The Set Protocol

Robert Smallshire
COFOUNDER - SIXTY NORTH

@robsmallshire

Austin Bingham
COFOUNDER - SIXTY NORTH

@austin_bingham

Abstract Base Class

collections.abc.Set

Inherits from

```
Collection(
  Sized
  Iterable
  Container)
```

Abstract methods

```
__contains__
__iter__
__len__
```

Mixin methods

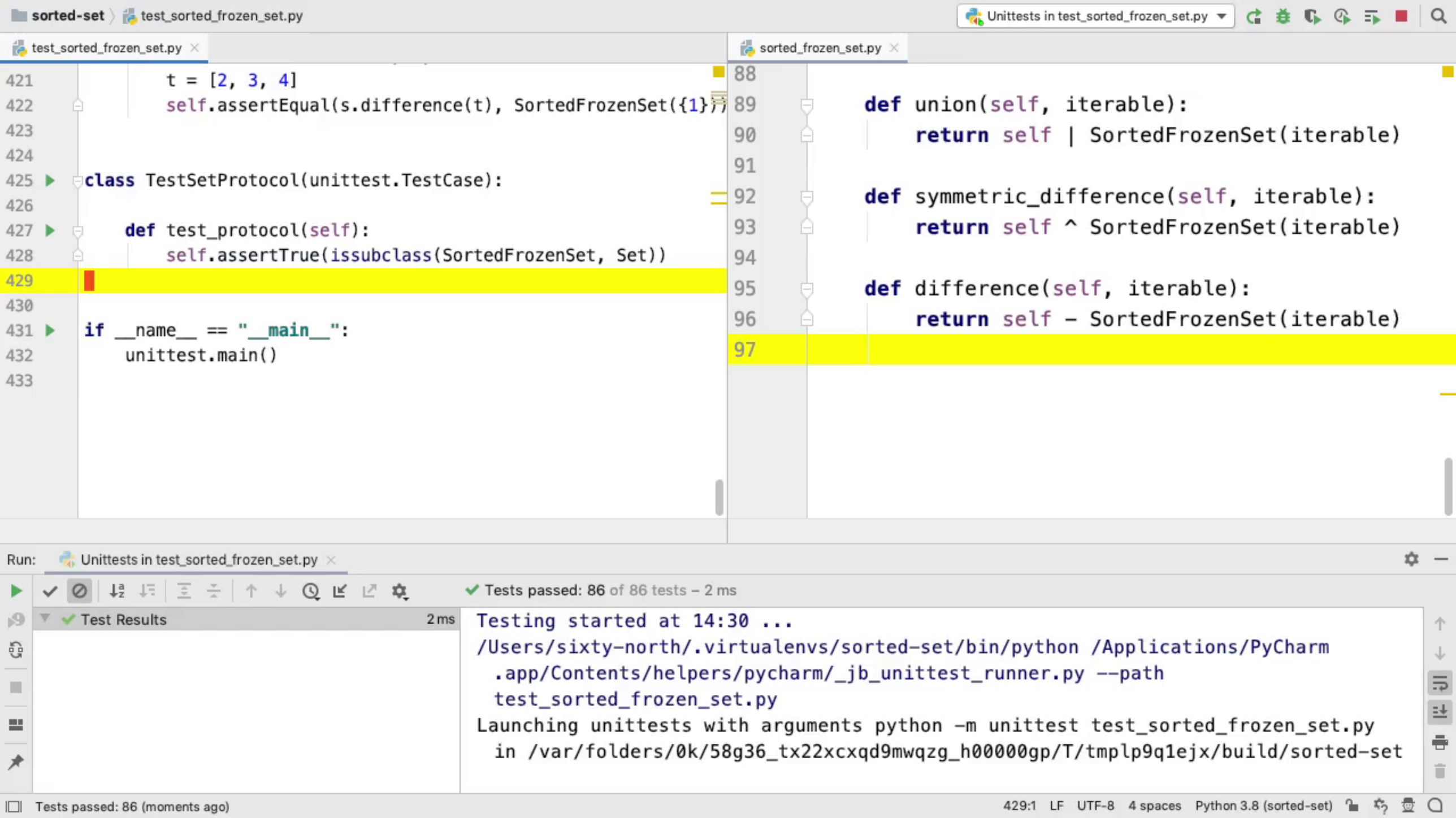
```
__le__    __lt__
__eq__    __ne__
__gt__    __ge__
__and__   __or__
__sub__   __xor__
    isdisjoint
```

