第四天

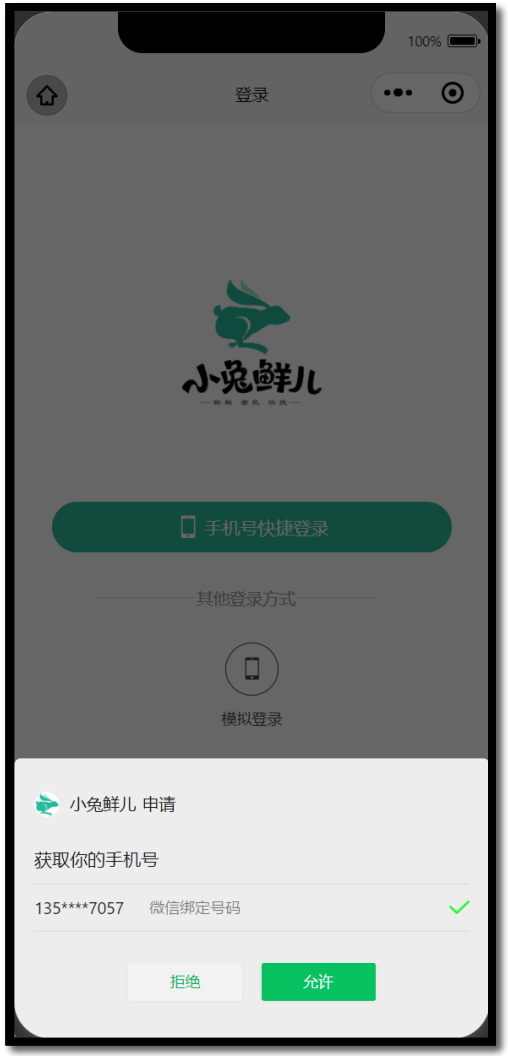
登录模块 + 用户模块

黑马程序员
www.itheima.com

传智教育旗下
高端IT教育品牌

登录模块 - 小程序快捷登录

当前接口所需参数



官方文档

[获取登录凭证(code)]

[获取手机号]

登录模块 - 小程序快捷登录

定义接口

```ts
// src/services/login.ts
export type LoginParams = {
  code: string
  encryptedData: string
  iv: string
}

/**
 * 小程序登录
 * @param data 请求参数
 */
export const postLoginWxMin = (data: LoginParams) => {
  return http({
    method: 'POST',
    url: '/login/wxMin',
    data,
  })
}
```

获取登录凭证 code

```ts
// src/pages/login/login.vue
let code = ''
onLoad(async () => {
  // 获取登录凭证
  const res = await wx.login()
  code = res.code
})
```

获取手机号并登录

```vue
<button open-type="getPhoneNumber" @getphonenumber="onGetphonenumber">
  手机号快捷登录
</button>
```

```ts
// 获取手机号码
const onGetphonenumber: UniHelper.ButtonOnGetphonenumber = async (ev) => {
  const encryptedData = ev.detail!.encryptedData!
  const iv = ev.detail!.iv!
  // 小程序登录
  const res = await postLoginWxMin({ code, encryptedData, iv })
  // 成功提示
  uni.showToast({ icon: 'success', title: '登录成功' })
}
```

