Skip to main content

__Back to top

__ `Ctrl`+`K`

[![conda 25.1.2.dev29 documentation -

Home](../../../../../_static/conda_logo_full.svg)](../../../../../index.html)

  * [Conda](https://docs.conda.io/projects/conda/)
  * [Conda-build](https://docs.conda.io/projects/conda-build)
  * [Miniconda](https://docs.anaconda.com/free/miniconda/)
  * conda.org

  * [__ GitHub](https://github.com/conda/conda "GitHub")
  * [![Element](../../../../../_static/element_logo.svg)](http://bit.ly/conda-chat-room "Element")
  * [__ Discourse](https://conda.discourse.group/ "Discourse")

  * [Conda](https://docs.conda.io/projects/conda/)
  * [Conda-build](https://docs.conda.io/projects/conda-build)
  * [Miniconda](https://docs.anaconda.com/free/miniconda/)
  * conda.org

  * [__ GitHub](https://github.com/conda/conda "GitHub")
  * [![Element](../../../../../_static/element_logo.svg)](http://bit.ly/conda-chat-room "Element")
  * [__ Discourse](https://conda.discourse.group/ "Discourse")

Navigation

* [Using non-standard certificates](../../../../../user-guide/configuration/non-standard-certs.html)

* [Using Custom Locations for Environment and Package Cache](../../../../../user-guide/configuration/custom-env-and-pkg-locations.html)

* [Improving interoperability with pip](../../../../../user-guide/configuration/pip-interoperability.html)

* [Using the free channel](../../../../../user-guide/configuration/free-channel.html)

* [Concepts](../../../../../user-guide/concepts/index.html) __

* [Commands](../../../../../user-guide/concepts/conda-commands.html)

* [Packages](../../../../../user-guide/concepts/packages.html)

* [Package specification](../../../../../user-guide/concepts/pkg-specs.html)

* [Package search and install specifications](../../../../../user-guide/concepts/pkg-search.html)

* [Channels](../../../../../user-guide/concepts/channels.html)

* [Environments](../../../../../user-guide/concepts/environments.html)

* [Installing with conda](../../../../../user-guide/concepts/installing-with-conda.html)

* [Performance](../../../../../user-guide/concepts/conda-performance.html)

* [Conda for data scientists](../../../../../user-guide/concepts/data-science.html)

* [Plugins](../../../../../user-guide/concepts/conda-plugins.html)

* [Troubleshooting](../../../../../user-guide/troubleshooting.html)

* [Cheatsheet](../../../../../user-guide/cheatsheet.html)

* [Configuration](../../../../../configuration.html)

* [Commands](../../../../../commands/index.html) __

* [` conda clean`](../../../../../commands/clean.html)

* [`conda compare`](../../../../../commands/compare.html)

* [`conda config`](../../../../../commands/config.html)

* [`conda create`](../../../../../commands/create.html)

* [`conda doctor`](../../../../../commands/doctor.html)

* [`conda env`](../../../../../commands/env/index.html) __

* [` conda env config`](../../../../../commands/env/config/index.html) __

        * [` conda env config vars`](../../../../../commands/env/config/vars/index.html) __

          * [` conda env config vars list`](../../../../../commands/env/config/vars/list.html)

          * [`conda env config vars set`](../../../../../commands/env/config/vars/set.html)

          * [`conda env config vars unset`](../../../../../commands/env/config/vars/unset.html)

      * [`conda env create`](../../../../../commands/env/create.html)

      * [`conda env export`](../../../../../commands/env/export.html)

      * [`conda env list`](../../../../../commands/env/list.html)

      * [`conda env remove`](../../../../../commands/env/remove.html)

      * [`conda env update`](../../../../../commands/env/update.html)

    * [`conda info`](../../../../../commands/info.html)

    * [`conda init`](../../../../../commands/init.html)

    * [`conda install`](../../../../../commands/install.html)

    * [`conda list`](../../../../../commands/list.html)

    * [`conda notices`](../../../../../commands/notices.html)

    * [`conda package`](../../../../../commands/package.html)

    * [`conda remove`](../../../../../commands/remove.html)

    * [`conda rename`](../../../../../commands/rename.html)

    * [`conda run`](../../../../../commands/run.html)

    * [`conda search`](../../../../../commands/search.html)

    * [`conda update`](../../../../../commands/update.html)

  * [Release notes](../../../../../release-notes.html)

  * [Glossary](../../../../../glossary.html)

  * [Developer guide](../../../../index.html) __

    * [Architecture](../../../../architecture.html)

    * [Contributing to conda](../../../../contributing.html)

    * [Development Environment](../../../../development-environment.html)

