Funny JSON Explorer

Funny JSON Explorer（**FJE**），是一个JSON文件可视化的命令行界面小工具

```
fje -f <json file> -s <style> -i <icon family>
```

```
{
    oranges: {
        'mandarin': {
            clementine: null,
            tangerine: 'cheap & juicy!'
        }
    },
    apples: {
        'gala': null,
        'pink lady': null
    }
}
```

```
├─ oranges
│  └─ mandarin
│     ├─ clementine
│     └─ tangerine: cheap & juicy!
└─ apples
   ├─ gala
   └─ pink lady
```

FJE可以快速切换**风格**（style），包括：树形（tree）、矩形（rectangle）；

```
├─ oranges
│  └─ mandarin
│     ├─ clementine
│     └─ tangerine: cheap & juicy!
└─ apples
   └─ gala
```

树形（tree）

```
┌─ oranges
│  ├─ mandarin
│  │  ├─ clementine
│  │  ├─ tangerine: cheap & juicy!
│  ├─ apples
│  │  └─┴─ gala
```

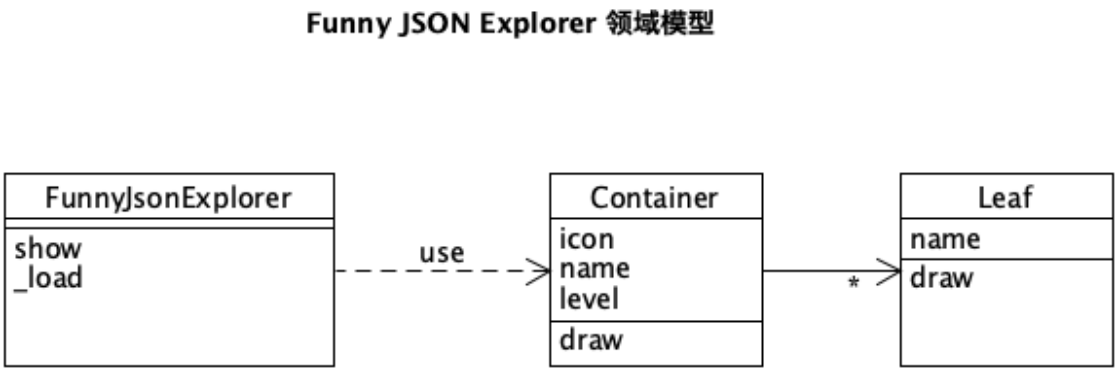
矩形（rectangle）

也可以指定**图标族**（icon family），为中间节点或叶节点指定一套icon

```
├─◊oranges
│  └─◊mandarin
│     ├─♤clementine
│     └─♤tangerine: cheap & juicy!
└─◊apples
   └─♤gala

poker-face-icon-family: 中间节点icon：◊ 叶节点icon：♤
```

领域模型



作业要求

基于上述需求描述和领域模型，按照设计模式要求，进行软件设计，并编码实现（任何语言均可）。

设计模式

使用**工厂方法**（Factory）、**抽象工厂**（Abstract Factory）、**建造者**（Builder）模式、**组合模式**（Composition），完成功能的同时，使得程序易于扩展和维护。

