

---

**pybiopax**

*Release 0.1.4*

**Benjamin M. Gyori**

**Jul 11, 2023**



# CONTENTS

<b>1</b>	<b>PyBioPAX API</b>	<b>1</b>
<b>2</b>	<b>BioPAX object model</b>	<b>5</b>
2.1	Base classes . . . . .	6
2.2	Interactions . . . . .	7
2.3	Physical entities . . . . .	12
2.4	Utility classes . . . . .	14
<b>3</b>	<b>Pathway Commons client</b>	<b>25</b>
<b>4</b>	<b>Path finding</b>	<b>27</b>
<b>5</b>	<b>XML/OWL processing utilities</b>	<b>29</b>
	<b>Python Module Index</b>	<b>31</b>
	<b>Index</b>	<b>33</b>



## PYBIOPAX API

`pybiopax.api.model_from_biocyc(identifier)`

Return a BioPAX model from a [BioCyc](#) entry.

BioCyc contains pathways for model eukaryotes and microbes.

**Parameters**

**identifier** (str) – The BioCyc identifier for a pathway (e.g., P105-PWY for [TCA cycle IV](#) (2-oxoglutarate decarboxylase))

**Return type**

[\*BioPaxModel\*](#)

**Returns**

A BioPAX model obtained from the BioCyc pathway.

`pybiopax.api.model_from_ecocyc(identifier)`

Return a BioPAX model from a [EcoCyc](#) entry.

EcoCyc contains pathways for Escherichia coli K-12 MG1655.

**Parameters**

**identifier** (str) – The EcoCyc identifier for a pathway (e.g., TCA for [TCA cycle I](#) (prokaryotic))

**Return type**

[\*BioPaxModel\*](#)

**Returns**

A BioPAX model obtained from the EcoCyc pathway.

`pybiopax.api.model_from_humancyc(identifier)`

Return a BioPAX model from a HumanCyc entry.

**Parameters**

**identifier** (str) – The HumanCyc identifier for a pathway (e.g., PWY66-398 for [TCA cycle](#))

**Return type**

[\*BioPaxModel\*](#)

**Returns**

A BioPAX model obtained from the HumanCyc pathway.

`pybiopax.api.model_from_metacyc(identifier)`

Return a BioPAX model from a [MetaCyc](#) entry.

MetaCyc contains pathways for all organisms

**Parameters**

**identifier** (str) – The MetaCyc identifier for a pathway (e.g., TCA for TCA cycle I (prokaryotic))

**Return type**

*BioPaxModel*

**Returns**

A BioPAX model obtained from the MetaCyc pathway.

`pybiopax.api.model_from_netpath(identifier)`

Return a BioPAX model from a [NetPath](#) entry.

**Parameters**

**identifier** (str) – The NetPath identifier for a pathway (e.g., 22 for the [leptin signaling pathway](#))

**Return type**

*BioPaxModel*

**Returns**

A BioPAX model obtained from the NetPath resource.

`pybiopax.api.model_from_owl_file(fname, encoding=None)`

Return a BioPAX Model from an OWL string.

**Parameters**

- **fname** (Union[str, Path, PathLike]) – A path to an OWL file of BioPAX content.
- **encoding** (Optional[str]) – The encoding type to be passed to `open()`.

**Return type**

*BioPaxModel*

**Returns**

A BioPAX Model deserialized from the OWL file.

`pybiopax.api.model_from_owl_gz(path)`

Return a BioPAX Model from an OWL file (gzipped).

**Parameters**

**path** (Union[str, Path, PathLike]) – A path to a gzipped OWL file of BioPAX content.

**Return type**

*BioPaxModel*

**Returns**

A BioPAX Model deserialized from the OWL file.

`pybiopax.api.model_from_owl_str(owl_str)`

Return a BioPAX Model from an OWL string.

**Parameters**

**owl\_str** (str) – A OWL string of BioPAX content.

**Returns**

A BioPAX Model deserialized from the OWL string.

**Return type**

`pybiopax.biopax.BioPaxModel`

`pybiopax.api.model_from_owl_url(url, request_params=None)`

Return a BioPAX Model from an URL pointing to an OWL file.

#### Parameters

- **url** (*str*) – A OWL URL with BioPAX content.
- **request\_params** (*Optional[Mapping[str, Any]]*) – Additional keyword arguments to pass to `requests.get()`

#### Return type

*BioPaxModel*

#### Returns

A BioPAX Model deserialized from the OWL file.

`pybiopax.api.model_from_pc_query(kind, source, target=None, **query_params)`

Return a BioPAX Model from a Pathway Commons query.

For more information on these queries, see <http://www.pathwaycommons.org/pc2/#graph>

#### Parameters

- **kind** (*str*) – The kind of graph query to perform. Currently 3 options are implemented, ‘neighborhood’, ‘pathsbetween’ and ‘pathsfromto’.
- **source** (*list[str]*) – A single gene name or a list of gene names which are the source set for the graph query.
- **target** (*Optional[list[str]]*) – A single gene name or a list of gene names which are the target set for the graph query. Only needed for ‘pathsfromto’ queries.
- **limit** (*Optional[int]*) – This limits the length of the longest path considered in the graph query. Default: 1
- **organism** (*Optional[str]*) – The organism used for the query. Default: ‘9606’ corresponding to human.
- **datasource** (*Optional[list[str]]*) – A list of database sources that the query results should include. Example: [‘pid’, ‘panther’]. By default, all databases are considered.

#### Returns

A BioPAX Model obtained from the results of the Pathway Commons query.

#### Return type

`pybiopax.biopax.BioPaxModel`

`pybiopax.api.model_from_reactome(identifier)`

Return a BioPAX model from a Reactome entry (pathway, event, etc.).

#### Parameters

**identifier** (*str*) – The Reactome identifier for a pathway (e.g., 177929 for [Signaling by EGFR](#)) or reaction (e.g., 177946 for [Pro-EGF is cleaved to form mature EGF](#)). For human pathways, the identifier for the BioPAX download is the same as the part that comes after R-HSA-. For non-human pathways, this is not so clear.

#### Return type

*BioPaxModel*

#### Returns

A BioPAX model obtained from the Reactome resource.

`pybiopax.api.model_to_owl_file(model, fname)`

Write an OWL string serialized from a BioPaxModel object into a file.

**Parameters**

- **model** (*BioPaxModel*) – The BioPaxModel to serialize into an OWL file.
- **fname** (Union[str, Path, PathLike]) – The path to the target OWL file.

`pybiopax.api.model_to_owl_str(model)`

Return an OWL string serialized from a BioPaxModel object.

**Parameters**

- **model** (*BioPaxModel*) – The BioPaxModel to serialize into an OWL string.

**Return type**

str

**Returns**

The OWL string for the model.



## BIOPAX OBJECT MODEL

This module implements the BioPAX Level 3 object model as a set of classes with inheritance. At the top of the class hierarchy is the generic BioPaxObject.

```
class pybiopax.biopax.model.BioPaxModel(objects,  
                                         xml_base='http://www.biopax.org/release/biopax-level3.owl#')
```

Bases: object

BioPAX Model.

