pycram.designators.object_designator

Contents

Classes

BelieveObject	Description for Objects that are only believed in.
ObjectPart	Object Designator Descriptions for Objects that are part of some other object.
LocatedObject	Description for KnowRob located objects.
RealObject	Object designator representing an object in the real world, when resolving this object designator description]

Module Contents

class pycram.designators.object_designator.**BelieveObject**(*names: typing_extensions.Optional[typing_extensions.List[str]] = None, types: typing_extensions.Optional[typing_extensions.List[pycram.datastructures.enums.ObjectType]] = None, resolver: typing_extensions.Optional[typing_extensions.Callable] = None, ontology_concept_holders: typing_extensions.Optional[typing_extensions.List[owlready2.Thing]] = None*)

Bases: pycram.external_interfaces.robokudo.ObjectDesignatorDescription

Description for Objects that are only believed in.

class **Object**

Bases: pycram.external_interfaces.robokudo.ObjectDesignatorDescription.Object

Concrete object that is believed in.

to_sql() → pycram.orm.object_designator.BelieveObject

Create an ORM object that corresponds to this description.

Returns:

The created ORM object.

insert(*session: sqlalchemy.orm.session.Session*) → pycram.orm.object_designator.BelieveObject

Add and commit this and all related objects to the session. Auto-Incrementing primary keys and foreign key be filled by this method.

Parameters:

session – Session with a database that is used to add and commit the objects

Returns:

The completely instanced ORM object

class pycram.designators.object_designator.**ObjectPart**(*names: pycram.external_interfaces.robokudo.List[str], part_of: pycram.external_interfaces.robokudo.ObjectDesignatorDescription.Object, type: pycram.external_interfaces.robokudo.Optional[pycram.datastructures.enums.ObjectType] = None, resolver: pycram.external_interfaces.robokudo.Optional[pycram.external_interfaces.robokudo.Callable] = None*)

Bases: pycram.external_interfaces.robokudo.ObjectDesignatorDescription

Object Designator Descriptions for Objects that are part of some other object.

class **Object**

Bases: pycram.external_interfaces.robokudo.ObjectDesignatorDescription.Object

A single element that fits the description.

part_pose: *pycram.external_interfaces.robokudo.Pose*

to_sql() → pycram.orm.object_designator.ObjectPart

Create an ORM object that corresponds to this description.

Returns:

The created ORM object.

insert(*session: sqlalchemy.orm.session.Session*) → pycram.orm.object_designator.ObjectPart

Add and commit this and all related objects to the session. Auto-Incrementing primary keys and foreign keys have to be filled by this method.

Parameters:

session – Session with a database that is used to add and commit the objects

Returns:

The completely instanced ORM object

type: *pycram.external_interfaces.robokudo.Optional[pycram.datastructures.enums.ObjectType]*

names: *pycram.external_interfaces.robokudo.Optional[pycram.external_interfaces.robokudo.List[str]]*

latest

part_of

ground() → Object

Default specialized_designators, returns the first result of the iterator of this instance.

Returns:

A resolved object designator

__iter__()

Iterates through every possible solution for the given input parameter.

Yield:

A resolved Object designator

class pycram.designators.object_designator.**LocatedObject**(*names: pycram.external_interfaces.robokudo.List[str], types: pycram.external_interfaces.robokudo.List[str], reference_frames: pycram.external_interfaces.robokudo.List[str], timestamps: pycram.external_interfaces.robokudo.List[float], resolver: pycram.external_interfaces.robokudo.Optional[pycram.external_interfaces.robokudo.Callable] = None, ontology_concept_holders: pycram.external_interfaces.robokudo.Optional[pycram.external_interfaces.robokudo.List[owlready2.Thing]] = None*)

Bases: pycram.external_interfaces.robokudo.ObjectDesignatorDescription

Description for KnowRob located objects. **Currently has no specialized_designators**

class **Object**

Bases: pycram.external_interfaces.robokudo.ObjectDesignatorDescription.Object

A single element that fits the description.

reference_frame: *str*

Reference frame in which the position is given

timestamp: *float*

Timestamp at which the position was valid

reference_frames: *pycram.external_interfaces.robokudo.List[str]*

timestamps: *pycram.external_interfaces.robokudo.List[float]*

class pycram.designators.object_designator.**RealObject**(*names: pycram.external_interfaces.robokudo.Optional[pycram.external_interfaces.robokudo.List[str]] = None, types: pycram.external_interfaces.robokudo.Optional[pycram.external_interfaces.robokudo.List[str None, world_object: pycram.world_concepts.world_object.Object = None, resolver:*

pycram.external_interfaces.robokudo.Optional[pycram.external_interfaces.robokudo.Callable] = *None*)

Bases: `pycram.external_interfaces.robokudo.ObjectDesignatorDescription`

Object designator representing an object in the real world, when resolving this object designator description] RoboKudo is queried to perceive an object fitting the given criteria. Afterward the specialized_designators tries to match the found object to an Object in the World.

class Object

Bases: `pycram.external_interfaces.robokudo.ObjectDesignatorDescription.Object`

A single element that fits the description.

pose: *pycram.external_interfaces.robokudo.Pose*

Pose of the perceived object

types: *pycram.external_interfaces.robokudo.Optional[pycram.external_interfaces.robokudo.List[str]]*

names: *pycram.external_interfaces.robokudo.Optional[pycram.external_interfaces.robokudo.List[str]]*

world_object: *pycram.world_concepts.world_object.Object*

__iter__()

Queries RoboKudo for objects that fit the description and then iterates over all World objects that have the same type to match a World object to the real object.

Yield:

A resolved object designator with reference world object

latest