抽象工厂

杨波~研发总监/资深架构师

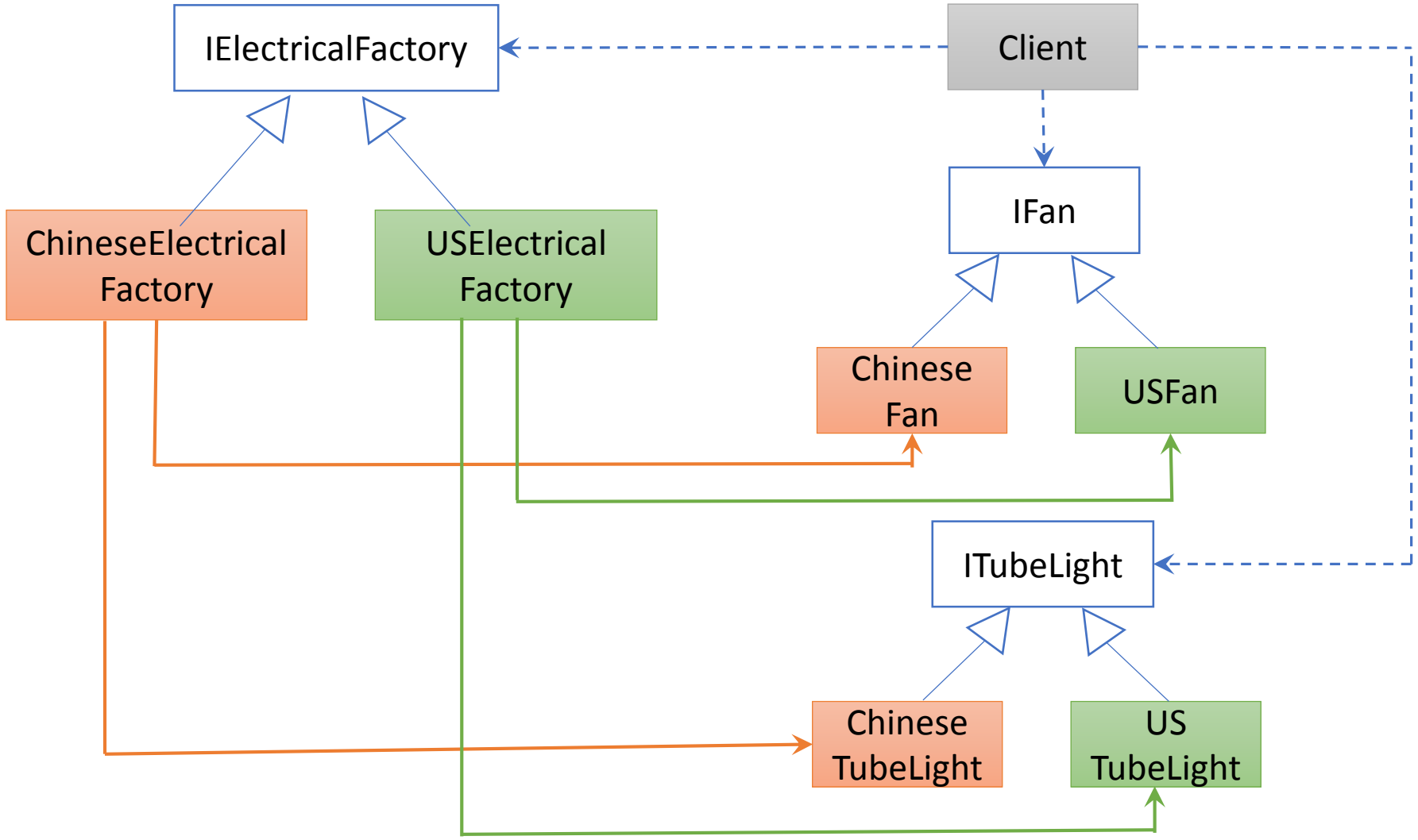
问题域

- 相关产品家族
  - 电器设备工厂
    - 电扇Fan
    - 日光灯TubeLight
    - 开关Switch
- 不同风格产品家族
  - 中国电器设备厂(China)
  - 美国电器设备分厂(US)

定义

- 提供一个接口，用于制造一族相关或者相互依赖的产品，无需指定具体的实现类。
- 创建工厂的工厂

关系图


IElectricalFactory
Client
IFan
ChineseElectrical Factory
USElectrical Factory
Chinese Fan
USFan
ITubeLight
Chinese TubeLight
US TubeLight

