
omnipath
Release master (1.0.4)

Michal Klein, Dénes Türei

Aug 16, 2021

GENERAL

1	Installation	3
1.1	PyPI	3
1.2	Development Version	3
2	API	5
2.1	Requests	5
2.2	Interactions	11
2.3	Other	27
3	Release Notes	31
3.1	Version 1.0	31
3.2	Version 0.0	32
4	References	33
	Bibliography	35
	Index	37

This package is a Python equivalent of an R package [OmnipathR](#) for accessing web service of *OmniPath* database developed by [Saezlab](#).

INSTALLATION

Omnipath requires Python version ≥ 3.7 to run.

1.1 PyPI

Omnipath is also available on PyPI:

```
pip install omnipath
```

Additionally, `omnipath` may sometimes require `networkx` to create an interaction graph. This dependency can be installed as:

```
pip install omnipath[graph]
```

1.2 Development Version

To stay up-to-date with the newest version, run:

```
pip install git+https://github.com/saezlab/omnipath
```


Import Omnipath as:

```
import omnipath as op
```

2.1 Requests

<i>requests.Annotations()</i>	Request annotations from [Omnipath].
<i>requests.Complexes()</i>	Request information about protein complexes from [Omnipath].
<i>requests.Enzsub()</i>	Request enzyme-substrate relationships from [Omnipath].
<i>requests.Intercell()</i>	Request <i>intercell</i> annotations from [Omnipath].
<i>requests.SignedPTMs()</i>	Request enzyme-substrate relationships and interactions from [Omnipath].

2.1.1 omnipath.requests.Annotations

class `omnipath.requests.Annotations`
Request annotations from [Omnipath].

Methods

<i>get</i> ([resources, force_full_download])	Import annotations from [Omnipath].
<i>params</i> ()	Return the available values for each parameter, if available.
<i>resources</i> (**kwargs)	Return the available resources for this query.

omnipath.requests.Annotations.get

classmethod `Annotations.get` (*resources=None, force_full_download=False, **kwargs*)

Import annotations from [OmniPath].

Retrieves protein annotations about function, localization, expression, structure and other properties of proteins from OmniPath.

Parameters

- **proteins** *¶* (`Union[str, Iterable[str], None]`) – Genes or proteins for which annotations will be retrieved (UniProt IDs, HGNC Gene Symbols or miRBase IDs).

In order to download annotations for proteins complexes, write ‘**COMPLEX:**’ before the gene symbols of the genes integrating the complex.

If *None*, fetch annotations for all available genes or proteins.

- **resources** *¶* (`Union[str, Iterable[str], None]`) – Load the annotations only from these databases. See *resources()* for available options. If *None*, use all available resources.
- **force_full_download** *¶* (`bool`) – Force the download of the entire annotations dataset. The full size of the data is ~1GB. We recommend to retrieve the annotations for a set of proteins or only from a few resources, depending on your interest.
- **kwargs** *¶* – Additional query parameters.

Returns A dataframe containing different gene/complex annotations.

Return type `pandas.DataFrame`

Notes

There might be also a few miRNAs annotated. A vast majority of protein complex annotations are inferred from the annotations of the members: if all members carry the same annotation the complex inherits.

omnipath.requests.Annotations.params

classmethod `Annotations.params()`

Return the available values for each parameter, if available.

Return type `Dict[str, Any]`

omnipath.requests.Annotations.resources

classmethod `Annotations.resources(**kwargs)`

Return the available resources for this query.

Return type `Tuple[str]`

2.1.2 omnipath.requests.Complexes

class `omnipath.requests.Complexes`

Request information about protein complexes from [OmniPath].

Methods

<code>complex_genes(genes[, total_match])</code>	<code>complexes,</code>	<code>to-</code>	Get all the molecular complexes for a given <code>genes</code> .
<code>get()</code>			Perform a request to the [OmniPath] web service.
<code>params()</code>			Return the available values for each parameter, if available.
<code>resources(**kwargs)</code>			Return the available resources for this query.

`omnipath.requests.Complexes.complex_genes`

classmethod `Complexes.complex_genes (genes, complexes=None, total_match=False)`

Get all the molecular complexes for a given `genes`.

This function returns all the molecular complexes where an input set of genes participate. User can choose to retrieve every complex where any of the input genes participate or just retrieve these complexes where all the genes in input set participate together.

Parameters

- **genes** `// (Union[str, Iterable[str]])` – The genes for which complexes will be retrieved (hgnc format).
- **complexes** `// (Optional[DataFrame])` – Complex data from `get ()`. If `None`, new request will be made.
- **total_match** `// (bool)` – If `True`, get only complexes where all the genes participate together, otherwise get complexes where any of the genes participate.

Returns The filtered `complexes`.

Return type `pandas.DataFrame`

`omnipath.requests.Complexes.get`

classmethod `Complexes.get ()`

Perform a request to the [OmniPath] web service.

Parameters **kwargs** `//` – Additional query parameters.

Returns The result which depends the type of the request and the supplied parameters.

Return type `pandas.DataFrame`

omnipath.requests.Complexes.params

classmethod `Complexes.params()`
 Return the available values for each parameter, if available.
Return type `Dict[str, Any]`

omnipath.requests.Complexes.resources

classmethod `Complexes.resources(**kwargs)`
 Return the available resources for this query.
Return type `Tuple[str]`

2.1.3 omnipath.requests.Enzsub

class `omnipath.requests.Enzsub`
 Request enzyme-substrate relationships from [OmniPath].
 Imports the enzyme-substrate (more exactly, enzyme-PTM) relationships `database`.