注意：获取手机号功能针对非个人开发者，且完成认证的小程序开放

工作场景：使用企业小程序 appid ，且把微信号添加到开发者列表中

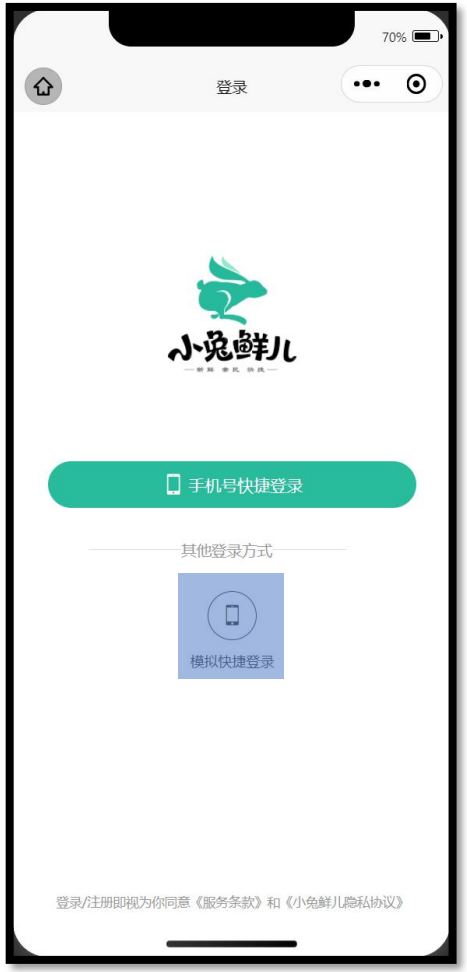
登录模块 - 模拟快捷登录

封装模拟登录API → 调用模拟登录

说明：获取手机号功能对个人开发者**不开放** 官方文档

```
// src/services/login.ts
/**
 * 小程序登录_内测版(模拟快捷登录)
 * @param phoneNumber 模拟手机号
 */
export const postLoginWxMinSimpleAPI = (phoneNumber: string) => {
  return http({
    method: 'POST',
    url: '/login/wxMin/simple',
    data: {
      phoneNumber,
    },
  })
}
```

```
// src/pages/login/login.vue
<button @tap="onGetphonenumberSimple">
  <text class="icon icon-phone">模拟快捷登录</text>
</button>

// 模块快捷登录
const onGetphonenumberSimple = async () => {
  const res = await postLoginWxMinSimpleAPI('13123456789')
  console.log(res)
  uni.showToast({ icon: 'success', title: '登录成功' })
}
```



登录模块 - 保存登录会员信息

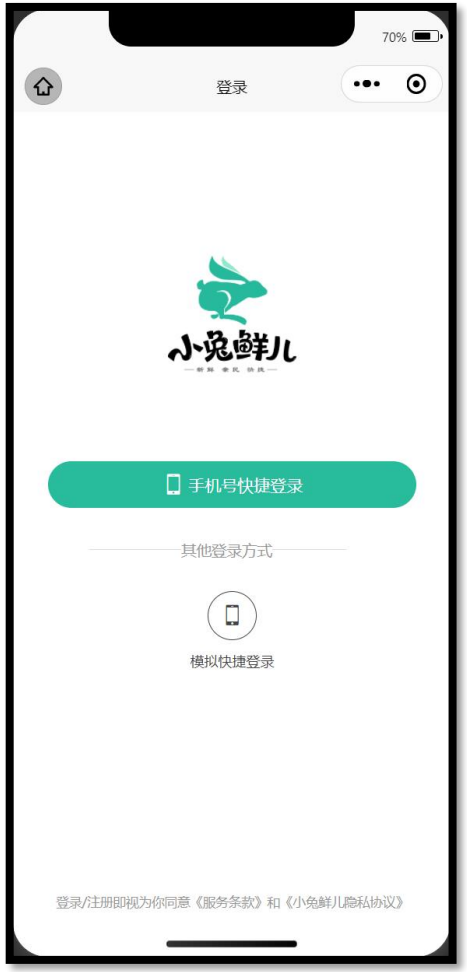
类型声明 → 状态管理 → 成功提示 → 页面跳转

```ts
// src/types/member.ts
/** 小程序登录 登录用户信息 */
export type LoginResult = {
  /** 账户名  */
  accout: string
  /** 头像  */
  avatar: string
  /** 用户ID */
  id: number
  /** 手机号 */
  mobile: string
