优医问诊day03

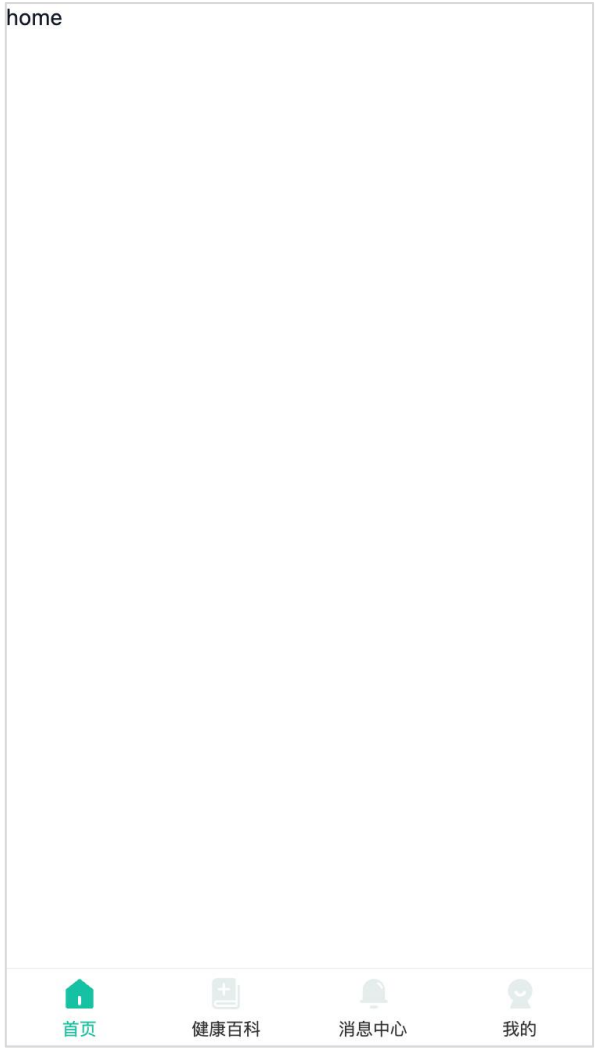
布局容器、个人中心、家庭档案

黑马程序员
www.itheima.com
传智教育旗下
高端IT教育品牌

01 布局容器–组件路由配置

布局容器-组件与路由配置

布局容器-组件与路由配置



一级路由：布局容器 /

- 二级路由：首页组件 /home
- 二级路由：健康百科 /article
- 二级路由：消息通知 /notify
- 二级路由：个人中心 /user

```
{
  path: '/',
  redirect: '/home',
  component: () => import('@/views/Layout/index.vue'),
  children: [
    { path: '/home', component: () => import('@/views/Home/index.vue') },
    {
      path: '/article',
      component: () => import('@/views/Article/index.vue')
    },
    {
      path: '/notify',
      component: () => import('@/views/Notify/index.vue')
    },
    { path: '/user', component: () => import('@/views/User/index.vue') }
  ]
}
```

02 布局容器–Tabbar切换效果

布局容器-Tabbar切换效果

tabbar组件 → 支持路由切换 → 自定义图标 → 图标切换

```
<van-tabbar route>  1
  <van-tabbar-item to="/home">
    首页
    <template #icon="{ active }">  2
      <cp-icon :name="`home-index-${active ? 'active' : 'default'}`">
      </cp-icon>
    </template>
  </van-tabbar-item>
  <van-tabbar-item to="/article">…
  </van-tabbar-item>
  <van-tabbar-item to="/notify">…
  </van-tabbar-item>
  <van-tabbar-item to="/user">…
  </van-tabbar-item>
```