Methods

<code>get()</code>	Perform a request to the [OmniPath] web service.
<code>params()</code>	Return the available values for each parameter, if available.
<code>resources(**kwargs)</code>	Return the available resources for this query.

omnipath.requests.Enzsub.get

classmethod `Enzsub.get()`
 Perform a request to the [OmniPath] web service.
Parameters `kwargs` – Additional query parameters.
Returns The result which depends the type of the request and the supplied parameters.
Return type `pandas.DataFrame`

omnipath.requests.Enzsub.params

classmethod `Enzsub.params()`
 Return the available values for each parameter, if available.
Return type `Dict[str, Any]`

omnipath.requests.Enzsub.resources

classmethod Enzsub.**resources** (**kwargs)
Return the available resources for this query.

Return type Tuple[str]

2.1.4 omnipath.requests.Intercell

class omnipath.requests.**Intercell**

Request *intercell* annotations from [OmniPath].

Imports the [OmniPath] inter-cellular communication role annotation database.

It provides information on the roles in inter-cellular signaling, e.g. if a protein is a ligand, a receptor, an extracellular matrix (ECM) component, etc.

Methods

<i>categories</i> ()	Return categories from the <i>intercell</i> database.
<i>generic_categories</i> ()	Return generic categories from the <i>intercell</i> database.
<i>get</i> ()	Perform a request to the [OmniPath] web service.
<i>params</i> ()	Return the available values for each parameter, if available.
<i>resources</i> ([generic_categories])	Return the resources falling into the specified generic categories.

omnipath.requests.Intercell.categories

classmethod Intercell.**categories** ()
Return categories from the *intercell* database.

Return type Tuple[str]

omnipath.requests.Intercell.generic_categories

classmethod Intercell.**generic_categories** ()
Return generic categories from the *intercell* database.

Return type Tuple[str]

omnipath.requests.Intercell.get

classmethod `Intercell.get()`

Perform a request to the [OmniPath] web service.

Parameters `kwargs` – Additional query parameters.

Returns The result which depends the type of the request and the supplied parameters.

Return type `pandas.DataFrame`

omnipath.requests.Intercell.params

classmethod `Intercell.params()`

Return the available values for each parameter, if available.

Return type `Dict[str, Any]`

omnipath.requests.Intercell.resources

classmethod `Intercell.resources (generic_categories=None)`

Return the resources falling into the specified generic categories.

Parameters `generic_categories` (`Union[str, Sequence[str], None]`) – For valid options, see `generic_categories`.

Returns The filtered resources according to `generic_categories`.

Return type `tuple`

2.1.5 omnipath.requests.SignedPTMs

class `omnipath.requests.SignedPTMs`

Request enzyme-substrate relationships and interactions from [OmniPath].

PTM data does not contain sign (activation/inhibition), we generate this information based on the interaction network.

Methods

<code>get([interactions])</code>	Get signs for enzyme-PTM interactions.
<code>graph([data])</code>	Create a graph.
<code>params()</code>	Return the available values for each parameter, if available.
<code>resources(**kwargs)</code>	Return the available resources for this query.

omnipath.requests.SignedPTMs.get

classmethod SignedPTMs.get (*interactions=None*)

Get signs for enzyme-PTM interactions.

Parameters

- **ptms** (Optional[DataFrame]) – Data generated by *omnipath.requests.Enzsub.get()*. If *None*, a new request will be performed.
- **interactions** (Optional[DataFrame]) – Data generated by *omnipath.interactions.OmniPath.get()*. If *None*, a new request will be performed.

Returns The signed PTMs with columns ‘is_inhibition’ and ‘is_stimulation’.

Return type pandas.DataFrame

omnipath.requests.SignedPTMs.graph

classmethod SignedPTMs.graph (*data=None, **kwargs*)

Create a graph.

Parameters

- **data** (Optional[DataFrame]) – The interaction data. If *None*, create a new request.
- **kwargs** – Keyword arguments for *get()* if *data = None*.

Returns The interaction graph.

Return type networkx.DiGraph

omnipath.requests.SignedPTMs.params

classmethod SignedPTMs.params ()

Return the available values for each parameter, if available.

Return type Dict[str, Any]

omnipath.requests.SignedPTMs.resources

classmethod SignedPTMs.resources (***kwargs*)

Return the available resources for this query.

Return type Tuple[str]

2.2 Interactions

<i>interactions.AllInteractions</i> ([include, exclude])	Request all [OmniPath] interaction datasets.
<i>interactions.Dorothea</i> ()	Request interactions from the <i>dorothea</i> dataset.
<i>interactions.KinaseExtra</i> ()	Request interactions from the <i>kinase extra</i> dataset.
<i>interactions.LigRecExtra</i> ()	Request interactions from the <i>ligrec extra</i> dataset.
<i>interactions.OmniPath</i> ()	Request interactions from the <i>omnipath</i> dataset.

continues on next page

Table 7 – continued from previous page

<code>interactions.PathwayExtra()</code>	Request interactions from the <i>pathway extra</i> dataset.
<code>interactions.PostTranslational([exclude])</code>	Request all post-translational interactions from [OmniPath].
<code>interactions.TFmiRNA()</code>	Request interactions from the <i>TF-miRNA</i> dataset.
<code>interactions.TFtarget()</code>	Request interactions from the <i>TF-target</i> dataset.
<code>interactions.Transcriptional()</code>	Request all <i>TF-target</i> interactions from [OmniPath].
<code>interactions.lncRNAmRNA()</code>	Request interactions from the <i>lncRNA-mRNA</i> dataset.
<code>interactions.miRNA()</code>	Request interactions from the <i>miRNA-target</i> dataset.
<code>interactions.import_intercell_network([. Import intercellular network combining intercellular annotations and protein interactions.</code>	

2.2.1 omnipath.interactions.AllInteractions

class `omnipath.interactions.AllInteractions` (*include=None, exclude=None*)
Request all [OmniPath] interaction datasets.