Set Protocol: Relational Predicates

special method	infix operator	named method	meaning
`__le__`	`<=`	`issubset`	subset
`__lt__`	`<`		proper subset
`__eq__`	`==`		equal
`__ne__`	`!=`		not equal
`__gt__`	`>`		proper superset
`__ge__`	`>=`	`issuperset`	superset
		`isdisjoint`	disjoint

Provided by `collections.abc.Set`

Provided by built-in `set`


sorted-set
test_sorted_frozen_set.py
Unittests in test_sorted_frozen_set.py
test_sorted_frozen_set.py
sorted_frozen_set.py
t = [2, 3, 4]
self.assertEqual(s.difference(t), SortedFrozenSet({1}))
class TestSetProtocol(unittest.TestCase):
def test_protocol(self):
self.assertTrue(issubclass(SortedFrozenSet, Set))
if __name__ == "__main__":
unittest.main()
def union(self, iterable):
return self | SortedFrozenSet(iterable)
def symmetric_difference(self, iterable):
return self ^ SortedFrozenSet(iterable)
def difference(self, iterable):
return self - SortedFrozenSet(iterable)
Run:
Unittests in test_sorted_frozen_set.py
Tests passed: 86 of 86 tests – 2 ms
Test Results
2 ms
Testing started at 14:30 ...
/Users/sixty-north/.virtualenvs/sorted-set/bin/python /Applications/PyCharm .app/Contents/helpers/pycharm/_jb_unittest_runner.py --path test_sorted_frozen_set.py
Launching unittests with arguments python -m unittest test_sorted_frozen_set.py in /var/folders/0k/58g36_tx22xcxqd9mwqzg_h00000gp/T/tmplp9q1ejx/build/sorted-set
Tests passed: 86 (moments ago)
429:1
LF
UTF-8
4 spaces
Python 3.8 (sorted-set)

Set Algebra Operations

special method	infix operator	named method
`__and__`	`&`	`intersection`
`__or__`	`|`	`union`
`__xor__`	`^`	`symmetric_difference`
`__sub__`	`-`	`difference`

Provided by `collections.abc.Set`

Provided by built-in `set`

Going Further on Your Own - Mutability

`SortedFrozenSet` – as its name suggests – is immutable

Immutability is sufficient and preferred

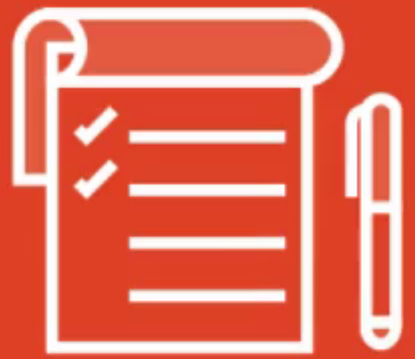
For a mutable version inherit `MutableSet`

Only two overrides needed: `add()` and `discard()`

Optionally include other `set` methods, like `update()` or `copy()`

Beware of lurking traps!