### Parameters

- **objects** (*dict or list*) – A dict of BioPaxObject instances keyed by their URI string or a list of BioPaxObject instances, which will get converted into a dict keyed their URI strings
- **xml\_base** (*str*) – The XML base namespace for the content being represented.

### objects

A dict of BioPaxObject instances keyed by their URI string that are part of the model.

### Type

dict

### xml\_base

The XML base namespace for the content being represented. If not provided, the default BioPAX Level 3 base namespace is used.

### Type

Optional[str]

### classmethod from\_xml(*tree*)

Return a BioPAX Model from an OWL/XML element tree.

### Parameters

**tree** – An element tree from which the model is extracted

### Return type

*BioPaxModel*

### Returns

A BioPAX Model deserialized from the OWL XML tree.

### to\_xml()

Return an OWL string from the content of the model.

### Return type

str

```
pybiopax.biopax.model.PYBIOPAX_TQDM_CONFIG = {'unit_scale': True}
```

Default configuration for tqdm progress bars in pybiopax. To modify the tqdm configuration, modify this module-level variable. For example, to disable the progress bars, set the `disable` key to `True`.

## 2.1 Base classes

```
class pybiopax.biopax.base.BioPaxObject(uid, comment=None, **kwargs)
```

Bases: object

Generic BioPAX Object. It is the parent class of all more specific BioPAX classes.

```
class pybiopax.biopax.base.Controller(**kwargs)
```

Bases: object

BioPAX Controller.

```
class pybiopax.biopax.base.Entity(availability=None, data_source=None, **kwargs)
```

Bases: *BioPaxObject*, *Observable*, *Named*

BioPAX Entity.

**availability**

Type

str

**data\_source**

Type

List[*Provenance*]

**property name**

All names associated with the object including the standard and display name, if available.

```
class pybiopax.biopax.base.Gene(organism, **kwargs)
```

Bases: *Entity*

BioPAX Gene

**organism**

Type

*BioSource*

**property name**

All names associated with the object including the standard and display name, if available.

```
class pybiopax.biopax.base.Named(display_name=None, standard_name=None, name=None, **kwargs)
```

Bases: *XReferrable*

A mixin class to add names to a BioPaxObject.

**display\_name**

Type

str

**standard\_name**

**Type**  
str

**name**

**Type**  
str

**property name**

All names associated with the object including the standard and display name, if available.

**class** pybiopax.biopax.base.**Observable**(evidence=None, \*\*kwargs)

Bases: object

A mixin class to add evidence to a BioPaxObject.

**evidence**

**Type**  
List[Evidence]

**class** pybiopax.biopax.base.**Pathway**(pathway\_component=None, pathway\_order=None, organism=None, \*\*kwargs)

Bases: *Entity*, *Controller*

BioPAX Pathway.

**property name**

All names associated with the object including the standard and display name, if available.

**class** pybiopax.biopax.base.**Unresolved**(obj\_id)

Bases: object

A placeholder class used while deserializing BioPAX models.

**class** pybiopax.biopax.base.**XReferrable**(xref=None, \*\*kwargs)

Bases: object

A mixin class to add xrefs to a BioPaxObject.

**xref**

**Type**  
List[Xref]

## 2.2 Interactions

**class** pybiopax.biopax.interaction.**BiochemicalReaction**(delta\_s=None, delta\_h=None, delta\_g=None, k\_e\_q=None, e\_c\_number=None, \*\*kwargs)

Bases: *Conversion*

BioPAX BiochemicalReaction.

**delta\_s**

**Type**  
List[float]

**delta\_h**

Type  
List[float]

**delta\_g**

Type  
List[*DeltaG*]

**k\_e\_q**

Type  
List[*KPrime*]

**e\_c\_number**

Type  
List[str]

**property name**

All names associated with the object including the standard and display name, if available.

**class** pybiopax.biopax.interaction.**Catalysis**(*catalysis\_direction=None*, *cofactor=None*, *\*\*kwargs*)

Bases: *Control*

BioPAX Catalysis.

**catalysis\_direction**

Type  
str

**cofactor**

Type  
List[*PhysicalEntity*]

**property name**

All names associated with the object including the standard and display name, if available.

**class** pybiopax.biopax.interaction.**ComplexAssembly**(*left=None*, *right=None*,  
*conversion\_direction=None*,  
*participant\_stoichiometry=None*,  
*spontaneous=None*, *\*\*kwargs*)

Bases: *Conversion*

BioPAX ComplexAssembly.

**property name**

All names associated with the object including the standard and display name, if available.

**class** pybiopax.biopax.interaction.**Control**(*control\_type=None*, *controller=None*, *controlled=None*,  
*\*\*kwargs*)

Bases: *Interaction*

BioPAX Control.

**control\_type**

Type  
str

**controller**

Type  
List[*Process*]

**controlled**

Type  
*Process*

**property name**

All names associated with the object including the standard and display name, if available.

```
class pybiopax.biopax.interaction.Conversion(left=None, right=None, conversion_direction=None,
                                             participant_stoichiometry=None, spontaneous=None,
                                             **kwargs)
```

Bases: *Interaction*

BioPAX Conversion.

**left**

Type  
List[*PhysicalEntity*]

**right**

Type  
List[*PhysicalEntity*]

**conversion\_direction**

Type  
str

**participant\_stoichiometry**

Type  
List[*Stoichiometry*]

**spontaneous**

Type  
bool

**property name**

All names associated with the object including the standard and display name, if available.

```
class pybiopax.biopax.interaction.Degradation(left=None, right=None, conversion_direction=None,
                                                participant_stoichiometry=None, spontaneous=None,
                                                **kwargs)
```

Bases: *Conversion*

BioPAX Degradation.

**property name**

All names associated with the object including the standard and display name, if available.

```
class pybiopax.biopax.interaction.GeneticInteraction(participant=None, interaction_type=None,
**kwargs)
```

Bases: [Interaction](#)

BioPAX GeneticInteraction.

**property name**

All names associated with the object including the standard and display name, if available.

```
class pybiopax.biopax.interaction.Interaction(participant=None, interaction_type=None, **kwargs)
```

Bases: [Process](#)

BioPAX Interaction.

**participant**

Type

List[[Entity](#)]

**interaction\_type**

Type

List[str]

**property name**

All names associated with the object including the standard and display name, if available.

```
class pybiopax.biopax.interaction.Modulation(control_type=None, controller=None, controlled=None,
**kwargs)
```

Bases: [Control](#)

BioPAX Modulation.

**property name**

All names associated with the object including the standard and display name, if available.

```
class pybiopax.biopax.interaction.MolecularInteraction(participant=None, interaction_type=None,
**kwargs)
```

Bases: [Interaction](#)

BioPAX MolecularInteraction.