The available interaction datasets are `omnipath.constants.InteractionDataset`.

Parameters

- **include** `//` (`Union[str, InteractionDataset, Sequence[str], Sequence[InteractionDataset], None]`) – Interactions datasets to include from the [OmniPath] database. If *None*, include everything.
- **exclude** `//` (`Union[str, InteractionDataset, Sequence[str], Sequence[InteractionDataset], None]`) – Interaction datasets to exclude from the [OmniPath] database. If *None*, don't exclude anything.

Methods

<code>get([exclude])</code>	Perform a request to the [OmniPath] web service.
<code>graph([data])</code>	Create a graph.
<code>params()</code>	Return the available values for each parameter, if available.
<code>resources(**kwargs)</code>	Return the available resources for this query.

omnipath.interactions.AllInteractions.get

classmethod `AllInteractions.get` (*exclude=None, **kwargs*)
Perform a request to the [OmniPath] web service.

The available interaction datasets are `omnipath.constants.InteractionDataset`.

Parameters

- **include** `//` (`Union[str, InteractionDataset, Sequence[str], Sequence[InteractionDataset], None]`) – Interactions datasets to include from the [OmniPath] database. If *None*, include everything.
- **exclude** `//` (`Union[str, InteractionDataset, Sequence[str], Sequence[InteractionDataset], None]`) – Interaction datasets to exclude from the [OmniPath] database. If *None*, don't exclude anything.

- **kwargs** – Additional query parameters.

Returns The result which depends the type of the request and the supplied parameters.

Return type `pandas.DataFrame`

omnipath.interactions.AllInteractions.graph

classmethod `AllInteractions.graph` (*data=None, **kwargs*)

Create a graph.

Parameters

- **data** (Optional[DataFrame]) – The interaction data. If *None*, create a new request.
- **kwargs** – Keyword arguments for `get()` if *data = None*.

Returns The interaction graph.

Return type `networkx.DiGraph`

omnipath.interactions.AllInteractions.params

classmethod `AllInteractions.params` ()

Return the available values for each parameter, if available.

Return type `Dict[str, Any]`

omnipath.interactions.AllInteractions.resources

classmethod `AllInteractions.resources` (***kwargs*)

Return the available resources for this query.

Return type `Tuple[str]`

2.2.2 omnipath.interactions.Dorothea

class `omnipath.interactions.Dorothea`

Request interactions from the *dorothea* dataset.

Imports the `dataset` which contains transcription factor (TF)-target interactions from `DoRothEA`.

Methods

<code>get()</code>	Perform a request to the [OmniPath] web service.
<code>graph([data])</code>	Create a graph.
<code>params()</code>	Return the available values for each parameter, if available.
<code>resources(**kwargs)</code>	Return the available resources for this query.

omnipath.interactions.Dorothea.get

classmethod `Dorothea.get()`

Perform a request to the [OmniPath] web service.

Parameters `kwargs` – Additional query parameters.

Returns The result which depends the type of the request and the supplied parameters.

Return type `pandas.DataFrame`

omnipath.interactions.Dorothea.graph

classmethod `Dorothea.graph(data=None, **kwargs)`

Create a graph.

Parameters

- `data` (Optional[DataFrame]) – The interaction data. If `None`, create a new request.
- `kwargs` – Keyword arguments for `get()` if `data = None`.

Returns The interaction graph.

Return type `networkx.DiGraph`

omnipath.interactions.Dorothea.params

classmethod `Dorothea.params()`

Return the available values for each parameter, if available.

Return type `Dict[str, Any]`

omnipath.interactions.Dorothea.resources

classmethod `Dorothea.resources(**kwargs)`

Return the available resources for this query.

Return type `Tuple[str]`

2.2.3 omnipath.interactions.KinaseExtra

class `omnipath.interactions.KinaseExtra`

Request interactions from the *kinase extra* dataset.

Imports the `dataset` which contains enzyme-substrate interactions without literature reference.

The enzyme-substrate interactions supported by literature references are part of the `omnipath.requests.AllInteractions`.

Methods

<code>get()</code>	Perform a request to the [OmniPath] web service.
<code>graph([data])</code>	Create a graph.
<code>params()</code>	Return the available values for each parameter, if available.
<code>resources(**kwargs)</code>	Return the available resources for this query.

omnipath.interactions.KinaseExtra.get

classmethod `KinaseExtra.get()`

Perform a request to the [OmniPath] web service.

Parameters `kwargs` – Additional query parameters.

Returns The result which depends the type of the request and the supplied parameters.

Return type `pandas.DataFrame`

omnipath.interactions.KinaseExtra.graph

classmethod `KinaseExtra.graph(data=None, **kwargs)`

Create a graph.

Parameters

- `data` (Optional[DataFrame]) – The interaction data. If *None*, create a new request.
- `kwargs` – Keyword arguments for `get()` if `data = None`.

Returns The interaction graph.