Summary



Collection protocols are at the heart of Python

`Container` for membership tests

`Sized` for length determination

`Iterable` and `Iterator` for collection traversal

`Sequence` for random-access by index

`Set` for distinct elements

Unified into `SortedFrozenSet` through Test-Driven Development

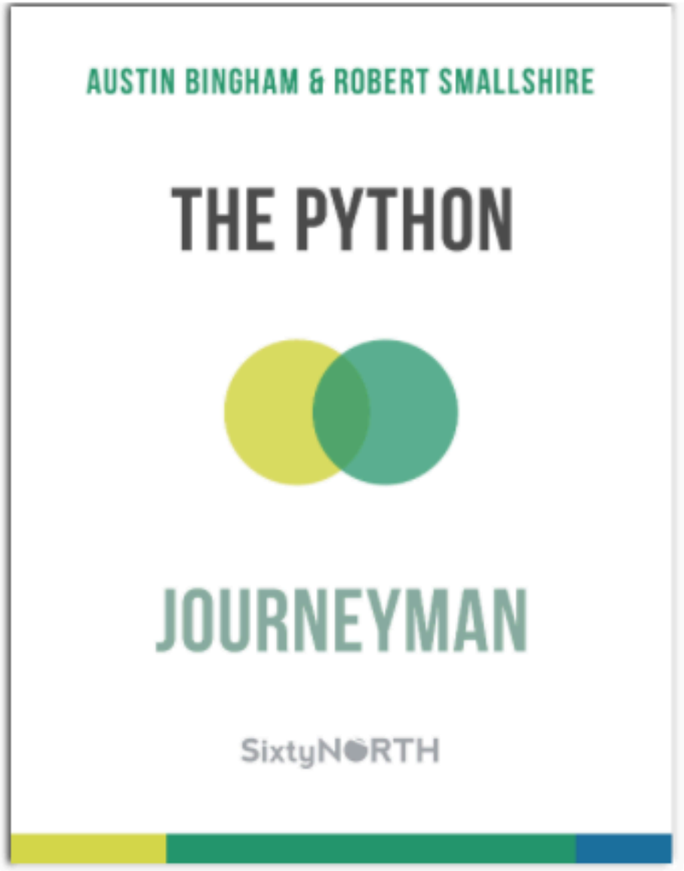
Well done!

Core Python

on

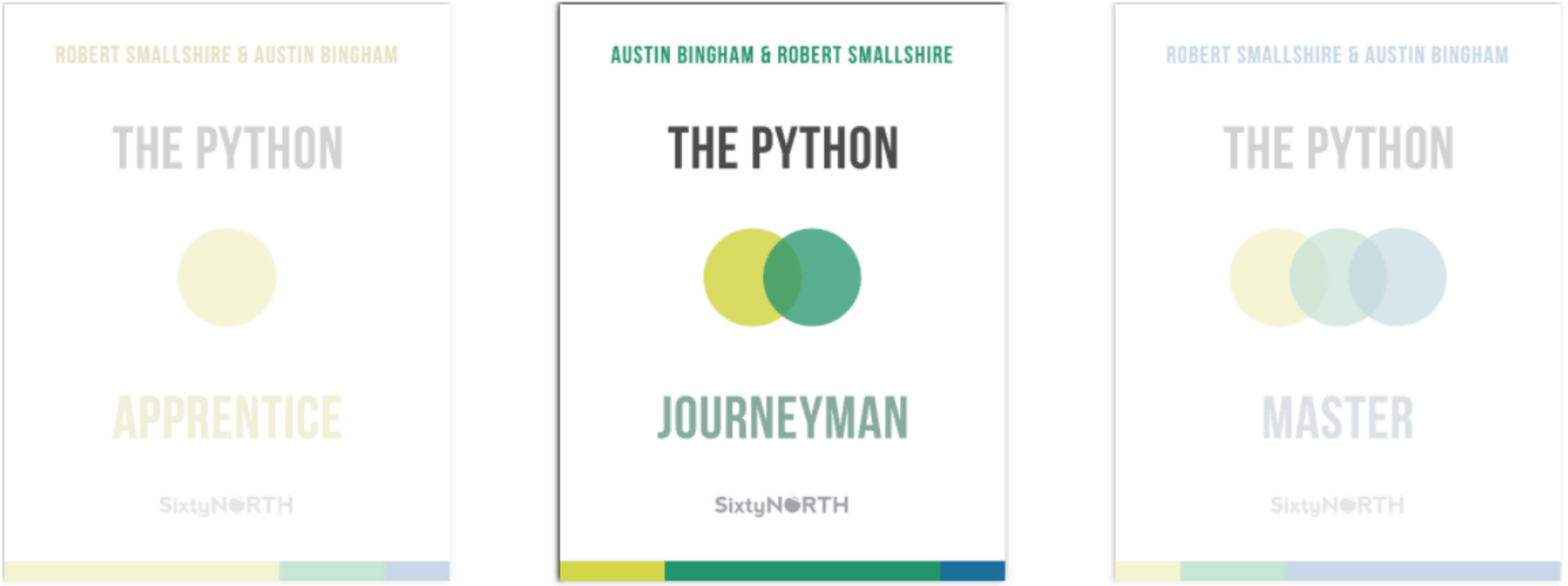
PLURALSIGHT

The Python Craftsman



leanpub.com/b/python-craftsman

The Python Journeyman



leanpub.com/python-journeyman

Happy Programming!