**property name**

All names associated with the object including the standard and display name, if available.

```
class pybiopax.biopax.interaction.Process(**kwargs)
```

Bases: [Entity](#)

BioPAX Process.

**property name**

All names associated with the object including the standard and display name, if available.

```
class pybiopax.biopax.interaction.TemplateReaction(template=None, product=None,
template_direction=None, **kwargs)
```

Bases: [Interaction](#)

BioPAX TemplateReaction.

**template**

**Type**

NucleicAcid

**product**

**Type**

List[PhysicalEntity]

**template\_direction**

**Type**

str

**property name**

All names associated with the object including the standard and display name, if available.

```
class pybiopax.biopax.interaction.TemplateReactionRegulation(control_type=None,
                                                             controller=None, controlled=None,
                                                             **kwargs)
```

Bases: *Control*

BioPAX TemplateReactionRegulation.

**property name**

All names associated with the object including the standard and display name, if available.

```
class pybiopax.biopax.interaction.Transport(left=None, right=None, conversion_direction=None,
                                             participant_stoichiometry=None, spontaneous=None,
                                             **kwargs)
```

Bases: *Conversion*

BioPAX Transport.

**property name**

All names associated with the object including the standard and display name, if available.

```
class pybiopax.biopax.interaction.TransportWithBiochemicalReaction(delta_s=None,
                                                                    delta_h=None,
                                                                    delta_g=None, k_e_q=None,
                                                                    e_c_number=None,
                                                                    **kwargs)
```

Bases: *BiochemicalReaction*

BioPAX TransportWithBiochemicalReaction.

**property name**

All names associated with the object including the standard and display name, if available.

## 2.3 Physical entities

```
class pybiopax.biopax.physical_entity.Complex(component=None, component_stoichiometry=None,
                                                **kwargs)
```

Bases: *PhysicalEntity*

BioPAX Complex.

**component**

Type

List[*PhysicalEntity*]

**component\_stoichiometry**

Type

List[*Stoichiometry*]

**property name**

All names associated with the object including the standard and display name, if available.

```
class pybiopax.biopax.physical_entity.Dna(entity_reference=None, **kwargs)
```

Bases: *SimplePhysicalEntity*

BioPAX Dna.

**property name**

All names associated with the object including the standard and display name, if available.

```
class pybiopax.biopax.physical_entity.DnaRegion(entity_reference=None, **kwargs)
```

Bases: *SimplePhysicalEntity*

BioPAX DnaRegion

**property name**

All names associated with the object including the standard and display name, if available.

```
class pybiopax.biopax.physical_entity.PhysicalEntity(feature=None, not_feature=None,
                                                       member_physical_entity=None,
                                                       cellular_location=None, **kwargs)
```

Bases: *Entity*, *Controller*

BioPAX PhysicalEntity.

**feature**

Type

List[*EntityFeature*]

**not\_feature**

Type

List[*EntityFeature*]

**member\_physical\_entity**

Type

List[*PhysicalEntity*]



**cellular\_location****Type***CellularLocationVocabulary***property name**

All names associated with the object including the standard and display name, if available.

**class** pybiopax.biopax.physical\_entity.**Protein**(entity\_reference=None, \*\*kwargs)

Bases: *SimplePhysicalEntity*

BioPAX Protein.

**property name**

All names associated with the object including the standard and display name, if available.

**class** pybiopax.biopax.physical\_entity.**Rna**(entity\_reference=None, \*\*kwargs)

Bases: *SimplePhysicalEntity*

BioPAX Rna.

**property name**

All names associated with the object including the standard and display name, if available.

**class** pybiopax.biopax.physical\_entity.**RnaRegion**(entity\_reference=None, \*\*kwargs)

Bases: *SimplePhysicalEntity*

BioPAX RnaRegion

**property name**

All names associated with the object including the standard and display name, if available.

**class** pybiopax.biopax.physical\_entity.**SimplePhysicalEntity**(entity\_reference=None, \*\*kwargs)

Bases: *PhysicalEntity*

BioPAX SimplePhysicalEntity.

**entity\_reference****Type***EntityReference***property name**

All names associated with the object including the standard and display name, if available.

**class** pybiopax.biopax.physical\_entity.**SmallMolecule**(entity\_reference=None, \*\*kwargs)

Bases: *SimplePhysicalEntity*

BioPAX SmallMolecule.

**property name**

All names associated with the object including the standard and display name, if available.

## 2.4 Utility classes

**class** pybiopax.biopax.util.**BindingFeature**(*binds\_to=None, intra\_molecular=None, \*\*kwargs*)

Bases: *EntityFeature*

BioPAX BindingFeature.

**binds\_to**

Type

*BindingFeature*

**intra\_molecular**

Type

bool

**class** pybiopax.biopax.util.**BioSource**(*cell\_type=None, tissue=None, taxon\_xref=None, \*\*kwargs*)

Bases: *UtilityClass, Named*

BioPAX BioSource.

**cell\_type**

Type

*CellVocabulary*

**tissue**

Type

*TissueVocabulary*

**property name**

All names associated with the object including the standard and display name, if available.

**class** pybiopax.biopax.util.**BiochemicalPathwayStep**(*step\_conversion=None, step\_direction=None, \*\*kwargs*)

Bases: *PathwayStep*

BioPAX BiochemicalPathwayStep.

**step\_conversion**

Type

*Conversion*

**step\_direction**

Type

str

**class** pybiopax.biopax.util.**CellVocabulary**(*term=None, \*\*kwargs*)

Bases: *ControlledVocabulary*

BioPAX CellVocabulary.

**class** pybiopax.biopax.util.**CellularLocationVocabulary**(*term=None, \*\*kwargs*)

Bases: *ControlledVocabulary*

BioPAX CellularLocationVocabulary.

```
class pybiopax.biopax.util.ChemicalConstant(ionic_strength=None, ph=None, p_mg=None,
                                             tempterature=None, **kwargs)
```

Bases: *UtilityClass*

BioPAX ChemicalConstant.

**ionic\_strength**

Type  
float

**ph**

Type  
float

**p\_mg**

Type  
float

**temperature**

Type  
float

```
class pybiopax.biopax.util.ChemicalStructure(structure_format=None, structure_data=None,
                                              **kwargs)
```

Bases: *UtilityClass*

BioPAX ChemicalStructure.

**structure\_format**

Type  
str

**structure\_data**

Type  
str

```
class pybiopax.biopax.util.ControlledVocabulary(term=None, **kwargs)
```

Bases: *UtilityClass*, *XReferrable*

BioPAX ControlledVocabulary.

**term**

Type  
List[str]

```
class pybiopax.biopax.util.DeltaG(delta_g_prime0=None, **kwargs)
```

Bases: *ChemicalConstant*

BioPAX DeltaG.

**delta\_g\_prime**

Type  
float

```
class pybiopax.biopax.util.DnaReference(sub_region=None, **kwargs)
```

Bases: *NucleicAcidReference*

BioPAX DnaReference.

