Atitit json数据查询法 jsonpath

目录

[1.1. 1.概述 1](#_Toc31472)

[1.2. 3.2。经营者特殊符号 1](#_Toc18187)

[1.3. # JSONPath expressions 2](#_Toc29988)

[1.4. Xpath vs jsonpath 4](#_Toc4744)

[1.4.2. Usage 5](#_Toc26240)

[1.5. Filters 5](#_Toc15519)

[1.6. 聚合运算3.3。功能和过滤器 9](#_Toc220)

[1.7. jsonpath的函数 9](#_Toc27209)

[1.8. jsonpath 操作符 9](#_Toc11722)

[1.9. 与xpath对照 10](#_Toc15711)

[1.10. Javascript Example: 10](#_Toc13317)

[1.11. # Issues 11](#_Toc1237)

## **1.概述**

XML的优点之一是处理的可用性-包括XPath-它被定义为[W3C标准](https://www.w3.org/TR/xpath/)。对于JSON，出现了一个类似的名为JSONPath的工具。

本文将**介绍Jayway JsonPath**，它是[JSONPath规范](http://goessner.net/articles/JsonPath/)的Java实现。它描述了设置，语法，通用API以及用例的演示。

## ****3.2。经营者特殊符号****

在JsonPath中，我们有几个有用的运算符：

****根节点（$）****：此符号表示JSON结构的根成员，无论它是对象还是数组。它的用法示例包含在前面的小节中。

****当前节点（@）****：表示正在处理的节点，通常用作谓词的输入表达式的一部分。假设我们在上面的JSON文档中处理book数组，表达式book [？（@。price == 49.99）]引用该数组中的第一本书。

****通配符（\*）****：表示指定范围内的所有元素。例如，book [\*]表示book数组内的所有节点

以$为root，.操作符或[]索引的方式获取指定 JsonPath 数据

## [#](https://goessner.net/articles/JsonPath/index.html" \l "e2" \o "permanent link) JSONPath expressions

JSONPath expressions always refer to a JSON structure in the same way as XPath expression are used in combination with an XML document. Since a JSON structure is usually anonymous and doesn't necessarily have a "root member object" JSONPath assumes the abstract name $ assigned to the outer level object.

JSONPath expressions can use the dot–notation

$.store.book[0].title

or the bracket–notation

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

for input pathes. Internal or output pathes will always be converted to the more general bracket–notation.

JSONPath allows the wildcard symbol \* for member names and array indices. It borrows the descendant operator '..' from [E4X](http://en.wikipedia.org/wiki/E4X) and the [array slice syntax](http://developer.mozilla.org/es4/proposals/slice_syntax.html) proposal [start:end:step] from [ECMASCRIPT 4](http://www.ecmascript.org/).

Expressions of the underlying scripting language (<expr>) can be used as an alternative to explicit names or indices as in

$.store.book[(@.length-1)].title

using the symbol '@' for the current object. Filter expressions are supported via the syntax ?(<boolean expr>) as in

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

Here is a complete overview and a side by side comparison of the JSONPath syntax elements with its XPath counterparts.

|  |  |  |
| --- | --- | --- |
| **XPath** | **JSONPath** | **Description** |
| / | $ | the root object/element |
| . | @ | the current object/element |
| / | . or [] | child operator |
| .. | n/a | parent operator |
| // | .. | recursive descent. JSONPath borrows this syntax from E4X. |
| \* | \* | wildcard. All objects/elements regardless their names. |
| @ | n/a | attribute access. JSON structures don't have attributes. |
| [] | [] | subscript operator. XPath uses it to iterate over element collections and for [predicates](http://www.w3.org/TR/xpath" \l "predicates). In Javascript and JSON it is the native array operator. |
| | | [,] | Union operator in XPath results in a combination of node sets. JSONPath allows alternate names or array indices as a set. |
| n/a | [start:end:step] | array slice operator borrowed from ES4. |
| [] | ?() | applies a filter (script) expression. |
| n/a | () | script expression, using the underlying script engine. |
| () | n/a | grouping in Xpath |

XPath has a lot more to offer (Location pathes in not abbreviated syntax, operators and functions) than listed here. Moreover there is a remarkable difference how the subscript operator works in Xpath and JSONPath.

* Square brackets in XPath expressions always operate on the node set resulting from the previous path fragment. Indices always start by 1.
* With JSONPath square brackets operate on the object or array addressed by the previous path fragment. Indices always start by 0.

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" \l "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" \l "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 address object, 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" \l "multiple). |
| [:*n*] | Selects the first *n* elements of the array. Returns a [list](https://support.smartbear.com/alertsite/docs/monitors/api/endpoint/jsonpath.html" \l "multiple). |
| [*-n*:] | Selects the last *n* elements of the array. Returns a [list](https://support.smartbear.com/alertsite/docs/monitors/api/endpoint/jsonpath.html" \l "multiple). |
| [?(*expression*)] | [Filter expression](https://support.smartbear.com/alertsite/docs/monitors/api/endpoint/jsonpath.html" \l "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" \l "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 |