  /** 昵称 */
  nickname: string
  /** 登录凭证 */
  token: string
}
```

```ts
// src/services/login.ts
/**
 * 小程序登录_内测版
 * @param phoneNumber 模拟手机号码
 */
export const postLoginWxMinSimpleAPI = (phoneNumber: string) => {
  return http<LoginResult>({
    method: 'POST',
    url: '/login/wxMin/simple',
    data: { phoneNumber },
  })
}
```

```ts
// src/stores/modules/member.ts
export const useMemberStore = defineStore('member', () => {
  // 会员信息
  const profile = ref<LoginResult>()
  // 保存会员信息，登录时使用
  const setProfile = (val: LoginResult) => {
    profile.value = val
  }
  …省略
}
```

```ts
// 模拟手机号码快捷登录
const onGetphonenumberSimple = async () => {
  const res = await postLoginWxMinSimpleAPI('13123456789')
  loginSuccess(res.result)
}

// 登录成功逻辑
const loginSuccess = (profile: LoginResult) => {
  // 保存登录信息到 Store 中
  const memberStore = useMemberStore()
  memberStore.setProfile(profile)
  // 登录成功提示
  uni.showToast({ icon: 'success', title: '登录成功' })
  // 页面跳转
  setTimeout(() => {
    uni.switchTab({ url: '/pages/my/my' })
  }, 500)
}
```