**property name**

All names associated with the object including the standard and display name, if available.

```
class pybiopax.biopax.util.DnaRegionReference(absolute_region=None, region_type=None, **kwargs)
```

Bases: *NucleicAcidRegionReference*

BioPAX DnaRegionReference.

**property name**

All names associated with the object including the standard and display name, if available.

```
class pybiopax.biopax.util.EntityFeature(owner_entity_reference=None, feature_location=None,
                                          member_feature=None, feature_location_type=None,
                                          **kwargs)
```

Bases: *UtilityClass, Observable*

BioPAX UtilityClass.

**owner\_entity\_reference**

Type

*EntityReference*

**feature\_location**

Type

*SequenceLocation*

**member\_feature**

Type

List[*EntityFeature*]

**feature\_location\_type**

Type

*SequenceRegionVocabulary*

```
class pybiopax.biopax.util.EntityReference(entity_feature=None, entity_reference_type=None,
                                          member_entity_reference=None,
                                          owner_entity_reference=None, **kwargs)
```

Bases: *UtilityClass, Named, Observable*

BioPAX EntityReference.

**entity\_feature**

Type

List[*EntityFeature*]

**entity\_reference\_type**

Type

List[*EntityReferenceTypeVocabulary*]

**member\_entity\_reference**

Type  
List[*EntityReference*]

**owner\_entity\_reference**

Type  
List[*EntityReference*]

**property name**

All names associated with the object including the standard and display name, if available.

**class** pybiopax.biopax.util.**EntityReferenceTypeVocabulary**(term=None, \*\*kwargs)

Bases: *ControlledVocabulary*

BioPAX EntityReferenceTypeVocabulary.

**class** pybiopax.biopax.util.**Evidence**(confidence=None, evidence\_code=None, experimental\_form=None, \*\*kwargs)

Bases: *UtilityClass*, *XReferrable*

BioPAX Evidence.

**confidence**

Type  
List[*Score*]

**evidence\_code**

Type  
List[*EvidenceCodeVocabulary*]

**experimental\_form**

Type  
List[*ExperimentalForm*]

**class** pybiopax.biopax.util.**EvidenceCodeVocabulary**(term=None, \*\*kwargs)

Bases: *ControlledVocabulary*

BioPAX EvidenceCodeVocabulary.

**class** pybiopax.biopax.util.**ExperimentalForm**(experimental\_form\_entity=None, experimental\_form\_description=None, experimental\_feature=None, \*\*kwargs)

Bases: *UtilityClass*

BioPAX ExperimentalForm.

**experimental\_form\_entity**

Type  
*Entity*

**experimental\_form\_description**

Type  
List[*ExperimentalFormVocabulary*]

**experimental\_feature**

**Type**

List[*EntityFeature*]

**class** pybiopax.biopax.util.**ExperimentalFormVocabulary**(term=None, \*\*kwargs)

Bases: *ControlledVocabulary*

BioPAX ExperimentalFormVocabulary.

**class** pybiopax.biopax.util.**FragmentFeature**(owner\_entity\_reference=None, feature\_location=None, member\_feature=None, feature\_location\_type=None, \*\*kwargs)

Bases: *EntityFeature*

BioPAX FragmentFeature.

**class** pybiopax.biopax.util.**InteractionVocabulary**(term=None, \*\*kwargs)

Bases: *ControlledVocabulary*

BioPAX InteractionVocabulary.

**class** pybiopax.biopax.util.**KPrime**(k\_prime, \*\*kwargs)

Bases: *ChemicalConstant*

BioPAX KPrime.

**k\_prime**

**Type**

float

**class** pybiopax.biopax.util.**ModificationFeature**(modification\_type=None, \*\*kwargs)

Bases: *EntityFeature*

BioPAX ModificationFeature.

**modification\_type**

**Type**

*SequenceModificationVocabulary*

**class** pybiopax.biopax.util.**NucleicAcidReference**(sub\_region=None, \*\*kwargs)

Bases: *SequenceEntityReference*

BioPAX NucleicAcidReference

**sub\_region**

**Type**

List[*NucleicAcidRegionReference*]

**property name**

All names associated with the object including the standard and display name, if available.

**class** pybiopax.biopax.util.**NucleicAcidRegionReference**(absolute\_region=None, region\_type=None, \*\*kwargs)

Bases: *NucleicAcidReference*

BioPAX NucleicAcidRegionReference

**absolute\_region**

**Type**

*SequenceLocation*

**region\_type**

**Type**

List[*SequenceRegionVocabulary*]

**property name**

All names associated with the object including the standard and display name, if available.

**class** pybiopax.biopax.util.**PathwayStep**(step\_process=None, next\_step=None, \*\*kwargs)

Bases: *UtilityClass*, *Observable*

BioPAX PathwayStep.

**step\_process**

**Type**

List[*Process*]

**next\_step**

**Type**

List[*Process*]

**class** pybiopax.biopax.util.**PhenotypeVocabulary**(term=None, \*\*kwargs)

Bases: *ControlledVocabulary*

BioPAX PhenotypeVocabulary.

**class** pybiopax.biopax.util.**ProteinReference**(organism=None, sequence=None, \*\*kwargs)

Bases: *SequenceEntityReference*

BioPAX ProteinReference.

**property name**

All names associated with the object including the standard and display name, if available.

**class** pybiopax.biopax.util.**Provenance**(\*\*kwargs)

Bases: *UtilityClass*, *Named*

BioPAX Provenance.

**property name**

All names associated with the object including the standard and display name, if available.

**class** pybiopax.biopax.util.**PublicationXref**(title=None, url=None, source=None, author=None, year=None, \*\*kwargs)

Bases: *Xref*

BioPAX PublicationXref.

**title**

**Type**

str

```
url
    Type
        List[str]

source
    Type
        List[str]

author
    Type
        List[str]

year
    Type
        int

class pybiopax.biopax.util.RelationshipTypeVocabulary(term=None, **kwargs)
    Bases: ControlledVocabulary
    BioPAX RelationshipTypeVocabulary.

class pybiopax.biopax.util.RelationshipXref(relationship_type=None, **kwargs)
    Bases: Xref
    BioPAX RelationshipXref.

relationship_type
    Type
        RelationshipTypeVocabulary

class pybiopax.biopax.util.RnaReference(sub_region=None, **kwargs)
    Bases: NucleicAcidReference
    BioPAX RnaReference.

property name
    All names associated with the object including the standard and display name, if available.

class pybiopax.biopax.util.RnaRegionReference(absolute_region=None, region_type=None, **kwargs)
    Bases: NucleicAcidRegionReference
    BioPAX RnaRegionReference.

property name
    All names associated with the object including the standard and display name, if available.

class pybiopax.biopax.util.Score(score_source=None, value=None, **kwargs)
    Bases: UtilityClass, XReferrable
    BioPAX Score.

score_source
    Type
        Provenance
```



**value**

**Type**

str

**class** pybiopax.biopax.util.**SequenceEntityReference**(*organism=None, sequence=None, \*\*kwargs*)

Bases: [EntityReference](#)

BioPAX SequenceEntityReference.