    * [Deep dives](../../../../deep-dives/index.html) __

  * [` conda install`](../../../../deep-dives/install.html)
  * [`conda init` and `conda activate`](../../../../deep-dives/activation.html)
  * [`conda config` and context](../../../../deep-dives/context.html)
  * [Solvers](../../../../deep-dives/solvers.html)
  * [Logging](../../../../deep-dives/logging.html)
* [Writing Tests](../../../../writing-tests/index.html) __
  * [Integration Tests](../../../../writing-tests/integration-tests.html)
* [Deprecations](../../../../deprecations.html)
* [Releasing](../../../../releasing.html)
* [Plugins](../../../../plugins/index.html) __
  * [Auth Handlers](../../../../plugins/auth_handlers.html)
  * [Health Checks](../../../../plugins/health_checks.html)
  * [Request Headers](../../../../plugins/request_headers.html)
  * [Post-commands](../../../../plugins/post_commands.html)
  * [Pre-commands](../../../../plugins/pre_commands.html)
  * [Reporter Backends](../../../../plugins/reporter_backends.html)
  * [Settings](../../../../plugins/settings.html)
  * [Solvers](../../../../plugins/solvers.html)
  * [Subcommands](../../../../plugins/subcommands.html)
  * [Virtual Packages](../../../../plugins/virtual_packages.html)
* [Specifications](../../../../specs/index.html) __
  * [Technical specification: solver state](../../../../specs/solver-state.html)
* [API](../../../../api.html) __
  * [` conda`](../../index.html) __
    * [` __main__`](../../__main__/index.html)
    * [`_vendor`](../../_vendor/index.html) __
      * [` frozendict`](../../_vendor/frozendict/index.html)

* [`_version`](../../_version/index.html)
* [`activate`](../../activate/index.html)
* [`api`](../../api/index.html)
* [`auxlib`](../index.html) __
  * [` collection`](../collection/index.html)
  * [`compat`](../compat/index.html)
  * [`decorators`](../decorators/index.html)
  * `entity`
  * [`exceptions`](../exceptions/index.html)
  * [`ish`](../ish/index.html)
  * [`logz`](../logz/index.html)
  * [`type_coercion`](../type_coercion/index.html)
* [`base`](../../base/index.html) __
  * [` constants`](../../base/constants/index.html)
  * [`context`](../../base/context/index.html)
* [`cli`](../../cli/index.html) __
  * [` actions`](../../cli/actions/index.html)
  * [`common`](../../cli/common/index.html)
  * [`conda_argparse`](../../cli/conda_argparse/index.html)
  * [`find_commands`](../../cli/find_commands/index.html)
  * [`helpers`](../../cli/helpers/index.html)
  * [`install`](../../cli/install/index.html)
  * [`main`](../../cli/main/index.html)
  * [`main_clean`](../../cli/main_clean/index.html)
  * [`main_commands`](../../cli/main_commands/index.html)
  * [`main_compare`](../../cli/main_compare/index.html)
  * [`main_config`](../../cli/main_config/index.html)

  * [`main_create`](../../cli/main_create/index.html)
  * [`main_env`](../../cli/main_env/index.html)
  * [`main_env_config`](../../cli/main_env_config/index.html)
  * [`main_env_create`](../../cli/main_env_create/index.html)
  * [`main_env_export`](../../cli/main_env_export/index.html)
  * [`main_env_list`](../../cli/main_env_list/index.html)
  * [`main_env_remove`](../../cli/main_env_remove/index.html)
  * [`main_env_update`](../../cli/main_env_update/index.html)
  * [`main_env_vars`](../../cli/main_env_vars/index.html)
  * [`main_export`](../../cli/main_export/index.html)
  * [`main_info`](../../cli/main_info/index.html)
  * [`main_init`](../../cli/main_init/index.html)
  * [`main_install`](../../cli/main_install/index.html)
  * [`main_list`](../../cli/main_list/index.html)
  * [`main_mock_activate`](../../cli/main_mock_activate/index.html)
  * [`main_mock_deactivate`](../../cli/main_mock_deactivate/index.html)
  * [`main_notices`](../../cli/main_notices/index.html)
  * [`main_package`](../../cli/main_package/index.html)
  * [`main_pip`](../../cli/main_pip/index.html)
  * [`main_remove`](../../cli/main_remove/index.html)
  * [`main_rename`](../../cli/main_rename/index.html)
  * [`main_run`](../../cli/main_run/index.html)
  * [`main_search`](../../cli/main_search/index.html)
  * [`main_update`](../../cli/main_update/index.html)
  * [`python_api`](../../cli/python_api/index.html)
* [`common`](../../common/index.html) __
  * [` _logic`](../../common/_logic/index.html)

* [`_os`](../../common/_os/index.html) __
  * [` linux`](../../common/_os/linux/index.html)
  * [`osx`](../../common/_os/osx/index.html)
  * [`unix`](../../common/_os/unix/index.html)
  * [`windows`](../../common/_os/windows/index.html)
* [`compat`](../../common/compat/index.html)
* [`configuration`](../../common/configuration/index.html)
* [`constants`](../../common/constants/index.html)
* [`disk`](../../common/disk/index.html)
* [`io`](../../common/io/index.html)
* [`iterators`](../../common/iterators/index.html)
* [`logic`](../../common/logic/index.html)
* [`path`](../../common/path/index.html) __
  * [` _cygpath`](../../common/path/_cygpath/index.html)
  * [`directories`](../../common/path/directories/index.html)
  * [`python`](../../common/path/python/index.html)
  * [`windows`](../../common/path/windows/index.html)
* [`pkg_formats`](../../common/pkg_formats/index.html) __
  * [` python`](../../common/pkg_formats/python/index.html)
* [`serialize`](../../common/serialize/index.html)
* [`signals`](../../common/signals/index.html)
* [`toposort`](../../common/toposort/index.html)
* [`url`](../../common/url/index.html)