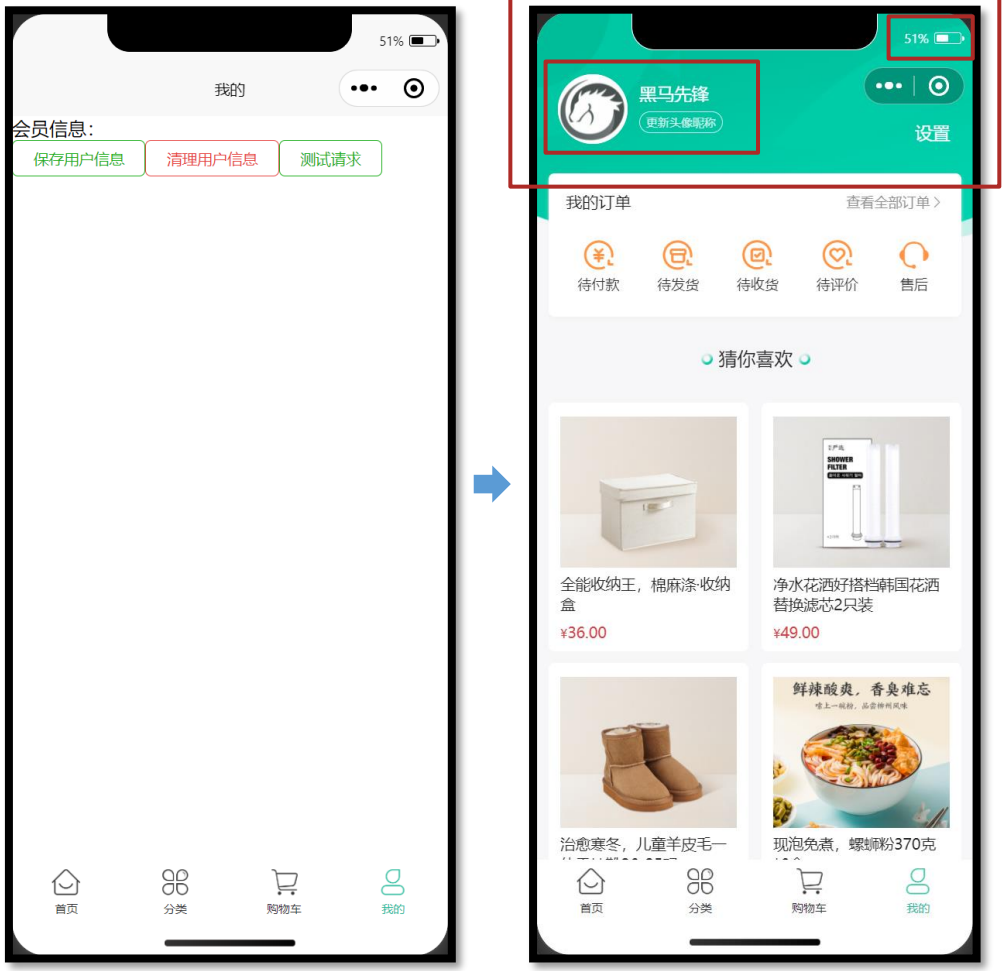


会员中心 - 会员信息展示

静态结构 → 自定义导航 → 渲染会员信息

```
// src/pages.json
{
  "path": "pages/my/my",
  "style": {
    "navigationStyle": "custom",
    "navigationBarTextStyle": "white",
    "navigationBarTitleText": "我的"
  }
},
```

```
// src/pages/my/my.vue
const memberStore = useMemberStore()

<view class="overview" v-if="memberStore.profile">
  <navigator url="/pagesMember/profile/profile" hover-class="none">
    <image class="avatar" :src="memberStore.profile.avatar"></image>
  </navigator>
  <view class="meta">
    <view class="nickname">
      {{ memberStore.profile.nickname || memberStore.profile.account }}
    </view>
    <navigator class="extra" url="/pagesMember/profile/profile">
      <text class="update">更新头像昵称</text>
    </navigator>
  </view>
</view>
```



会员中心 - 猜你喜欢分页加载

获取组件实例 → 滚动触底事件 → 加载分页数据 → 封装组合式函数

组合式函数官方文档

```ts
// src/composables/index.ts
import type { XtxGuessInstance } from '@/types/components'
import { ref } from 'vue'

/** 猜你喜欢组合式函数 */
export const useGuessList = () => {
  // 获取猜你喜欢组件实例
  const guessRef = ref<XtxGuessInstance>()
  // 滚动触底获取分页数据
  const onScrolltolower = () => {
    guessRef.value?.getMore()
  }
  // 返回封装的ref和函数
  return { guessRef, onScrolltolower }
}
```

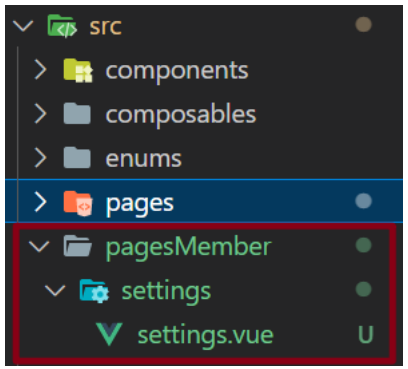
```ts
// 调用猜你喜欢组合式函数
const { guessRef, onScrolltolower } = useGuessList()

<!-- 猜你喜欢组件 -->
<XtxGuess ref="guessRef" />

<!- 滚动容器 -->
<scroll-view @scrolltolower="onScrolltolower"></scroll-view>
```

