

# 第46期-规约模式

2020年11月30日 10:35

## 规约模式

规约模式用于定义可重用、可组合和、可测试和有意义的过滤器，简单地说，规约模式就是对查询条件表达式用类的形式进行封装。

```
var query = _repository.Where(s => s.Age >= 18 && s.Balance < 1000);
```

## 安装规模模块

```
abp add-package Volo.Abp.Specifications
```

## 定义规约

```
public class AgeSpecification: Specification<Student>
{
    public override Expression<Func<Student, bool>> ToExpression()
    {
        return s => s.Age >= 18;
    }
}
```

## 使用规约

### 检查对象是否满足某个业务条件

```
public async Task ApplyingCredit(Student student)
{
    if (!new AgeSpecification().IsSatisfiedBy(student))
    {
        throw new Exception("This student doesn't satisfy the Age specification!");
    }

    await Task.CompletedTask;
}
```

### 使用规约查询数据

```
public async Task<IEnumerable<Student>> StudentsByCanApplying()
```

```

{
    var repository = ServiceProvider.GetRequiredService<IRepository<Student,
Guid>>>();

    var query = repository.Where(new AgeSpecification());

    var queryList = await AsyncExecuter.ToListAsync(query);

    await Task.CompletedTask;

    return queryList;
}

```

## 组合规约

规约的一个强大的功能是它们可组合使用，提供 And, Or, Not 和 AndNot 扩展方法。

```

var spec = new AgeSpecification().And(new BalanceSpecification());
var query = repository.Where(spec.ToExpression());

```

## 使用继承组合规约

```

public class AgeBalanceSpecification : AndSpecification<Student>
{
    public AgeBalanceSpecification() : base(new AgeSpecification(), new
BalanceSpecification())
    {
    }

    public override Expression<Func<Student, bool>> ToExpression()
    {
        return base.ToExpression().And(s => s.Age > 0);
    }
}

```

## 关于 LINQ 表达式

```

var query = _repository.Where(s => s.Age >= 18 && s.Balance < 1000);

```

## 什么时候使用

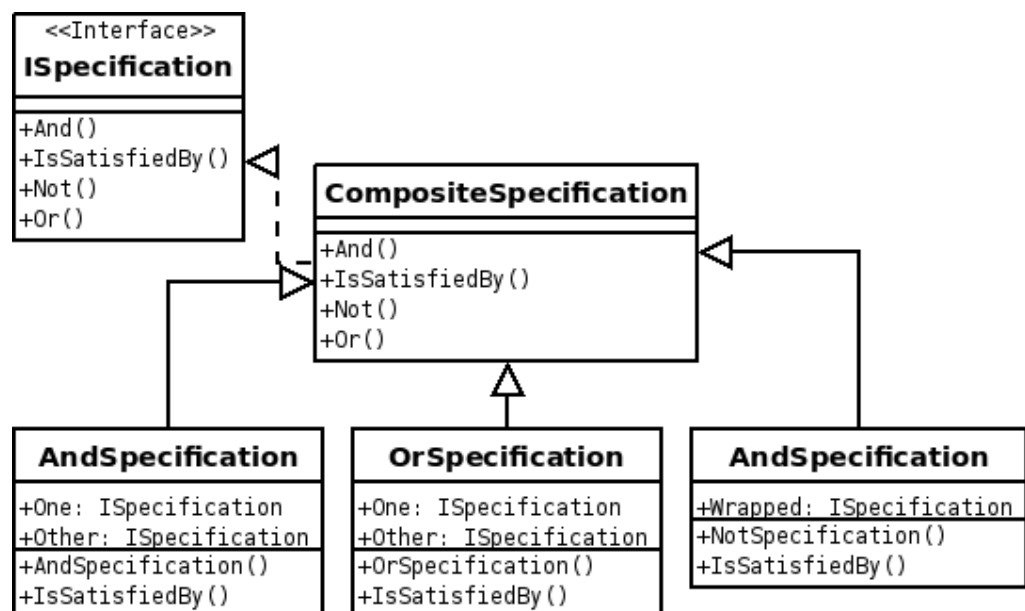
使用规约的好处：可重用性，可组合性，有意义的名称，可测试性。

## 什么时候不用

与业务无关的表达式不需要抽取成规约，比如导出根据一个条件导出一个报表。

## 实现原理

ISpecification -> Specification -> CompositeSpecification -> And, Or, Not 和 AndNot



## 参考资料

规约模式: Specification Pattern

<https://martinfowler.com/apsupp/spec.pdf>