String Representation of Objects

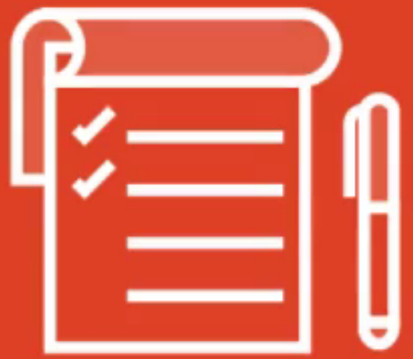
Robert Smallshire
COFOUNDER - SIXTY NORTH

@robsmallshire

Austin Bingham
COFOUNDER - SIXTY NORTH

@austin_bingham

Overview



Three built-in functions:

- `r = repr(obj)`
- `s = str(obj)`
- `f = format(obj)`

Customization gives:

- maintainability
- debuggability
- usability

position 〉 position.py

position.py

```python
class Position:

    def __init__(self, latitude, longitude):
        if not (-90 <= latitude <= +90):
            raise ValueError(f"Latitude {latitude} out of range")

        if not (-180 <= longitude <= +180):
            raise ValueError(f"Longitude {longitude} out of range")

        self._latitude = latitude
```

Position

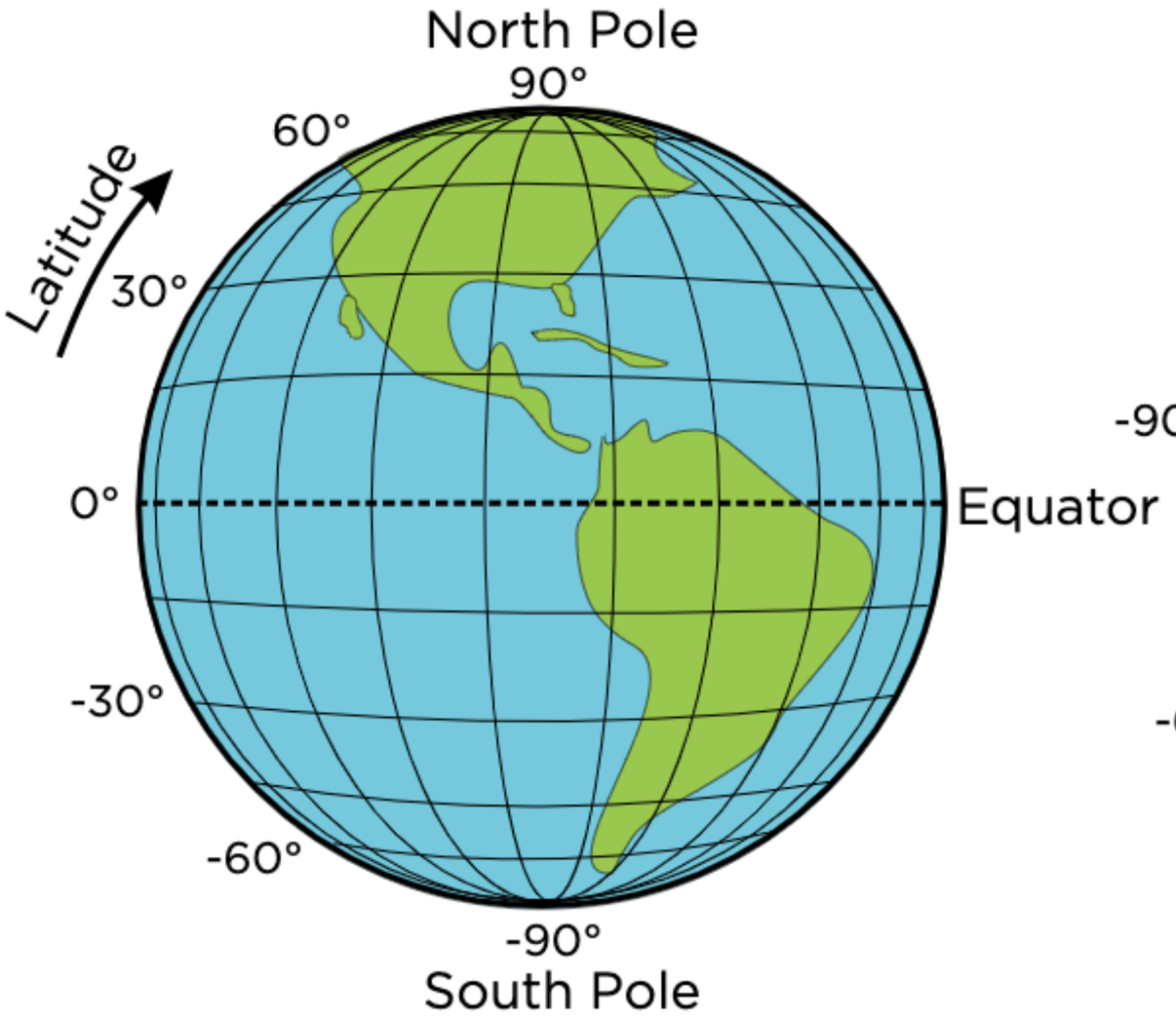
Python Console

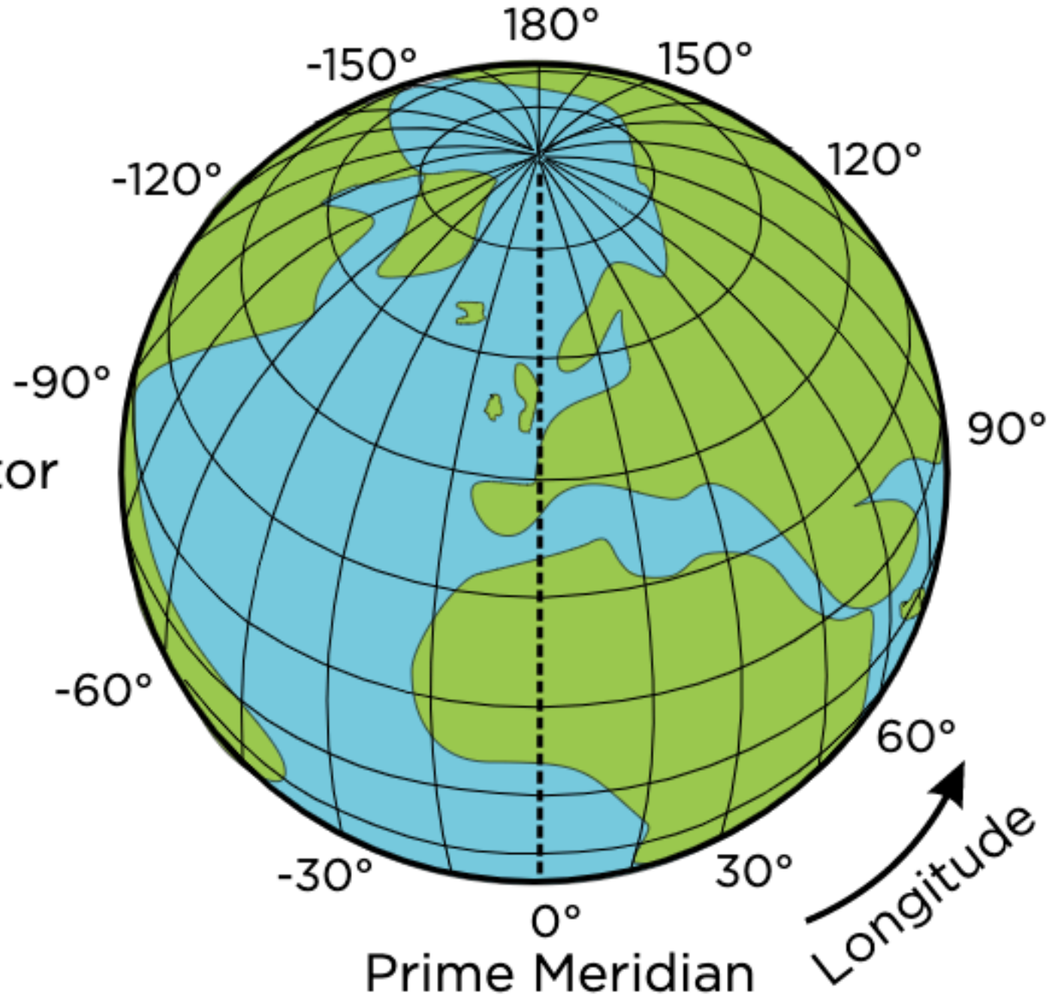
```
'<position.Position object at 0x104a9efa0>'
>>> str(oslo)
'<position.Position object at 0x104a9efa0>'
>>> format(oslo)
'<position.Position object at 0x104a9efa0>'
>>> dir(object)
['__class__', '__delattr__', '__dir__', '__doc__', '__eq__', '__format__', '__ge__', '__getattribute__', '__gt__',
 '__hash__', '__init__', '__init_subclass__', '__le__', '__lt__', '__ne__', '__new__', '__reduce__', '__reduce_ex__',
 '__repr__', '__setattr__', '__sizeof__', '__str__', '__subclasshook__']

>>>
```