* [`core`](../../core/index.html) __
  * [` envs_manager`](../../core/envs_manager/index.html)
  * [`index`](../../core/index/index.html)
  * [`initialize`](../../core/initialize/index.html)

  * [`link`](../../core/link/index.html)
  * [`package_cache_data`](../../core/package_cache_data/index.html)
  * [`path_actions`](../../core/path_actions/index.html)
  * [`portability`](../../core/portability/index.html)
  * [`prefix_data`](../../core/prefix_data/index.html)
  * [`solve`](../../core/solve/index.html)
  * [`subdir_data`](../../core/subdir_data/index.html)
* [`deprecations`](../../deprecations/index.html)
* [`env`](../../env/index.html) __
  * [` env`](../../env/env/index.html)
  * [`installers`](../../env/installers/index.html) __
    * [` base`](../../env/installers/base/index.html)
    * [`conda`](../../env/installers/conda/index.html)
    * [`pip`](../../env/installers/pip/index.html)
  * [`pip_util`](../../env/pip_util/index.html)
  * [`specs`](../../env/specs/index.html) __
    * [` binstar`](../../env/specs/binstar/index.html)
    * [`requirements`](../../env/specs/requirements/index.html)
    * [`yaml_file`](../../env/specs/yaml_file/index.html)
* [`exception_handler`](../../exception_handler/index.html)
* [`exceptions`](../../exceptions/index.html)
* [`exports`](../../exports/index.html)
* [`gateways`](../../gateways/index.html) __
  * [` anaconda_client`](../../gateways/anaconda_client/index.html)
  * [`connection`](../../gateways/connection/index.html) __
    * [` adapters`](../../gateways/connection/adapters/index.html) __
      * [` ftp`](../../gateways/connection/adapters/ftp/index.html)

        * [`http`](../../gateways/connection/adapters/http/index.html)
        * [`localfs`](../../gateways/connection/adapters/localfs/index.html)
        * [`s3`](../../gateways/connection/adapters/s3/index.html)
      * [`download`](../../gateways/connection/download/index.html)
      * [`session`](../../gateways/connection/session/index.html)
    * [`disk`](../../gateways/disk/index.html) __
      * [` create`](../../gateways/disk/create/index.html)
      * [`delete`](../../gateways/disk/delete/index.html)
      * [`link`](../../gateways/disk/link/index.html)
      * [`lock`](../../gateways/disk/lock/index.html)
      * [`permissions`](../../gateways/disk/permissions/index.html)
      * [`read`](../../gateways/disk/read/index.html)
      * [`test`](../../gateways/disk/test/index.html)
      * [`update`](../../gateways/disk/update/index.html)
    * [`logging`](../../gateways/logging/index.html)
    * [`repodata`](../../gateways/repodata/index.html) __
      * [` jlap`](../../gateways/repodata/jlap/index.html) __
        * [` core`](../../gateways/repodata/jlap/core/index.html)
        * [`fetch`](../../gateways/repodata/jlap/fetch/index.html)
        * [`interface`](../../gateways/repodata/jlap/interface/index.html)
      * [`lock`](../../gateways/repodata/lock/index.html)
    * [`subprocess`](../../gateways/subprocess/index.html)
  * [`history`](../../history/index.html)
  * [`instructions`](../../instructions/index.html)
  * [`misc`](../../misc/index.html)
  * [`models`](../../models/index.html) __
    * [` channel`](../../models/channel/index.html)

  * [`dist`](../../models/dist/index.html)
  * [`enums`](../../models/enums/index.html)
  * [`leased_path_entry`](../../models/leased_path_entry/index.html)
  * [`match_spec`](../../models/match_spec/index.html)
  * [`package_info`](../../models/package_info/index.html)
  * [`prefix_graph`](../../models/prefix_graph/index.html)
  * [`records`](../../models/records/index.html)
  * [`version`](../../models/version/index.html)
* [`notices`](../../notices/index.html) __
  * [` cache`](../../notices/cache/index.html)
  * [`core`](../../notices/core/index.html)
  * [`fetch`](../../notices/fetch/index.html)
  * [`types`](../../notices/types/index.html)
  * [`views`](../../notices/views/index.html)
* [`plan`](../../plan/index.html)
* [`plugins`](../../plugins/index.html) __
  * [` hookspec`](../../plugins/hookspec/index.html)
  * [`manager`](../../plugins/manager/index.html)
  * [`post_solves`](../../plugins/post_solves/index.html) __
    * [` signature_verification`](../../plugins/post_solves/signature_verification/index.html)
  * [`reporter_backends`](../../plugins/reporter_backends/index.html) __
    * [` console`](../../plugins/reporter_backends/console/index.html)
    * [`json`](../../plugins/reporter_backends/json/index.html)
  * [`solvers`](../../plugins/solvers/index.html)
  * [`subcommands`](../../plugins/subcommands/index.html) __
    * [` doctor`](../../plugins/subcommands/doctor/index.html) __
      * [` health_checks`](../../plugins/subcommands/doctor/health_checks/index.html)

