Практика

Как найти нужный нам сервис?

1. Заходим на сайт <https://www.metaweather.com/api/>

2. Ищем сервис с именем Location Search  
  
Требования к сервису Location Search

### 1) **URL** :

* /api/location/search/?query=(query)

### 2) **Arguments** : Either query or latt/long need to be present:

* **query**Text to search for.

### 3) **Response Fields**

|  |  |  |  |
| --- | --- | --- | --- |
| **Field** | **Type** | **Unit** | **Description** |
| title | string |  | Name of the location |
| location\_type | string | (City|Region / State / Province|Country|Continent) |  |
| latt\_long | floats, comma separated |  |  |
| woeid | integer |  | Where On Earth ID |
|  |  |  |  |

### Examples

* [/api/location/search/?query=san](https://www.metaweather.com/api/location/search/?query=san)
* [/api/location/search/?query=london](https://www.metaweather.com/api/location/search/?query=london)

3. Составим чеклист

|  |  |  |  |
| --- | --- | --- | --- |
| Test case | Expected result | Actual result | Comments |
| Create request with valid data Ex: [/api/location/search/?query=london](https://www.metaweather.com/api/location/search/?query=london) | Verify that response contains: 1) Status code = 200 2) Response message = “OK” | 1)Status code = 200 2) Response message = “OK” | pass |
| Create request with valid data | Verify that Response body contains the next parameters: - title - location\_type - latt\_long - woeid - distance | body contains the next parameters: - title - location\_type - latt\_long - woeid - distance | pass |
| Create request with valid data | Verify that response body parameters should have the following type of data: - title - string - location\_type - string - latt\_long - floats, comma separated - woeid - integer - distance - integer | body parameters should have the following type of data: - title - string - location\_type - string - latt\_long - floats, comma separated - woeid - integer - distance - integer | pass |
| Create request with invalid data(add additional param) Ex: [/api/location/search/?query=](https://www.metaweather.com/api/location/search/?query=london)lon&& | Verify that response contains: 1) Status code = 403 2) Response message = “Forbidden” 3) Body is empty | 1)Status code = 200 2) Response message = “OK”  3)Body is not empty | fail |
| Create request with invalid data (empty ‘query’ value) Ex: [/api/location/search/?query=](https://www.metaweather.com/api/location/search/?query=london) | Verify that response contains: 1) Status code = 403 2) Response message = “Forbidden” 3) Body is empty | 1) Status code = 403 2) Response message = “Forbidden” 3) Body is empty | pass |
| Create request with invalid data (digit ‘query’ value) Ex: [/api/location/search/?query=](https://www.metaweather.com/api/location/search/?query=london)7 | Verify that response contains: 1) Status code = 200 2) Response message = “Ok” 3) Body is [] | 1) Status code = 200 2) Response message = “Ok” 3) Body is [] | pass |
| Create request with invalid data (specific symbol ‘query’ value) Ex: [/api/location/search/?query=](https://www.metaweather.com/api/location/search/?query=london){ | Verify that response contains: 1) Status code = 200 2) Response message = “Ok” 3) Body is [] | 1) Status code = 200 2) Response message = “Ok” 3) Body is [] | pass |
| Create request with valid data (work with space) Ex: [/api/location/search/?query=](https://www.metaweather.com/api/location/search/?query=london) l | Verify that response contains: 1) Status code = 200 2) Response message = “Ok” 3) Body has valid values | 1) Status code = 200 2) Response message = “Ok” 3) Body has valid values | pass |

Домашнее задание

Как найти нужный нам сервис?

1. Заходим на сайт <https://www.metaweather.com/api/>

2. Ищем сервис с именем Location  
3. Составляем чек-лист и тестируем  
  
Требования к сервису Location

**1) URL** :

* /api/location/(woeid)/

**2) Arguments:**

* **woeid**Where On Earth ID. [Docs](http://developer.yahoo.com/geo/geoplanet/guide/concepts.html).

### **Examples**

* [/api/location/44418/](https://www.metaweather.com/api/location/44418/) - London
* [/api/location/2487956/](https://www.metaweather.com/api/location/2487956/) - San Francisco

