

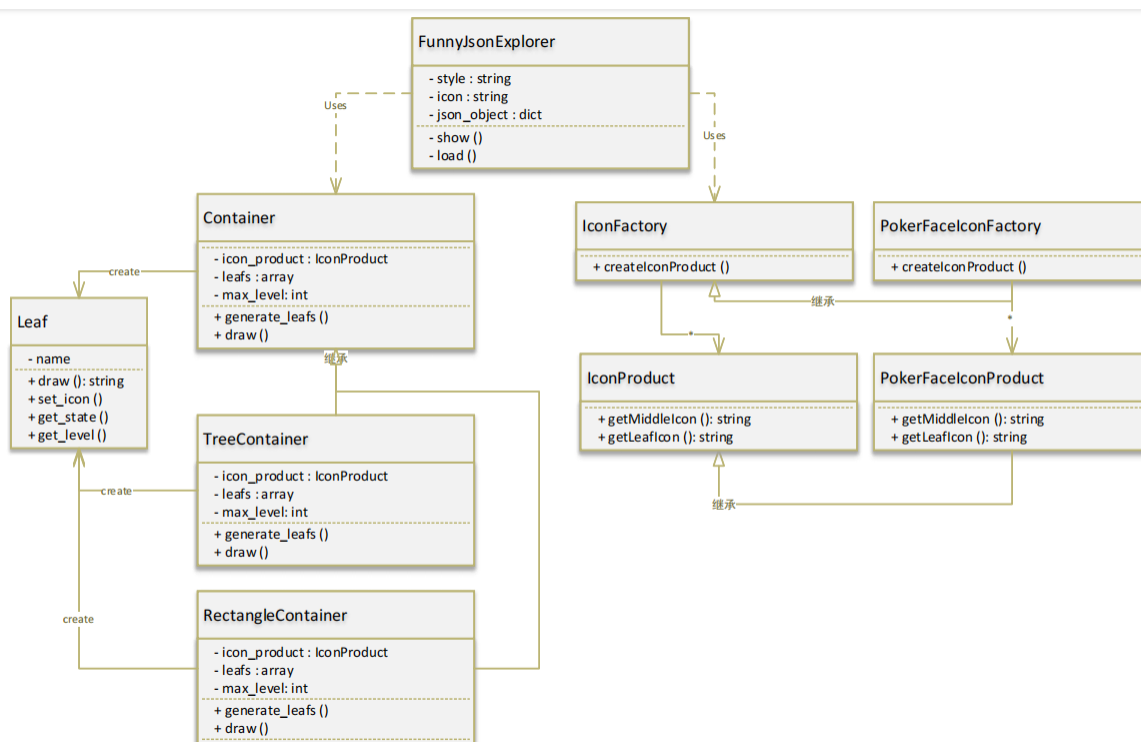
Funny JSON Explorer

设计模式

使用**工厂方法模式**，将对象的创建延迟到工厂的子类中，由子类决定创建哪个具体的对象，使用工厂方法模式时，新增新的产品也就是新增新的风格只需要添加对应的工厂类，不需要修改现有代码，对应于Icon-Family风格的使用，每个新的Icon-Family对应新建的一个工厂类和产品类。

使用**建造者模式**，能够将一个复杂对象的创建过程分解成小步骤，即将对象的创建和表示过程分离，在使用同样的过程时，只需要修改这个过程的小步骤就能构建不同的对象，这里对应Container和Leaf的关系，Container根据不同的表示类型（tree/rectangle）构建不同的类，但继承的类都创建同样的Leaf类对象。

类图



添加新的风格

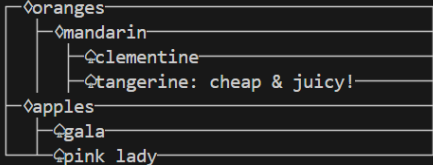
添加style，仅需继承Container类到一个具体类，为其编写draw函数；

添加icon_family仅需继承IconFactory类和IconProduct类成一个新的icon_family类。在FunnyJsonExplorer中加入条件判断调用新的类即可

运行结果

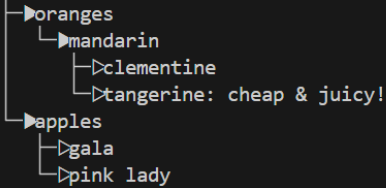
```
PS C:\Users\asus\Desktop\大三下\软件工程\3-design-pattern> python FJE.py -f test.json -s tree -i poker-face
├── oranges
│   ├── mandarin
│   │   ├── clementine
│   │   └── tangerine: cheap & juicy!
└── apples
    ├── gala
    └── pink lady
```

```
PS C:\Users\asus\Desktop\大三下\软件工程\3-design-pattern> python FJE.py -f test.json -s rectangle -i poker-face
```



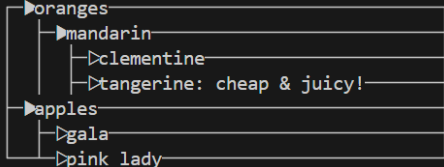
```
graph LR
    oranges --> mandarin
    oranges --> apples
    mandarin --> clementine
    mandarin --> tangerine["tangerine: cheap & juicy!"]
    apples --> gala
    apples --> pink_lady[pink lady]
```

```
PS C:\Users\asus\Desktop\大三下\软件工程\3-design-pattern> python FJE.py -f test.json -s tree -i myicon
```



```
graph LR
    oranges --> mandarin
    oranges --> apples
    mandarin --> clementine
    mandarin --> tangerine["tangerine: cheap & juicy!"]
    apples --> gala
    apples --> pink_lady[pink lady]
```

```
PS C:\Users\asus\Desktop\大三下\软件工程\3-design-pattern> python FJE.py -f test.json -s rectangle -i myicon
```



```
graph LR
    oranges --> mandarin
    oranges --> apples
    mandarin --> clementine
    mandarin --> tangerine["tangerine: cheap & juicy!"]
    apples --> gala
    apples --> pink_lady[pink lady]
```

Github repo URL

[diffwoak/Funny-JSON-Explorer \(github.com\)](https://github.com/diffwoak/Funny-JSON-Explorer)