Replay server listening on port 14415: You just opened position (4 minutes ago)

1:1 UTF-8 4 spaces Python 3.8 (position)

Latitude and Longitude





Customizing `repr()`

position 〉 position.py

position.py

```python
    def __repr__(self):
        return f"{typename(self)}(latitude={self.latitude}, longitude={self.longitude})"


class EarthPosition(Position):
    pass


class MarsPosition(Position):
    pass
```

Python Console

```
>>> mauna_kea
EarthPosition(latitude=19.82, longitude=-155.47)
>>> olympus_mons = MarsPosition(18.65, -133.8)
>>> olympus_mons
MarsPosition(latitude=18.65, longitude=-133.8)
>>> str(olympus_mons)
'MarsPosition(latitude=18.65, longitude=-133.8)'
>>> format(olympus_mons)
'MarsPosition(latitude=18.65, longitude=-133.8)'

>>>
```

Replay server listening on port 14415: You just opened position (9 minutes ago)

1:1 LF UTF-8 4 spaces Python 3.8 (position)

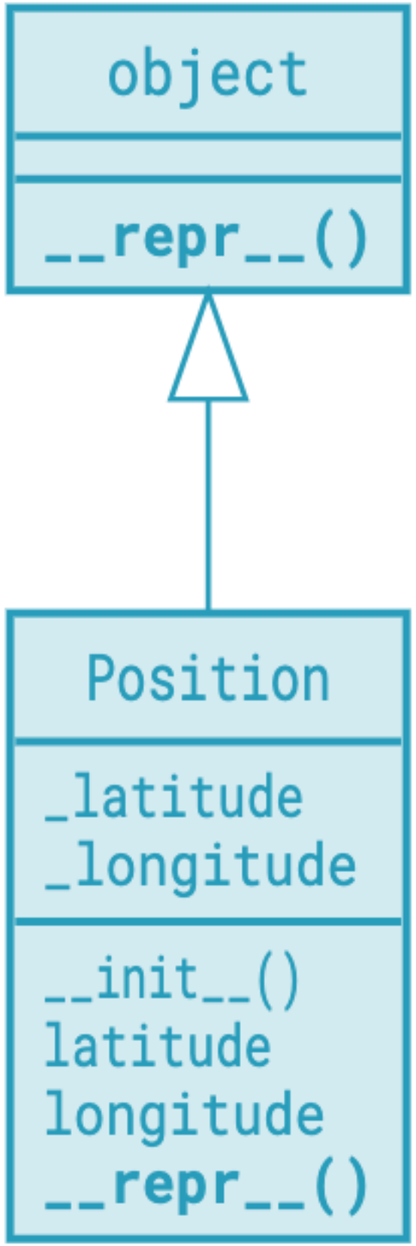
Consider the Target Audience

For whom is `repr()` intended?

Conventions for Good `__repr__` Results

Include necessary state, but be prepared to compromise.

Format as constructor invocation source code.



Overriding `__repr__()`

The default `__repr__` inherited from `object` is not much use.

Override `__repr__` to return a more useful string, which ideally formatted as source code for a constructor call.

You should almost always override `__repr__()` in your classes.

