mslookup service Documentation

Release latest

Apr 30, 2021

Contents

CHAPTER 1

Contents

1.1 MsLookup Webservice

The MSLoopkup web service is a Restful API that provides a programatic interface to access to peptide evidences previously identified in mass spectrometry experiments.

Note: The API is published in the following URL: https://www.ebi.ac.uk/pride/multiomics/ws

1.1.1 Mass spectrum JSON structure

The MSLookup service provides mass spectrometry evidences for peptides, with special focus on modified peptides (PTMs - posttranslational modifications) and single aminoacid variants. The structure of the spectra provided on each endpoint is the following:

```
{
  "id": "NIST:cptac2_human_hcd_itraq_selected_part1_2015.msp:index:80003",
  "usi": "NIST:cptac2_human_hcd_itraq_selected_part1_2015.msp:index:80003",
  "pepSequence": "AQLGVQAFADALLIIPK",
  "proteinAccessions": [ "P40227-2", "ENSP00000275603.4"],
  "geneAccessions": [CCT6A","ENSG00000146731.11","ENST00000335503.3"],
  "precursorMz": 514.8157,
  "precursorCharge": 4,
  "projectAssays": null,
  "pxProjects": null,
  "species": [],
  "modifications": [
      {
        "neutralLoss": null,
        "positionMap": [
           {
             "key": 0,
```

(continues on next page)

(continued from previous page)


```
            "value": []
          },
          {
            "key": 16,
            "value": []
          }
        ],
       "modification": {
          "cvLabel": "UNIMOD",
          "accession": "UNIMOD:214",
          "name": "iTRAQ4plex",
          "value": "144.102063"
       },
      "attributes": null
    }
  ],
  "masses": [145.1084,199.1806,458.2939],
  "intensities": [5123.7,6716.8,2049.7],
  "retentionTime": null,
  "properties": null,
  "missedCleavages": 0,
  "annotations": null,
  "qualityEstimationMethods": [],
  "text": null
}
```


1.1.2 Main spectra attributes

The mass spectrum attributes can be divided in three main groups:

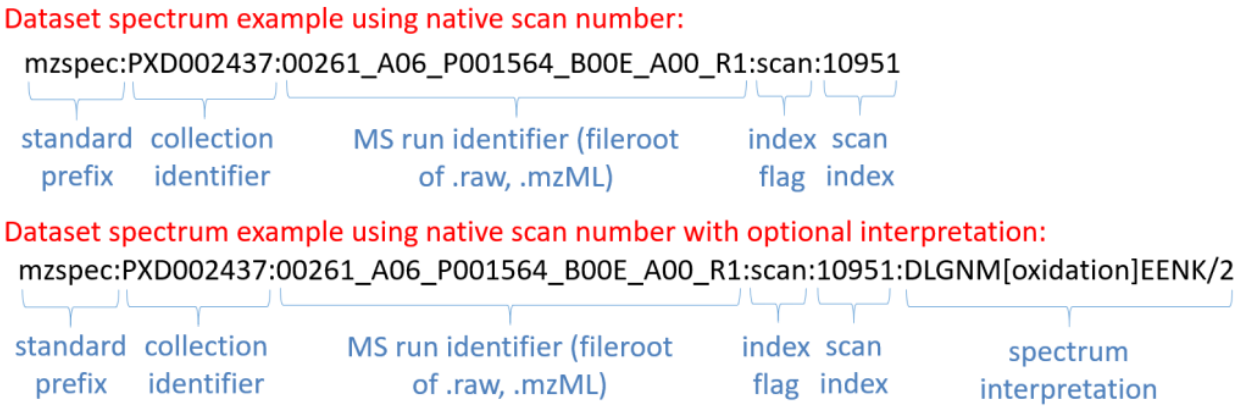
- **biology properties:**
  - Protein accessions in ENSEMBL and UNIPROT that contains the corresponding peptides; gene accessions which represent a list of gene names, gene and transcript accessions from ENSEMBL that contains the corresponding peptides.
  - Post-translational modifications: a list of post-translational modifications identified by mass spectrometry including position, monoisotopic mass, and UNIMOD accession if available.
  - Additional metadata: species, sample conditions, tissue, cell-line, proteomeXchange project accessions.
- mass spectrometry properties: Spectrum information including (precursor mz, charge and peak list), additional information such as retention time and missed-cleavages.
- statistical assessment: additional quality and statistical assessment scores such as search engine scores, p-values, q-values.

