

The background image shows a tropical beach at dusk or dawn. Large, smooth, grey granite boulders of various sizes are scattered across the sandy shore. In the background, several palm trees stand tall against a sky filled with soft, white clouds. The water is a calm, light blue-green color.

谈一个编程范式

The background image shows a tropical beach at dusk or dawn. Large, smooth, grey granite boulders of various sizes are scattered across the sandy shore. In the distance, more boulders are visible along the coastline. Several palm trees stand tall in the background, their fronds silhouetted against a bright sky. The water is a calm, light blue-green color.

函数式编程

- 先看一段代码

```
//过滤传入数组并返回data数组
function getData(col) {
  var results = [];
  for (var i=0; i < col.length; i++) {
    if (col[i] && col[i].data) {
      results.push(col[i].data);
    }
  }
  return results;
}
```

```
function getData(col) {
  return col.filter(item => item && item.data).map(item => item.data);
}
```

- 看看命令式与函数编程区别

- 看看新需求来了，怎么重构

```
function extract(filterFn, mapFn, col) {
  return col.filter(filterFn).map(mapFn);
}

const validData = item => item && item.data;

const getData = extract.bind(this, validData, item => item.data);
//对col执行getData
getData(col);

const getDataLength = extract.bind(this, validData, item => item.data.length)
//对col执行getDataLength
getDataLength(col);
```

- 我们大致已经能看出函数式编程的一些特点

1. 提倡组合（composition）
2. 每个函数尽可能完成单一的功能
3. 屏蔽细节，告诉计算机我要做什么，而不是怎么做。
4. 尽可能不引入或者少引入状态

- 这些特点运用得当的话能够为软件带来什么

1. 更好的设计和实现
2. 更加清晰可读的代码。由于状态被大大减少，代码更容易维护，也带来更强的稳定性
3. 在分布式系统下有更好的性能。函数式编程一般都在一个较高的层次进行抽象，`map` / `filter` / `reduce` 就是其基础指令，如果这些指令为分布式而优化，那么系统无需做任何改动，就可以提高性能
4. 使得惰性运算成为可能。在命令式编程中，由于你明确告诉了 CPU 一步步该怎么操作，能优化的空间被挤压；而在函数式编程里，每个函数只是封装了运算，一组数据从输入经历一系列运算到输出，如果没有人处理这些输出，则运算不会被真正执行

高阶函数

- 一个函数可以接收另一个函数作为参数，这种函数就称之为高阶函数

```
//民用住宅面积
public Func<int,int,decimal> SquareForCivil()
{
    return (width,length)=>width*length;
}
//商业住宅面积
public Func<int, int, decimal> SquareForBusiness()
{
    return (width, length) => width * length*1.2m;
}
//物业费
public decimal PropertyFee(decimal price,int width,int length, Func<int, int, decimal> square)
{
    return price*square(width, length);
}

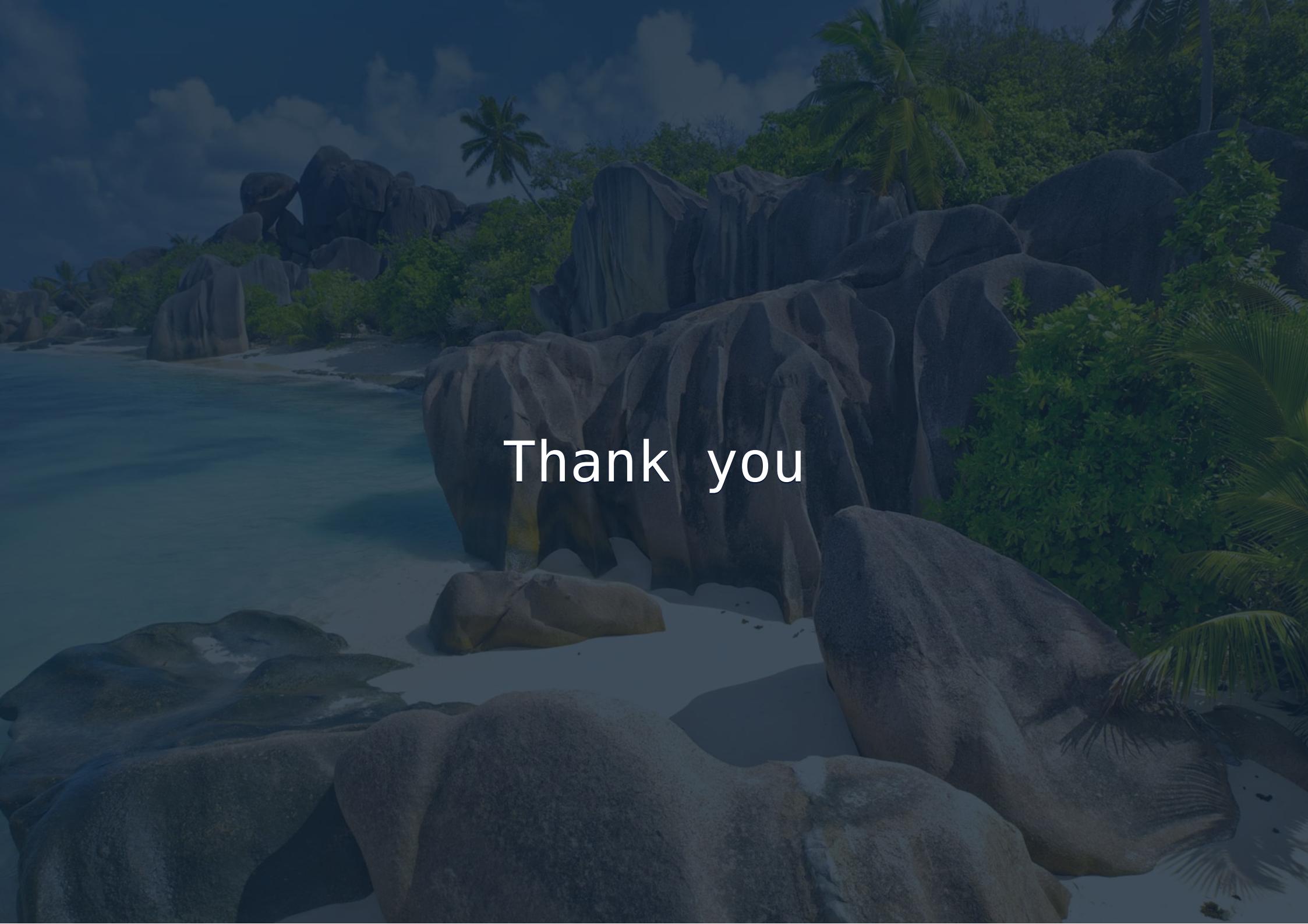
//计算60平的商住房屋物业费
PropertyFee(1.2, 4, 15, SquareForBusiness());
```

函数柯里化

- 是把接受多个参数的函数转换成接受一个单一参数(最初函数的第一个参数)的函数，并且返回接受余下的参数且返回结果的新函数
- 进行函数式编程的时候，函数参数的位置很有讲究，需要精心安排，把辅助性的，可以柯里化的参数放在前面，以方便绑定

总结

- 函数式编程还衍生出很多概念，本质都是为了组合（composition），在函数式编程里，组合是王道。

A scenic tropical beach at dusk or dawn. Large, smooth, grey granite boulders of various sizes are scattered across the white sand. Some boulders are partially submerged in the shallow, turquoise water. In the background, more boulders are stacked along the shore, and several tall palm trees stand against a dark, cloudy sky.

Thank you