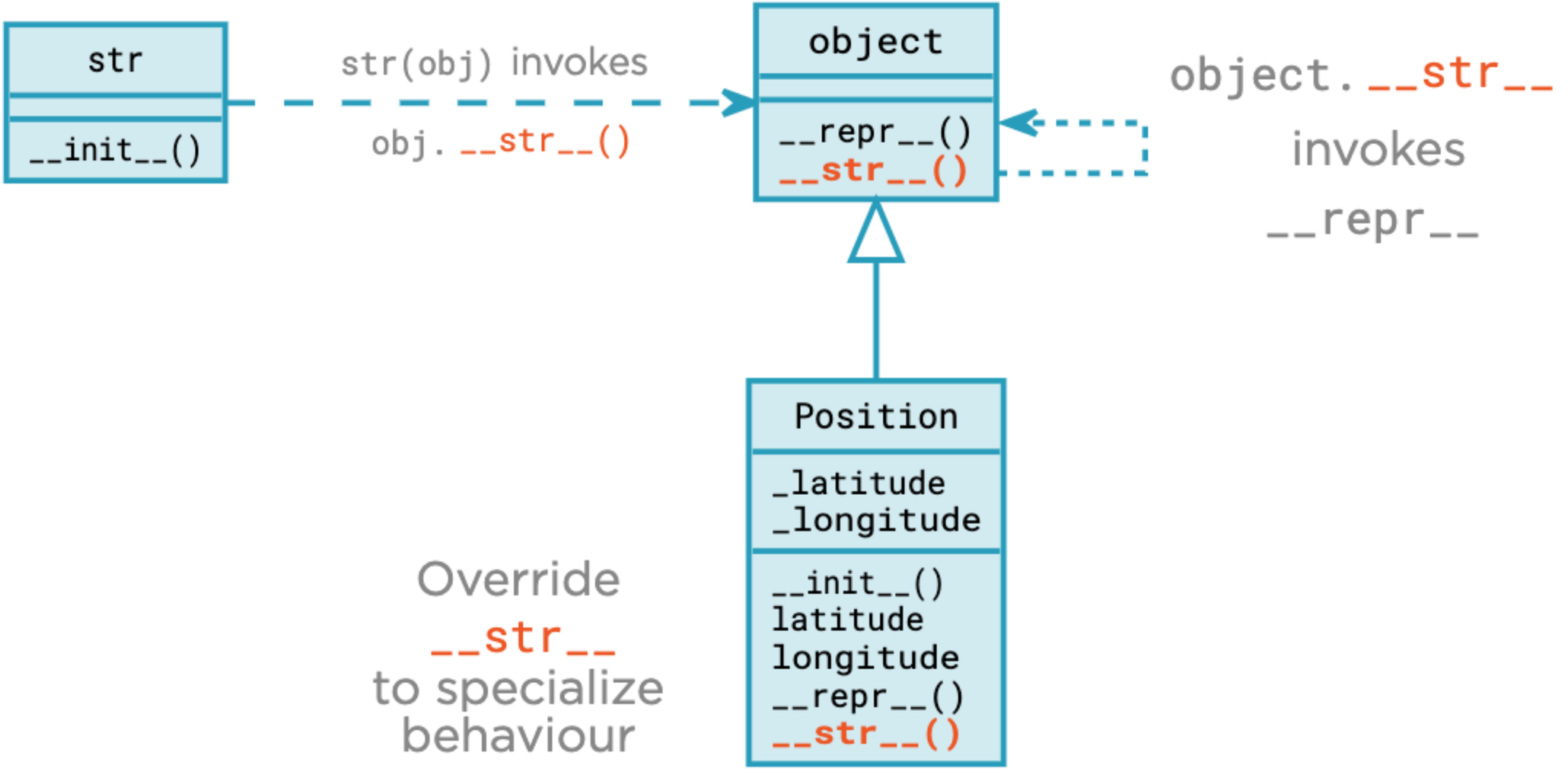
Customizing `str()`

What Is the `str()` Function?

```
str(obj)
```

The string class is callable

String Constructor Delegation



Consider the Target Audience

For whom is `str()` intended?

`str()` Is for System Consumers

Users, people.
In user interfaces.

Other systems.

Think of the User

~~Position(latitude=-34.8, longitude=14.4)~~

Geographic Positions

77.5° S, 167.2° E

Mount Erebus, Antarctica

position 〉 position.py

position.py

```python
        return "E" if self.longitude >= 0 else "W"

    def __repr__(self):
        return f"{typename(self)}(latitude={self.latitude}, longitude={self.longitude})"

    def __str__(self):
        return (
            f"{abs(self.latitude)}° {self.latitude_hemisphere}, "
            f"{abs(self.longitude)}° {self.longitude_hemisphere}"
        )
```

Position 〉 __str__()