If the information is not available empty lists or *null* values are provided.

1.2 Get by USI

USI(universal spectrum identifier) is a unique representation for a mass spectrum within a resource or a dataset Documentation. In summary, a USI is a combination of a dataset accession or collection (e.g. spectral library); a file within

that collection and an scan or index number withing the file. Inm addition, the USI can contains the information about the peptide sequence and PTMs.



If you want to know more about USI, USI Specification.

1.2.1 Swagger

Note: https://www.ebi.ac.uk/pride/multiomics/ws/swagger-ui/index.html?url=/pride/multiomics/ws/api-docs&configUrl=/pride/multiomics/ws/api-docs/swagger-config#/Spectra/findByUsi

1.2.2 Curl

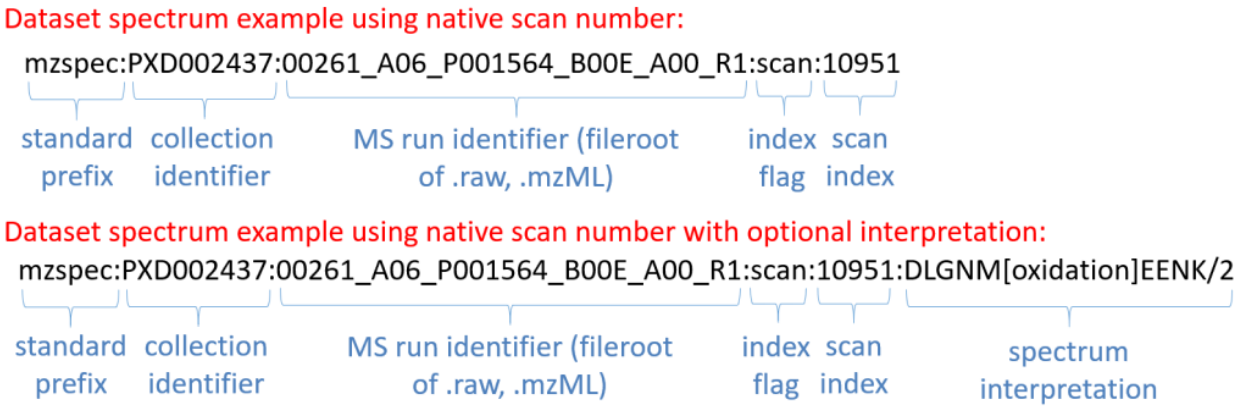
```
curl -X GET "https://www.ebi.ac.uk/pride/multiomics/ws/spectra/findByUsi?usi=NIST
↪%3Acptac2_human_hcd_itraq_selected_part1_2015.msp%3Aindex%3A80003" -H "accept: */*"
```

1.2.3 Python sample code

```
import requests

def main():
 url = 'https://www.ebi.ac.uk/pride/multiomics/ws/spectra/findByUsi?usi=NIST:cptac2_
↪human_hcd_itraq_selected_part1_2015.msp:index:80003'
 response = requests.get(url)
 if response.status_code != 200:
   text = str(response.status_code) + ': ' + response.text
   raise Exception(text)

print(response.text)

if __name__ == "__main__":
 main()
```

1.3 Get Spectra using list of USIs

USI(universal spectrum identifier) is a unique representation for a mass spectrum within a resource or a dataset Documentation. In summary, a USI is a combination of a dataset accession or collection (e.g. spectral library); a file within that collection and an scan or index number withing the file. Inm addition, the USI can contains the information about the peptide sequence and PTMs.



If you want to know more about USI, USI Specification.