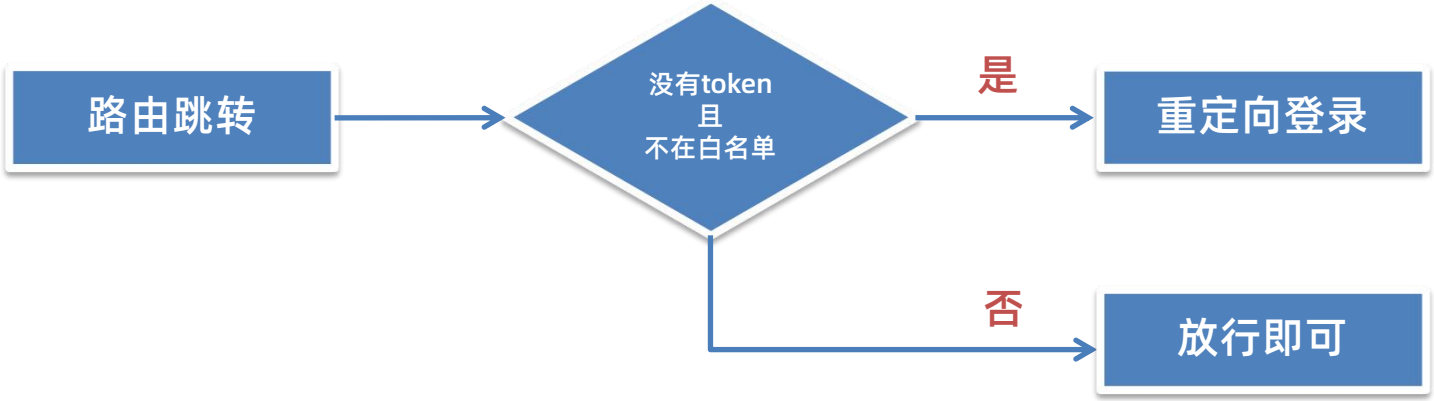
03 布局容器–访问权限控制

布局容器-访问权限控制

全局前置守卫变化：

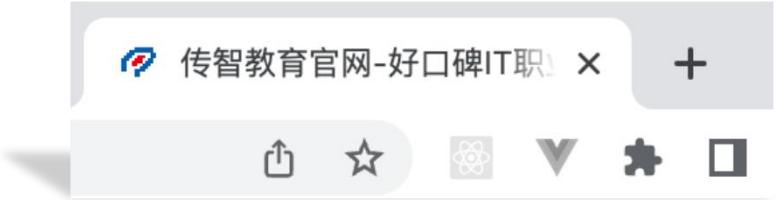
```
router.beforeEach((to, from) => {
  // next 现在可选，next() 放行  next('/login') 跳转指定路由（重定向）
  // ------------------------------------------------------------
  // 返回 false 取消
  // 返回 true | undefined 放行
  // 返回 路由地址（对象格式） 重定向
  return true
})
```

访问权限控制实现：



04 布局容器–页面标题

布局容器-页面标题



- 路由配置元信息 meta 中定义标题
- 路由全局后置守卫设置标题给页面，扩展元信息类型

```
{
  path: '/login',
  component: () => import('@/views/Login/index.vue'),
  meta: { title: '登录' }
},
```

05 布局容器–加载进度

布局容器-加载进度

开启进度条 beforeEach → 解析路由，加载异步组件 → 结束进度条 afterEach

安装：pnpm i nprogress 和 pnpm i @types/nprogress -D

导入：

```
import NProgress from 'nprogress'
import 'nprogress/nprogress.css'
```

控制：NProgress.start() NProgress.done()

调整：不显示环形进度图标，颜色改成优医问诊主题色

```
NProgress.configure({
  showSpinner: false
})
```

```
#nprogress .bar {
  background-color: var(--cp-primary) !important;
}
```

06 个人中心–用户信息类型

个人中心-用户信息类型

id	string	✓	用户id
avatar	string	✓	头像
mobile	string	✓	手机号
account	string	✓	用户名
likeNumber	number	✓	关注数量
collectionNumber	number	✓	收藏数量
score	number	✓	我的总积分
couponNumber	number	✓	我的优惠券数量
orderInfo	object {4}	✓	
paidNumber	number	✓	待付款
receivedNumber	number	✓	待发货
shippedNumber	number	✓	待收货
finishedNumber	number	✓	已完成