Python Console

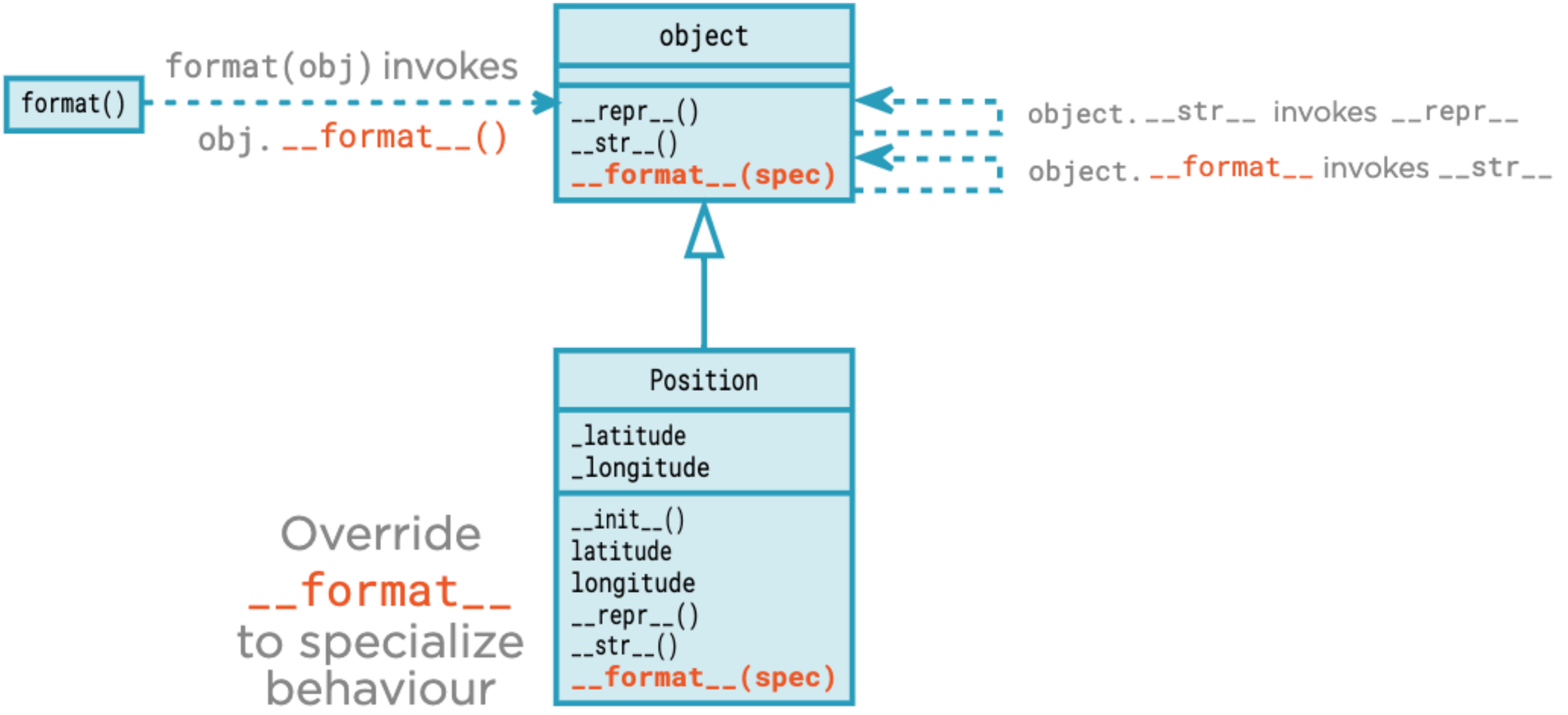
```
'77.5° S, 167.2° E'
>>> print("Mount Erebus is located at", mount_erebus)
Mount Erebus is located at 77.5° S, 167.2° E
>>> repr(mount_erebus)
'EarthPosition(latitude=-77.5, longitude=167.2)'
>>> str(mount_erebus)
'77.5° S, 167.2° E'
>>> format(mount_erebus)
'77.5° S, 167.2° E'

>>>
```

Replay server listening on port 14415: You just opened position (3 minutes ago)

1:1 UTF-8 4 spaces Python 3.8 (position)

Customizing `format()`

format() Function Delegation



position / position.py

position.py

```python
        return (
            f"{abs(self.latitude)}° {self.latitude_hemisphere}, "
            f"{abs(self.longitude)}° {self.longitude_hemisphere}"
        )

    def __format__(self, format_spec):
        return "FORMATTED POSITION"

class EarthPosition(Position):
    pass
```

