FJE进阶

> 21307259 彭璇

对已有FJE实现进行设计重构，改用迭代器+访问者模式，或者迭代器 +策略模式（根据所用语言和技术栈，恰当选定）

📢 目录

设计重构

根据上一个实验的语言与技术，决定使用迭代器 +策略模式进行重构

- 迭代器模式提供一种方法来顺序访问一个聚合对象中的各个元素，而不暴露其内部的表示，主要用于集合类的遍历。
- 策略模式定义一系列算法，把它们一个个封装起来，并且使它们可互相替换，这样算法可独立于使用它的客户而变化。

将这两种模式结合起来，可以让遍历（迭代）和具体操作（策略）相分离。适合用于遍历复杂数据结构（如JSON）并对每个元素执行不同的操作（如不同的可视化方式），所以选择改模式进行进阶。

设计解析

使用迭代器模式来遍历JSON数据，使用策略模式来定义不同的可视化方法。下面是实现这种组合的步骤：

1. **JSONIterator**：用于遍历JSON数据。
2. **VisualizationStrategy**：定义各种可视化策略。
3. **Visualizer**：接受一个JSON数据和一个可视化策略，使用迭代器遍历数据，并应用策略进行可视化。

`JSONIterator`实现迭代器，用于深度优先遍历嵌套的JSON数据结构（包括字典和列表）。使用一个栈来跟踪当前遍历的位置。每次调用`__next__`方法时，迭代器返回当前键、值及其所在的层级，直到遍历完所有元素为止：

```
class JSONIterator:
    def __init__(self, data):
        self.data = data
        self.stack = [(None, data, iter(data.items()))] if
isinstance(data, dict) else [(None, data, iter(enumerate(da
ta)))]

    def __iter__(self):
        return self

    def __next__(self):
        while self.stack:
            key, current_data, current_iter = self.stack[-
1]
            try:
                key, value = next(current_iter)
                if isinstance(value, (dict, list)):
                    self.stack.append((key, value, iter(val
ue.items()) if isinstance(value, dict) else iter(enumerate
(value))))
                return key, value, len(self.stack) - 1
            except StopIteration:
                self.stack.pop()
        raise StopIteration
```

`visualization_strategy`中，定义`VisualizationStrategy`作为抽象基类，定义一个抽象方法`visualize`用于定义可视化策略。`TreeVisualizationStrategy`类和`RectangleVisualizationStrategy`类作为该抽象方法的子类，在内部具体实现了`visualize`方法来以树型和矩形的形式打印字典数据。这两种可视化策略可以根据数据的不同形式来选择使用：

```
from abc import ABC, abstractmethod
from json_iterator import JSONIterator
```

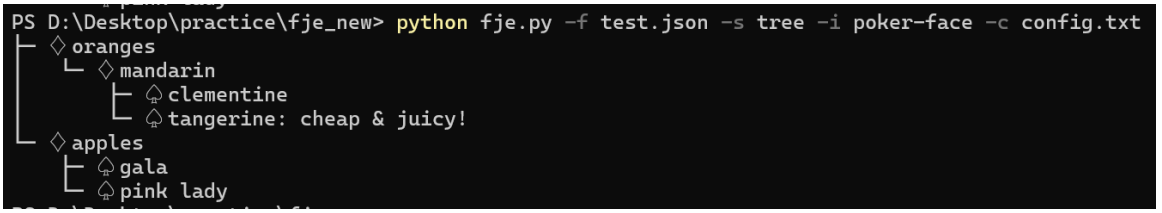
```
class VisualizationStrategy(ABC):
    def __init__(self, icon_factory):
        self.icon_factory = icon_factory

    @abstractmethod
    def visualize(self, data: dict) -> None:
        pass
```

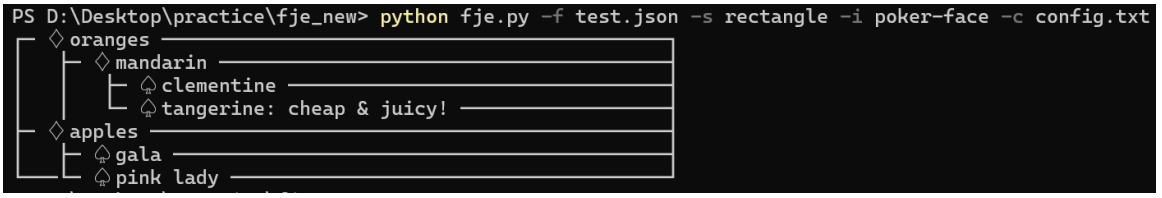
运行截图

使用`python fje.py [-h] -f FILE -s {tree,rectangle} -i ICON [-c CONFIG]`运行。

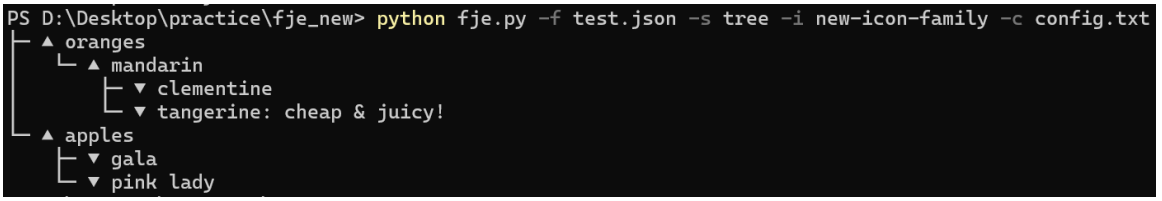
扑克图标+树形结构：



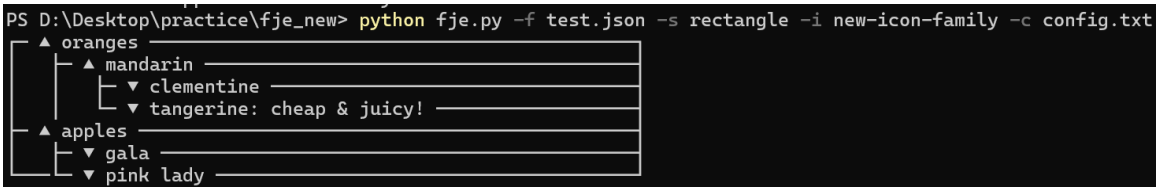
扑克图标+矩形结构：



自定义图标+树形结构：



自定义图标+矩形结构：


PS D:\Desktop\practice\fje_new> python fje.py -f test.json -s rectangle -i new-icon-family -c config.txt
▲ oranges
▲ mandarin
▼ clementine
▼ tangerine: cheap & juicy!
▲ apples
▼ gala
▼ pink lady