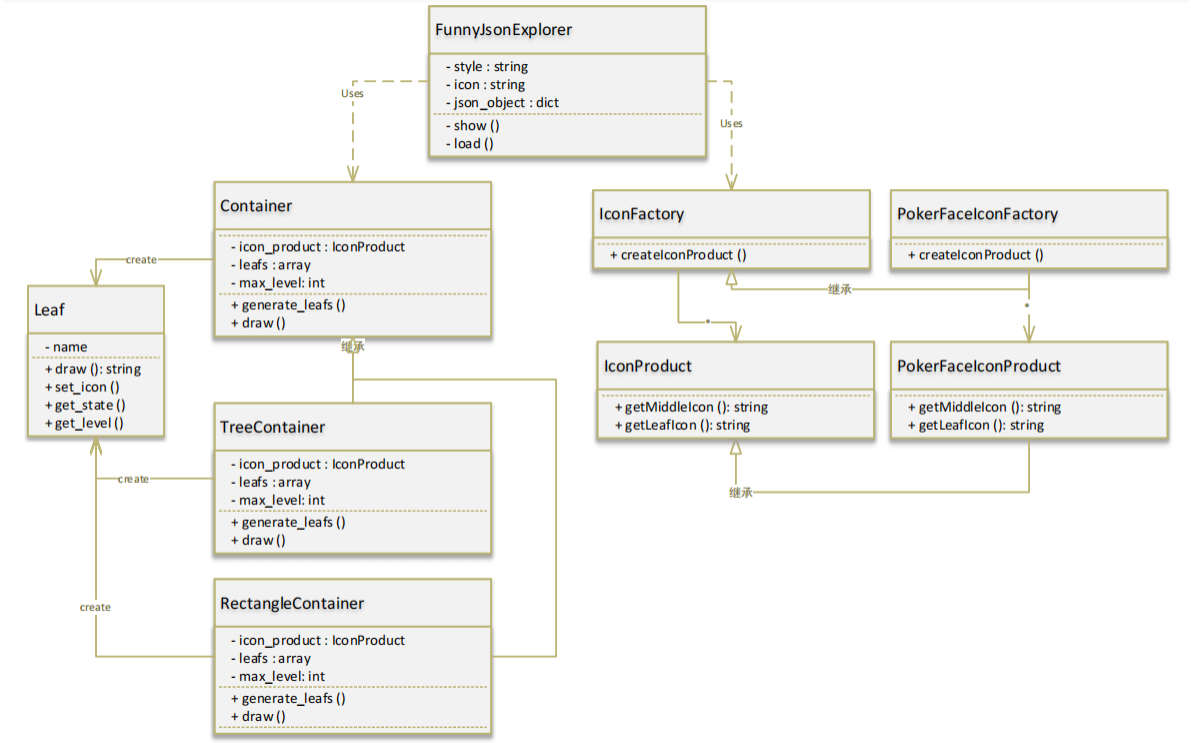
1. （必做）：不改变现有代码，只需添加新的抽象工厂，即可添加新的风格
2. （选做）：通过配置文件，可添加新的图标族

作业提交

1. 设计文档：类图与说明，说明使用的设计模式及作用
2. 运行截图：两种风格，两种图标族，共计4次运行fje的屏幕截图
3. 源代码库：公开可访问的Github repo URL

思路

使用抽象工厂设计模式，能够创建相关或依赖对象的家族，不需要指定具体的类，可派生多个具体工厂类和产品类，一个工厂实例能够创建多个该工厂对应的产品实例，这里对应了领域模型中Container和Leaf的关系。将Container根据不同的style抽象成不同的具体类，再定义另一种IconFactory工厂类，抽象不同icon family类型。



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

```
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
┌─♢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 tree -i myicon

├─▶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 myicon
┌─▶oranges──────────────┐
│  ├─▶mandarin──────────┤
│  │  ├─▷clementine─────┤
│  │  └─▷tangerine: cheap & juicy!┤
└─▶apples─────────────┤
   ├─▷gala────────────┤
   └─▷pink lady───────┘
```

还差：

1. 说明设计模式
2. github推送

Repository search results (github.com)

设计模式之-3种常见的工厂模式简单工厂模式、工厂方法模式和抽象工厂模式，每一种模式的概念、使用场景和优缺点。 工厂方法 抽象工厂 代码-CSDN博客

Java设计模式：工厂模式——图文+代码示例（通俗易懂）-CSDN博客

【深入设计模式】建造者模式—带你彻底弄懂建造者模式 解析真实建造者模式-CSDN博客

【设计模式】组合模式 (简介 | 适用场景 | 优缺点 | 代码示例)-CSDN博客

参考资料

1. unicode 制表符与图标： https://unicode.yunser.com/