Return type `networkx.DiGraph`

omnipath.interactions.KinaseExtra.params

classmethod `KinaseExtra.params()`

Return the available values for each parameter, if available.

Return type `Dict[str, Any]`

omnipath.interactions.KinaseExtra.resources

classmethod `KinaseExtra.resources(**kwargs)`

Return the available resources for this query.

Return type `Tuple[str]`

2.2.4 omnipath.interactions.LigRecExtra

class `omnipath.interactions.LigRecExtra`

Request interactions from the *ligrec extra* dataset.

Imports the `dataset` which contains ligand-receptor interactions without literature reference.

Methods

<code>get()</code>	Perform a request to the [OmniPath] web service.
<code>graph([data])</code>	Create a graph.
<code>params()</code>	Return the available values for each parameter, if available.
<code>resources(**kwargs)</code>	Return the available resources for this query.

`omnipath.interactions.LigRecExtra.get`

classmethod `LigRecExtra.get()`

Perform a request to the [OmniPath] web service.

Parameters `kwargs` – Additional query parameters.

Returns The result which depends the type of the request and the supplied parameters.

Return type `pandas.DataFrame`

`omnipath.interactions.LigRecExtra.graph`

classmethod `LigRecExtra.graph(data=None, **kwargs)`

Create a graph.

Parameters

- `data` (Optional[DataFrame]) – The interaction data. If `None`, create a new request.
- `kwargs` – Keyword arguments for `get()` if `data = None`.

Returns The interaction graph.

Return type `networkx.DiGraph`

`omnipath.interactions.LigRecExtra.params`

classmethod `LigRecExtra.params()`

Return the available values for each parameter, if available.

Return type `Dict[str, Any]`

omnipath.interactions.LigRecExtra.resources

classmethod `LigRecExtra.resources (**kwargs)`
Return the available resources for this query.

Return type `Tuple[str]`

2.2.5 omnipath.interactions.OmniPath

class `omnipath.interactions.OmniPath`
Request interactions from the *omnipath* dataset.

Imports the `database`, which contains only interactions supported by literature references.

This part of the interaction database was compiled in a similar way as it has been presented in [OmniPath16].

Methods

<code>get()</code>	Perform a request to the [OmniPath] web service.
<code>graph([data])</code>	Create a graph.
<code>params()</code>	Return the available values for each parameter, if available.
<code>resources(**kwargs)</code>	Return the available resources for this query.

omnipath.interactions.OmniPath.get

classmethod `OmniPath.get()`
Perform a request to the [OmniPath] web service.

Parameters `kwargs` – Additional query parameters.

Returns The result which depends the type of the request and the supplied parameters.

Return type `pandas.DataFrame`

omnipath.interactions.OmniPath.graph

classmethod `OmniPath.graph (data=None, **kwargs)`
Create a graph.

Parameters

- `data` (Optional[DataFrame]) – The interaction data. If *None*, create a new request.
- `kwargs` – Keyword arguments for `get()` if `data = None`.

Returns The interaction graph.

Return type `networkx.DiGraph`

omnipath.interactions.OmniPath.params

classmethod `OmniPath.params()`
 Return the available values for each parameter, if available.
Return type `Dict[str, Any]`

omnipath.interactions.OmniPath.resources

classmethod `OmniPath.resources(**kwargs)`
 Return the available resources for this query.
Return type `Tuple[str]`

2.2.6 omnipath.interactions.PathwayExtra

class `omnipath.interactions.PathwayExtra`
 Request interactions from the *pathway extra* dataset.
 Imports the `dataset` which contains activity flow interactions without literature reference.

Methods

<code>get()</code>	Perform a request to the [OmniPath] web service.
<code>graph([data])</code>	Create a graph.
<code>params()</code>	Return the available values for each parameter, if available.
<code>resources(**kwargs)</code>	Return the available resources for this query.

omnipath.interactions.PathwayExtra.get

classmethod `PathwayExtra.get()`
 Perform a request to the [OmniPath] web service.
Parameters `kwargs` – Additional query parameters.
Returns The result which depends the type of the request and the supplied parameters.
Return type `pandas.DataFrame`

omnipath.interactions.PathwayExtra.graph

classmethod `PathwayExtra.graph(data=None, **kwargs)`
 Create a graph.
Parameters

- `data` (Optional[DataFrame]) – The interaction data. If *None*, create a new request.
- `kwargs` – Keyword arguments for `get()` if `data = None`.

Returns The interaction graph.
Return type `networkx.DiGraph`

omnipath.interactions.PathwayExtra.params

classmethod PathwayExtra.params ()
Return the available values for each parameter, if available.

Return type Dict[str, Any]

omnipath.interactions.PathwayExtra.resources

classmethod PathwayExtra.resources (**kwargs)
Return the available resources for this query.

Return type Tuple[str]

2.2.7 omnipath.interactions.PostTranslational

class omnipath.interactions.PostTranslational (exclude=None)
Request all post-translational interactions from [OmniPath] .

Imports the `dataset` which contains post-transcriptional (i.e. protein-protein) interactions. This query requests the interactions from the following datasets:

- `omnipath.constants.InteractionDataset.OMNIPATH`
- `omnipath.constants.InteractionDataset.PATHWAY_EXTRA`
- `omnipath.constants.InteractionDataset.KINASE_EXTRA`
- `omnipath.constants.InteractionDataset.LIGREC_EXTRA`

Parameters `exclude` (Union[str, InteractionDataset, Sequence[str], Sequence[InteractionDataset], None]) – Post-translational interaction datasets to exclude. If *None*, don't exclude anything.