会员中心 - 设置页分包和预下载

小程序分包：将小程序的**代码分割成多个部分**，分别打包成多个小程序包，**减少**小程序的**加载时间**，提高用户体验。

分包预下载：在进入小程序某个页面时，由框架**自动预下载**可能需要的分包，**提升**进入后续分包页面时的**启动速度**。

新建分包页面 → 配置分包预下载



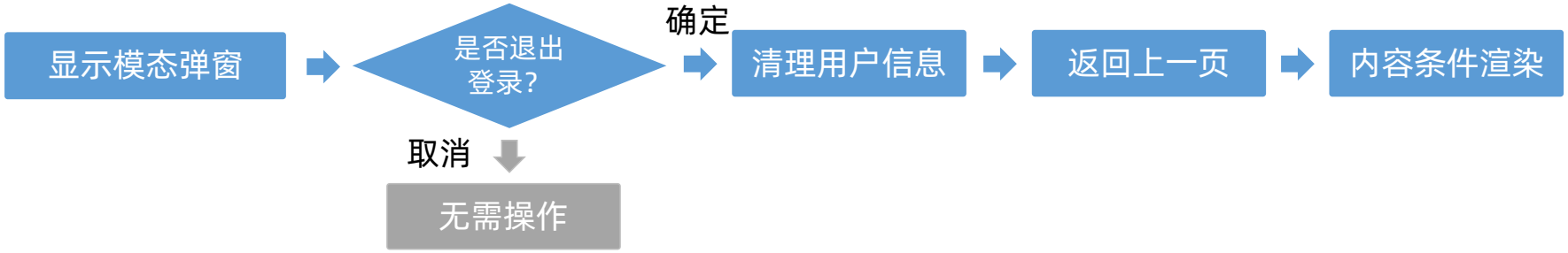

```
// 分包加载规则
"subPackages": [
  {
    // 子包的根目录
    "root": "pagesMember",
    // 子包页面路径和窗口表现
    "pages": [
      {
        "path": "settings/settings",
        "style": {
          "navigationBarTitleText": "设置"
        }
      }
    ]
  }
],
```

```
// 分包预下载规则
"preloadRule": {
  // 进入 我的 页面时，预下载分包
  "pages/my/my": {
    "network": "all",
    "packages": ["pagesMember"]
  }
}
```

官方文档

经验：分包一般是按照项目的**业务模块划分**，如会员模块分包，订单模块分包等。

会员中心 - 退出登录

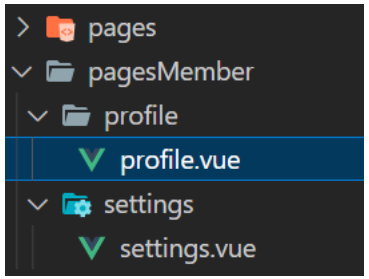



```
const memberStore = useMemberStore()
// 退出登录
const onLogout = () => {
  // 显示模态框
  uni.showModal({
    content: '是否退出登录',
    success(res) {
      // 如果点击了确定
      if (res.confirm) {
        // 清理用户信息
        memberStore.clearProfile()
        // 返回上一页
        uni.navigateBack()
      }
    },
  })
}

<view @tap="onLogout" class="button">退出登录</view>
```

```
<view class="action" v-if="memberStore.profile">
  <view @tap="onLogout" class="button">退出登录</view>
</view>


<view class="list" v-if="memberStore.profile">
  <navigator url="./address/address">
    我的收货地址
  </navigator>
</view>
```

会员中心 - 个人信息页准备工作

新建分包页面 → 静态结构 → 自定义导航




```
<template>
  <view class="viewport">
    <!-- 导航栏 -->
    <view class="navbar">
      …省略
    </view>
    <!-- 头像 -->
    <view class="avatar">
      …省略
    </view>
    </view>
    <!-- 表单 -->
    <view class="form">
      …省略
    </view>
  </view>
</template>
```

会员中心 - 个人信息展示

封装API接口 → 初始化调用 → 定义类型 → 页面渲染

```ts
/** 获取个人信息 */
export const getMemberProfileAPI = () => {
  return http<ProfileDetail>({
    method: 'GET',
    url: '/member/profile',
  })
}
```

```ts
// 获取用户信息
const profile = ref<ProfileDetail>()
const getMemberProfileData = async () => {
  const res = await getMemberProfileAPI()
  profile.value = res.result
}

onLoad(() => {
  getMemberProfileData()
})
```

```ts
/** 基本用户信息 */
type BaseProfile = {
  id: number
  avatar: string
  account: string
  nickname?: string
}
/** 小程序登录 登录用户信息 */
export type LoginResult = BaseProfile & {
  mobile: string
  token: string
}
/** 用户信息详情 */
export type ProfileDetail = BaseProfile & {
  gender?: Gender
  birthday?: string
  fullLocation?: string
  profession?: string
}
export type Gender = '女' | '男'
```

