**JSONPath Notation**

A JSONPath expression specifies a path to an element (or a set of elements) in a JSON structure. Paths can use the dot notation:

$.store.book[0].title

or the bracket notation:

$['store']['book'][0]['title']

The leading $ represents the root object or array and can be omitted. For example, $.foo.bar and foo.bar are the same, and so are $[0].status and [0].status.

Other syntax elements are described below.

| **Expression** | **Description** |
| --- | --- |
| $ | The root object or array. |
| .*property* | Selects the specified property in a parent object. |
| ['*property*'] | Selects the specified property in a parent object. Be sure to put single quotes around the property name.  **Tip:**Use this notation if the property name contains special characters such as spaces, or begins with a character other than A..Za..z\_. |
| [*n*] | Selects the *n*-th element from an array. Indexes are 0-based. |
| [*index1*,*index2*,*…*] | Selects array elements with the specified indexes. Returns a [list](https://support.smartbear.com/alertsite/docs/monitors/api/endpoint/jsonpath.html#multiple). |
| ..*property* | Recursive descent: Searches for the specified property name recursively and returns an array of all values with this property name. Always returns a [list](https://support.smartbear.com/alertsite/docs/monitors/api/endpoint/jsonpath.html#multiple), even if just one property is found. |
| \* | Wildcard selects all elements in an object or an array, regardless of their names or indexes. For example, address.\* means all properties of the addressobject, and book[\*] means all items of the book array. |
| [*start*:*end*] [*start*:] | Selects array elements from the *start* index and up to, but not including, *end*index. If *end* is omitted, selects all elements from *start* until the end of the array. Returns a [list](https://support.smartbear.com/alertsite/docs/monitors/api/endpoint/jsonpath.html#multiple). |
| [:*n*] | Selects the first *n* elements of the array. Returns a [list](https://support.smartbear.com/alertsite/docs/monitors/api/endpoint/jsonpath.html#multiple). |
| [*-n*:] | Selects the last *n* elements of the array. Returns a [list](https://support.smartbear.com/alertsite/docs/monitors/api/endpoint/jsonpath.html#multiple). |
| [?(*expression*)] | [Filter expression](https://support.smartbear.com/alertsite/docs/monitors/api/endpoint/jsonpath.html#filters). Selects all elements in an object or array that match the specified filter. Returns a [list](https://support.smartbear.com/alertsite/docs/monitors/api/endpoint/jsonpath.html#multiple). |
| [(*expression*)] | Script expressions can be used instead of explicit property names or indexes. An example is [(@.length-1)] which selects the last item in an array. Here, length refers to the length of the current array rather than a JSON field named length. |
| @ | Used in filter expressions to refer to the current node being processed. |

Notes:

* JSONPath expressions, including property names and values, are **case-sensitive**.
* Unlike XPath, JSONPath does not have operations for accessing parent or sibling nodes from the given node.

Filters

Filters are logical expressions used to filter arrays. An example of a JSONPath expression with a filter is

$.store.book[?(@.price < 10)]

where @ represents the current array item or object being processed. Filters can also use $ to refer to the properties outside of the current object:

$.store.book[?(@.price < $.expensive)]

An expression that specifies just a property name, such as [?(@.isbn)], matches all items that have this property, regardless of the value.

Additionally, filters support the following operators:

| **Operator** | **Description** |
| --- | --- |
| == | Equals to. 1 and '1' are considered equal. String values must be enclosed in single quotes (not double quotes): [?(@.color=='red')]. |
| != | Not equal to. String values must be enclosed in single quotes. |
| > | Greater than. |
| >= | Greater than or equal to. |
| < | Less than. |
| <= | Less than or equal to. |
| =~ | Match a [JavaScript regular expression](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Guide/Regular_Expressions). For example, [?(@.description =~ /cat.\*/i)] matches items whose description starts with *cat* (case-insensitive).  **Note:**Not supported at [locations that use Ready! API 1.1](https://support.smartbear.com/alertsite/docs/monitors/api/readyapi-versions.html). |
| ! | Use to negate a filter: [?(!@.isbn)] matches items that do not have the isbn property.  **Note:**Not supported at [locations that use Ready! API 1.1](https://support.smartbear.com/alertsite/docs/monitors/api/readyapi-versions.html). |
| && | Logical AND, used to combine multiple filter expressions:  [?(@.category=='fiction' && @.price < 10)] |
| || | Logical OR, used to combine multiple filter expressions:  [?(@.category=='fiction' || @.price < 10)]  **Note:**Not supported at [locations that use Ready! API 1.1](https://support.smartbear.com/alertsite/docs/monitors/api/readyapi-versions.html). |