Methods

<code>get(**kwargs)</code>	Perform a request to the [OmniPath] web service.
<code>graph([data])</code>	Create a graph.
<code>params()</code>	Return the available values for each parameter, if available.
<code>resources(**kwargs)</code>	Return the available resources for this query.

omnipath.interactions.PostTranslational.get

classmethod PostTranslational.get (**kwargs)
Perform a request to the [OmniPath] web service.

The available interaction datasets are `omnipath.constants.InteractionDataset`.

Parameters

- **exclude** (Union[str, InteractionDataset, Sequence[str], Sequence[InteractionDataset], None]) – Post-translational interaction datasets to exclude. If *None*, don't exclude anything.

- `kwargs` – Additional query parameters.

Returns The result which depends the type of the request and the supplied parameters.

Return type `pandas.DataFrame`

omnipath.interactions.PostTranslational.graph

classmethod `PostTranslational.graph` (`data=None, **kwargs`)

Create a graph.

Parameters

- `data` (Optional[DataFrame]) – The interaction data. If `None`, create a new request.
- `kwargs` – Keyword arguments for `get()` if `data = None`.

Returns The interaction graph.

Return type `networkx.DiGraph`

omnipath.interactions.PostTranslational.params

classmethod `PostTranslational.params` ()

Return the available values for each parameter, if available.

Return type `Dict[str, Any]`

omnipath.interactions.PostTranslational.resources

classmethod `PostTranslational.resources` (`**kwargs`)

Return the available resources for this query.

Return type `Tuple[str]`

2.2.8 omnipath.interactions.TFmiRNA

class `omnipath.interactions.TFmiRNA`

Request interactions from the *TF-miRNA* dataset.

Imports the `dataset` which contains transcription factor-miRNA gene interactions.

Methods

<code>get()</code>	Perform a request to the [OmniPath] web service.
<code>graph([data])</code>	Create a graph.
<code>params()</code>	Return the available values for each parameter, if available.
<code>resources(**kwargs)</code>	Return the available resources for this query.

omnipath.interactions.TFmiRNA.get

classmethod `TFmiRNA.get()`

Perform a request to the [OmniPath] web service.

Parameters `kwargs` – Additional query parameters.

Returns The result which depends the type of the request and the supplied parameters.

Return type `pandas.DataFrame`

omnipath.interactions.TFmiRNA.graph

classmethod `TFmiRNA.graph(data=None, **kwargs)`

Create a graph.

Parameters

- `data` (Optional[DataFrame]) – The interaction data. If `None`, create a new request.
- `kwargs` – Keyword arguments for `get()` if `data = None`.

Returns The interaction graph.

Return type `networkx.DiGraph`

omnipath.interactions.TFmiRNA.params

classmethod `TFmiRNA.params()`

Return the available values for each parameter, if available.

Return type `Dict[str, Any]`

omnipath.interactions.TFmiRNA.resources

classmethod `TFmiRNA.resources(**kwargs)`

Return the available resources for this query.

Return type `Tuple[str]`

2.2.9 omnipath.interactions.TFtarget

class `omnipath.interactions.TFtarget`

Request interactions from the *TF-target* dataset.

Imports the `dataset` which contains transcription factor-target protein coding gene interactions.

Other TF-target datasets in `omnipath` are `omnipath.interactions.Dorothea` and `omnipath.interactions.TFmiRNA` which provides TF-miRNA gene interactions.

Methods

<code>get()</code>	Perform a request to the [OmniPath] web service.
<code>graph([data])</code>	Create a graph.
<code>params()</code>	Return the available values for each parameter, if available.
<code>resources(**kwargs)</code>	Return the available resources for this query.

omnipath.interactions.TFtarget.get

classmethod `TFtarget.get()`

Perform a request to the [OmniPath] web service.

Parameters `kwargs` – Additional query parameters.

Returns The result which depends the type of the request and the supplied parameters.

Return type `pandas.DataFrame`

omnipath.interactions.TFtarget.graph

classmethod `TFtarget.graph(data=None, **kwargs)`

Create a graph.

Parameters

- `data` (Optional[DataFrame]) – The interaction data. If *None*, create a new request.
- `kwargs` – Keyword arguments for `get()` if `data = None`.

Returns The interaction graph.

Return type `networkx.DiGraph`

omnipath.interactions.TFtarget.params

classmethod `TFtarget.params()`

Return the available values for each parameter, if available.

Return type `Dict[str, Any]`

omnipath.interactions.TFtarget.resources

classmethod `TFtarget.resources(**kwargs)`

Return the available resources for this query.

Return type `Tuple[str]`

2.2.10 omnipath.interactions.Transcriptional

class `omnipath.interactions.Transcriptional`

Request all *TF-target* interactions from [OmniPath].

Imports the `dataset` which contains transcription factor-target protein coding gene interactions.

Methods

<code>get()</code>	Perform a request to the [OmniPath] web service.
<code>graph([data])</code>	Create a graph.
<code>params()</code>	Return the available values for each parameter, if available.
<code>resources(**kwargs)</code>	Return the available resources for this query.

omnipath.interactions.Transcriptional.get

classmethod `Transcriptional.get()`

Perform a request to the [OmniPath] web service.

Parameters `kwargs` – Additional query parameters.

Returns The result which depends the type of the request and the supplied parameters.

Return type `pandas.DataFrame`

omnipath.interactions.Transcriptional.graph

classmethod `Transcriptional.graph(data=None, **kwargs)`

Create a graph.

