05.ref

学习要点：

1.ref

本节课我们来要了解一下 Vue3.x 中的 ref 响应式用法。

一. ref

1. 在 setup()中我们可以创建一个静态的数据，比如：

```
const count = 0
```

2. 使用 ref 可以接受一个内部值，返回一个响应式且可变的 ref 对象；
3. 这个 ref 对象具有指向内部单个 property：.value(修改这个即可)；

```
import { ref } from 'vue'

// 响应式返回一个“RefImpl”的 ref 对象，值存储在内部的.value 属性上，且可变；
const count = ref(0)

// 累加时需要显示的指向.value
const increment = () => {
    count.value++
}
```

PS：ref 支持简单数据，包括字符串、数值、布尔等单一数据；但也并不是说不能用对象；

4. 如果使用 ref 对对象进行响应式化，那么该如何理解？

```
// ref 包装下的对象，语法成立
const obj = ref({
    name : 'Mr.Lee',
    age  : 100
})

// obj.value 下返回的还是 reactive 的 Proxy，obj 本身还是 RefImpl
console.log(obj.value)
// 输出 name
console.log(obj.value.name)
```

5. 最终 return 返回的时候，并不需要显示的指明.value，它会自动解开；

```
return {
    count,                    // 还有其它情况会自动展开，以后再探讨
}
```

6. ref 周边的方法：isRef、unref、toRef 和 toRefs 等；

```
// 判断是否是 RefImpl
console.log(isRef(obj))
// unref 是一个语法糖：val = isRef(val) ? val.value : val
console.log(unref(obj))

const info = reactive({
    name : 'Mr.Lee',
    age  : 100
})

// toRef：即将 infoRef 转换成 ref 对象，并将指定的属性绑定到.value 上
const infoRef = toRef(info, 'name')

// 这里的.value 输出 name
console.log(infoRef.value)

// toRef 将响应式对象转换为普通对象，内部的属性转化为 ref 对象
const infoRefs = toRefs(info)
console.log(infoRefs.name.value)
console.log(infoRefs.age.value)
```

PS：当我们要对响应式对象解构时，属性也是响应式 ref，就适合使用 toRefs，手册有例子；