    * [`types`](../../plugins/types/index.html)
    * [`virtual_packages`](../../plugins/virtual_packages/index.html) __
      * [` archspec`](../../plugins/virtual_packages/archspec/index.html)
      * [`conda`](../../plugins/virtual_packages/conda/index.html)
      * [`cuda`](../../plugins/virtual_packages/cuda/index.html)
      * [`freebsd`](../../plugins/virtual_packages/freebsd/index.html)
      * [`linux`](../../plugins/virtual_packages/linux/index.html)
      * [`osx`](../../plugins/virtual_packages/osx/index.html)
      * [`windows`](../../plugins/virtual_packages/windows/index.html)
  * [`reporters`](../../reporters/index.html)
  * [`resolve`](../../resolve/index.html)
  * [`testing`](../../testing/index.html) __
    * [` cases`](../../testing/cases/index.html)
    * [`fixtures`](../../testing/fixtures/index.html)
    * [`gateways`](../../testing/gateways/index.html) __
      * [` fixtures`](../../testing/gateways/fixtures/index.html)
    * [`helpers`](../../testing/helpers/index.html)
    * [`integration`](../../testing/integration/index.html)
    * [`notices`](../../testing/notices/index.html) __
      * [` fixtures`](../../testing/notices/fixtures/index.html)
      * [`helpers`](../../testing/notices/helpers/index.html)
    * [`solver_helpers`](../../testing/solver_helpers/index.html)
  * [`trust`](../../trust/index.html) __
    * [` constants`](../../trust/constants/index.html)
    * [`signature_verification`](../../trust/signature_verification/index.html)
  * [`utils`](../../utils/index.html)
* [`conda_env`](../../../conda_env/index.html) __

        * [` cli`](../../../conda_env/cli/index.html)
        * [`installers`](../../../conda_env/installers/index.html)

  * [__](../../../../../index.html)
  * [Developer guide](../../../../index.html)
  * __
  * [` auxlib`](../index.html)
  *

`entity`#

This module provides serializable, validatable, type-enforcing domain objects
and data transfer objects. It has many of the same motivations as the python
[Marshmallow](http://marshmallow.readthedocs.org/en/latest/why.html) package.
It is most similar to [Schematics](http://schematics.readthedocs.io/).

Tutorial#

Chapter 1: Entity and Field Basics#

```
    >>> class Color(Enum):
    ...     blue = 0
    ...     black = 1
    ...     red = 2
    >>> class Car(Entity):
```

```
...     weight = NumberField(required=False)
...     wheels = IntField(default=4, validation=lambda x: 3 <= x <= 4)
...     color = EnumField(Color)



>>> # create a new car object
>>> car = Car(color=Color.blue, weight=4242.46)
>>> car
Car(weight=4242.46, color=0)



>>> # it has 4 wheels, all by default
>>> car.wheels
4



>>> # but a car can't have 5 wheels!
>>> #  the `validation=` field is a simple callable that returns a
>>> #  boolean based on validity
>>> car.wheels = 5
Traceback (most recent call last):
ValidationError: Invalid value 5 for wheels
```

```
>>> # we can call .dump() on car, and just get back a standard

>>> #  python dict actually, it's an ordereddict to match attribute

>>> #  declaration order

>>> type(car.dump())

<class '...OrderedDict'>

>>> car.dump()

OrderedDict([('weight', 4242.46), ('wheels', 4), ('color', 0)])




>>> # and json too (note the order!)

>>> car.json()

'{"weight": 4242.46, "wheels": 4, "color": 0}'




>>> # green cars aren't allowed

>>> car.color = "green"

Traceback (most recent call last):

ValidationError: 'green' is not a valid Color




>>> # but black cars are!

>>> car.color = "black"

>>> car.color
```

```
<Color.black: 1>



>>> # car.color really is an enum, promise
>>> type(car.color)
<enum 'Color'>



>>> # enum assignment can be with any of (and preferentially)
>>> #   (1) an enum literal,
>>> #   (2) a valid enum value, or
>>> #   (3) a valid enum name
>>> car.color = Color.blue; car.color.value
0
>>> car.color = 1; car.color.name
'black'



>>> # let's do a round-trip marshalling of this thing
>>> same_car = Car.from_json(car.json())  # or equally Car.from_json(json.dumps(car.dump()))
>>> same_car == car
True
```

```
>>> # actually, they're two different instances
>>> same_car is not car
True



>>> # this works too
>>> cloned_car = Car(**car.dump())
>>> cloned_car == car
True



>>> # while we're at it, these are all equivalent too
>>> car == Car.from_objects(car)
True
>>> car == Car.from_objects({"weight": 4242.46, "wheels": 4, "color": 1})
True
>>> car == Car.from_json('{"weight": 4242.46, "color": 1}')
True



>>> # .from_objects() even lets you stack and combine objects
>>> class DumbClass:
...     color = 0
```

```
...     wheels = 3
>>> Car.from_objects(DumbClass(), dict(weight=2222, color=1))
Car(weight=2222, wheels=3, color=0)
>>> # and also pass kwargs that override properties pulled
>>> #  off any objects
>>> Car.from_objects(DumbClass(), {'weight': 2222, 'color': 1}, color=2, weight=33)
Car(weight=33, wheels=3, color=2)
```

