



This PDF is generated from authoritative online content, and is provided for convenience only. This PDF cannot be used for legal purposes. For authoritative understanding of what is and is not supported, always use the online content. To copy code samples, always use the online content.

Web Services API Reference

Subresources

12/14/2025

Subresources

This is part of the [API Basics](#) section of the [Web Services API](#).

Contents

- [1 Subresources](#)
 - [1.1 Overview](#)
 - [1.2 Selecting Subresources](#)
 - [1.3 Filtering and Sorting by Subresource Properties](#)

Overview

Web Services includes a feature called subresources, which allows reading the subresources of an object.

Example - Show all subresources

For example, if we have a user object which has one or more skills and one or more devices associated with it, and we want to read all of those in one request we need to do the following:

Request

```
GET .../api/v2/users/<user_id>?subresources=*
```

Response

```
{
  "id":<user_id>,
  "firstName":<first_name>,
  ...
  "skills":[{
    "id":<skill_1_id>,
    ...
  },
  ...
  {
    "id":<skill_N_id>,
    ...
  }],
  "devices":[{
    "id":<device_1_id>,
    ...
  },
  ...
  {
    "id":<device_M_id>,
    ...
  }]
}
```

If the subresources parameter is not included in the request, you receive everything except the "skills" collection and "devices" collection.

Important

It is also possible to apply subresources feature to object settings and request both an object and its settings in one request.

Selecting Subresources

In the example above, we specify `subresources=*` to get all available subresources.

If the object we are interested in has several types of subresources, we can choose which subresources to be returned. This could be achieved by specifying a comma-separated list of subresources.

Example

Request

```
GET .../api/v2/users/<user_id>?subresources=skills,devices
```

Response

```
{
  "id":<user_id>,
  "firstName":<first_name>,
  ...
  "skills":[{
    "id":<skill_1_id>,
    ...
  },
  ...
  {
    "id":<skill_N_id>,
    ...
  }],
  "devices":[{
    "id":<device_1_id>,
    ...
  },
  ...
  {
    "id":<device_M_id>,
    ...
  }]
}
```

Example 2

Request

```
GET .../api/v2/users/<user_id>?subresources=skills
```

Response

```
{
  "id":<user_id>,
  "firstName":<first_name>,
  ...
  "skills":[{
    "id":<skill_1_id>,
    ...
  },
  ...
  {
    "id":<skill_N_id>,
    ...
  }]
```

```
    },
    ...
    {
        "id": <skill_N_id>,
        ...
    }
}
```

Filtering and Sorting by Subresource Properties

If an object has subresources, it is possible to filter and sort by their properties. For example: show all users with skill "Sales". This could be achieved by using subresource name (for example "skills") followed by a dot and its property (for example ".name").

Example - Sorting by Skill Name

Request

```
GET .../api/v2/users?subresources=skills&fields=id,userName&skills.name=Sales
```

Response

```
{
    "statusCode": 0,
    "users": [{
        "id": "3454353254324",
        "userName": "cmburns@springfieldnclear.com",
        "skills": [{
            "id": "0890689",
            "name": "Sales",
            "level": 2
        }
    ],
    {
        "id": "3567365736736",
        "userName": "hsimpson@springfieldnclear.com",
        "skills": [{
            "id": "0890689",
            "name": "Sales",
            "level": 4
        }
    ]
}]
}
```

Example 2

Request

```
GET .../api/v2/users?subresources=skills&fields=id,userName&sortBy=skills.level=Descending
```

Response

```
{
  "statusCode": 0,
  "users": [{
    "id": "3567365736736",
    "userName": "hsimpson@springfieldnclear.com",
    "skills": [{
      "id": "0890689",
      "name": "Sales",
      "level": 4
    }
  ],
  {
    "id": "3454353254324",
    "userName": "cmburns@springfieldnclear.com",
    "skills": [{
      "id": "0890689",
      "name": "Sales",
      "level": 2
    }
  ]
}]
}
```