Funny JSON Explorer 设计文档

21307362 郑博铿

设计要求

Funny JSON Explorer（**FJE**），是一个JSON文件可视化的命令行界面小工具，FJE可以快速切换**风格**（style），包括：树形（tree）、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。

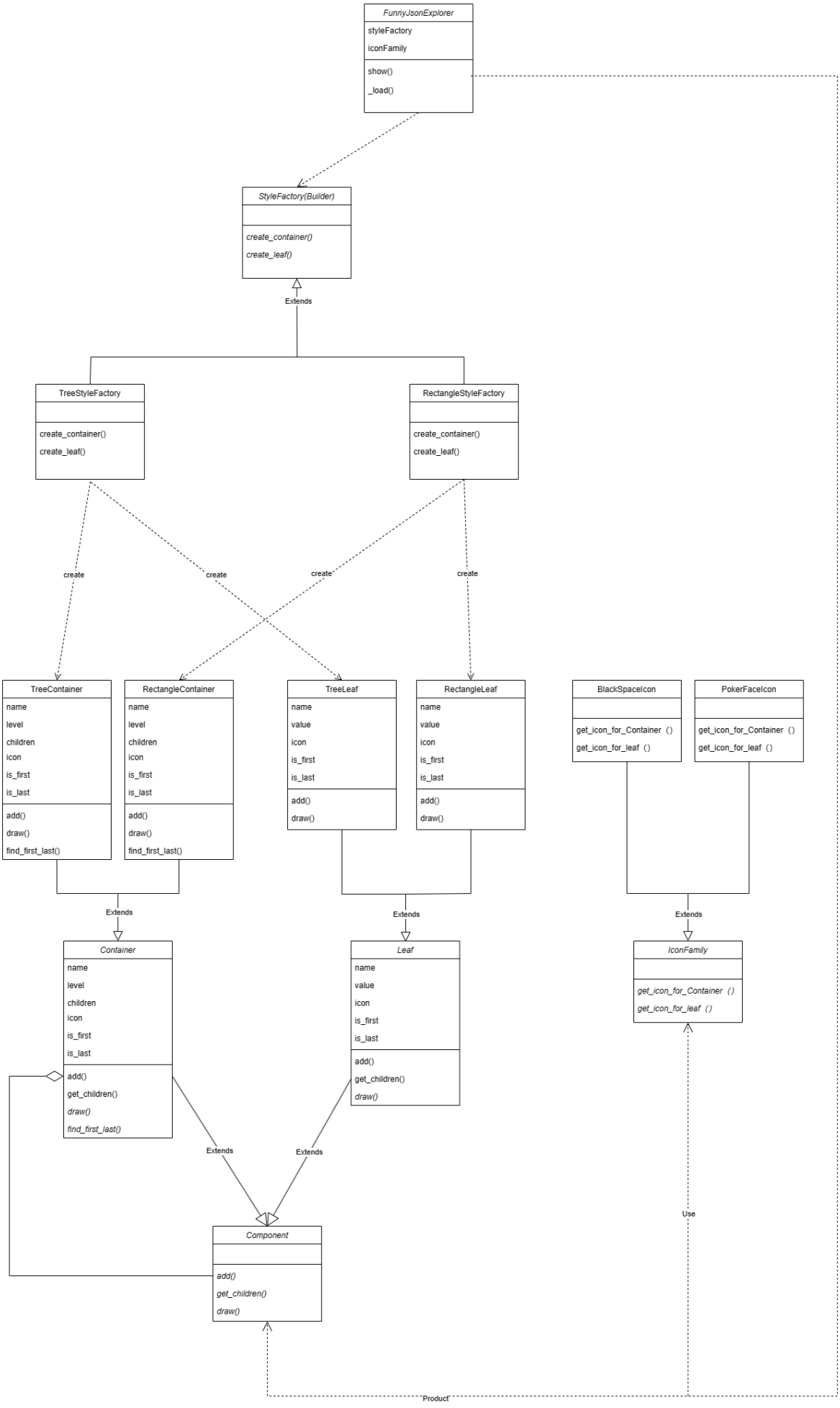
使用**工厂方法**（Factory）、**抽象工厂**（Abstract Factory）、**建造者**（Builder）模式、**组合模式**（Composition），完成功能的同时，使得程序易于扩展和维护。

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

引言

本设计文档描述了一个用于展示JSON数据的系统，该系统实现了多个设计模式，包括工厂方法模式、抽象工厂模式、建造者模式和组合模式。以下内容将详细介绍类图、类说明以及各设计模式的作用和实现。

类图


FunnyJsonExplorer
styleFactory
iconFamily
show()
_load()
StyleFactory(Builder)
create_container()
create_leaf()
Extends
TreeStyleFactory
create_container()
create_leaf()
RectangleStyleFactory
create_container()
create_leaf()
create
create
create
create
TreeContainer
name
level
children
icon
is_first
is_last
add()
draw()
find_first_last()
RectangleContainer
name
level
children
icon
is_first
is_last
add()
draw()
find_first_last()
TreeLeaf
name
value
icon
is_first
is_last
add()
draw()
RectangleLeaf
name
value
icon
is_first
is_last
add()
draw()
BlackSpaceIcon
get_icon_for_Container ()
get_icon_for_leaf ()
PokerFaceIcon
get_icon_for_Container ()
get_icon_for_leaf ()
Extends
Extends
Extends
Container
name
level
children
icon
is_first
is_last
add()
get_children()
draw()
find_first_last()
Leaf
name
value
icon
is_first
is_last
add()
get_children()
draw()
IconFamily
get_icon_for_Container ()
get_icon_for_leaf ()
Extends
Extends
Component
add()
get_children()
draw()
Use
Product