Chapter 2: Entity and Field Composition#

```
>>> # now let's get fancy
>>> # a ComposableField "nests" another valid Entity
>>> # a ListField's first argument is a "generic" type,
>>> #   which can be a valid Entity, any python primitive
>>> #   type, or a list of Entities/types
>>> class Fleet(Entity):
...     boss_car = ComposableField(Car)
...     cars = ListField(Car)


>>> # here's our fleet of company cars
>>> company_fleet = Fleet(boss_car=Car(color='red'), cars=[car, same_car, cloned_car])
>>> company_fleet.pretty_json()
```

```
{
  "boss_car": {
    "wheels": 4
    "color": 2,
  },
  "cars": [
    {
      "weight": 4242.46,
      "wheels": 4
      "color": 1,
    },
    {
      "weight": 4242.46,
      "wheels": 4
      "color": 1,
    },
    {
      "weight": 4242.46,
      "wheels": 4
      "color": 1,
    }
  ]
}
```


```
>>> # the boss' car is red of course (and it's still an Enum)
```

```
>>> company_fleet.boss_car.color.name
'red'



>>> # and there are three cars left for the employees
>>> len(company_fleet.cars)
3
```

Chapter 3: Immutability#

```
>>> class ImmutableCar(ImmutableEntity):
...     wheels = IntField(default=4, validation=lambda x: 3 <= x <= 4)
...     color = EnumField(Color)
>>> icar = ImmutableCar.from_objects({'wheels': 3, 'color': 'blue'})
>>> icar
ImmutableCar(wheels=3, color=0)



>>> icar.wheels = 4
Traceback (most recent call last):
AttributeError: Assignment not allowed. ImmutableCar is immutable.
```

```
>>> class FixedWheelCar(Entity):
...     wheels = IntField(default=4, immutable=True)
...     color = EnumField(Color)
>>> fwcar = FixedWheelCar.from_objects(icar)
>>> fwcar.json()
'{"wheels": 3, "color": 0}'



>>> # repainting the car is easy
>>> fwcar.color = Color.red
>>> fwcar.color.name
'red'



>>> # can't really change the number of wheels though
>>> fwcar.wheels = 18
Traceback (most recent call last):
AttributeError: The wheels field is immutable.
```

Chapter X: The del and null Weeds#

```
>>> old_date = lambda: isoparse('1982-02-17')
>>> class CarBattery(Entity):
...     # NOTE: default value can be a callable!
...     first_charge = DateField(required=False)  # default=None, nullable=False
...     latest_charge = DateField(default=old_date, nullable=True)  # required=True
...     expiration = DateField(default=old_date, required=False, nullable=False)



>>> # starting point
>>> battery = CarBattery()
>>> battery
CarBattery()
>>> battery.json()
'{"latest_charge": "1982-02-17T00:00:00", "expiration": "1982-02-17T00:00:00"}'



>>> # first_charge is not assigned a default value. Once one is assigned, it can be deleted,
>>> #   but it can't be made null.
>>> battery.first_charge = isoparse('2016-03-23')
>>> battery
CarBattery(first_charge=datetime.datetime(2016, 3, 23, 0, 0))
>>> battery.first_charge = None
Traceback (most recent call last):
ValidationError: Value for first_charge not given or invalid.
```

```
>>> del battery.first_charge
>>> battery
CarBattery()



>>> # latest_charge can be null, but it can't be deleted. The default value is a callable.
>>> del battery.latest_charge
Traceback (most recent call last):
AttributeError: The latest_charge field is required and cannot be deleted.
>>> battery.latest_charge = None
>>> battery.json()
'{"latest_charge": null, "expiration": "1982-02-17T00:00:00"}'



>>> # expiration is assigned by default, can't be made null, but can be deleted.
>>> battery.expiration
datetime.datetime(1982, 2, 17, 0, 0)
>>> battery.expiration = None
Traceback (most recent call last):
ValidationError: Value for expiration not given or invalid.
>>> del battery.expiration
>>> battery.json()
'{"latest_charge": null}'
```

Classes#

`Field` | Fields are doing something very similar to boxing and unboxing

---|---

`BooleanField` | Fields are doing something very similar to boxing and unboxing

`IntegerField` | Fields are doing something very similar to boxing and unboxing

`NumberField` | Fields are doing something very similar to boxing and unboxing

`StringField` | Fields are doing something very similar to boxing and unboxing

`DateField` | Fields are doing something very similar to boxing and unboxing

`EnumField` | Fields are doing something very similar to boxing and unboxing

`ListField` | Fields are doing something very similar to boxing and unboxing

`MapField` | Fields are doing something very similar to boxing and unboxing

`ComposableField` | Fields are doing something very similar to boxing and unboxing

`Entity` |

`ImmutableEntity` |

Attributes#

`BoolField` |

---|---

`IntField` |

_class _Field(_default =NULL_, _required =True_, _validation =None_, _in_dump

=True_, _default_in_dump =True_, _nullable =False_, _immutable =False_,

aliases =())#

Fields are doing something very similar to boxing and unboxing of c#/java primitives. __set__ should take a "primitive" or "raw" value and create a "boxed" or "programmatically usable" value of it. While __get__ should return the boxed value, dump in turn should unbox the value into a primitive or raw value.