=

```
export type User = {
  /* token令牌 */
  token: string
  /* 用户ID */
  id: string
  /* 用户名称 */
  account: string
  /* 手机号 */
  mobile: string
  /* 头像 */
  avatar: string
}
```

去掉token

```
Omit<User, 'token'>
```

+

likeNumber
collectionNumber
score
couponNumber
orderInfo
paidNumber
receivedNumber
shippedNumber
finishedNumber

TS内置类型 Pick 与 Omit

```
type Person = {
  name: string
  age: number
}
type PickPerson = Pick<Person, 'age'>
// PickPerson === { age: string }
```

```
type Person = {
  name: string
  age: number
}
type OmitPerson = Omit<Person, 'age'>
// OmitPerson === { name: string }
```

Pick 可以从一个对象类型中取出某些属性

Omit 可以从一个对象类型中排出某些属性

07 个人中心–用户信息展示

个人中心-用户信息展示

准备基础结构 → 定义API函数 → 获取数据 → 完成渲染

```
// 获取用户信息
export const getUserInfo = () => request<UserInfo>('patient/myUser')
```

```
const user = ref<UserInfo>()
onMounted(async () => {
  const res = await getUserInfo()
  user.value = res.data
})
```

```
<van-badge :content="user?.orderInfo.paidNumber || ''">
  <cp-icon name="user-paid" />
</van-badge>
```

08 个人中心–快捷工具和退出登录

个人中心-快捷工具和退出登录

准备基础结构 → 初始化数据 → 完成渲染

```
<van-cell
  v-for="(item, i) in tools"
  :key="item.label"
  :title="item.label"
  :to="item.path"
  is-link
  :border="false"
>
  <template #icon><cp-icon :name="`user-tool-0${i + 1}`" /></template>
</van-cell>
```

准备按钮 → 弹窗提示 → 完成退出

```
await showConfirmDialog({
  title: '温馨提示',
  message: '您是否确认退出优医问诊？'
})
// 点击的确认
store.delUser()
router.push('/login')
```

09 家庭档案–展示患者列表

家庭档案

患者列表 → 添加患者 → 编辑患者 → 删除患者

家庭档案

患者列表 → 添加患者 → 编辑患者 → 删除患者

家庭档案-展示患者列表

路由和组件结构 → 定义患者类型 → 患者列表API → 获取数据渲染

1

```
{
  path: '/user/patient',
  component: () => import('@/views/User/PatientPage.vue'),
  meta: { title: '家庭答案' }
},
```

2

```
// 家庭档案-患者信息
export type Patient = {
  /** 患者ID */
  id: string
  /** 患者名称 */
  name: string
  /** 身份证号 */
  idCard: string
  /** 0不默认  1默认 */
  defaultFlag: 0 | 1
  /** 0 女  1 男 */
  gender: 0 | 1
  /** 性别文字 */
  genderValue: string
  /** 年龄 */
  age: number
}

// 家庭档案-患者信息列表
export type PatientList = Patient[]
```

3

```
// 获取患者列表
export const getPatientList = () => request<PatientList>('/patient/mylist')
```

4

```
// 获取患者列表数据
const list = ref<PatientList>([])
const loadList = async () => {
  const res = await getPatientList()
  list.value = res.data
}

onMounted(() => {
  loadList()
})
```

```
<div class="patient-item" v-for="item in list" :key="item.id">
  <div class="info">
    <span class="name">{{ item.name }}</span>
    <span class="id">
      {{ item.idCard.replace(/^(.{6}).+(.{4})$/, '$1******$2') }}
    </span>
    <span>{{ item.genderValue }}</span>
    <span>{{ item.age }}岁</span>
  </div>
  <div class="icon"
    <cp-icon name="user-edit" />
  </div>
  <div class="tag" v-if="item.defaultFlag === 1">默认</div>
</div>
<div class="patient-add" v-if="list.length < 6">
```