类说明

Component 类

- **描述**：定义组合中所有对象的通用接口，声明了访问和管理子组件的方法。
- **方法**：
  - `add(component)`：添加子组件。
  - `get_children()`：获取子组件列表。
  - `draw(prefix)`：绘制组件。

Container 类

- **继承**：继承自 Component 类。
- **描述**：表示复合节点对象，包含子节点并实现组件接口的方法。
- **属性**：
  - `name`：名称。
  - `level`：层级。
  - `children`：子组件列表。
  - `icon`：图标。
  - `is_first`：是否是第一个子节点。
  - `is_last`：是否是最后一个子节点。
- **方法**：
  - `add(child)`：添加子组件。
  - `get_children()`：获取子组件列表。
  - `draw(prefix)`：绘制组件。
  - `find_first_last(container, is_execute=False)`：查找并设置第一个和最后一个子节点。

具体Container 类

- **TreeContainer** 和 **RectangleContainer**：继承自 Container，具体实现了绘制树形和矩形风格容器的方法。

Leaf 类

- **继承**：继承自 Component 类。
- **描述**：表示叶子节点对象，没有子节点。
- **属性**：
  - `name`：名称。
  - `value`：值。
  - `icon`：图标。
  - `is_first`：是否是第一个节点。
  - `is_last`：是否是最后一个节点。
- **方法**：
  - `draw(prefix)`：绘制叶子节点。

具体Leaf类

- **TreeLeaf** 和 **RectangleLeaf**：继承自 Leaf，具体实现了绘制树形和矩形风格叶节点的方法。

IconFamily 类

- **描述**：抽象类，定义了获取容器和叶子节点图标的方法。
- **方法**：
  - `get_icon_for_container()`: 获取容器图标。
  - `get_icon_for_leaf()`: 获取叶子节点图标。

具体图标类

- **BlaceSpaceIcon** 和 **PokerFaceIcon**：继承自 IconFamily，具体实现了获取容器和叶子节点图标的方法。

抽象工厂类

- **描述**：StyleFactory抽象类，定义了创建风格（容器，叶节点）的方法。
- **方法**：
  - `create_container()`: 创建容器节点。
  - `create_leaf()`: 创建叶子节点。

具体工厂类

- **TreeStyleFactory**、**RectangleStyleFactory**：分别实现了不同风格的创建方法。

FunnyJsonExplorer 类

- **描述**：客户端类，通过组件接口与组合结构进行交互。
- **属性**：
  - `styleFactory`: 风格工厂实例
  - `iconFamily`： 图标实例
- **方法**：
  - `show(json_data)`: 显示JSON数据。
  - `_load(json_data, container)`: 加载JSON数据。

设计模式及作用

工厂方法模式（Factory Method）

- **描述**：定义一个用于创建对象的接口，让子类决定实例化哪一个类。
- **实现**：`StyleFactory` 类中的 `create_container`，`create_leaf` 等 `create` 方法是Factory模式的方法。

抽象工厂模式（Abstract Factory）

- **描述**：提供一个创建一系列相关或互相依赖对象的接口，而无需指定它们具体的类。
- **实现**：`StyleFactory` 是抽象工厂接口，和 `TreeStyleFactory`、`RectangleStyleFactory` 等具体的抽象工厂，它们一起实现了抽象工厂模式。

建造者模式（Builder）

- **描述**：将一个复杂对象的构建与它的表示分离，使得同样的构建过程可以创建不同的表示。
- **实现**：`StyleFactory` 类及其子类负责创建不同风格和图标的组合对象，`StyleFactory` 类中的 `create_container`，`create_leaf` 等 `create` 方法是Builder模式的部分方法(`buildPartX`)。

组合模式（Composite）

- **描述**：将对象组合成树形结构以表示“部分-整体”的层次结构，使客户端对单个对象和组合对象的使用具有一致性。
- **实现**：`Component` 抽象类定义了组合中所有对象的通用接口，`Container` 类和 `Leaf` 类分别表示组合中的复合节点和叶子节点对象。

扩展

1. **添加新的风格**：只需要在 `StyleFactory.py` 中添加新的抽象工厂，新的风格包括新的容器节点和叶节点，需要分别实现新的风格对应容器节点（参考 `Container.py` 中 `TreeContainer` 实现）和叶节点（参考 `Leaf.py` 中 `TreeLeaf` 实现），并在新的抽象工厂中create即可，并且需要在 `fje.py` 中的中注册新的风格（具体步骤见代码中注释）。
2. **添加新的图标族**：只需要在 `icon.py` 中添加新的图标族（继承 `iconFamily`），并在 `fje.py` 中的中注册新的图标族即可。

总结

通过使用工厂方法模式、抽象工厂模式、建造者模式和组合模式，系统实现了灵活的对象创建和管理机制，使客户端能够一致地对待整体和部分，并支持不同风格的展示。这样设计的系统具有较好的扩展性和维护性。