1.3.1 Swagger

https://www.ebi.ac.uk/pride/multiomics/ws/swagger-ui/index.html?url=/pride/multiomics/ws/api-docs&configUrl=/pride/multiomics/ws/api-docs/swagger-config#/Spectra/findByMultipleUsisSse

OR

https://www.ebi.ac.uk/pride/multiomics/ws/swagger-ui/index.html?url=/pride/multiomics/ws/api-docs&configUrl=/pride/multiomics/ws/api-docs/swagger-config#/Spectra/findByMultipleUsisStream

1.3.2 Curl

```
curl -X POST "https://www.ebi.ac.uk/pride/multiomics/ws/spectra/sse/findByMultipleUsis
↪" -H "accept: */*" -H "Content-Type: application/json" -d '["NIST:cptac2_human_hcd_
↪itraq_selected_part1_2015.msp:index:80003","NIST:cptac2_human_hcd_itraq_selected_
↪part1_2015.msp:index:80016"]'

OR

curl -X POST "https://www.ebi.ac.uk/pride/multiomics/ws/spectra/stream/
↪findByMultipleUsis" -H "accept: */*" -H "Content-Type: application/json" -d '[
↪"NIST:cptac2_human_hcd_itraq_selected_part1_2015.msp:index:80003","NIST:cptac2_
↪human_hcd_itraq_selected_part1_2015.msp:index:80016"]'
```

1.3.3 Python sample code

Using SSEs

Note: pip install sseclient-py

```
from sseclient import SSEClient   #pip install sseclient-py
import requests

url = 'https://www.ebi.ac.uk/pride/multiomics/ws/spectra/sse/findByMultipleUsis'
headers = {"Content-Type": "application/json"}
data = '["NIST:cptac2_human_hcd_itraq_selected_part1_2015.msp:index:80003",
↪"NIST:cptac2_human_hcd_itraq_selected_part1_2015.msp:index:80016"]'

def main():
    response = requests.post(url, data=data, headers=headers, stream=True)
    if response.status_code != 200:
        text = str(response.status_code) + ': ' + response.text
        raise Exception(text)
    client = SSEClient(response)
    for event in client.events():
        if event.event.lower() == "spectrum":
            print(event.data)
        elif event.event.lower() == "done":
            client.close()
            break


if __name__ == "__main__":
    main()
```

Using Stream

```
import requests

url = 'https://www.ebi.ac.uk/pride/multiomics/ws/spectra/stream/findByMultipleUsis'
headers = {"Content-Type": "application/json"}
data = '["NIST:cptac2_human_hcd_itraq_selected_part1_2015.msp:index:80003",
↪"NIST:cptac2_human_hcd_itraq_selected_part1_2015.msp:index:80016"]'

def main1():
    response = requests.post(url, data=data, headers=headers, stream=True)
    if response.status_code != 200:
        text = str(response.status_code) + ': ' + response.text
        raise Exception(text)
    for line in response.iter_lines():
        if line:
            print(line)


if __name__ == "__main__":
    main1()
```

1.4 Search by peptide sequence

1.4.1 Swagger

We recommend not to use browser for this as the amount of data could be really huge

> **Warning:** 'peptideSequenceRegex' parameter should contain at-least 4 valid characters
>
> valid: AS*DF, ASDF* etc.,
>
> invalid: AS*F, ASF* etc.,

1.4.2 Curl

```
curl -X GET "https://www.ebi.ac.uk/pride/multiomics/ws/spectra/sse/findByPepSequence?
↪peptideSequenceRegex=AVC*KR" -H "accept: */*"

OR

curl -X GET "https://www.ebi.ac.uk/pride/multiomics/ws/spectra/stream/
↪findByPepSequence?peptideSequenceRegex=AVC*KR" -H "accept: */*"
```

1.4.3 Python sample code

Using SSEs

Note: pip install sseclient-py

```
from sseclient import SSEClient   #pip install sseclient-py
import requests

url = 'https://www.ebi.ac.uk/pride/multiomics/ws/spectra/sse/findByPepSequence?
↪peptideSequenceRegex=AVC*KR'

def main():
    response = requests.get(url, stream=True)
    if response.status_code != 200:
        text = str(response.status_code) + ': ' + response.text
        raise Exception(text)
    client = SSEClient(response)
    for event in client.events():
        if event.event.lower() == "spectrum":
            print(event.data)
        elif event.event.lower() == "done":
            client.close()
            break


if __name__ == "__main__":
    main()
```