**organism**

**Type**

[BioSource](#)

**sequence**

**Type**

str

**property name**

All names associated with the object including the standard and display name, if available.

**class** pybiopax.biopax.util.**SequenceInterval**(*sequence\_interval\_begin=None, sequence\_interval\_end=None, \*\*kwargs*)

Bases: [SequenceLocation](#)

BioPAX SequenceInterval.

**sequence\_interval\_begin**

**Type**

[SequenceSite](#)

**sequence\_interval\_end**

**Type**

[SequenceSite](#)

**class** pybiopax.biopax.util.**SequenceLocation**(*region\_type=None, \*\*kwargs*)

Bases: [UtilityClass](#)

BioPAX SequenceLocation.

**region\_type**

**Type**

List[[SequenceRegionVocabulary](#)]

**class** pybiopax.biopax.util.**SequenceModificationVocabulary**(*term=None, \*\*kwargs*)

Bases: [ControlledVocabulary](#)

BioPAX SequenceModificationVocabulary.

**class** pybiopax.biopax.util.**SequenceRegionVocabulary**(*term=None, \*\*kwargs*)

Bases: [ControlledVocabulary](#)

BioPAX SequenceRegionVocabulary.

**class** pybiopax.biopax.util.**SequenceSite**(*position\_status=None, sequence\_position=None, \*\*kwargs*)

Bases: [SequenceLocation](#)

BioPAX SequenceSite.

**position\_status**

Type  
str

**sequence\_position**

Type  
int

```
class pybiopax.biopax.util.SmallMoleculeReference(structure=None, chemical_formula=None,
                                                    molecular_weight=None, **kwargs)
```

Bases: *EntityReference*

BioPAX SmallMoleculeReference.

**structure**

Type  
*ChemicalStructure*

**chemical\_formula**

Type  
str

**molecular\_weight**

Type  
float

**property name**

All names associated with the object including the standard and display name, if available.

```
class pybiopax.biopax.util.Stoichiometry(stoichiometric_coefficient=None, physical_entity=None,
                                          **kwargs)
```

Bases: *UtilityClass*

BioPAX Stoichiometry.

**stoichiometric\_coefficient**

Type  
float

**physical\_entity**

Type  
*PhysicalEntity*

```
class pybiopax.biopax.util.TissueVocabulary(term=None, **kwargs)
```

Bases: *ControlledVocabulary*

BioPAX TissueVocabulary.

```
class pybiopax.biopax.util.UnificationXref(db=None, id=None, db_version=None, id_version=None,
                                           **kwargs)
```

Bases: *Xref*

BioPAX UnificationXref.

```
class pybiopax.biopax.util.UtilityClass(uid, comment=None, **kwargs)
```

Bases: [BioPaxObject](#)

BioPAX UtilityClass.

```
class pybiopax.biopax.util.Xref(db=None, id=None, db_version=None, id_version=None, **kwargs)
```

Bases: [UtilityClass](#)

BioPAX Xref.

**db**

**Type**

str

**id**

**Type**

str

**db\_version**

**Type**

str

**id\_version**

**Type**

str



## PATHWAY COMMONS CLIENT

A client to the PathwayCommons REST API. For more details about the service, see the documentation at <https://www.pathwaycommons.org/pc2/>.

`pybiopax.pc_client.graph_query(kind, source, target=None, **query_params)`

Perform a graph query on PathwayCommons.

For more information on these queries, see <http://www.pathwaycommons.org/pc2/#graph>

### Parameters

- **kind** (*str*) – The kind of graph query to perform. Currently 3 options are implemented, ‘neighborhood’, ‘pathsbetween’ and ‘pathsfromto’.
- **source** (*list[str]*) – A single gene name or a list of gene names which are the source set for the graph query.
- **target** (*Optional[list[str]]*) – A single gene name or a list of gene names which are the target set for the graph query. Only needed for ‘pathsfromto’ queries.
- **limit** (*Optional[int]*) – This limits the length of the longest path considered in the graph query. Default: 1
- **organism** (*Optional[str]*) – The organism used for the query. Default: ‘9606’ corresponding to human.
- **datasource** (*Optional[list[str]]*) – A list of database sources that the query results should include. Example: [‘pid’, ‘panther’]. By default, all databases are considered.

### Returns

A BioPAX OWL string that can then be deserialized into a BioPaxModel.

### Return type

str



## PATH FINDING

This module implements finding paths in a BioPaxModel starting from a given object using a path constraint string.

**exception** `pybiopax.paths.BiopaxClassConstraintError(cls_str)`

Bases: `KeyError`

`pybiopax.paths.find_objects(start_obj, path_str)`

Return objects matching the given path specification.

### Parameters

- **start\_obj** (*BioPaxObject*) – The object to start the search from.
- **path\_str** (*str*) – A path specification string which consists of one or more parts separated by /. Each part is the name of an object attribute, and can optionally contain a class name as well, separated by : to constrain the class of the target of the attribute to consider. Optionally, each attribute can also have a \* suffix to make the search recursive.

### Return type

`List[BioPaxObject]`

### Returns

A list of BioPaxObjects satisfying the given path specification.





## XML/OWL PROCESSING UTILITIES

`pybiopax.xml_util.camel_to_snake(txt)`  
Return snake case from camel case

`pybiopax.xml_util.get_attr_tag(element)`  
Return the tag of an element as an attribute name.

`pybiopax.xml_util.get_datatype(attrib)`  
Return the RDF data type of an element attribute.

`pybiopax.xml_util.get_id_or_about(element)`  
Return the ID or the about associated with an element

`pybiopax.xml_util.get_ns(element)`  
Return the name space of a given element.

`pybiopax.xml_util.get_resource(attrib)`  
Return the resource associated with an element attribute.

`pybiopax.xml_util.get_tag(element)`  
Return the tag of an element.

`pybiopax.xml_util.has_ns(element, ns)`  
Return True if the element is from a given name space.

`pybiopax.xml_util.is_datatype(attrib, prefix, datatype)`  
Return True if the given attribute is of a given type.

`pybiopax.xml_util.is_url(txt)`  
Return true if the given string is an URL.

`pybiopax.xml_util.nselem(ns, elem)`  
Return a full namespaced string with curly brackets with a suffix.

`pybiopax.xml_util.nssuffix(ns, suffix)`  
Return a full namespaced string with a suffix.

`pybiopax.xml_util.snake_to_camel(txt)`  
Return camel case from snake case.

`pybiopax.xml_util.wrap_xml_elements(elements, xml_base)`  
Return a valid BioPAX OWL wrapping XML-serialized BioPAX objects.

`pybiopax.xml_util.xml_to_file(xml, fname)`  
Write an XML element tree to a given file.

`pybiopax.xml_util.xml_to_str(xml)`

Return the OWL string for an XML element tree.