Parameters:

* **types** (_primitive literal_ _or_[_type_](https://docs.python.org/3/library/functions.html#type "\(in Python v3.13\)") _or_ _sequence_ _of_ _types_)

* **default** (_any_ _,__callable_ _,__optional_) -- If default is callable, it's guaranteed to return a valid value at the time of Entity creation.

* **required** (_boolean_ _,__optional_)

* **validation** (_callable_ _,__optional_)

* **dump** (_boolean_ _,__optional_)

_property _name#

_property _required#

_property _type#

_property _default#

_property _in_dump#

_property _default_in_dump#

_property _nullable#

_property _is_nullable#

_property _immutable#

_order_helper _ = 0_#

set_name(_name_)#

__get__(_instance_ , _instance_type_)#

__set__(_instance_ , _val_)#

__delete__(_instance_)#

box(_instance_ , _instance_type_ , _val_)#

unbox(_instance_ , _instance_type_ , _val_)#

dump(_instance_ , _instance_type_ , _val_)#

validate(_instance_ , _val_)#

Returns:

if val is valid

Return type:

True

Raises:

[**ValidationError**](../exceptions/index.html#conda.auxlib.exceptions.ValidationError "conda.auxlib.exceptions.ValidationError") \--

_class _BooleanField(_default =NULL_, _required =True_, _validation =None_, _in_dump =True_, _default_in_dump =True_, _nullable =False_, _immutable =False_, _aliases =()_)#

Bases: `Field`

Fields are doing something very similar to boxing and unboxing of c#/java primitives. __set__ should take a "primitive" or "raw" value and create a "boxed" or "programmatically usable" value of it. While __get__ should return the boxed value, dump in turn should unbox the value into a primitive or raw value.

Parameters:

* **types** (_primitive literal_ _or_[_type_](https://docs.python.org/3/library/functions.html#type "\(in Python v3.13\)") _or_ _sequence_ _of_ _types_)

* **default** (_any_ _,__callable_ _,__optional_) -- If default is callable, it's guaranteed to return a valid value at the time of Entity creation.

* **required** (_boolean_ _,__optional_)

* **validation** (_callable_ _,__optional_)

* **dump** (_boolean_ _,__optional_)

_type#

box(_instance_ , _instance_type_ , _val_)#

BoolField#

_class _IntegerField(_default =NULL_, _required =True_, _validation =None_, _in_dump =True_, _default_in_dump =True_, _nullable =False_, _immutable =False_, _aliases =()_)#

Bases: `Field`

Fields are doing something very similar to boxing and unboxing of c#/java primitives. __set__ should take a "primitive" or "raw" value and create a "boxed" or "programmatically usable" value of it. While __get__ should return the boxed value, dump in turn should unbox the value into a primitive or raw value.

Parameters:

* **types** (_primitive literal_ _or_[_type_](https://docs.python.org/3/library/functions.html#type "\(in Python v3.13\)") _or_ _sequence_ _of_ _types_)

* **default** (_any_ _,__callable_ _,__optional_) -- If default is callable, it's guaranteed to return a valid value at the time of Entity creation.

* **required** (_boolean_ _,__optional_)

* **validation** (_callable_ _,__optional_)

* **dump** (_boolean_ _,__optional_)

_type#

IntField#

_class _NumberField(_default =NULL_, _required =True_, _validation =None_, _in_dump =True_, _default_in_dump =True_, _nullable =False_, _immutable =False_, _aliases =()_)#

Bases: `Field`

Fields are doing something very similar to boxing and unboxing of c#/java primitives. __set__ should take a "primitive" or "raw" value and create a "boxed" or "programmatically usable" value of it. While __get__ should return the boxed value, dump in turn should unbox the value into a primitive or raw value.

Parameters:

* **types** (_primitive literal_ _or_[_type_](https://docs.python.org/3/library/functions.html#type "\(in Python v3.13\)") _or_ _sequence_ _of_ _types_)

* **default** (_any_ _,__callable_ _,__optional_) -- If default is callable, it's guaranteed to return a

valid value at the time of Entity creation.

* **required** (_boolean_ _,__optional_)

* **validation** (_callable_ _,__optional_)

* **dump** (_boolean_ _,__optional_)

_type _ = ()_#

_class _StringField(_default =NULL_, _required =True_, _validation =None_, _in_dump =True_, _default_in_dump =True_, _nullable =False_, _immutable =False_, _aliases =()_)#

Bases: `Field`

Fields are doing something very similar to boxing and unboxing of c#/java primitives. __set__ should take a "primitive" or "raw" value and create a "boxed" or "programmatically usable" value of it. While __get__ should return the boxed value, dump in turn should unbox the value into a primitive or raw value.

Parameters:

* **types** (_primitive literal_ _or_[_type_](https://docs.python.org/3/library/functions.html#type "\(in Python v3.13\)") _or_ _sequence_ _of_ _types_)

* **default** (_any_ _,__callable_ _,__optional_) -- If default is callable, it's guaranteed to return a valid value at the time of Entity creation.