Parameters

- `data` (Optional[DataFrame]) – The interaction data. If `None`, create a new request.
- `kwargs` – Keyword arguments for `get()` if `data = None`.

Returns The interaction graph.

Return type `networkx.DiGraph`

omnipath.interactions.Transcriptional.params

classmethod `Transcriptional.params()`

Return the available values for each parameter, if available.

Return type `Dict[str, Any]`

omnipath.interactions.Transcriptional.resources

classmethod `Transcriptional.resources (**kwargs)`
 Return the available resources for this query.

Return type `Tuple[str]`

2.2.11 omnipath.interactions.lncRNAmRNA

class `omnipath.interactions.lncRNAmRNA`
 Request interactions from the *lncRNA-mRNA* dataset.
 Imports the `dataset` which contains lncRNA-mRNA interactions.

Methods

<code>get()</code>	Perform a request to the [OmniPath] web service.
<code>graph([data])</code>	Create a graph.
<code>params()</code>	Return the available values for each parameter, if available.
<code>resources(**kwargs)</code>	Return the available resources for this query.

omnipath.interactions.lncRNAmRNA.get

classmethod `lncRNAmRNA.get ()`
 Perform a request to the [OmniPath] web service.

Parameters `kwargs` – Additional query parameters.

Returns The result which depends the type of the request and the supplied parameters.

Return type `pandas.DataFrame`

omnipath.interactions.lncRNAmRNA.graph

classmethod `lncRNAmRNA.graph (data=None, **kwargs)`
 Create a graph.

Parameters

- `data` (Optional[DataFrame]) – The interaction data. If *None*, create a new request.
- `kwargs` – Keyword arguments for `get ()` if `data = None`.

Returns The interaction graph.

Return type `networkx.DiGraph`

omnipath.interactions.lncRNAmRNA.params**classmethod** `lncRNAmRNA.params()`

Return the available values for each parameter, if available.

Return type `Dict[str, Any]`**omnipath.interactions.lncRNAmRNA.resources****classmethod** `lncRNAmRNA.resources(**kwargs)`

Return the available resources for this query.

Return type `Tuple[str]`**2.2.12 omnipath.interactions.miRNA****class** `omnipath.interactions.miRNA`Request interactions from the *miRNA-target* dataset.Imports the `dataset`. which contains miRNA-mRNA interactions.**Methods**

<code>get()</code>	Perform a request to the [OmniPath] web service.
<code>graph([data])</code>	Create a graph.
<code>params()</code>	Return the available values for each parameter, if available.
<code>resources(**kwargs)</code>	Return the available resources for this query.

omnipath.interactions.miRNA.get**classmethod** `miRNA.get()`

Perform a request to the [OmniPath] web service.

Parameters `kwargs` – Additional query parameters.**Returns** The result which depends the type of the request and the supplied parameters.**Return type** `pandas.DataFrame`**omnipath.interactions.miRNA.graph****classmethod** `miRNA.graph(data=None, **kwargs)`

Create a graph.

Parameters

- `data` (Optional[DataFrame]) – The interaction data. If *None*, create a new request.
- `kwargs` – Keyword arguments for `get()` if `data = None`.

Returns The interaction graph.**Return type** `networkx.DiGraph`

omnipath.interactions.miRNA.params**classmethod** `miRNA.params()`

Return the available values for each parameter, if available.

Return type `Dict[str, Any]`**omnipath.interactions.miRNA.resources****classmethod** `miRNA.resources(**kwargs)`

Return the available resources for this query.

Return type `Tuple[str]`**2.2.13 omnipath.interactions.import_intercell_network**

```
omnipath.interactions.import_intercell_network (include=(<InteractionDataset.OMNIPATH>,
<Interaction-
Dataset.PATHWAY_EXTRA>, <Interac-
tionDataset.KINASE_EXTRA>, <Inter-
actionDataset.LIGREC_EXTRA>),
interactions_params=None,
transmitter_params=None,      re-
ceiver_params=None)
```

Import intercellular network combining intercellular annotations and protein interactions.

First, it imports a network of protein-protein interactions. Then, it retrieves annotations about the proteins intercellular communication roles, once for the transmitter (delivering information from the expressing cell) and second, the receiver (receiving signal and relaying it towards the expressing cell) side.

These 3 queries can be customized by providing parameters which will be passed to `omnipath.interactions.OmniPath.get()` for the network and `omnipath.requests.Intercell()` for the annotations.

Finally the 3 `pandas.DataFrame` are combined in a way that the source proteins in each interaction annotated by the transmitter, and the target proteins by the receiver categories. If undirected interactions present (these are disabled by default) they will be duplicated, i.e. both partners can be both receiver and transmitter.

Parameters

- **include** `// (Union[str, InteractionDataset, Sequence[str], Sequence[InteractionDataset]])` – Interaction datasets to include for `omnipath.interactions.AllInteractions.get()`.
- **interactions_params** `// (Optional[Mapping[str, Any]])` – Parameters for the `omnipath.interactions.AllInteractions.get()`.
- **transmitter_params** `// (Optional[Mapping[str, Any]])` – Parameters defining the transmitter side of intercellular connections. See `omnipath.interactions.AllInteractions.params()` for available values.
- **receiver_params** `// (Optional[Mapping[str, Any]])` – Parameters defining the receiver side of intercellular connections. See `omnipath.interactions.AllInteractions.params()` for available values.