### **3) Response Fields**

|  |  |  |  |
| --- | --- | --- | --- |
| **Field** | **Type** | **Unit** | **Description** |
| title | string |  | Name of the location |
| location\_type | string | (City|Region / State / Province|Country|Continent) |  |
| latt\_long | floats, comma separated |  |  |
| time | datetime |  | Time in location |
| sun\_(rise|set) | datetime |  |  |
| timezone\_name | string |  | Name of the timezone that the location is in |
| parent | | | |
| title | string |  | Name of the parent location |
| location\_type | string | (City|Region / State / Province|Country|Continent) |  |
| latt\_long | floats, comma separated |  |  |
| woeid | integer |  | Where On Earth ID |
| consolidated\_weather | | | |
| id | integer |  | Internal identifier for the forecast |
| applicable\_date | date |  | Date that the forecast or observation pertains to |
| weather\_state\_name | string |  | Text description of the weather state |
| weather\_state\_abbr | string |  | One or two letter abbreviation of the weather state |
| wind\_speed | float | mph |  |
| wind\_direction | float | degrees |  |
| wind\_direction\_compass | string | [compass point](http://en.wikipedia.org/wiki/Boxing_the_compass#Compass_points) | Compass point of the wind direction |
| (min|max|the)\_temp | integer | centigrade |  |
| air\_pressure | float | mbar |  |
| humidity | float | percentage |  |
| visibility | float | miles |  |
| predictability | integer | percentage | Our interpretation of the level to which the forecasters agree with each other - 100% being a complete consensus. |
| sources | | | |
| title | string |  | Name of the source |
| url | string |  | URL of the source |

3. Составим чеклист

|  |  |  |  |
| --- | --- | --- | --- |
| Test case | Expected result | Actual result | Comments |
| Create request with valid data Ex: [/api/location/44418/](https://www.metaweather.com/api/location/44418/) | Verify that response contains: 1) Status code = 200 2) Response message = “OK” | 1)Status code = 200 2) Response message = “OK” | pass |
| Create request with valid data Ex: [/api/location/44418/](https://www.metaweather.com/api/location/44418/) | Verify that Response body contains the next parameters: "id" "weather\_state\_name" "weather\_state\_abbr"  "wind\_direction\_compas"  "created" "applicable\_date"  "min\_temp"  "max\_temp"  "the\_temp" "wind\_speed"  "wind\_direction" "air\_pressure"  "humidity"  "visibility" "predictability" | Response body contains the next parameters: "id" "weather\_state\_name" "weather\_state\_abbr"  "wind\_direction\_compas"  "created" "applicable\_date"  "min\_temp"  "max\_temp"  "the\_temp" "wind\_speed"  "wind\_direction" "air\_pressure"  "humidity"  "visibility" "predictability" | pass |
| Create request with valid data Ex: [/api/location/44418/](https://www.metaweather.com/api/location/44418/) | Verify that Response body contains the next parameters: "id" - integer "weather\_state\_name" - string "weather\_state\_abbr" - string  "wind\_direction\_compas" - string  "created" - datetime "applicable\_date" - date  "min\_temp" - float  "max\_temp"- float  "the\_temp" - float "wind\_speed" - float  "wind\_direction" - float "air\_pressure" - float  "humidity" - integer "visibility" float "predictability" - integer | Response body contains the next parameters: "id" - integer "weather\_state\_name" - string "weather\_state\_abbr" - string  "wind\_direction\_compas" - string  "created" - datetime "applicable\_date" - date  "min\_temp" - float  "max\_temp"- float  "the\_temp" - float "wind\_speed" - float  "wind\_direction" - float "air\_pressure" - float  "humidity" - integer "visibility" float "predictability" - integer | pass |
| Create request with invalid data(w/o param) Ex: [/api/location/](https://www.metaweather.com/api/location/) | Verify that response contains: 1) Status code = 404 2) Response message = “Not Found” | 1)Status code = 200 2) Response message = “OK” | pass |

Протестировать post запрос [searchSettlements](https://devcenter.novaposhta.ua/docs/services/556d7ccaa0fe4f08e8f7ce43/operations/58e5ebeceea27017bc851d67) (сервис для онлайн поиска населенных пунктов)  
  
Шаги:  
1. Создать чек-лист, по которому будет проходить тестирование  
2. Протестировать и записать результаты  
  
