that can be added to an incident using POST /rest/orgs/:id/incident/:incid/tasks. The task does not have an ID until the server assigns one upon creation.

For a full description of the fields, refer to taskDTO in the REST API documentation.

```
{
  "name": "Revoke external accounts",
  "instr_text": "An ad-hoc task",
  "due_date": 1442610000000,
  "phase_id": "Respond",
  "status": "O"
}
```

### 5.3. Task (full)

A task retrieved by GET /rest/orgs/:id/tasks/:taskid?handle\_format=names includes additional information about the task and incident, HTML content for the instructions, and text values for object handles:

```
{
  "owner_fname": "Jernau",
  "creator": {
    "status": "A",
    "cell": null,
    "locked": false,
    "title": null,
    "notes": null,
    "email": "admin@example.com",
    "lname": "Sysadmin",
    "phone": null,
    "last_login": 1439943704668,
    "fname": "Resilient",
    "id": 2
  },
  "perms": {
```

```

        "comment": true,
        "read_attachments": true,
        "read": true,
        "change_members": true,
        "write": true,
        "attach_file": true,
        "delete_attachments": true,
        "close": true,
        "assign": true
    },
    "actions": [],
    "active": true,
    "init_date": 1439279145605,
    "id": 2253611,
    "description": "",
    "closed_date": null,
    "cat_name": "Engage",
    "custom": false,
    "rollup_id": 1000,
    "inc_training": false,
    "owner_id": "morat@example.com",
    "status": "O",
    "due_date": 1442610000000,
    "inc_name": "Phishing mails",
    "form": null,
    "regs": {
        "Security Incident Best Practices": "Security Incident Best Practices"
    },
    "members": null,
    "src_name": null,
    "name": "Interview key individuals",
    "frozen": false,
    "user_notes": null,
    "notes": [],
    "required": true,
    "inc_owner_id": "chiark@example.com",
    "instr_text": "Interview key individuals such as end users, system administrators and affected system/application owners. Determine:
    \n<ul>\n<li>Any recent unusual activities noted?</li>\n<li>Who has admin access to the affected systems involved?</li>\n<li>Any recent configuration changes or patches applied?</li>\n<li>Any recent suspicious emails or documents received and/or opened?</li>\n<li>Any recent suspicious web browsing activity?</li>\n<li>What logging is provided by the affected systems/applications?</li>\n<li>Do the affected systems contain any sensitive personal or confidential information?</li>\n</ul>\nAdd notes to this task, or attach interview notes, capturing all of your activities and findings.",
    "auto_task_id": null,
    "owner_lname": "Gurgeh",
    "inc_id": 2268,
    "category_id": null,
    "phase_id": "Engage"
}

```