Returns A dataframe containing information about protein-protein interactions and the inter-cellular roles of the proteins involved in those interactions.

Return type `pandas.DataFrame`

2.3 Other

2.3.1 Constants

<code>constants.InteractionDataset(*args, **kw)</code>	Available interaction datasets in [OmniPath].
<code>constants.License(*args, **kw)</code>	License types.
<code>constants.Organism(*args, **kw)</code>	Organism types.

omnipath.constants.InteractionDataset

enum `omnipath.constants.InteractionDataset` (*value*)
Available interaction datasets in [OmniPath].

Member Type `ErrorFormatter`

Valid values are as follows:

DOROTHEA

KINASE_EXTRA

LIGREC_EXTRA

LNCRNA_MRNA

MIRNA_TARGET

OMNIPATH

PATHWAY_EXTRA

TF_MIRNA

TF_REGULONS

TF_TARGET

omnipath.constants.License

enum `omnipath.constants.License` (*value*)
License types.

Member Type `ErrorFormatter`

Valid values are as follows:

ACADEMIC

Academic license.

COMMERCIAL

Commercial license.

NON_PROFIT

Non-profit license.

FOR_PROFIT

For-profit license.

IGNORE

Ignore the license type.

omnipath.constants.Organism

enum `omnipath.constants.Organism` (*value*)

Organism types.

Member Type `ErrorFormatter`

Valid values are as follows:

HUMAN

NCIB taxonomy id: 9606.

MOUSE

NCIB taxonomy id: 10090.

RAT

NCIB taxonomy id: 10116.

2.3.2 Options

<code>omnipath.clear_cache()</code>	Remove all cached data from <code>omnipath.options.cache</code> .
<code>omnipath.options</code>	Class defining various <code>omnipath</code> options.

omnipath.clear_cache

`omnipath.clear_cache()`

Remove all cached data from `omnipath.options.cache`.

Return type `None`

omnipath.options

`omnipath.options = Options(url='https://omnipathdb.org', license=None, cache=<FileCache[si`

Class defining various `omnipath` options.

Parameters

- **url** – URL of the web service.
- **license** – License to use when fetching the data.
- **password** – Password used when performing requests.
- **cache** – Type of a cache. If `None`, cache files to memory. If a `str`, persist files into a directory.
- **autoload** – Whether to contact the server at `url` during import to get the server version and the most up-to-date query parameters and their valid options.

- **convert_dtypes** – Whether to convert the data types of the resulting `pandas.DataFrame`.
- **num_retries** – Number of retries before giving up.
- **timeout** – Timeout in seconds when awaiting response.
- **chunk_size** – Size in bytes in which to read the data.
- **progress_bar** – Whether to show the progress bar when downloading data.

RELEASE NOTES

3.1 Version 1.0

3.1.1 1.0.3 2020-12-08

- Add *omnipath.interactions.PostTranslational*
- Add possibility to download all *omnipath.requests.Annotations*

3.1.2 1.0.2 2020-11-29

- Fix small bug when converting boolean values
- Fix typos
- Add option to create interaction graphs

3.1.3 1.0.1 2020-11-29

- Fix bug of not correctly passing datasets in interactions
- Fix the way the progress bar is getting content size
- Add comparison tests with OmnipathR

3.1.4 1.0.0 2020-11-23

- Fix minor bugs
- Add options improvements
- Add tests

3.2 Version 0.0

3.2.1 0.0.0 2020-11-15

- Initial release

REFERENCES

BIBLIOGRAPHY

- [OmniPath] Türei, D., Valdeolivas, A. *et al.* (2020), *Integrated intra- and intercellular signaling knowledge for multicellular omics analysis*, [bioRxiv 2020.08.03.221242](https://doi.org/10.1101/2020.08.03.221242).
- [OmniPath16] Türei, D., Korcsmáros, T. & Saez-Rodriguez, J. (2016), *OmniPath: guidelines and gateway for literature-curated signaling pathway resources.*, *Nat Methods* 13, 966–967.

A

ACADEMIC (*omnipath.constants.License attribute*), 27
 AllInteractions (*class in omnipath.interactions*), 12
 Annotations (*class in omnipath.requests*), 5

C

categories() (*omnipath.requests.Intercell class method*), 9
 clear_cache() (*in module omnipath*), 28
 COMMERCIAL (*omnipath.constants.License attribute*), 27
 complex_genes() (*omnipath.requests.Complexes class method*), 7
 Complexes (*class in omnipath.requests*), 7

D

Dorothea (*class in omnipath.interactions*), 13
 DOROTHEA (*omnipath.constants.InteractionDataset attribute*), 27

E

Enzsub (*class in omnipath.requests*), 8

F

FOR_PROFIT (*omnipath.constants.License attribute*), 27

G

generic_categories() (*omnipath.requests.Intercell class method*), 9
 get() (*omnipath.interactions.AllInteractions class method*), 12
 get() (*omnipath.interactions.Dorothea class method*), 14
 get() (*omnipath.interactions.KinaseExtra class method*), 15
 get() (*omnipath.interactions.LigRecExtra class method*), 16
 get() (*omnipath.interactions.lncRNAmRNA class method*), 24

