# 主页面布局

现在的管理系统页面都有一个通用的布局结构，比如左边菜单栏加右边的主显示区域，在开发主体页面之前可以先确定下这个主体页面布局，后面开发就比较方便。

https://element-plus.gitee.io/#/zh-CN/component/container

这里可以用element的现成的架构。

改动下，布局在其他页面会经常用到，改动下，保存到components/Layout.vue文件中

```
<template>
  <el-container>
    <el-header>Header</el-header>
    <el-container>
      <el-aside width="200px">Aside</el-aside>
      <el-main>Main</el-main>
    </el-container>
  </el-container>
</template>

<script>
export default {
  name: 'Layout'
}
</script>

<style>
  .el-header,
  .el-footer {
    background-color: #b3c0d1;
    color: var(--el-text-color-primary);
    text-align: center;
    line-height: 60px;
  }

  .el-aside {
    background-color: #d3dce6;
    color: var(--el-text-color-primary);
    text-align: center;
    line-height: 200px;
  }

  .el-main {
    background-color: #e9eef3;
    color: var(--el-text-color-primary);
    text-align: center;
    line-height: 160px;
  }

  body > .el-container {
    margin-bottom: 40px;
  }

</style>
```

引入到Home.vue组件中看看效果

```
<template>
  <layout></layout>
</template>

<script>
import Layout from '../components/Layout.vue'
// @ is an alias to /src
import {logout} from '../httplib'
export default {
  name: 'Home',
  components: {
    Layout
  },
  setup(){
    function quit(){
      logout()
    }
    return{
      quit
    }
  }
}
</script>
```



# 顶栏组件开发

接下来开发一个顶部菜单栏，填充到页面布局中。

https://element-plus.gitee.io/#/zh-CN/component/menu

同样可以参考现有的组件样式，获取并做出改动。

## navbar.vue

借助下拉菜单https://element-plus.gitee.io/#/zh-CN/component/dropdown

和头像组件https://element-plus.gitee.io/#/zh-CN/component/avatar

以及图标组件https://element-plus.gitee.io/#/zh-CN/component/icon

```
<template>
<div class="logo">测试平台</div>
<el-menu
  class="navbar"
  mode="horizontal"
  background-color="#545c64"
  text-color="#fff"
  active-text-color="#ffd04b"
>
  <el-menu-item id="avatar">
    <el-dropdown>
      <el-avatar :size="32" :src="circleUrl"></el-avatar>
      <template #dropdown>
```

```
        <el-dropdown-menu size="medium">
          <el-dropdown-item icon="el-icon-setting">设置</el-dropdown-item>
          <el-dropdown-item icon="el-icon-turn-off" @click="submitExit()">退出
</el-dropdown-item>
        </el-dropdown-menu>
      </template>
    </el-dropdown>
  </el-menu-item>
</el-menu>
</template>

<script>
  export default {
    setup(){
      return {
        circleUrl:
          'https://cube.elemecdn.com/3/7c/3ea6beec64369c2642b92c6726f1epng.png',
      }
    }
  }
</script>

<style>
.navbar{
  text-align: center;
}
.logo{
  width: 200px;
  background-color: rgb(255, 208, 75);
  float: left;
}
#avatar{
  width: 96px;
  text-align: center;
  float: right;
}

</style>
```



## 具名插槽

我们希望组件做成可插拔式，这样扩展效果更好，此时可以借助插槽 | Vue.js (vuejs.org)

具名插槽就提供了此功能。

先在布局组件留出插槽：Layout.vue

```
<template>
  <el-container>
    <el-header>
      <slot name="header"></slot>
    </el-header>
    <el-container>
      <el-aside width="200px">
        <slot name="aside"></slot>
      </el-aside>
```

```
      <el-main>
        <slot name="main"></slot>
      </el-main>
    </el-container>
  </el-container>
</template>

<script>
import Navbar from '../components/Navbar.vue'

export default {
  name: 'Layout',
  components:{
    Navbar
  }
}
</script>

<style>
  .el-header,
  .el-aside,
  .el-main{
    padding:0px;
  }
  .el-header,
  .el-footer {
    background-color: #b3c0d1;
    color: var(--el-text-color-primary);
    text-align: center;
    line-height: 60px;
  }

  .el-aside {
    background-color: #d3dce6;
    color: var(--el-text-color-primary);
    text-align: center;
    line-height: 200px;
  }

  .el-main {
    background-color: #e9eef3;
    color: var(--el-text-color-primary);
    text-align: center;
    line-height: 160px;
  }

  body > .el-container {
    margin-bottom: 40px;
  }

</style>
```

### 主页这里插入组件

Home.vue

```
<template>
  <layout>
    <template v-slot:header>
      <navbar></navbar>
    </template>
  </layout>
</template>

<script>
import Layout from '../components/Layout.vue'
import Navbar from '../components/Navbar.vue'
// @ is an alias to /src
import {logout} from '../httplib'
export default {
  name: 'Home',
  components: {
    Layout,
    Navbar
  },
  setup(){
    function quit(){
      logout()
    }
    return{
      quit
    }
  }
}
</script>
```