## Xpath vs jsonpath

|  |  |  |
| --- | --- | --- |
| **XPath** | **JSONPath** | **Result** |
| /store/book/author | $.store.book[\*].author | the authors of all books in the store |
| //author | $..author | all authors |
| /store/\* | $.store.\* | all things in store, which are some books and a red bicycle. |
| /store//price | $.store..price | the price of everything in the store. |
| //book[3] | $..book[2] | the third book |
| //book[last()] | $..book[(@.length-1)] $..book[-1:] | the last book in order. |
| //book[position()<3] | $..book[0,1] $..book[:2] | the first two books |
| //book[isbn] | $..book[?(@.isbn)] | filter all books with isbn number |
| //book[price<10] | $..book[?(@.price<10)] | filter all books cheapier than 10 |
| //\* | $..\* | all Elements in XML document. All members of JSON structure. |

#### ***|2007-08-22| e4***[#](https://goessner.net/articles/JsonPath/index.html" \l "e4" \o "permanent link) JSONPath implementation

JSONPath is implemented in Javascript for clientside usage and ported over to PHP for use on the server.

### Usage

All you need to do is downloading either of the files

* [jsonpath.js](http://code.google.com/p/jsonpath/)
* [jsonpath.php](http://code.google.com/p/jsonpath/)

include it in your program and use the simple API consisting of one single function.

jsonPath(obj, expr [, args])

## Filters

.. ：深层扫描操作

?(<expression>) ：表达式

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" \t "https://support.smartbear.com/alertsite/docs/monitors/api/endpoint/_blank). 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). |

## 聚合运算****3.3。**功能和过滤器**

JsonPath还具有可用于路径末尾以综合该路径的输出表达式的函数：min（），max（），avg（），stddev（），length（）。

最后–我们有过滤器；这些是布尔表达式，用于将返回的节点列表限制为仅调用方法所需的节点列表。

一些示例包括等式（==），正则表达式匹配（=〜），包含（in），检查是否为空（empty）。过滤器主要用于谓词。

有关不同运算符，函数和过滤器的完整列表和详细说明，请参阅[JsonPath GitHub](https://github.com/jayway/JsonPath)项目

## jsonpath的函数

| **名称** | **描述** | **输出** |
| --- | --- | --- |
| min() | 获取数值类型数组的最小值 | Double |
| max() | 获取数值类型数组的最大值 | Double |
| avg() | 获取数值类型数组的平均值 | Double |
| stddev() | 获取数值类型数组的标准差 | Double |
| length() | 获取数值类型数组的长度 | Integer |

## jsonpath 操作符

| **操作符** | **描述** |
| --- | --- |
| == | 等于符号，但数字1不等于字符1(note that 1 is not equal to ‘1’) |
| != | 不等于符号 |
| < | 小于符号 |
| <= | 小于等于符号 |
| > | 大于符号 |
| >= | 大于等于符号 |
| =~ | 判断是否符合正则表达式，例如[?(@.name =~ /foo.\*?/i)] |
| in | 所属符号，例如[?(@.size in [‘S’, ‘M’])] |
| nin | 排除符号 |
| size | size of left (array or string) should match right |
| empty | 判空符号 |

## 与xpath对照

| **XPath** | **JSONPath** | **Description** |
| --- | --- | --- |
| / | $ | 根结点 |
| . | @ | 当前结点 |
| / | . or [] | 取子结点 |
| .. | n/a | 取父节点 |
| // | .. | 选择所有符合条件的 |
| \* | \* | 匹配所有元素 |
| @ | n/a | 根据属性访问 |
| [] | [] | 迭代器标示. XPath 用来选择集合元素. js或json中用作数组下标. |
| | | [,] | 迭代器中多选 |
| n/a | [start:end:step] | 数组分隔 |
| [] | ?() | 过滤操作 |
| n/a | () | 表达式计算 |
| () | n/a | xpath中分组 |

## ****Javascript Example****:

    <script src="jquery/3.4.1/jquery.js"></script>

        <script src="jsonpath.jquery.js"></script>

        <Script>

        var accList = [{

                'accnum': '83457834758947598', 'holdername': '李一', 'bank': '中国银行', 'branch': '上海分行xxx支行'

            },

            {

                'accnum': '22222222222',

                'holdername': '王er',

                'bank': '农业银行',

                'branch': '上海分行农业银行第一支行'

            },

            {

                'accnum': '287488347958940',

                'holdername': '李三',

                'bank': '招商银行',

                'branch': '上海分行招商银行第2支行'

            },

From root where accnum=’5555’

  var path = $.JSONPath({ data: accList, keepHistory: false });

    var rs = path.query('$[?(@.accnum=="5555555555555555555555")]');  // filter all books cheapier

        alert(rs);

        $("#txt\_bank").val(rs[0].bank);

## [#](https://goessner.net/articles/JsonPath/index.html" \l "e5" \o "permanent link) Issues

* Currently only single quotes allowed inside of JSONPath expressions.
* Script expressions inside of JSONPath locations are currently not recursively evaluated by jsonPath. Only the global $ and local @ symbols are expanded by a simple regular expression.
* An alternative for jsonPath to return false in case of no match may be to return an empty array in future.

JSONPath - XPath for JSON.html

JSONPath的使用 - 豆芽丝.html

JSONPath Syntax \_ AlertSite Documentation.html

kubectl 的 JSONPath 查询支持 \_ 天青色等烟雨.html

(···条消息)JsonPath教程\_JsonPath\_koflance的博客-CSDN博客.html

json数据查询的方法，JsonSQL数据查询，jfunk数据查询.html