note.md

# Vue组件库 - Unit03

# 1.CSS组件

## • TabContainer组件

TabContainer组件用于实现面板，其语法结构是：

<mt-tab-container v-model="变量名称" :swipeable="是否可滑动">
  
 <mt-tab-container-item id="当前面板的ID">
  
 ...
  
 </mt-tab-container-item>
  
 ...
  
</mt-tab-container>

示例代码如下：

<template>
  
 <div>
  
 <!-- 顶部选项卡开始 -->
  
 <mt-navbar v-model="active">
  
 <mt-tab-item id="a">生活</mt-tab-item>
  
 <mt-tab-item id="b">娱乐</mt-tab-item>
  
 <mt-tab-item id="c">汽车</mt-tab-item>
  
 <mt-tab-item id="d">游戏</mt-tab-item>
  
 </mt-navbar>
  
 <!-- 顶部选项卡结束 -->
  
 <!-- 面板开始 -->
  
 <mt-tab-container v-model="active" :swipeable="true">
  
 <mt-tab-container-item id="a">
  
 文章列表1<br>
  
 文章列表1<br>
  
 文章列表1<br>
  
 文章列表1<br>
  
 </mt-tab-container-item>
  
 <mt-tab-container-item id="b">
  
 文章列表2<br>
  
 文章列表2<br>
  
 文章列表2<br>
  
 文章列表2<br>
  
 文章列表2<br>
  
 </mt-tab-container-item>
  
 <mt-tab-container-item id="c">
  
 文章列表3<br>
  
 文章列表3<br>
  
 文章列表3<br>
  
 文章列表3<br>
  
 文章列表3<br>
  
 </mt-tab-container-item>
  
 <mt-tab-container-item id="d">
  
 文章列表4<br>
  
 文章列表4<br>
  
 文章列表4<br>
  
 文章列表4<br>
  
 </mt-tab-container-item>
  
 </mt-tab-container>
  
 <!-- 面板结束 -->
  
 </div>
  
</template>
  
<script>
  
export default {
  
 data(){
  
 return {
  
 active:'c'
  
 }
  
 }
  
}
  
</script>

# • Tabbar组件

Tabbar组件用于实现底部选项卡，其语法结构是：

<mt-tabbar>
  
 <mt-tab-item>
  
 ...
  
 </mt-tab-item>
  
 ...
  
</mt-tabbar>

# 2.JS组件

## • swipe组件

swipe组件用于实现轮播图，其语法结构是：

<mt-swipe
  
 :showIndicators="是否显示标志"
  
 :auto="图像的显示时长(单位为毫秒)"
  
 :speed="图像的切换时长(单位为毫秒)"
  
 :continuous="是否循环播放">
  
 <mt-swipe-item>
  
 ...
  
 </mt-swipe-item>
  
</mt-swipe>

必须为<mt-swipe>或父元素设置明确的高度

示例代码如下：

<template>
  
 <div>
  
 <div class="swipe">
  
 <mt-swipe
  
 :showIndicators="true"
  
 :auto="3000"
  
 :speed="300"
  
 :continuous="true">
  
 <mt-swipe-item>
  
 <img src="../../assets/images/ad/886x315-1592552506.jpg">
  
 </mt-swipe-item>
  
 <mt-swipe-item>
  
 <img src="../../assets/images/ad/886x315-1592556291.jpg">
  
 </mt-swipe-item>
  
 </mt-swipe>
  
 </div>
  
 </div>
  
</template>
  
<style scoped>
  
.swipe{
  
 height:120px;
  
}
  
</style>

## • Lazy Load指令

Lazy Load指令用于实现图像的懒加载，其语法结构是：

<img v-lazy="item">

item为图像的URL地址，图像需要通过require进行引入

* import是编译时调用，所以必须放在文件开头
* require是运行时调用，可以放在任意位置

必须为懒加载的图像设置CSS样式，如：

<style scoped>
  
 img{
  
 display: block;
  
 width: 100%;
  
 }
  
</style>

示例代码如下：

<template>
  
 <div>
  
 <p v-for="(image,index) of images" :key="index">
  
 <img v-lazy="image">
  
 </p>
  
 </div>
  
</template>
  
   
<style scoped>
  
img{
  
 display: block;
  
 width: 100%;
  
}
  
</style>
  
<script>
  
export default {
  
 data(){
  
 return {
  
 images:[
  
 require('../../assets/images/590x470.jpg'),
  
 require('../../assets/images/590x47011.jpg'),
  
 require('../../assets/images/739d9b28bc8b5587.jpg'),
  
 require('../../assets/images/2231df5a1de265c0.jpg'),
  
 require('../../assets/images/418665f516b1a85b.jpg'),
  
 require('../../assets/images/bb7709198b4e78d0.jpg'),
  
 ]
  
 }
  
 }
  
}
  
</script>