## 首页测边栏组件开发

接下来可以开发测边栏组件，和顶栏组件一样，也是可以采用现成的element-puls组件，

构成测边栏和顶栏的元素是一样的，区别在于`mode`属性，horizontal 使菜单为水平模式，vertical为垂直模式（默认效果）。

### sidebar.vue

删减掉多余的标签，采用深色模式的那个，菜单的图形小标签可以引用[组件 | Element (gitee.io)]

总有一款适合你。

```
<template>
    <el-menu
      :uniqueOpened="true"
      default-active="2"
      class="siderbar"
      background-color="#545c64"
      text-color="#fff"
```

```
      active-text-color="#ffd04b"
    >
      <el-menu-item>
        <i class="el-icon-s-order"></i>
        <template #title>测试用例</template>
      </el-menu-item>
      <el-menu-item>
        <i class="el-icon-s-promotion"></i>
        <template #title>web接口</template>
      </el-menu-item>
      <el-menu-item>
        <i class="el-icon-s-flag"></i>
        <template #title>测试计划</template>
      </el-menu-item>
      <el-menu-item>
        <i class="el-icon-s-data"></i>
        <template #title>测试报告</template>
      </el-menu-item>
    </el-menu>
</template>

<script>
  export default {

  }
</script>

<style>

</style>
```

## 加入动态效果

如果希望达到自动收起菜单功能

```
<template>
    <el-menu
      :uniqueOpened="true"
      default-active="2"
      class="siderbar"
      background-color="#545c64"
      text-color="#fff"
      active-text-color="#ffd04b"
      :collapse="iscollapse"
    >
      <el-menu-item>
        <i class="el-icon-s-order"></i>
        <template #title>测试用例</template>
      </el-menu-item>
      <el-menu-item>
        <i class="el-icon-s-promotion"></i>
        <template #title>web接口</template>
      </el-menu-item>
      <el-menu-item>
        <i class="el-icon-s-flag"></i>
        <template #title>测试计划</template>
```

```
      </el-menu-item>
      <el-menu-item>
        <i class="el-icon-s-data"></i>
        <template #title>测试报告</template>
      </el-menu-item>
      <el-menu-item @click="switch_sidebar" class="switch_bar">
        <i class="el-icon-arrow-left" v-if="iscollapse===false"></i>
        <i class="el-icon-arrow-right" v-else></i>
      </el-menu-item>
    </el-menu>
</template>

<script>
  import {ref} from 'vue'
  export default {
    setup(){
      const iscollapse = ref(false)
      function switch_sidebar(){
        iscollapse.value=!iscollapse.value
      }
      return{
        iscollapse,
        switch_sidebar
      }
    }
  }
</script>

<style>
.siderbar{
  /* 撑开测边栏 */
  height: 100vh;
}
</style>
```



## 合并到主页

加入到主页组件插槽中Home.vue

```
<template>
  <layout>
    <template v-slot:header>
      <navbar></navbar>
    </template>
    <template #aside>
      <sidebar></sidebar>
    </template>
  </layout>
</template>

<script>
import Layout from '../components/Layout.vue'
import Navbar from '../components/Navbar.vue'
import Sidebar from '../components/Sidebar.vue'
```

```
// @ is an alias to /src
import {logout} from '../httplib'
export default {
  name: 'Home',
  components: {
    Layout,
    Navbar,
    Sidebar
  },
  setup(){
    function quit(){
      logout()
    }
    return{
      quit
    }
  }
}
</script>
```

## 菜单路由效果

目前菜单项点击没有反应，我们希望点击菜单，右边就出现对应的页面。

### 设计路由对应的组件

pages/Cases.vue

```
<template>
    <div>测试用例</div>
</template>
```

pages/Plans.vue

```
<template>
    <div>测试计划</div>
</template>
```

pages/Request.vue

```
<template>
    <div>测试报告</div>
</template>
```

pages/Reports.vue

```
<template>
    <div>Request</div>
</template>
```

### 设计菜单对应的路由

注意，我们要在Home下面去嵌套路由，这样路由对应的组件才能显示在对应的区域内

```
const routes = [  //路由列表
  {
    path: '/',   // 请求的路径从Host之后开始计算
    name: 'Home',  // 路由名称，方便后续引用
    component: Home,  // 组件
    children:[{
      path: 'cases',
      component: () => import('../pages/Cases.vue')
    },{
      path: 'request',
      component: () => import('../pages/Request.vue')
    },{
      path: 'plans',
      component: () => import('../pages/Plans.vue')
    },{
      path: 'reports',
      component: () => import('../pages/Reports.vue')
    }]
  },
  ...
  ]
```

### 设置主页面的显示区

Home.vue

```
<template>
  <layout>
    <template #header>
      <navbar></navbar>
    </template>
    <template #aside>
      <sidebar></sidebar>
    </template>
    <template #main>
     <router-view></router-view>
    </template>
  </layout>
</template>
```