Examples

For these examples, we will use a modified version of JSON from <http://goessner.net/articles/JsonPath/index.html#e3>:

{  
  "store": {  
    "book": [  
      {  
        "category": "reference",  
        "author": "Nigel Rees",  
        "title": "Sayings of the Century",  
        "price": 8.95  
      },  
      {  
        "category": "fiction",  
        "author": "Herman Melville",  
        "title": "Moby Dick",  
        "isbn": "0-553-21311-3",  
        "price": 8.99  
      },  
      {  
        "category": "fiction",  
        "author": "J.R.R. Tolkien",  
        "title": "The Lord of the Rings",  
        "isbn": "0-395-19395-8",  
        "price": 22.99  
      }  
    ],  
    "bicycle": {  
      "color": "red",  
      "price": 19.95  
    }  
  },  
  "expensive": 10  
}

In all these examples, the leading $. is optional and can be omitted.

| **Expression** | **Meaning** |
| --- | --- |
| $.store.\* | All direct properties of store (not recursive). |
| $.store.bicycle.color | The color of the bicycle in the store.  Result: red |
| $.store..price $..price | The prices of all items in the store.  Result: [8.95, 8.99, 22.99, 19.95] |
| $.store.book[\*] $..book[\*] | All books in the store. |
| $..book[\*].title | The titles of all books in the store.  Result: [Sayings of the Century, Moby Dick, The Lord of the Rings] |
| $..book[0] | The first book.  Result: [{"category":"reference","author":"Nigel Rees","title":"Sayings of the Century","price":8.95}] |
| $..book[0].title | The title of the first book.  Result: Sayings of the Century |
| $..book[0,1].title $..book[:2].title | The titles of the first two books.  Result: [Sayings of the Century, Moby Dick] |
| $..book[-1:].title $..book[(@.length-1)].title | The title of the last book.  Result: [The Lord of the Rings]  The result is a [list](https://support.smartbear.com/alertsite/docs/monitors/api/endpoint/jsonpath.html#multiple), because [*-n*:] always returns lists. |
| $..book[?(@.author=='J.R.R. Tolkien')].title | The titles of all books by *J.R.R. Tolkien* (exact match, case-sensitive).  Result: [The Lord of the Rings]  The result is a list, because filters always return lists. |
| $..book[?(@.isbn)] | All books that have the isbn property. |
| $..book[?(!@.isbn)] | All books without the isbn property. |
| $..book[?(@.price < 10)] | All books cheaper than 10. |
| $..book[?(@.price > $.expensive)] | All expensive books. |
| $..book[?(@.author =~ /.\*Tolkien/i)] | All books whose author name ends with *Tolkien* (case-insensitive). |
| $..book[?(@.category == 'fiction' || @.category == 'reference')] | All fiction and reference books. |
| $..\* | All members of the JSON structure beneath the root (child objects, individual property values, array items), combined into an array. |

Considerations for JSONPath Expressions That Return Multiple Elements

JSONPath queries can return not just a single element, but also a list of matching elements. For example, given this JSON:

{  
  "name": "Rose Kolodny",  
  "phoneNumbers": [  
    {  
      "type": "home",  
      "number": "954-555-1234"  
    },  
    {  
      "type": "work",  
      "number": "754-555-5678"  
    }  
  ]  
}

the JSONPath expression

phoneNumbers[\*].number

returns a list containing two phone numbers:

[954-555-1234, 754-555-5678]