10

通用组件–cp–radio–btn

- v–model语法糖

家庭档案-添加患者

- **v-model语法糖** — 需要封装支持 v-model 的组件
- **封装单选组件** — Vant组件库没有单选按钮组件
- **侧滑添加弹层** — 实现侧滑显示添加患者界面
- **表单数据绑定** — 为表单绑定响应式数据
- **表单校验** — 为表单添加表单校验
- **实现添加** — 通过请求完成添加患者功能

单选按钮组件-v-model 语法糖

Vue2 的 v-model 默认解析成 :value 与 @input

Vue3 的 v-model 默认解析成 :modelValue 与 @update:modelValue

```
<cp-radio-btn v-model="count"></cp-radio-btn>
```

```
<cp-radio-btn
  :model-value="count"
  @update:model-value="count = $event"
></cp-radio-btn>
```

```
<script setup lang="ts">
defineProps<{ modelValue: number }>()
defineEmits<{ (e: 'update:modelValue', value: number): void }>()
</script>

<template>
  <div class="cp-radio-btn">
    计数器: {{ modelValue }}
    <button @click="$emit('update:modelValue', modelValue + 1)">+1</button>
  </div>
</template>
```

Vue2 的 attr.sync 默认解析成 :attr 与 @update:attr

Vue3 的 v-model:attr 默认解析成 :attr 与 @update:attr

```
<cp-radio-btn v-model:count="count"></cp-radio-btn>
```

```
<cp-radio-btn :count="count" @update:count="count = $event"></cp-radio-btn>
```

```
<script setup lang="ts">
defineProps<{ count: number }>()
defineEmits<{ (e: 'update:count', value: number): void }>()
</script>

<template>
  <div class="cp-radio-btn">
    计数器: {{ count }}
    <button @click="$emit('update:count', count + 1)">+1</button>
  </div>
</template>
```

11

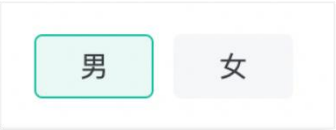
通用组件–cp–radio–btn

- 组件封装

家庭档案-添加患者

- v-model语法糖：需要封装支持 v-model 的组件
- 封装单选组件：Vant组件库没有单选按钮组件
- 侧滑添加弹层：实现侧滑显示添加患者界面
- 表单数据绑定：为表单绑定响应式数据
- 表单校验：为表单添加表单校验
- 实现添加：通过请求完成添加患者功能

单选按钮组件-实现封装



组件结构 → 渲染选项 → 选中和切换 → 支持v-model

```
const options = [
  { label: '男', value: 1 },
  { label: '女', value: 0 }
]
const gender = ref(1)
```

```
<cp-radio-btn :options="options" v-model="gender"></cp-radio-btn>
```

```
<script setup lang="ts">
defineProps<{
  options: {
    label: string
    value: string | number
  }[]
  modelValue?: string | number
}>()
const emit = defineEmits<{
  (e: 'update:modelValue', value: string | number): void
}>()
</script>

<template>
  <div class="cp-radio-btn">
    <a
      class="item"
      href="javascript:;"
      v-for="item in options"
      :key="item.value"
      :class="{ active: modelValue === item.value }"
      @click="emit('update:modelValue', item.value)"
    >
      {{ item.label }}
    </a>
  </div>
</template>
```

12 家庭档案–侧滑层控制

家庭档案-添加患者

- **v-model语法糖** — 需要封装支持 v-model 的组件
- **封装单选组件** — Vant组件库没有单选按钮组件
- **侧滑添加弹层** — 实现侧滑显示添加患者界面
- **表单数据绑定** — 为表单绑定响应式数据
- **表单校验** — 为表单添加表单校验
- **实现添加** — 通过请求完成添加患者功能