- `genindex`
- `modindex`

## PYTHON MODULE INDEX

### p

- `pybiopax.api`, 1
- `pybiopax.biopax`, 5
- `pybiopax.biopax.base`, 6
- `pybiopax.biopax.interaction`, 7
- `pybiopax.biopax.model`, 5
- `pybiopax.biopax.physical_entity`, 12
- `pybiopax.biopax.util`, 14
- `pybiopax.paths`, 27
- `pybiopax.pc_client`, 25
- `pybiopax.xml_util`, 29



## A

`absolute_region` (pybiopax.biopax.util.NucleicAcidRegionReference attribute), 18  
`author` (pybiopax.biopax.util.PublicationXref attribute), 20  
`availability` (pybiopax.biopax.base.Entity attribute), 6

## B

`BindingFeature` (class in pybiopax.biopax.util), 14  
`binds_to` (pybiopax.biopax.util.BindingFeature attribute), 14  
`BiochemicalPathwayStep` (class in pybiopax.biopax.util), 14  
`BiochemicalReaction` (class in pybiopax.biopax.interaction), 7  
`BiopaxClassConstraintError`, 27  
`BioPaxModel` (class in pybiopax.biopax.model), 5  
`BioPaxObject` (class in pybiopax.biopax.base), 6  
`BioSource` (class in pybiopax.biopax.util), 14

## C

`camel_to_snake()` (in module pybiopax.xml\_util), 29  
`Catalysis` (class in pybiopax.biopax.interaction), 8  
`catalysis_direction` (pybiopax.biopax.interaction.Catalysis attribute), 8  
`cell_type` (pybiopax.biopax.util.BioSource attribute), 14  
`cellular_location` (pybiopax.biopax.physical\_entity.PhysicalEntity attribute), 12  
`CellularLocationVocabulary` (class in pybiopax.biopax.util), 14  
`CellVocabulary` (class in pybiopax.biopax.util), 14  
`chemical_formula` (pybiopax.biopax.util.SmallMoleculeReference attribute), 22  
`ChemicalConstant` (class in pybiopax.biopax.util), 14  
`ChemicalStructure` (class in pybiopax.biopax.util), 15

`cofactor` (pybiopax.biopax.interaction.Catalysis attribute), 8  
`Complex` (class in pybiopax.biopax.physical\_entity), 12  
`ComplexAssembly` (class in pybiopax.biopax.interaction), 8  
`component` (pybiopax.biopax.physical\_entity.Complex attribute), 12  
`component_stoichiometry` (pybiopax.biopax.physical\_entity.Complex attribute), 12  
`confidence` (pybiopax.biopax.util.Evidence attribute), 17  
`Control` (class in pybiopax.biopax.interaction), 8  
`control_type` (pybiopax.biopax.interaction.Control attribute), 8  
`controlled` (pybiopax.biopax.interaction.Control attribute), 9  
`ControlledVocabulary` (class in pybiopax.biopax.util), 15  
`Controller` (class in pybiopax.biopax.base), 6  
`controller` (pybiopax.biopax.interaction.Control attribute), 8  
`Conversion` (class in pybiopax.biopax.interaction), 9  
`conversion_direction` (pybiopax.biopax.interaction.Conversion attribute), 9

## D

`data_source` (pybiopax.biopax.base.Entity attribute), 6  
`db` (pybiopax.biopax.util.Xref attribute), 23  
`db_version` (pybiopax.biopax.util.Xref attribute), 23  
`Degradation` (class in pybiopax.biopax.interaction), 9  
`delta_g` (pybiopax.biopax.interaction.BiochemicalReaction attribute), 8  
`delta_g_prime` (pybiopax.biopax.util.DeltaG attribute), 15  
`delta_h` (pybiopax.biopax.interaction.BiochemicalReaction attribute), 7  
`delta_s` (pybiopax.biopax.interaction.BiochemicalReaction attribute), 7  
`DeltaG` (class in pybiopax.biopax.util), 15  
`display_name` (pybiopax.biopax.base.Named attribute),

- 6  
Dna (class in *pybiopax.biopax.physical\_entity*), 12  
DnaReference (class in *pybiopax.biopax.util*), 15  
DnaRegion (class in *pybiopax.biopax.physical\_entity*), 12  
DnaRegionReference (class in *pybiopax.biopax.util*), 16
- E**  
e\_c\_number (*pybiopax.biopax.interaction.BiochemicalReaction* attribute), 8  
Entity (class in *pybiopax.biopax.base*), 6  
entity\_feature (*pybiopax.biopax.util.EntityReference* attribute), 16  
entity\_reference (*pybiopax.biopax.physical\_entity.SimplePhysicalEntity* attribute), 13  
entity\_reference\_type (*pybiopax.biopax.util.EntityReference* attribute), 16  
EntityFeature (class in *pybiopax.biopax.util*), 16  
EntityReference (class in *pybiopax.biopax.util*), 16  
EntityReferenceTypeVocabulary (class in *pybiopax.biopax.util*), 17  
Evidence (class in *pybiopax.biopax.util*), 17  
evidence (*pybiopax.biopax.base.Observable* attribute), 7  
evidence\_code (*pybiopax.biopax.util.Evidence* attribute), 17  
EvidenceCodeVocabulary (class in *pybiopax.biopax.util*), 17  
experimental\_feature (*pybiopax.biopax.util.ExperimentalForm* attribute), 17  
experimental\_form (*pybiopax.biopax.util.Evidence* attribute), 17  
experimental\_form\_description (*pybiopax.biopax.util.ExperimentalForm* attribute), 17  
experimental\_form\_entity (*pybiopax.biopax.util.ExperimentalForm* attribute), 17  
ExperimentalForm (class in *pybiopax.biopax.util*), 17  
ExperimentalFormVocabulary (class in *pybiopax.biopax.util*), 18
- F**  
feature (*pybiopax.biopax.physical\_entity.PhysicalEntity* attribute), 12  
feature\_location (*pybiopax.biopax.util.EntityFeature* attribute), 16  
feature\_location\_type (*pybiopax.biopax.util.EntityFeature* attribute), 16
- find\_objects() (in module *pybiopax.paths*), 27  
FragmentFeature (class in *pybiopax.biopax.util*), 18  
from\_xml() (*pybiopax.biopax.model.BioPaxModel* class method), 5
- G**  
Gene (class in *pybiopax.biopax.base*), 6  
GeneticInteraction (class in *pybiopax.biopax.interaction*), 9  
get\_attr\_tag() (in module *pybiopax.xml\_util*), 29  
get\_datatype() (in module *pybiopax.xml\_util*), 29  
get\_id\_or\_about() (in module *pybiopax.xml\_util*), 29  
get\_ns() (in module *pybiopax.xml\_util*), 29  
get\_resource() (in module *pybiopax.xml\_util*), 29  
get\_tag() (in module *pybiopax.xml\_util*), 29  
graph\_query() (in module *pybiopax.pc\_client*), 25
- H**  
has\_ns() (in module *pybiopax.xml\_util*), 29
- I**  
id (*pybiopax.biopax.util.Xref* attribute), 23  
id\_version (*pybiopax.biopax.util.Xref* attribute), 23  
Interaction (class in *pybiopax.biopax.interaction*), 10  
interaction\_type (*pybiopax.biopax.interaction.Interaction* attribute), 10  
InteractionVocabulary (class in *pybiopax.biopax.util*), 18  
intra\_molecular (*pybiopax.biopax.util.BindingFeature* attribute), 14  
ionic\_strength (*pybiopax.biopax.util.ChemicalConstant* attribute), 15  
is\_datatype() (in module *pybiopax.xml\_util*), 29  
is\_url() (in module *pybiopax.xml\_util*), 29
- K**  
k\_e\_q (*pybiopax.biopax.interaction.BiochemicalReaction* attribute), 8  
k\_prime (*pybiopax.biopax.util.KPrime* attribute), 18  
KPrime (class in *pybiopax.biopax.util*), 18
- L**  
left (*pybiopax.biopax.interaction.Conversion* attribute), 9
- M**  
member\_entity\_reference (*pybiopax.biopax.util.EntityReference* attribute), 16  
member\_feature (*pybiopax.biopax.util.EntityFeature* attribute), 16