Python Console

```
'EarthPosition(latitude=-32.7, longitude=-70.1)'
>>> str(aconcagua)
'32.7° S, 70.1° W'
>>> format(aconcagua)
'FORMATTED POSITION'
>>> f"The highest mountain in South America is located at {aconcagua}"
'The highest mountain in South America is located at FORMATTED POSITION'
>>> "The highest mountain in South America is located at {}".format(aconcagua)
'The highest mountain in South America is located at FORMATTED POSITION'

>>>
```

Replay server listening on port 14415: You just opened position (2 minutes ago)

1:1 UTF-8 4 spaces Python 3.8 (position)

Floating-point Format Specifications

```
'      +0.00007748091'
>>> format("Format this!", ">+20.11f")
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
ValueError: Unknown format code 'f' for object of type 'str'
>>> f"The conductance quantum is {q}"
'The conductance quantum is 7.748091e-05'
>>> f"The conductance quantum is {q:.6f}"
'The conductance quantum is 0.000077'
>>> f"The conductance quantum is {q:.2e}"
'The conductance quantum is 7.75e-05'
>>>
```

position 〉 position.py

position.py

```python
    def latitude_hemisphere(self):
        return "N" if self.latitude >= 0 else "S"

    @property
    def longitude_hemisphere(self):
        return "E" if self.longitude >= 0 else "W"

    def __repr__(self):
        return f"{typename(self)}(latitude={self.latitude}, longitude={self.longitude})"

    def __str__(self):
        return format(self)

    def __format__(self, format_spec):
        component_format_spec = ".2f"
        prefix, dot, suffix = format_spec.partition(".")
        if dot:
            num_decimal_places = int(suffix)
            component_format_spec = f".{num_decimal_places}f"
        latitude = format(abs(self.latitude), component_format_spec)
        longitude = format(abs(self.longitude), component_format_spec)
        return (
```

Position

Replay server listening on port 14415: You just opened position (5 minutes ago)

34:1 LF UTF-8 4 spaces Python 3.8 (position)

position 〉 position.py

position.py

```python
    def __format__(self, format_spec):
        component_format_spec = ".2f"
        prefix, dot, suffix = format_spec.partition(".")
        if dot:
            num_decimal_places = int(suffix)
            component_format_spec = f".{num_decimal_places}f"
        latitude = format(abs(self.latitude), component_format_spec)
        longitude = format(abs(self.longitude), component_format_spec)
        return (
            f"{latitude}° {self.latitude_hemisphere}, "
            f"{longitude}° {self.longitude_hemisphere}"
        )

```

Position

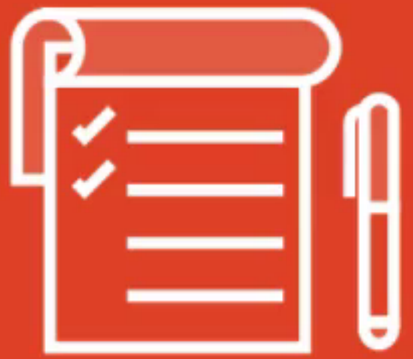
Python Console

```
'The everest object is EarthPosition(latitude=27.988056, longitude=86.925278)'
>>> f"The everest object is {everest!s}"
'The everest object is 27.99° N, 86.93° E'
>>> f"{everest=}"
'everest=EarthPosition(latitude=27.988056, longitude=86.925278)'

>>>
```

Replay server listening on port 14415: You just opened position (2 minutes ago)

1:1 UTF-8 4 spaces Python 3.8 (position)

Summary

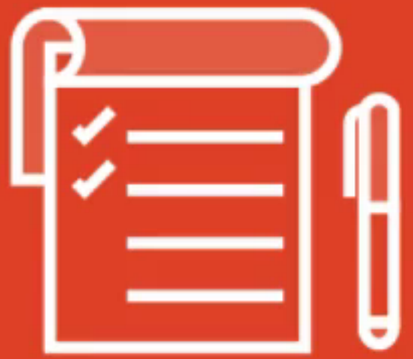


`repr()` gives a string for developers

`str()` – the string constructor – gives a string for users

`format()` gives more control

Summary



`repr(obj)` delegates to `obj.__repr__()`

`str(obj)` delegates to `obj.__str__()`

`format(obj, spec)` delegates to `obj.__format__(spec)`

Summary

All classes inherit default `__repr__()`, `__str__()` and `__format__()`

The default `__repr__()` isn't very helpful

Most classes should override `__repr__()`