

Immutable Data Structures

Moshe Zadka

Pyninsula November 2017

Are Squares Rectangles?

And what does that have to do with anything?

What is a Rectangle?

```
class IRectangle(Interface):  
    def get_length(self):  
        """Squares can do that"""  
    def get_width(self):  
        """Squares can do that"""  
    def set_dimensions(self, length, width):  
        """Uh oh"""
```

What is a Rectangle? (V2)

```
class IRectangle(Interface):  
    def get_length(self):  
        """Squares can do that"""  
    def get_width(self):  
        """Squares can do that"""  
    def with_dimensions(self, length, width):  
        """Returns a new rectangle"""
```

What is an array?

```
class IArrayOfThing(Interface):  
    def get_element(self, i):  
        """ Returns Thing """  
    def set_element(self, i, thing):  
        """ 'thing' can be any Thing """
```

What is an array?

```
class IArrayOfThing(Interface):  
    def get_element(self, i):  
        """ Returns Thing """  
    def set_element(self, i, thing):  
        """ 'thing' can be any Thing """  
  
class IArrayOfSuperthing(Interface):  
    def get_element(self, i):  
        """ Returns Superthing """  
    def set_element(self, i, superthing):  
        """ 'superthing' can be any Superthing """
```

What is an array?

```
class IArrayOfThing(Interface):  
    def get_element(self, i):  
        """ Returns Thing """  
    def set_element(self, i, thing):  
        """ 'thing' can be any Thing """  
  
class IArrayOfSuperthing(Interface):  
    def get_element(self, i):  
        """ Returns Superthing """  
    def set_element(self, i, superthing):  
        """ 'superthing' can be any Superthing """
```

You cannot implement both IArrayOfThing and IArrayOfSuperthing

Global Mutable State

- ▶ Evil
- ▶ More likely than you think

Immutability helps!

```
@attr.s(frozen=True)
class Rectangle(object):
    length = attr.ib()
    width = attr.ib()
    @classmethod
    def with_dimensions(cls, length, width):
        return cls(length, width)
```

Immutability helps!

```
@attr.s(frozen=True)
class Rectangle(object):
    length = attr.ib()
    width = attr.ib()
    @classmethod
    def with_dimensions(cls, length, width):
        return cls(length, width)
```

```
@attr.s(frozen=True)
class Square(object):
    side = attr.ib()
    @classmethod
    def with_dimensions(cls, length, width):
        return Rectangle(length, width)
```

Easy Modification

```
too_long = Rectangle(100, 4)
reasonable = attr.evolve(too_long, length=10)
```

Pyrsistent

```
# Vector of integers  
a = pyrsistent.v(1, 2, 3)  
# Not a vector of integers  
b = a.set(1, "hello")
```

Persistent performance

- ▶ $O(1)$ time
- ▶ $O(1)$ space

Transformers!

```
blog = pyrsistent.m(  
    title="My_blog",  
    links=pyrsistent.v(" github", " twitter"),  
    posts=pyrsistent.v(  
        pyrsistent.m(title="no_updates",  
                       content="I'm_busy"),  
        pyrsistent.m(title="still_no_updates",  
                       content="still_busy"))  
blog = blog.transform(["posts", 1, "content"],  
                      "pretty_busy")
```

Transformers!

```
{ 'links': [ 'github', 'twitter' ],  
  'posts': [ { 'content': "I'm busy",  
                'title': 'no updates' },  
              { 'content': 'pretty busy',  
                'title': 'still no updates' } ],  
  'title': 'My blog' }
```

This is safe

```
def silly_sum(a, b, extra=v(1, 2)):
    extra.extend([a, b])
    return sum(extra)
```