家庭档案-添加患者-侧滑弹层

用popup组件 → nav支持关闭 → 准备表单

```
<van-popup v-model:show="show" position="right">
  <cp-nav-bar title="添加患者" right-text="保存"></cp-nav-bar>
  <!-- 表单 -->
</van-popup>
```

```
const show = ref(false)
const showPopup = () => {
  show.value = true
}
```

```
<div class="patient-add" v-if="list.length < 6" @click="showPopup()">
  <cp-icon name="user-add" />
  <p>添加患者</p>
</div>
```

```
.patient-page {
  padding: 46px 0 80px;
  :deep() {
    0 references
    .van-popup {
      width: 100%;
      height: 100%;
      padding-top: 46px;
      box-sizing: border-box;
    }
  }
}
```

```
<van-form autocomplete="off">
  <van-field label="真实姓名" placeholder="请输入真实姓名" />
  <van-field label="身份证号" placeholder="请输入身份证号" />
  <van-field label="性别" class="pb4">
    <!-- 单选按钮组件 -->
    <template #input>
      <cp-radio-btn :options="options"></cp-radio-btn>
    </template>
  </van-field>
  <van-field label="默认就诊人">
    <template #input>
      <van-checkbox :icon-size="18" round />
    </template>
  </van-field>
</van-form>
```

```
const props = defineProps<{
  title?: string
  rightText?: string
  back?: () => void
}>()
```

```
const onClickLeft = () => {
  if (props.back) return props.back()
  if (history.state?.back) {
    router.back()
  } else {
    router.push('/')
  }
}
```

```
<cp-nav-bar
  title="添加患者"
  right-text="保存"
  :back="() => (show = false)"
></cp-nav-bar>
```

13 家庭档案–表单数据绑定

家庭档案-添加患者

- **v-model语法糖** — 需要封装支持 v-model 的组件
- **封装单选组件** — Vant组件库没有单选按钮组件
- **侧滑添加弹层** — 实现侧滑显示添加患者界面
- **表单数据绑定** — 为表单绑定响应式数据
- **表单校验** — 为表单添加表单校验
- **实现添加** — 通过请求完成添加患者功能

家庭档案-添加患者-表单数据绑定

修改类型 → 数据绑定 → 支持复选框 → 重置表单

```
// 家庭档案-患者信息
export type Patient = {
  /** 患者ID */
  id?: string
  /** 患者名称 */
  name: string
  /** 身份证号 */
  idCard: string
  /** 0不默认  1默认 */
  defaultFlag: 0 | 1
  /** 0 女  1 男 */
  gender: 0 | 1
  /** 性别文字 */
  genderValue?: string
  /** 年龄 */
  age?: number
}
```

修改类型

```
const patient = ref<Patient>({
  name: '',
  idCard: '',
  gender: 1,
  defaultFlag: 0
})
```

```
<van-field
  v-model="patient.name"
  label="真实姓名"
  placeholder="请输入真实姓名"
/>
<van-field
  v-model="patient.idCard"
  label="身份证号"
  placeholder="请输入身份证号"
/>
<van-field label="性别" class="pb4">
  <!-- 单选按钮组件 -->
  <template #input>
    <cp-radio-btn
      v-model="patient.gender"
      :options="options"
    ></cp-radio-btn>
  </template>
</van-field>
<van-field label="默认就诊人">
  <template #input>
    <van-checkbox v-model="defaultFlag" :icon-size="18" round />
  </template>
</van-field>
```

数据绑定

```
const defaultFlag = computed({
  get: () => (patient.value.defaultFlag === 1 ? true : false),
  set: (value) => (patient.value.defaultFlag = value ? 1 : 0)
})
```

支持复选框

```
const showPopup = () => {
  patient.value = { ...initPatient }
  show.value = true
}

const initPatient: Patient = {
  name: '',
  idCard: '',
  gender: 1,
  defaultFlag: 0
}
const patient = ref<Patient>({ ...initPatient })
```

重置表单

14 家庭档案–表单校验

家庭档案-添加患者

- v-model语法糖：需要封装支持 v-model 的组件
- 封装单选组件：Vant组件库没有单选按钮组件
- 侧滑添加弹层：实现侧滑显示添加患者界面
- 表单数据绑定：为表单绑定响应式数据
- 表单校验：为表单添加表单校验
- 实现添加：通过请求完成添加患者功能