member\_physical\_entity (py-  
     biopax.biopax.physical\_entity.PhysicalEntity  
     attribute), 12  
 model\_from\_biocyc() (in module pybiopax.api), 1  
 model\_from\_ecocyc() (in module pybiopax.api), 1  
 model\_from\_humancyc() (in module pybiopax.api), 1  
 model\_from\_metacyc() (in module pybiopax.api), 1  
 model\_from\_netpath() (in module pybiopax.api), 2  
 model\_from\_owl\_file() (in module pybiopax.api), 2  
 model\_from\_owl\_gz() (in module pybiopax.api), 2  
 model\_from\_owl\_str() (in module pybiopax.api), 2  
 model\_from\_owl\_url() (in module pybiopax.api), 2  
 model\_from\_pc\_query() (in module pybiopax.api), 3  
 model\_from\_reactome() (in module pybiopax.api), 3  
 model\_to\_owl\_file() (in module pybiopax.api), 3  
 model\_to\_owl\_str() (in module pybiopax.api), 4  
 modification\_type (py-  
     biopax.biopax.util.ModificationFeature at-  
     tribute), 18  
 ModificationFeature (class in pybiopax.biopax.util),  
     18  
 Modulation (class in pybiopax.biopax.interaction), 10  
 module  
     pybiopax.api, 1  
     pybiopax.biopax, 5  
     pybiopax.biopax.base, 6  
     pybiopax.biopax.interaction, 7  
     pybiopax.biopax.model, 5  
     pybiopax.biopax.physical\_entity, 12  
     pybiopax.biopax.util, 14  
     pybiopax.paths, 27  
     pybiopax.pc\_client, 25  
     pybiopax.xml\_util, 29  
 molecular\_weight (py-  
     biopax.biopax.util.SmallMoleculeReference  
     attribute), 22  
 MolecularInteraction (class in py-  
     biopax.biopax.interaction), 10

## N

name (pybiopax.biopax.base.Entity property), 6  
 name (pybiopax.biopax.base.Gene property), 6  
 name (pybiopax.biopax.base.Named attribute), 7  
 name (pybiopax.biopax.base.Named property), 7  
 name (pybiopax.biopax.base.Pathway property), 7  
 name (pybiopax.biopax.interaction.BiochemicalReaction  
     property), 8  
 name (pybiopax.biopax.interaction.Catalysis property), 8  
 name (pybiopax.biopax.interaction.ComplexAssembly  
     property), 8  
 name (pybiopax.biopax.interaction.Control property), 9  
 name (pybiopax.biopax.interaction.Conversion property),  
     9  
 name (pybiopax.biopax.interaction.Degradation prop-  
     erty), 9  
 name (pybiopax.biopax.interaction.GeneticInteraction  
     property), 10  
 name (pybiopax.biopax.interaction.Interaction property),  
     10  
 name (pybiopax.biopax.interaction.Modulation property),  
     10  
 name (pybiopax.biopax.interaction.MolecularInteraction  
     property), 10  
 name (pybiopax.biopax.interaction.Process property), 10  
 name (pybiopax.biopax.interaction.TemplateReaction  
     property), 11  
 name (pybiopax.biopax.interaction.TemplateReactionRegulation  
     property), 11  
 name (pybiopax.biopax.interaction.Transport property),  
     11  
 name (pybiopax.biopax.interaction.TransportWithBiochemicalReaction  
     property), 11  
 name (pybiopax.biopax.physical\_entity.Complex prop-  
     erty), 12  
 name (pybiopax.biopax.physical\_entity.Dna property), 12  
 name (pybiopax.biopax.physical\_entity.DnaRegion prop-  
     erty), 12  
 name (pybiopax.biopax.physical\_entity.PhysicalEntity  
     property), 13  
 name (pybiopax.biopax.physical\_entity.Protein property),  
     13  
 name (pybiopax.biopax.physical\_entity.Rna property), 13  
 name (pybiopax.biopax.physical\_entity.RnaRegion prop-  
     erty), 13  
 name (pybiopax.biopax.physical\_entity.SimplePhysicalEntity  
     property), 13  
 name (pybiopax.biopax.physical\_entity.SmallMolecule  
     property), 13  
 name (pybiopax.biopax.util.BioSource property), 14  
 name (pybiopax.biopax.util.DnaReference property), 16  
 name (pybiopax.biopax.util.DnaRegionReference prop-  
     erty), 16  
 name (pybiopax.biopax.util.EntityReference property), 17  
 name (pybiopax.biopax.util.NucleicAcidReference prop-  
     erty), 18  
 name (pybiopax.biopax.util.NucleicAcidRegionReference  
     property), 19  
 name (pybiopax.biopax.util.ProteinReference property),  
     19  
 name (pybiopax.biopax.util.Provenance property), 19  
 name (pybiopax.biopax.util.RnaReference property), 20  
 name (pybiopax.biopax.util.RnaRegionReference prop-  
     erty), 20  
 name (pybiopax.biopax.util.SequenceEntityReference  
     property), 21  
 name (pybiopax.biopax.util.SmallMoleculeReference  
     property), 22

Named (class in *pybiopax.biopax.base*), 6  
 next\_step (*pybiopax.biopax.util.PathwayStep* attribute), 19  
 not\_feature (*pybiopax.biopax.physical\_entity.PhysicalEntity* attribute), 12  
 nselect() (in module *pybiopax.xml\_util*), 29  
 nssuffix() (in module *pybiopax.xml\_util*), 29  
 NucleicAcidReference (class in *pybiopax.biopax.util*), 18  
 NucleicAcidRegionReference (class in *pybiopax.biopax.util*), 18