* **required** (_boolean_ _,__optional_)

* **validation** (_callable_ _,__optional_)

* **dump** (_boolean_ _,__optional_)

_type#

box(_instance_ , _instance_type_ , _val_)#

_class _DateField(_default =NULL_, _required =True_, _validation =None_, _in_dump =True_, _default_in_dump =True_, _nullable =False_, _immutable =False_, _aliases =()_)#

Bases: `Field`

Fields are doing something very similar to boxing and unboxing of c#/java primitives. __set__ should take a "primitive" or "raw" value and create a "boxed" or "programmatically usable" value of it. While __get__ should return the boxed value, dump in turn should unbox the value into a primitive or raw value.

Parameters:

* **types** (_primitive literal_ _or_[_type_](https://docs.python.org/3/library/functions.html#type "\(in Python v3.13\)") _or_ _sequence_ _of_ _types_)

* **default** (_any_ _,__callable_ _,__optional_) -- If default is callable, it's guaranteed to return a valid value at the time of Entity creation.

* **required** (_boolean_ _,__optional_)

* **validation** (_callable_ _,__optional_)

* **dump** (_boolean_ _,__optional_)

_type#

box(_instance_ , _instance_type_ , _val_)#

dump(_instance_ , _instance_type_ , _val_)#

_class _EnumField(_enum_class_ , _default =NULL_, _required =True_, _validation =None_, _in_dump =True_, _default_in_dump =True_, _nullable =False_, _immutable =False_, _aliases =()_)#

Bases: `Field`

Fields are doing something very similar to boxing and unboxing of c#/java primitives. __set__ should take a "primitive" or "raw" value and create a "boxed" or "programmatically usable" value of it. While __get__ should return the boxed value, dump in turn should unbox the value into a primitive or raw value.

Parameters:

* **types** (_primitive literal_ _or_[_type_](https://docs.python.org/3/library/functions.html#type "\(in Python v3.13\)") _or_ _sequence_ _of_ _types_)

* **default** (_any_ _,__callable_ _,__optional_) -- If default is callable, it's guaranteed to return a valid value at the time of Entity creation.

* **required** (_boolean_ _,__optional_)

* **validation** (_callable_ _,__optional_)

* **dump** (_boolean_ _,__optional_)

box(_instance_ , _instance_type_ , _val_)#

dump(_instance_ , _instance_type_ , _val_)#

_class _ListField(_element_type_ , _default =NULL_, _required =True_, _validation =None_, _in_dump =True_, _default_in_dump =True_, _nullable =False_, _immutable =False_, _aliases =()_)#

Bases: `Field`

Fields are doing something very similar to boxing and unboxing of c#/java primitives. __set__ should take a "primitive" or "raw" value and create a "boxed" or "programmatically usable" value of it. While __get__ should return the boxed value, dump in turn should unbox the value into a primitive or raw value.

Parameters:

* **types** (_primitive literal_ _or_[_type_](https://docs.python.org/3/library/functions.html#type "\(in Python v3.13\)") _or_ _sequence_ _of_ _types_)

* **default** (_any_ _,__callable_ _,__optional_) -- If default is callable, it's guaranteed to return a valid value at the time of Entity creation.

* **required** (_boolean_ _,__optional_)

* **validation** (_callable_ _,__optional_)

* **dump** (_boolean_ _,__optional_)

_type#

box(_instance_ , _instance_type_ , _val_)#

unbox(_instance_ , _instance_type_ , _val_)#

dump(_instance_ , _instance_type_ , _val_)#

validate(_instance_ , _val_)#

Returns:

if val is valid

Return type:

True

Raises:

[**ValidationError**](../exceptions/index.html#conda.auxlib.exceptions.ValidationError "conda.auxlib.exceptions.ValidationError") \--

_class _MapField(_default =NULL_, _required =True_, _validation =None_, _in_dump =True_, _default_in_dump =True_, _nullable =False_, _immutable =True_, _aliases =()_)#

Bases: `Field`

Fields are doing something very similar to boxing and unboxing of c#/java primitives. __set__ should take a "primitive" or "raw" value and create a "boxed" or "programmatically usable" value of it. While __get__ should return the boxed value, dump in turn should unbox the value into a primitive or raw value.

Parameters:

* **types** (_primitive literal_ _or_[_type_](https://docs.python.org/3/library/functions.html#type "\(in Python v3.13\)") _or_ _sequence_ _of_ _types_)

* **default** (_any_ _,__callable_ _,__optional_) -- If default is callable, it's guaranteed to return a valid value at the time of Entity creation.

* **required** (_boolean_ _,__optional_)

* **validation** (_callable_ _,__optional_)

* **dump** (_boolean_ _,__optional_)

_type#

box(_instance_ , _instance_type_ , _val_)#

_class _ComposableField(_field_class_ , _default =NULL_, _required =True_, _validation =None_, _in_dump =True_, _default_in_dump =True_, _nullable =False_, _immutable =False_, _aliases =()_)#

Bases: `Field`

Fields are doing something very similar to boxing and unboxing of c#/java primitives. __set__ should take a "primitive" or "raw" value and create a "boxed" or "programmatically usable" value of it. While __get__ should return the boxed value, dump in turn should unbox the value into a primitive or raw value.

