MintUI-- Unit01

1.组件库基础

1.1 什么是组件库?

组件(Component)是企业为提升开发效率而开发的针对应用的、结构、表现及行为的封装。组件的优点在于**一次定义，多次使用。**

1.2 组件库的优势

- 提升开发效率
- 提升项目的可维护性
- 便于团队的协作开发

1.3 组件库的分类

- 移动端组件库
  - `Mint UI`(饿了吗) -- http://mint-ui.github.io/#!/zh-cn
  - `Vant UI`(有赞) -- https://youzan.github.io/vant/#/zh-CN/
  - `Cube UI`(滴滴) -- https://didi.github.io/cube-ui/#/zh-CN
- 桌面端组件库
  - `Element UI` (饿了吗) -- https://element.eleme.cn/#/zh-CN
  - `AT-UI`(凹凸实验室) -- https://at-ui.github.io/at-ui/#/zh
  - `View UI`(视图更新) -- https://www.iviewui.com/

2.Mint UI

2.1 安装

```
npm install --save mint-ui
```

> 该命令在脚手架的根目录下执行或者说在`package.json`文件所在的目录下执行
>
> 在命令行中实现安装的操作步骤如下:
>
> A.启动`WINDOWS`操作系统的命令行(`WIN+R`),然后输入`cmd`
>
> B.有可能需要切换盘符，输入`盘符:`，如`d:`
>
> C.输入`cd 路径`
>
> D.输入`npm install --save mint-ui`

2.2 引入Mint UI

在`src/main.js`中输入以下代码：

```
//导入Mint UI
import MintUI from 'mint-ui'

//导入Mint UI的样式表文件(library,库)
import 'mint-ui/lib/style.min.css'

//通过Vue.use()方法注册为插件
Vue.use(MintUI)
```

2.3 `Mint UI`组件库的组成

- `CSS`组件
- `JS`组件
- 表单组件

2.4 使用`Mint UI`

- `Header` 组件

  `Header`组件用于实现顶部导航，其语法结构是：

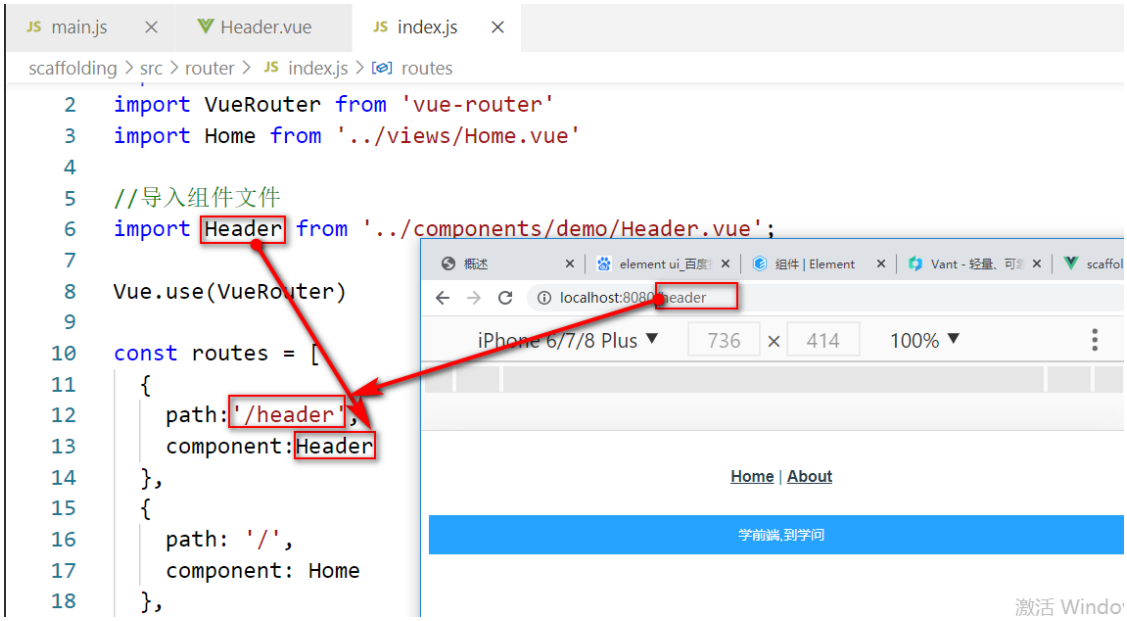
  ```
  <mt-header title="标题信息" fixed>
      ...
  </mt-header>
  ```