家庭档案-添加患者-表单校验

表单项校验 → 提交时校验 → 性别确认提示

```
const nameRules: FieldRule[] = [
  { required: true, message: '请输入姓名' },
  { pattern: /^(?:[\u4e00-\u9fa5·]{2,16})$/, message: '中文2-16个字符' }
]

const idCardRules: FieldRule[] = [
  { required: true, message: '请输入身份证号' },
  {
    pattern:
      /^[1-9]\d{5}(?:18|19|20)\d{2}(?:0[1-9]|10|11|12)(?:0[1-9]|[1-2]\d|30|31)\d{3}[\dXx]$/,
    message: '身份证号不正确'
  }
]
```

```
// 身份证倒数第二位，单数是男，双数是女
const gender = +patient.value.idCard.slice(-2, -1) % 2
if (gender !== patient.value.gender) {
  await showConfirmDialog({
    title: '温馨提示',
    message: '选择的性别和身份证信息中的不一致\n您确认提交吗？'
  })
}
```

性别确认提示

```
<van-field
  v-model="patient.name"
  label="真实姓名"
  placeholder="请输入真实姓名"
  :rules="nameRules"
/>
<van-field
  v-model="patient.idCard"
  label="身份证号"
  placeholder="请输入身份证号"
  :rules="idCardRules"
/>
```

表单项校验

```
const form = ref<FormInstance>()
const onSubmit = () => {
  form.value?.validate()
}
```

```
<van-form autocomplete="off" ref="form">
```

提交时校验

15 家庭档案–添加患者

家庭档案-添加患者

- v-model语法糖：需要封装支持 v-model 的组件
- 封装单选组件：Vant组件库没有单选按钮组件
- 侧滑添加弹层：实现侧滑显示添加患者界面
- 表单数据绑定：为表单绑定响应式数据
- 表单校验：为表单添加表单校验
- 实现添加：通过请求完成添加患者功能

家庭档案-添加患者-进行添加

添加患者API → 进行提交

```
// 添加患者
export const addPatient = (patient: Patient) =>
  request('/patient/add', 'POST', patient)
```

```
await addPatient(patient.value)
show.value = false
loadList()
showSuccessToast('添加成功')
```

16 家庭档案–编辑患者

家庭档案-编辑患者

回填表单 → 区分标题 → 编辑患者API → 合并编辑

```
const showPopup = (item?: Patient) => {
  if (item) {
    // 如果点的是编辑，解构出后台需要的数据
    const { id, gender, name, idCard, defaultFlag } = item
    patient.value = { id, gender, name, idCard, defaultFlag }
  } else {
    patient.value = { ...initPatient }
  }
  show.value = true
}
```

回填表单

```
// 编辑患者
export const editPatient = (patient: Patient) =>
  request('/patient/update', 'PUT', patient)
```

编辑患者API

```
patient.value.id
  ? await editPatient(patient.value)
  : await addPatient(patient.value)
show.value = false
loadList()
showSuccessToast(patient.value.id ? '编辑成功' : '添加成功')
```

合并编辑

```
<cp-nav-bar
  :title="patient.id ? '编辑患者' : '添加患者'"
  right-text="保存"
  :back="() => (show = false)"
  @click-right="onSubmit"
></cp-nav-bar>
```

标题区分

17 家庭档案–删除患者

家庭档案-删除患者

准备删除按钮 → 删除患者API → 进行删除

```
<van-action-bar v-if="patient.id">
  <van-action-bar-button @click="remove">删除</van-action-bar-button>
</van-action-bar>
```

```
// 删除患者信息
export const delPatient = (id: string) =>
  request(`/patient/del/${id}`, 'DELETE')
```

```
const remove = async () => {
  if (patient.value.id) {
    await showConfirmDialog({
      title: '温馨提示',
      message: `您确认要删除 ${patient.value.name} 患者信息吗 ? `
    })
    await delPatient(patient.value.id)
    show.value = false
    loadList()
    showSuccessToast('删除成功')
  }
}
```

黑马程序员

www.itheima.com

传智教育旗下高端IT教育品牌