Parameters:

* **types** (_primitive literal_ _or_[_type_](https://docs.python.org/3/library/functions.html#type "\(in Python v3.13\)") _or_ _sequence_ _of_ _types_)

* **default** (_any_ _,__callable_ _,__optional_) -- If default is callable, it's guaranteed to return a valid value at the time of Entity creation.

* **required** (_boolean_ _,__optional_)

* **validation** (_callable_ _,__optional_)

* **dump** (_boolean_ _,__optional_)

box(_instance_ , _instance_type_ , _val_)#

dump(_instance_ , _instance_type_ , _val_)#

_class _Entity(_** kwargs_)#

_property __initd#

__fields__#

_lazy_validate _ = False_#

_classmethod _from_objects(_* objects_, _** override_fields_)#

Construct a new object of type `cls` from existing objects or dicts.

Allows the creation of new objects of concrete `Entity` subclasses by
combining information from several sources. This can be any combination of
objects and dictionaries passed in as positional arguments. When looking for
the value of the fields of the `Entity` subclass, the first object that
provides an attribute (or, in the case of a dict an entry) that has the name
of the field or one of its aliases will take precedence. Any keyword arguments
passed in will override this and take precedence.

Parameters:

* **cls** (`Entity` subclass) -- The class to create, usually determined by call, e.g. `PrefixRecord.from_objects(...)`.

* ***objects** ([_tuple_](https://docs.python.org/3/library/stdtypes.html#tuple "\(in Python v3.13\)") _(_[_object_](https://docs.python.org/3/library/functions.html#object "\(in Python v3.13\)") _or_[_dict_](https://docs.python.org/3/library/stdtypes.html#dict "\(in Python v3.13\)") _)_) -- Any combination of objects and dicts in order of decending precedence.

* ****override_fields** ([_dict_](https://docs.python.org/3/library/stdtypes.html#dict "\(in Python v3.13\)") _(_[_str_](https://docs.python.org/3/library/stdtypes.html#str "\(in Python v3.13\)") _,_[_object_](https://docs.python.org/3/library/functions.html#object "\(in Python v3.13\)") _)_) -- Any individual fields overriding possible contents from `*objects`.

_classmethod _from_json(_json_str_)#

_classmethod _load(_data_dict_)#

validate()#

__repr__()#

Return repr(self).

_classmethod ___register__()#

json(_indent =None_, _separators =None_, _** kwargs_)#

pretty_json(_indent =2_, _separators =(',', ': ')_, _** kwargs_)#

dump()#

_classmethod ___dump_fields()#

__eq__(_other_)#

Return self==value.

__hash__()#

Return hash(self).

_class _ImmutableEntity(_** kwargs_)#

Bases: `Entity`

__setattr__(_attribute_ , _value_)#

Implement setattr(self, name, value).

__delattr__(_item_)#

Implement delattr(self, name).

__On this page

  * `Field.default_in_dump`
  * `Field.nullable`
  * `Field.is_nullable`
  * `Field.immutable`
  * `Field._order_helper`
  * `Field.set_name()`
  * `Field.__get__()`
  * `Field.__set__()`
  * `Field.__delete__()`
  * `Field.box()`
  * `Field.unbox()`
  * `Field.dump()`
  * `Field.validate()`
* `BooleanField`
  * `BooleanField._type`
  * `BooleanField.box()`
* `BoolField`
* `IntegerField`
  * `IntegerField._type`
* `IntField`
* `NumberField`
  * `NumberField._type`
* `StringField`
  * `StringField._type`
  * `StringField.box()`
* `DateField`
  * `DateField._type`

  * `DateField.box()`
  * `DateField.dump()`
* `EnumField`
  * `EnumField.box()`
  * `EnumField.dump()`
* `ListField`
  * `ListField._type`
  * `ListField.box()`
  * `ListField.unbox()`
  * `ListField.dump()`
  * `ListField.validate()`
* `MapField`
  * `MapField._type`
  * `MapField.box()`
* `ComposableField`
  * `ComposableField.box()`
  * `ComposableField.dump()`
* `Entity`
  * `Entity._initd`
  * `Entity.__fields__`
  * `Entity._lazy_validate`
  * `Entity.from_objects()`
  * `Entity.from_json()`
  * `Entity.load()`
  * `Entity.validate()`
  * `Entity.__repr__()`
  * `Entity.__register__()`

      * `Entity.json()`
      * `Entity.pretty_json()`
      * `Entity.dump()`
      * `Entity.__dump_fields()`
      * `Entity.__eq__()`
      * `Entity.__hash__()`
    * `ImmutableEntity`
      * `ImmutableEntity.__setattr__()`
      * `ImmutableEntity.__delattr__()`

[__Edit on GitHub](https://github.com/conda/conda/edit/main/docs/source/dev-guide/api/conda/auxlib/entity/index.rst)

[__Show Source](../../../../../_sources/dev-guide/api/conda/auxlib/entity/index.rst.txt)