  > `fixed`属性为布尔属性，用于控制顶部导航固定在顶部(不会随页面的滚动而滚动)
  >
  > 在`header`中可以嵌套子元素，可以为子元素添加`slot="left"`或`slot="right"`属性

  示例代码如下：

  ```
  <template>
      <div>
          <mt-header title="学前端,到学问">
          </mt-header>
      </div>
  </template>
  ```

  组件文件与路径的对应关系如下图所示：



- `reset.css`

  所有的`HTML`元素都有默认样式，而且不同的浏览器的默认样式是不相同的，为了保证所有浏览器的默认样式相同 -- `reset.css`

  操作步骤：

  A.将`reset.css`放置在`public/css`目录内

  B.编辑`public/index.html`，在头部书写以下语句

```
<link type="text/css" rel="stylesheet" href="/css/reset.css">
```

- `Button`组件

  `Button`组件用于实现按钮，其语法结构是：

```
<mt-button
    type="按钮类型"
    size="按钮尺寸"
    icon="按钮图标型"
    disabled
    plain>
    ...
</mt-button>
```

> 按钮类型包括：`default`(默认)、`primary`(主要的)、`danger`(危险的)
>
> 按钮尺寸包括：`small`(小的)、`normal`(标准的)、`large`(大的)
>
> 按钮图标包括：`back`(返回)、`more`(更多)
>
> 可以在按钮内嵌套图像，并且为图像添加`slot="icon"`属性，此时该图像将作为按钮的图标来呈现（优先级高于`icon`属性）
>
> 图标尺寸一般为：`24x24`、`32x32`、`48x48`、`64x64`
>
> `plain`属性为布尔属性，表示按钮是否为镂空按钮

- `Field`组件

  `Field` 组件用于实现表单控件，其语法结构是：

```
<mt-field
    type="表单控件的类型"
    label="标签"
    placeholder="占位符"
    state="检测状态"
    disableClear
    :attr="原生属性"
    readonly
    disabled
    v-model="变量名称">
</mt-field>
```

> 表单控件的类型包括：
>
> - `text`，单行文本框
> - `password`，密码框
> - `textarea`，多行文本框
> - `number`，数字
> - `url`,`URL`地址，如网域
>
> `state`属性用于标识表单控件的检测状态，其值为`success`(成功)、`warning`(警告)、`error`(错误)
>
> `disableClear`属性为布尔属性，表示是否禁用`clear`按钮
>
> `:attr`为组件的原生属性，对象类型，形如：
>
> ```
> <mt-field
>     type="text"
>     :attr="{maxlength:'10',autocomplete:'off'}">
> </mt-field>
> ```
>
> `readonly`属性为布尔属性，表示当前表单控件是否为只读
>
> `disabled`属性为布尔属性，表示当前表单控件是否禁用

  示例代码如下：

```
<template>
    <div>
        <mt-field
                  type="text"
                  label="用户名"
                  state="success"
                  placeholder="请输入用户名">
        </mt-field>
        <mt-field
                  type="password"
                  label="密码"
```

```
                    state="warning"
                    disableClear
                    placeholder="请输入密码">
            </mt-field>
            <mt-field
                    type="password"
                    label="确认密码"
                    state="error"
                    placeholder="请再次输入密码"
                    :attr="{maxlength:'10',autocomplete:'off'}">
            </mt-field>
        </div>
    </template>
```

- `Toast`组件

  `Toast`组件用于显示短消息示提示框，其语法结构是：

```
//简捷语法
this.$toast("提示内容")

//标准语法
this.$toast({
    message:"提示内容",
    position:"提示框的位置(top|middle|bottom)"
    duration:持续时间(单位为毫秒,默认为3000)
})
```

- `Messagebox`组件

  `Messagebox`组件用于显示消息提示框，语法结构是：

```
//简捷语法
this.$messagebox("标题信息","提示内容")
```

作业:

1:如何实时控制表单控件的状态

2:如何让表单控件在失去焦点时自动完成检测的业务(百度！百度！百度)