Python Dataclasses

`Dataclasses` are python classes, but are suited for storing data objects. This module provides a decorator and functions for automatically adding generated special methods such as `__init__()` and `__repr__()` to user-defined classes.

Features

1. They store data and represent a certain data type. Ex: A number. For people familiar with ORMs, a model instance is a data object. It represents a specific kind of entity. It holds attributes that define or represent the entity.
2. They can be compared to other objects of the same type. Ex: A number can be greater than, less than, or equal to another number.

Python 3.7 provides a decorator dataclass that is used to convert a class into a dataclass.

```
>>> class Number:
...     def __init__(self, val):
...         self.val = val
...
>>> obj = Number(2)
>>> obj.val
# 2
```

with dataclass

```
>>> @dataclass
... class Number:
...     val: int
...
>>> obj = Number(2)
>>> obj.val
# 2
```

Default values

It is easy to add default values to the fields of your data class.

```
>>> @dataclass
... class Product:
...     name: str
...     count: int = 0
...     price: float = 0.0
...
>>> obj = Product("Python")
>>> obj.name
# Python

>>> obj.count
# 0
```

```
>>> obj.price
# 0.0
```

Type hints

It is mandatory to define the data type in dataclass. However, If you would rather not specify the datatype then, use `typing.Any`.

```
>>> from dataclasses import dataclass
>>> from typing import Any

>>> @dataclass
... class WithoutExplicitTypes:
...     name: Any
...     value: Any = 42
```