16.provide.inject.上

学习要点：

1.provide.inject

本节课我们来要了解一下 Vue3.x 中的 provide.inject(提供.注入)功能；

一. provide.inject

1. 在更多复杂的场景中，组件可能有祖父孙三代组件关联，并且每一代有多个；
2. 而这三代组件，有需要自上而下提供数据进行 Props 传递，数据链条会越发复杂；
3. 在前面的例子中 Props.vue 和 Child.vue 形成了父子关系，并 Props 通信；
4. 现在我们创建一个通过 App.vue 给父和子传递信息；

```
// App.vue(祖父)
<router-view :title="title"/>

return {
    title : 'Vue3.x'
}

// Provide.vue(父)
<Child :title="title"></Child>
props : {
    title : String
},
components : {
    Child
},

// Child.vue(子)
<h3>Child Title : {{title}}</h3>
props : {
    title : String
},
```

PS：如果这种 Props 关系链的深度和广度再大一些，将变的难以理解和阅读；

5. 如果要使用 **提供/注入** 语法，它有 options 和 Composition API 两种；
6. 由于核心篇没讲到这个知识点，我们稍微提一下 options 方法，方便理解；
7. 先将刚才搭建的 Props 全部删除，再将:title=title 也删掉；

```
<router-view/>
<Child></Child>
```

8. 如果使用 provide/inject ，在祖父设置 provide，孙子设置 inject 即可；

```
// App.vue
export default {
    name : 'App',
    provide : {
        title : 'Vue3.x'
    },

// Child.vue
export default {
    name: "Child",
    inject : ['title'],
```

PS：此时数据传递，基本跳过了父组件，只需要在祖父设置提供数据，子组件获取数据即可；

PS：当然，这些本身是存在依赖关系的，祖父通过路由加载了父，父组件加载子组件；

PS：更多 options 的用法，参考手册；