Using Stream

```
import requests

url = 'https://www.ebi.ac.uk/pride/multiomics/ws/spectra/stream/findByPepSequence?
↪peptideSequenceRegex=AVC*KR'

def main1():
    response = requests.get(url, stream=True)
    if response.status_code != 200:
        text = str(response.status_code) + ': ' + response.text
        raise Exception(text)
    for line in response.iter_lines():
        if line:
            print(line)


if __name__ == "__main__":
    main1()
```

1.5 Search by protein accessions

1.5.1 Swagger

We recommend not to use browser for this as the amount of data could be really huge

1.5.2 Curl

```
curl -X POST "https://www.ebi.ac.uk/pride/multiomics/ws/spectra/sse/
↪findByProteinAccessions" -H "accept: */*" -H "Content-Type: application/json" -d '[
↪"ENSP00000382982.3","P68363","P68366"]'

OR

curl -X POST "https://www.ebi.ac.uk/pride/multiomics/ws/spectra/stream/
↪findByProteinAccessions" -H "accept: */*" -H "Content-Type: application/json" -d '[
↪"ENSP00000382982.3","P68363","P68366"]'
```

1.5.3 Python sample code

Using SSEs

Note: pip install sseclient-py

```
from sseclient import SSEClient    #pip install sseclient-py
import requests

url = 'https://www.ebi.ac.uk/pride/multiomics/ws/spectra/sse/findByProteinAccessions'
```

(continues on next page)

(continued from previous page)

```
headers = {"Content-Type": "application/json"}
data = '["ENSP00000382982.3","P68363","P68366"]'

def main():
    response = requests.post(url, data=data, headers=headers, stream=True)
    if response.status_code != 200:
        text = str(response.status_code) + ': ' + response.text
        raise Exception(text)
    client = SSEClient(response)
    for event in client.events():
        if event.event.lower() == "spectrum":
            print(event.data)
        elif event.event.lower() == "done":
            client.close()
            break


if __name__ == "__main__":
    main()
```

Using Stream

```
import requests

url = 'https://www.ebi.ac.uk/pride/multiomics/ws/spectra/stream/
↪findByProteinAccessions'
headers = {"Content-Type": "application/json"}
data = '["ENSP00000382982.3","P68363","P68366"]'

def main1():
    response = requests.post(url, data=data, headers=headers, stream=True)
    if response.status_code != 200:
        text = str(response.status_code) + ': ' + response.text
        raise Exception(text)
    for line in response.iter_lines():
        if line:
            print(line)


if __name__ == "__main__":
    main1()
```

1.6 Search by gene accessions

1.6.1 Swagger

We recommend not to use browser for this as the amount of data could be really huge

1.6.2 Curl

```
curl -X POST "https://www.ebi.ac.uk/pride/multiomics/ws/spectra/sse/
↪findByGeneAccessions" -H "accept: */*" -H "Content-Type: application/json" -d '[
↪"ENSG00000183785.15","TUBA4A","TUBA8"]'

OR

curl -X POST "https://www.ebi.ac.uk/pride/multiomics/ws/spectra/stream/
↪findByGeneAccessions" -H "accept: */*" -H "Content-Type: application/json" -d '[
↪"ENSG00000183785.15","TUBA4A","TUBA8"]'
```

1.6.3 Python sample code

Using SSEs

Note: pip install sseclient-py

```
from sseclient import SSEClient   #pip install sseclient-py
import requests

url = 'https://www.ebi.ac.uk/pride/multiomics/ws/spectra/sse/findByGeneAccessions'
headers = {"Content-Type": "application/json"}
data = '["ENSG00000183785.15","TUBA4A","TUBA8"]'

def main():
    response = requests.post(url, data=data, headers=headers, stream=True)
    if response.status_code != 200:
        text = str(response.status_code) + ': ' + response.text
        raise Exception(text)
    client = SSEClient(response)
    for event in client.events():
        if event.event.lower() == "spectrum":
            print(event.data)
        elif event.event.lower() == "done":
            client.close()
            break


if __name__ == "__main__":
    main()
```

Using Stream

```
import requests

url = 'https://www.ebi.ac.uk/pride/multiomics/ws/spectra/stream/findByGeneAccessions'
headers = {"Content-Type": "application/json"}
data = '["ENSG00000183785.15","TUBA4A","TUBA8"]'

def main1():
```

(continues on next page)