## O

objects (*pybiopax.biopax.model.BioPaxModel* attribute), 5  
 Observable (class in *pybiopax.biopax.base*), 7  
 organism (*pybiopax.biopax.base.Gene* attribute), 6  
 organism (*pybiopax.biopax.util.SequenceEntityReference* attribute), 21  
 owner\_entity\_reference (*pybiopax.biopax.util.EntityFeature* attribute), 16  
 owner\_entity\_reference (*pybiopax.biopax.util.EntityReference* attribute), 17

## P

p\_mg (*pybiopax.biopax.util.ChemicalConstant* attribute), 15  
 participant (*pybiopax.biopax.interaction.Interaction* attribute), 10  
 participant\_stoichiometry (*pybiopax.biopax.interaction.Conversion* attribute), 9  
 Pathway (class in *pybiopax.biopax.base*), 7  
 PathwayStep (class in *pybiopax.biopax.util*), 19  
 ph (*pybiopax.biopax.util.ChemicalConstant* attribute), 15  
 PhenotypeVocabulary (class in *pybiopax.biopax.util*), 19  
 physical\_entity (*pybiopax.biopax.util.Stoichiometry* attribute), 22  
 PhysicalEntity (class in *pybiopax.biopax.physical\_entity*), 12  
 position\_status (*pybiopax.biopax.util.SequenceSite* attribute), 21  
 Process (class in *pybiopax.biopax.interaction*), 10  
 product (*pybiopax.biopax.interaction.TemplateReaction* attribute), 11  
 Protein (class in *pybiopax.biopax.physical\_entity*), 13  
 ProteinReference (class in *pybiopax.biopax.util*), 19  
 Provenance (class in *pybiopax.biopax.util*), 19  
 PublicationXref (class in *pybiopax.biopax.util*), 19  
 pybiopax.api module, 1

*pybiopax.biopax* module, 5  
*pybiopax.biopax.base* module, 6  
*pybiopax.biopax.interaction* module, 7  
*pybiopax.biopax.model* module, 5  
*pybiopax.biopax.physical\_entity* module, 12  
*pybiopax.biopax.util* module, 14  
*pybiopax.paths* module, 27  
*pybiopax.pc\_client* module, 25  
*pybiopax.xml\_util* module, 29  
 PYBIOPAX\_TQDM\_CONFIG (in module *pybiopax.biopax.model*), 5

## R

region\_type (*pybiopax.biopax.util.NucleicAcidRegionReference* attribute), 19  
 region\_type (*pybiopax.biopax.util.SequenceLocation* attribute), 21  
 relationship\_type (*pybiopax.biopax.util.RelationshipXref* attribute), 20  
 RelationshipTypeVocabulary (class in *pybiopax.biopax.util*), 20  
 RelationshipXref (class in *pybiopax.biopax.util*), 20  
 right (*pybiopax.biopax.interaction.Conversion* attribute), 9  
 Rna (class in *pybiopax.biopax.physical\_entity*), 13  
 RnaReference (class in *pybiopax.biopax.util*), 20  
 RnaRegion (class in *pybiopax.biopax.physical\_entity*), 13  
 RnaRegionReference (class in *pybiopax.biopax.util*), 20

## S

Score (class in *pybiopax.biopax.util*), 20  
 score\_source (*pybiopax.biopax.util.Score* attribute), 20  
 sequence (*pybiopax.biopax.util.SequenceEntityReference* attribute), 21  
 sequence\_interval\_begin (*pybiopax.biopax.util.SequenceInterval* attribute), 21  
 sequence\_interval\_end (*pybiopax.biopax.util.SequenceInterval* attribute), 21  
 sequence\_position (*pybiopax.biopax.util.SequenceSite* attribute),



22

SequenceEntityReference (class in pybiopax.biopax.util), 21

SequenceInterval (class in pybiopax.biopax.util), 21

SequenceLocation (class in pybiopax.biopax.util), 21

SequenceModificationVocabulary (class in pybiopax.biopax.util), 21

SequenceRegionVocabulary (class in pybiopax.biopax.util), 21

SequenceSite (class in pybiopax.biopax.util), 21

SimplePhysicalEntity (class in pybiopax.biopax.physical\_entity), 13

SmallMolecule (class in pybiopax.biopax.physical\_entity), 13

SmallMoleculeReference (class in pybiopax.biopax.util), 22

snake\_to\_camel() (in module pybiopax.xml\_util), 29

source (pybiopax.biopax.util.PublicationXref attribute), 20

spontaneous (pybiopax.biopax.interaction.Conversion attribute), 9

standard\_name (pybiopax.biopax.base.Named attribute), 6

step\_conversion (pybiopax.biopax.util.BiochemicalPathwayStep attribute), 14

step\_direction (pybiopax.biopax.util.BiochemicalPathwayStep attribute), 14

step\_process (pybiopax.biopax.util.PathwayStep attribute), 19

stoichiometric\_coefficient (pybiopax.biopax.util.Stoichiometry attribute), 22

Stoichiometry (class in pybiopax.biopax.util), 22

structure (pybiopax.biopax.util.SmallMoleculeReference attribute), 22

structure\_data (pybiopax.biopax.util.ChemicalStructure attribute), 15

structure\_format (pybiopax.biopax.util.ChemicalStructure attribute), 15

sub\_region (pybiopax.biopax.util.NucleicAcidReference attribute), 18

**T**

temperature (pybiopax.biopax.util.ChemicalConstant attribute), 15

template (pybiopax.biopax.interaction.TemplateReaction attribute), 10

template\_direction (pybiopax.biopax.interaction.TemplateReaction attribute), 11

TemplateReaction (class in pybiopax.biopax.interaction), 10

TemplateReactionRegulation (class in pybiopax.biopax.interaction), 11

term (pybiopax.biopax.util.ControlledVocabulary attribute), 15

tissue (pybiopax.biopax.util.BioSource attribute), 14

TissueVocabulary (class in pybiopax.biopax.util), 22

title (pybiopax.biopax.util.PublicationXref attribute), 19

to\_xml() (pybiopax.biopax.model.BioPaxModel method), 5

Transport (class in pybiopax.biopax.interaction), 11

TransportWithBiochemicalReaction (class in pybiopax.biopax.interaction), 11

**U**

UnificationXref (class in pybiopax.biopax.util), 22

Unresolved (class in pybiopax.biopax.base), 7

url (pybiopax.biopax.util.PublicationXref attribute), 19

UtilityClass (class in pybiopax.biopax.util), 22

**V**

value (pybiopax.biopax.util.Score attribute), 20

**W**

wrap\_xml\_elements() (in module pybiopax.xml\_util), 29

**X**

xml\_base (pybiopax.biopax.model.BioPaxModel attribute), 5

xml\_to\_file() (in module pybiopax.xml\_util), 29

xml\_to\_str() (in module pybiopax.xml\_util), 29

Xref (class in pybiopax.biopax.util), 23

xref (pybiopax.biopax.base.XReferrable attribute), 7

XReferrable (class in pybiopax.biopax.base), 7

**Y**

year (pybiopax.biopax.util.PublicationXref attribute), 20