实现~接口

```java
package io.spring2go.corespring.abstractfactory;

// 电扇接口
public interface IFan {

    public void swithOn();

    public void switchOff();

}
```

```java
package io.spring2go.corespring.abstractfactory;

public interface IElectricalFactory {

    IFan createFan();

    ITubeLight createTubeLight();
}
```

```java
package io.spring2go.corespring.abstractfactory;

// 日光灯接口
public interface ITubeLight {
    public void swithOn();

    public void switchOff();

    public void tuneLight();
}
```

实现~中国工厂和产品

```java
package io.spring2go.corespring.abstractfactory;

public class ChineseFan implements IFan {

    public void swithOn() {
        System.out.println("The ChineseFan is swithed on ...");
    }

    public void switchOff() {
        System.out.println("The ChineseFan is swithed off ...");
    }

}
```

```java
package io.spring2go.corespring.abstractfactory;

public class ChineseElectricalFactory implements IElectricalFactory {

    public IFan createFan() {
        return new ChineseFan();
    }

    public ITubeLight createTubeLight() {
        return new ChineseTubeLight();
    }

}
```

```java
package io.spring2go.corespring.abstractfactory;

public class ChineseTubeLight implements ITubeLight {
    public void swithOn() {
        System.out.println("The ChineseTubeLight is swithed on ...");
    }

    public void switchOff() {
        System.out.println("The ChineseTubeLight is swithed off ...");
    }

    public void tuneLight() {
        System.out.println("The ChineseTubeLight is tuned ...");

    }
}
```

实现~美国工厂和产品

```java
package io.spring2go.corespring.abstractfactory;

public class USFan implements IFan {

    public void swithOn() {
        System.out.println("The USFan is swithed on ...");
    }

    public void switchOff() {
        System.out.println("The USFan is swithed off ...");
    }

}
```

```java
package io.spring2go.corespring.abstractfactory;

public class USElectricalFactory  implements IElectricalFactory {

    public IFan createFan() {
        return new USFan();
    }

    public ITubeLight createTubeLight() {
        return new USTubeLight();
    }

}
```

```java
package io.spring2go.corespring.abstractfactory;

public class USTubeLight implements ITubeLight {
    public void swithOn() {
        System.out.println("The USTubeLight is swithed on ...");
    }

    public void switchOff() {
        System.out.println("The USTubeLight is swithed off ...");
    }

    public void tuneLight() {
        System.out.println("The USTubeLight is tuned ...");
    }
}
```

客户端

```
package io.spring2go.corespring.abstractfactory;

public class AbstractFactoryMain {

    public static void main(String[] args) {

        // 国产
        IElectricalFactory electricalFactory = new ChineseElectricalFactory();

        IFan fan = electricalFactory.createFan();

        fan.swithOn();

        // 美产
        electricalFactory = new USElectricalFactory();

        ITubeLight tubeLight = electricalFactory.createTubeLight();
        tubeLight.swithOn();
        tubeLight.tuneLight();

    }

}
```

好处

- 解耦
  - 客户代码和具体产品解耦
  - 产品家族之间解耦
- 比工厂模式更高层的设计模式
- 标准化产品构造流程
- 易于替换产品家族

Spring框架应用

- FactoryBean接口基于抽象工厂模式
  - ProxyFactoryBean
  - JndiFactoryBean
  - LocalSessionFactoryBean
  - LocalContainerEntityManagerFactoryBean
- 构造具有很多依赖的复杂对象
- 构造逻辑易变且依赖于配置

思考和预习

- 简单工厂和工厂方法差异？
- 工厂方法和抽象工厂差异？
- 客户和工厂之间还是有耦合，如何进一步优化？
  - 依赖反转原理(Dependency Inversion Principle)
  - 控制反转(Inversion of Control)
  - 依赖注入(Dependency Injection)

参考

- Factory Patterns – Abstract Factory Pattern(by Snesh Prajapati)
  - https://www.codeproject.com/Articles/1137307/Factory-Patterns-Abstract-Factory-Pattern

代码

- https://github.com/spring2go/core-spring-patterns

波波微课

- 关于波波微课
  - 十多年研发经验老司机波波老师主导
  - 致力于使用新媒体技术提升学习成效
  - 主题面向Java，Spring，面向对象开发和微服务等
  - 关注工程师的成长
- 理念
  - 交互式的课程体验
  - 贴近一线企业实践
- 方法
  - 短视频，平均10分钟，最长不超过15分钟
  - 一个视频专注讲清楚一个主题
  - 50%原理+50%案例代码
  - 所有代码和ppt在github上可免费获得