(continued from previous page)

```
    response = requests.post(url, data=data, headers=headers, stream=True)
    if response.status_code != 200:
        text = str(response.status_code) + ': ' + response.text
        raise Exception(text)
    for line in response.iter_lines():
        if line:
            print(line)


if __name__ == "__main__":
    main1()
```

1.7 Search by ptm & peptide sequence

1.7.1 Swagger

We recommend not to use browser for this as the amount of data could be really huge

1.7.2 Sample request payloads

sample1

```
{
 "peptideSequenceRegex": "AQLG*",
 "positions": [9, 16],
 "ptmKey": "name",
 "ptmValue": "iTRAQ4plex",
 "proteinAccessions": ["P40227", "ENSP00000352019.2"],
 "geneAccessions": ["ENST00000335503.3", "CCT6A"]
}
```

sample2

```
{
 "peptideSequenceRegex": "AQLG*",
 "positions": [9, 16],
 "ptmKey": "accession",
 "ptmValue": "UNIMOD:214",
 "proteinAccessions": ["P40227", "ENSP00000352019.2"],
 "geneAccessions": ["ENST00000335503.3", "CCT6A"]
}
```

sample3

```
 {
  "peptideSequenceRegex": "AQLG*",
  "positions": [9, 16],
  "ptmKey": "mass",
  "ptmValue": "144.102063",
  "proteinAccessions": ["P40227", "ENSP00000352019.2"],
  "geneAccessions": ["ENST00000335503.3", "CCT6A"]
}
```

Note: 'proteinAccessions' & 'geneAccessions' are optional filters.

Warning: 'peptideSequenceRegex' parameter should contain at-least 4 valid characters

valid: AS*DF, ASDF* etc.,

invalid: AS*F, ASF* etc.,

Warning: 'ptmKey' should be one of these: 'name, accession, mass' and 'ptmValue' should be it's corresponding value

1.7.3 Curl

```
curl -X POST "https://www.ebi.ac.uk/pride/multiomics/ws/spectra/sse/findByPtm" -H
↪"accept: */*" -H "Content-Type: application/json" -d '{"peptideSequenceRegex":"AQLG*
↪","positions":[9,16],"ptmKey":"mass","ptmValue":"144.102063"}'

OR

curl -X POST "https://www.ebi.ac.uk/pride/multiomics/ws/spectra/stream/findByPtm" -H
↪"accept: */*" -H "Content-Type: application/json" -d '{"peptideSequenceRegex":"AQLG*
↪","positions":[9,16],"ptmKey":"mass","ptmValue":"144.102063"}'
```

1.7.4 Python sample code

Using SSEs

Note: pip install sseclient-py

```
from sseclient import SSEClient   #pip install sseclient-py
import requests

url = 'https://www.ebi.ac.uk/pride/multiomics/ws/spectra/sse/findByPtm'
headers = {"Content-Type": "application/json"}
data = '{"peptideSequenceRegex":"AQLG*","positions":[9,16],"ptmKey":"mass","ptmValue":
↪"144.102063"}'

def main():
    response = requests.post(url, data=data, headers=headers, stream=True)
    if response.status_code != 200:
        text = str(response.status_code) + ': ' + response.text
        raise Exception(text)
    client = SSEClient(response)
    for event in client.events():
        if event.event.lower() == "spectrum":
            print(event.data)
        elif event.event.lower() == "done":
```

(continues on next page)

(continued from previous page)

```
            client.close()
            break


if __name__ == "__main__":
    main()
```

Using Stream

```
import requests

url = 'https://www.ebi.ac.uk/pride/multiomics/ws/spectra/stream/findByPtm'
headers = {"Content-Type": "application/json"}
data = '{"peptideSequenceRegex":"AQLG*","positions":[9,16],"ptmKey":"mass","ptmValue":
↪"144.102063"}'

def main1():
    response = requests.post(url, data=data, headers=headers, stream=True)
    if response.status_code != 200:
        text = str(response.status_code) + ': ' + response.text
        raise Exception(text)
    for line in response.iter_lines():
        if line:
            print(line)


if __name__ == "__main__":
    main1()
```

1.8 Search by one or more filters

1.8.1 Swagger

https://www.ebi.ac.uk/pride/multiomics/ws/swagger-ui/index.html?url=/pride/multiomics/ws/api-docs&configUrl=/pride/multiomics/ws/api-docs/swagger-config#/Spectra/findByGenericRequest