get() (*omnipath.interactions.miRNA class method*), 25
 get() (*omnipath.interactions.OmniPath class method*), 17
 get() (*omnipath.interactions.PathwayExtra class method*), 18
 get() (*omnipath.interactions.PostTranslational class method*), 19
 get() (*omnipath.interactions.TFmiRNA class method*), 21
 get() (*omnipath.interactions.TFtarget class method*), 22
 get() (*omnipath.interactions.Transcriptional class method*), 23
 get() (*omnipath.requests.Annotations class method*), 6
 get() (*omnipath.requests.Complexes class method*), 7
 get() (*omnipath.requests.Enzsub class method*), 8
 get() (*omnipath.requests.Intercell class method*), 10
 get() (*omnipath.requests.SignedPTMs class method*), 11
 graph() (*omnipath.interactions.AllInteractions class method*), 13
 graph() (*omnipath.interactions.Dorothea class method*), 14
 graph() (*omnipath.interactions.KinaseExtra class method*), 15
 graph() (*omnipath.interactions.LigRecExtra class method*), 16
 graph() (*omnipath.interactions.lncRNAmRNA class method*), 24
 graph() (*omnipath.interactions.miRNA class method*), 25
 graph() (*omnipath.interactions.OmniPath class method*), 17
 graph() (*omnipath.interactions.PathwayExtra class method*), 18
 graph() (*omnipath.interactions.PostTranslational class method*), 20
 graph() (*omnipath.interactions.TFmiRNA class method*), 21
 graph() (*omnipath.interactions.TFtarget class method*), 22
 graph() (*omnipath.interactions.Transcriptional class method*), 23

method), 23
graph() (*omnipath.requests.SignedPTMs* class method), 11

H
HUMAN (*omnipath.constants.Organism* attribute), 28

I
IGNORE (*omnipath.constants.License* attribute), 28
import_intercell_network() (in module *omnipath.interactions*), 26
Intercell (class in *omnipath.requests*), 9

K
KINASE_EXTRA (*omnipath.constants.InteractionDataset* attribute), 27
KinaseExtra (class in *omnipath.interactions*), 14

L
LIGREC_EXTRA (*omnipath.constants.InteractionDataset* attribute), 27
LigRecExtra (class in *omnipath.interactions*), 16
LNCRNA_MRNA (*omnipath.constants.InteractionDataset* attribute), 27
LncRNAmRNA (class in *omnipath.interactions*), 24

M
miRNA (class in *omnipath.interactions*), 25
MIRNA_TARGET (*omnipath.constants.InteractionDataset* attribute), 27
MOUSE (*omnipath.constants.Organism* attribute), 28

N
NON_PROFIT (*omnipath.constants.License* attribute), 27

O
OmniPath (class in *omnipath.interactions*), 17
OMNIPATH (*omnipath.constants.InteractionDataset* attribute), 27
options (in module *omnipath*), 28

P
params() (*omnipath.interactions.AllInteractions* class method), 13
params() (*omnipath.interactions.Dorothea* class method), 14
params() (*omnipath.interactions.KinaseExtra* class method), 15
params() (*omnipath.interactions.LigRecExtra* class method), 16
params() (*omnipath.interactions.LncRNAmRNA* class method), 25
params() (*omnipath.interactions.miRNA* class method), 26
params() (*omnipath.interactions.OmniPath* class method), 18
params() (*omnipath.interactions.PathwayExtra* class method), 19
params() (*omnipath.interactions.PostTranslational* class method), 20
params() (*omnipath.interactions.TFmiRNA* class method), 21
params() (*omnipath.interactions.TFtarget* class method), 22
params() (*omnipath.interactions.Transcriptional* class method), 23
params() (*omnipath.requests.Annotations* class method), 6
params() (*omnipath.requests.Complexes* class method), 8
params() (*omnipath.requests.Enzsub* class method), 8
params() (*omnipath.requests.Intercell* class method), 10
params() (*omnipath.requests.SignedPTMs* class method), 11
PATHWAY_EXTRA (*omnipath.constants.InteractionDataset* attribute), 27
PathwayExtra (class in *omnipath.interactions*), 18
PostTranslational (class in *omnipath.interactions*), 19

R
RAT (*omnipath.constants.Organism* attribute), 28
resources() (*omnipath.interactions.AllInteractions* class method), 13
resources() (*omnipath.interactions.Dorothea* class method), 14
resources() (*omnipath.interactions.KinaseExtra* class method), 15
resources() (*omnipath.interactions.LigRecExtra* class method), 17
resources() (*omnipath.interactions.LncRNAmRNA* class method), 25
resources() (*omnipath.interactions.miRNA* class method), 26
resources() (*omnipath.interactions.OmniPath* class method), 18
resources() (*omnipath.interactions.PathwayExtra* class method), 19
resources() (*omnipath.interactions.PostTranslational* class

method), 20
resources() (*omnipath.interactions.TFmiRNA class method*), 21
resources() (*omnipath.interactions.TFtarget class method*), 22
resources() (*omnipath.interactions.Transcriptional class method*), 24
resources() (*omnipath.requests.Annotations class method*), 6
resources() (*omnipath.requests.Complexes class method*), 8
resources() (*omnipath.requests.Enzsub class method*), 9
resources() (*omnipath.requests.Intercell class method*), 10
resources() (*omnipath.requests.SignedPTMs class method*), 11

S

SignedPTMs (*class in omnipath.requests*), 10

T

TF_MIRNA (*omnipath.constants.InteractionDataset attribute*), 27
TF_REGULONS (*omnipath.constants.InteractionDataset attribute*), 27
TF_TARGET (*omnipath.constants.InteractionDataset attribute*), 27
TFmiRNA (*class in omnipath.interactions*), 20
TFtarget (*class in omnipath.interactions*), 21
Transcriptional (*class in omnipath.interactions*), 23