

Built-in Methods

Instance Method vs Static Method

Instance Methods (normal methods without declaring @) specifies with that particular declared instance. For ex. a unique event is specific to a specific calendar instance, and should not exist on other calendar instances.

Static Method doesn't care about any particular instance. For ex, a weekend on a calendar applies to all calendars since it is universally agreed upon how calendars work, regardless of instance

Also look into Class Method if need to consider inheritance.

```
class Calendar:
    def __init__(self):
        self.events = []

    def add_event(self, event):
        self.events.append(event)

    @staticmethod
    def is_weekend(dt):
        pass
```

Dataclasses

Without using Dataclass,

- If you print, it will be the memory address where the object resides.
 - To print a string, will have to modify the `__str__` dunder method
- A new object with identical information will be a new object

```
class Person:
    def __init__(self, name, job, age):
        self.name = name
        self.job = job
        self.age = age
```

```
person1 = Person("Geralt", "Witcher", 30)
person2 = Person("Yennefer", "Sorceress", 25)
person3 = Person("Yennefer", "Sorceress", 25)
```

```
print(id(person2))
print(id(person3))
print(person1)
```

```
print(person3 == person2)
```

```
"""
```

OUTPUT:

```
140244722433808
```

```
140244722433712
```

```
<__main__.Person object at 0x7f8d44dcbfd0>
```

```
False
```

```
"""
```

With Dataclass:

- Print string information instead of object, unless using id()
- A new object will still be a new object, but if information is identical it will return True
- If forgot to write @dataclass...
 - class variables instead of instance variables

```
from dataclasses import dataclass, field
```

```
@dataclass(order=True,frozen=False)
```

```
class Person:
```

```
    sort_index: int = field(init=False, repr=False)
```

```
    name: str
```

```
    job: str
```

```
    age: int
```

```
    strength: int = 100
```

```
    def __post_init__(self):
```

```
        object.__setattr__(self, 'sort_index', self.strength)
```

```
def __str__(self):  
    return f'{self.name}, {self.job} ({self.age})'
```

```
person1 = Person("Geralt", "Witcher", 30, 99)  
person2 = Person("Yennefer", "Sorceress", 25)  
person3 = Person("Yennefer", "Sorceress", 25)
```

```
print(person1)  
print(id(person2))  
print(id(person3))  
print(person3 == person2)  
print(person1 > person2)
```

```
""""  
Geralt, Witcher (30)  
140120080908048  
140120080907856  
True  
False  
""""
```

<https://www.youtube.com/watch?v=CvQ7e6yUtnw&t=389s>

Revision #4

Created 7 March 2024 00:49:37 by victor

Updated 7 March 2024 04:35:40 by victor