API key: f0a8ed3fd5f618970de71afbc1d9828c

Составим чеклист

|  |  |  |  |
| --- | --- | --- | --- |
| Test case | Expected result | Actual result | Comments |
| Create request with valid data Ex: <http://testapi.novaposhta.ua/v2.0/json/Address/searchSettlements/>  and next body:  {  "apiKey": "f0a8ed3fd5f618970de71afbc1d9828c",  "modelName": "Address",  "calledMethod": "searchSettlements",  "methodProperties": {  "CityName": "харьков",  "Limit": 1  }  } | Verify that response contains: 1) Status code = 200 2) Response message = “OK”  all requested data returned | 1)Status code = 200 2) Response message = “OK” | pass |
| Create request with valid data Ex: <http://testapi.novaposhta.ua/v2.0/json/Address/searchSettlements/>  and next body:  {  "apiKey": "f0a8ed3fd5f618970de71afbc1d9828c",  "modelName": "Address",  "calledMethod": "searchSettlements",  "methodProperties": {  "CityName": "харьков",  "Limit": 1  }  } | Verify that Response body contains the next parameters: - Warehouses - MainDescription - Area - Region - SettlementTypeCode  - Ref  - DeliveryCity | body contains the next parameters: - Warehouses - MainDescription - Area - Region - SettlementTypeCode  - Ref  - DeliveryCity | pass |
| Create request with valid data Ex: <http://testapi.novaposhta.ua/v2.0/json/Address/searchSettlements/>  and next body:  {  "apiKey": "f0a8ed3fd5f618970de71afbc1d9828c",  "modelName": "Address",  "calledMethod": "searchSettlements",  "methodProperties": {  "CityName": "харьков",  "Limit": 1  }  } | Verify that Response body contains the next parameters: - Warehouses - integer - MainDescription - string - Area - string - Region - string - SettlementTypeCode - string  - Ref - string  - DeliveryCity - string | body contains the next parameters: - Warehouses - integer - MainDescription - string - Area - string - Region - string - SettlementTypeCode - string  - Ref - string  - DeliveryCity - string | pass |
| Create request with invalid data(missing required values of required parameters) Ex: <http://testapi.novaposhta.ua/v2.0/json/Address/searchSettlements/>  and next body:  {  "apiKey": "",  "modelName": "",  "calledMethod": "",  "methodProperties": {  "CityName": "",  "Limit":  }  } | Verify that response contains: 1) Status code = 200 2) Response message = “OK” 3) Body contains next  “{  "success": false,  "data": [],  "errors": [  "Data is invalid"  ],  "warnings": [],  "info": [],  "messageCodes": [],  "errorCodes": [],  "warningCodes": [],  "infoCodes": []  }  ” | 1) Status code = 200 2) Response message = “OK” 3) Body contains next  “{  "success": false,  "data": [],  "errors": [  "Data is invalid"  ],  "warnings": [],  "info": [],  "messageCodes": [],  "errorCodes": [],  "warningCodes": [],  "infoCodes": []  }  ” | pass |
| Create request with invalid data (missing required ApiKey parameter and its value) Ex: <http://testapi.novaposhta.ua/v2.0/json/Address/searchSettlements/>  and next body:  {  "modelName": "Address",  "calledMethod": "searchSettlements",  "methodProperties": {  "CityName": "харьков",  "Limit": 1  }  } | Verify that response contains: 1) Status code = 200 2) Response message = “OK” 3) Body contains next  “{  "success": false,  "data": [],  "errors": [  "Data is invalid"  ],  "warnings": [],  "info": [],  "messageCodes": [],  "errorCodes": [],  "warningCodes": [],  "infoCodes": []  }  ” | 1) Status code = 200 2) Response message = “OK” 3) body contains the next parameters with their correct values: - Warehouses - MainDescription - Area - Region - SettlementTypeCode  - Ref  - DeliveryCity | fail |
| Create request with valid data (check ZIP code instead City name) Ex: <http://testapi.novaposhta.ua/v2.0/json/Address/searchSettlements/>  and next body:  {  "modelName": "Address",  "calledMethod": "searchSettlements",  "methodProperties": {  "CityName": "61007",  "Limit": 1  }  } | Verify that Response body contains the next parameters: - Warehouses - MainDescription - Area - Region - SettlementTypeCode  - Ref  - DeliveryCity | body contains the next parameters: - Warehouses - MainDescription - Area - Region - SettlementTypeCode  - Ref  - DeliveryCity | pass |
| Create request with invalid data (limit is string instead integer) Ex: <http://testapi.novaposhta.ua/v2.0/json/Address/searchSettlements/>  and next body:  {  "modelName": "Address",  "calledMethod": "searchSettlements",  "methodProperties": {  "CityName": "61007",  "Limit": "a"  }  } | Verify that response contains: 1) Status code = 200 2) Response message = “Ok” 3) Body {  "success": false,  "data": [],  "errors": [  "Limit has invalid characters"  ],  "warnings": [],  "info": [],  "messageCodes": [],  "errorCodes": [  "20000500653"  ],  "warningCodes": [],  "infoCodes": [] | 1) Status code = 200 2) Response message = “Ok” 3) Body {  "success": false,  "data": [],  "errors": [  "Limit has invalid characters"  ],  "warnings": [],  "info": [],  "messageCodes": [],  "errorCodes": [  "20000500653"  ],  "warningCodes": [],  "infoCodes": [] | pass |
| Create request with valid data Ex: <http://testapi.novaposhta.ua/v2.0/json/Address/searchSettlements/>  and next body:  {  "modelName": "Address",  "calledMethod": "searchSettlements",  "methodProperties": {  "CityName": "61007",  "Limit": 100000  }  } | Verify that response contains: 1) Status code = 200 2) Response message = “Ok” 3) Body has valid values | 1) Status code = 200 2) Response message = “Ok” 3) Body{  "success": false,  "data": [],  "errors": [  "DataSerialization::deserialize: DOMDocument::loadXML()"  ],  "warnings": [],  "info": [],  "messageCodes": [],  "errorCodes": [],  "warningCodes": [],  "infoCodes": []  } | fail |