Summing Up

Gerald Britton
IT SOLUTIONS DESIGNER

@GeraldBritton www.linkedin.com/in/geraldbritton

Façade Pattern

```
class GetEmployeesFacade(AbsFacade):
    def get_employees(self):
        connection = pyodbc.connect(CONNSTR)
        cursor = connection.cursor()
        cursor.execute(QUERY)
        for row in cursor:
            print(row.FirstName, row.LastName)
        connection.commit()
        connection.close()
```

Adapter

```
class VendAdapter(AbsAdapter):
    @property
    def name(self):
        return self.adaptee.name

    @property
    def address(self):
        return '{} {}'.format(
            self.adaptee.number,
            self.adaptee.street
        )
```

Decorator

```
class Inline4Cyl(AbsDecorator):
    @property
    def description(self):
        return self.car.description + ', inline 4 cylinder'

    @property
    def cost(self):
        return self.car.cost + 500.00
```

Template

```
class Airplane(AbsTransport):

    def start_engine(self):
        print('Starting the Rolls-Royce gas-turbine engines')

    def leave_terminal(self):
        print('Leaving terminal')
        print('Taxiing to runway')

    def travel_to_destination(self):
        print('Flying...')

    def entertainment(self):
        print('Playing in-flight movie')

    def arrive_at_destination(self):
        print('Landing at ' + self._destination)
```

Iterator

```
class Employees(Iterable):
    _employees = {}
    _headcount = 0

    def add_employee(self, employee):
        self._headcount += 1
        self._employees[self._headcount] = employee

    def __iter__(self):
        return (e for e in self._employees.values())
```

Composite

```
class Tree(Iterable, AbsComposite):

    def __init__(self, members):
        self.members = members

    def __iter__(self):
        return iter(self.members)

    def get_oldest(self):
        def f(t1, t2):
            t1_, t2_ = t1.get_oldest(), t2.get_oldest()
            return t1_ if t1_.birthdate < t2_.birthdate else t2_
        return reduce(f, self, NullPerson())
```

State

```
class ShoppingCart:
    def __init__(self):
        self.empty = Empty(self)
        self.not_empty = NotEmpty(self)
        self.check_out = AtCheckOut(self)
        self.paid_for = PaidFor(self)

        self.items = 0
        self.state = self.empty
```

Proxy

```
class Proxy(AbsEmployees):

    def __init__(self, employees, reqid):
        self._employees = employees
        self._reqid = reqid

    def get_employee_info(self, empids):
        reqid = self._reqid
        acc = AccessControls.get_access_control()

        for e in self._employees.get_employee_info(empids):

            if e.empid == reqid or \
                (reqid in acc and acc[reqid].can_see_personal):
                yield e
```

Summing up

Gerald Britton
IT SOLUTIONS DESIGNER

@GeraldBritton www.linkedin.com/in/geraldbritton