Single-Responsibility Principle

Single-Responsibility Principle

A principle in two parts:

1. Each class covers one piece of functionality. And
2. Everything about that piece is encapsulated inside the class.

An important organizational principle.

Model & View

This `StockModel` represents a data abstraction:

```
class StockModel:
    def __init__(self, symbol, open_price, close_price,
                 volume, average_volume):
        self.symbol = symbol
        self.open_price = open_price
        self.close_price = close_price
        self.volume = volume
        self.average_volume = average_volume

    def is_bullish(self):
        price_ratio = self.close_price / self.open_price
        volume_ratio = self.volume / self.average_volume
        return price_ratio > 1.02 and volume_ratio > 1.1
```

Model & View

This `StockView` encapsulates how to render it for the user:

```
class StockView:
    def params(self, model):
        if model.is_bullish():
            sentiment = 'Bullish'
        else:
            sentiment = 'Bearish'
        return {
            'name': model.symbol,
            'price': model.close_price,
            'sentiment': sentiment,
            }

    def render(self, model):
        params = self.params(model)
        return '{name}: ${price:0.2f} ({sentiment})'.format_map(params)
```

Demo

This default view renders it as brief text.

```
>>> model = StockModel('AAPL', 159.29, 163.05, 44035531, 22509937)
>>> view = StockView()
>>> view.render(model)
'AAPL: $163.05 (Bullish)'
```

params() and render()

This `StockView` encapsulates how to render it for the user:

```
class StockView:
    def params(self, model):
        if model.is_bullish():
            sentiment = 'Bullish'
        else:
            sentiment = 'Bearish'
        return {
            'name': model.symbol,
            'price': model.close_price,
            'sentiment': sentiment,
            }

    def render(self, model):
        params = self.params(model)
        return '{name}: ${price:0.2f} ({sentiment})'.format_map(params)
```