To get just total count : https://www.ebi.ac.uk/pride/multiomics/ws/swagger-ui/index.html?url=/pride/multiomics/ws/api-docs&configUrl=/pride/multiomics/ws/api-docs/swagger-config#/Spectra/findByGenericRequestCount

1.8.2 Sample request payloads

sample

```
{
 "peptideSequenceRegex": "AQLG*",
 "ptm": {
   "ptmKey": "name",
   "ptmValue": "iTRAQ4plex"
 },
  "proteinAccessions": ["P40227", "ENSP00000352019.2"],
```

(continues on next page)

(continued from previous page)

```
  "geneAccessions": ["ENST00000335503.3", "CCT6A"]
}
```

Warning: Any one filter is mandatory i.e., either 'peptideSequenceRegex' or 'ptm' or 'proteinAccessions' or 'geneAccessions'

Warning: 'ptmKey' should be one of these: 'name, accession, mass' and 'ptmValue' should be it's corresponding value

Warning: 'peptideSequenceRegex' parameter should contain at-least 4 valid characters

valid: AS*DF, ASDF* etc.,

invalid: AS*F, ASF* etc.,

1.8.3 Curl

```
SSE: curl -X POST "https://www.ebi.ac.uk/pride/multiomics/ws/spectra/see/
↪findByGenericRequest" -H "accept: */*" -H "Content-Type: application/json" -d '{
↪"peptideSequenceRegex":"AQLG*","ptm":{"ptmKey":"name","ptmValue":"iTRAQ4plex"},
↪"proteinAccessions":["P40227","ENSP00000352019.2"],"geneAccessions":[
↪"ENST00000335503.3","CCT6A"]}'

OR

Streams: curl -X POST "https://www.ebi.ac.uk/pride/multiomics/ws/spectra/stream/
↪findByGenericRequest" -H "accept: */*" -H "Content-Type: application/json" -d '{
↪"peptideSequenceRegex":"AQLG*","ptm":{"ptmKey":"name","ptmValue":"iTRAQ4plex"},
↪"proteinAccessions":["P40227","ENSP00000352019.2"],"geneAccessions":[
↪"ENST00000335503.3","CCT6A"]}'
```

1.8.4 Python sample code

Using SSEs

Note: pip install sseclient-py

```
from sseclient import SSEClient    #pip install sseclient-py
import requests

url = 'curl -X POST "https://www.ebi.ac.uk/pride/multiomics/ws/spectra/see/
↪findByGenericRequest'
headers = {"Content-Type": "application/json"}
data = '{"peptideSequenceRegex":"AQLG*","ptm":{"ptmKey":"name","ptmValue":"iTRAQ4plex
↪"},"proteinAccessions":["P40227","ENSP00000352019.2"],"geneAccessions":[
↪"ENST00000335503.3","CCT6A"]}'
```

(continues on next page)

(continued from previous page)

```
def main():
    response = requests.post(url, data=data, headers=headers, stream=True)
    if response.status_code != 200:
        text = str(response.status_code) + ': ' + response.text
        raise Exception(text)
    client = SSEClient(response)
    for event in client.events():
        if event.event.lower() == "spectrum":
            print(event.data)
        elif event.event.lower() == "done":
            client.close()
            break


if __name__ == "__main__":
    main()
```

Using Stream

```
import requests

url = 'curl -X POST "https://www.ebi.ac.uk/pride/multiomics/ws/spectra/stream/
↪findByGenericRequest'
headers = {"Content-Type": "application/json"}
data = '{"peptideSequenceRegex":"AQLG*","ptm":{"ptmKey":"name","ptmValue":"iTRAQ4plex
↪"},"proteinAccessions":["P40227","ENSP00000352019.2"],"geneAccessions":[
↪"ENST00000335503.3","CCT6A"]}'

def main1():
    response = requests.post(url, data=data, headers=headers, stream=True)
    if response.status_code != 200:
        text = str(response.status_code) + ': ' + response.text
        raise Exception(text)
    for line in response.iter_lines():
        if line:
            print(line)


if __name__ == "__main__":
    main1()
```