会员中心 - 个人信息展示

封装API接口 → 初始化调用 → 定义类型 → 页面渲染

```
<view class="form-item">
  <text class="label">账号</text>
  <text class="account">{{ profile?.account }}</text>
</view>
<view class="form-item">
  <text class="label">昵称</text>
  <input class="input" placeholder="请填写昵称" :value="profile?.nickname" />
</view>
<view class="form-item">
  <text class="label">性别</text>
  <radio-group>
    <label class="radio">
      <radio value="男" color="#27ba9b" :checked="profile?.gender === '男'" />
      男
    </label>
    <label class="radio">
      <radio value="女" color="#27ba9b" :checked="profile?.gender === '女'" />
      女
    </label>
  </radio-group>
</view>
```

```
result:
  account: "Dawn99"
  avatar: "http://yjy-xiaotu
  birthday: null
  fullLocation: ""
  gender: null
  id: "1427824787677253633"
  nickname: null
  profession: null
```

```
<view class="form-item">
  <text class="label">出生日期</text>
  <picker
    class="picker"
    mode="date"
    :value="profile?.birthday"
    start="1900-01-01"
    :end="new Date()"
  >
    <view v-if="profile?.birthday">{{ profile?.birthday }}</view>
    <view class="placeholder" v-else>请选择日期</view>
  </picker>
</view>
<view class="form-item">
  <text class="label">城市</text>
  <picker :value="profile?.fullLocation?.split(' ')" mode="region">
    <view v-if="profile?.fullLocation">{{ profile?.fullLocation }}</view>
    <view class="placeholder" v-else>请选择城市</view>
  </picker>
</view>
<view class="form-item">
  <text class="label">职业</text>
  <input class="input" placeholder="请填写职业" :value="profile?.profession" />
</view>
</view>
```

温馨提示：新注册的用户信息是缺失的，**个人信息展示**可使用账号 **13123456789** ，**个人信息修改**的时候换成**自己手机号**。

会员中心 - 修改用户头像

调用拍照/选择图片 → 获取图片路径 → 上传文件 → 更新头像

```
// 修改头像
const onAvatarChange = () => {
  // 调用拍照/选择图片
  uni.chooseMedia({
    // 文件个数
    count: 1,
    // 文件类型
    mediaType: ['image'],
    success(res) {
      // 图片的本地文件路径
      const { tempFilePath } = res.tempFiles[0]
      // 上传文件
    },
  })
}
```

```
// 上传文件
uni.uploadFile({
  url: '/member/profile/avatar', // 接口地址
  name: 'file', // 接口参数
  filePath: tempFilePath, // 文件路径
  success(res) {
    if (res.statusCode === 200) {
      // 更新头像
      profile.value!.avatar = JSON.parse(res.data).result.avatar
      uni.showToast({ icon: 'success', title: '头像更新成功' })
    } else {
      uni.showToast({ icon: 'error', title: '出现错误' })
    }
  },
})
```

温馨提示：调用拍照/选择图片时，模拟器和手机端有差异，**模拟器只能选择图片。**

会员中心 - 修改用户昵称

封装API接口 → 定义参数类型 → 点击保存调用 → 成功提示

```ts
/** 修改个人信息-请求参数 */
export type ProfileParams = Pick<
  ProfileDetail,
  'nickname' | 'gender' | 'birthday' | 'profession'
> & {
  provinceCode?: string
  cityCode?: string
  countyCode?: string
}

/** 修改个人信息*/
export const putMemberProfileAPI = (data: ProfileParams) => {
  return http<ProfileDetail>({
    method: 'PUT',
    url: '/member/profile',
    data,
  })
}
```

```ts
// 获取个人信息，初始化空对象用于修改
// const profile = ref<ProfileDetail>()
const profile = ref({} as ProfileDetail)

// 提交表单
const onSubmit = async () => {
  // 修改个人信息
  const res = await putMemberProfileAPI({
    nickname: profile.value.nickname,
  })
  // 成功提示
  uni.showToast({ icon: 'success', title: '保存成功' })
}
```

```vue
<!-- 输入框 -->
<input placeholder="请填写昵称" v-model="profile.nickname" />
<!-- 提交按钮 -->
<button @tap="onSubmit" class="form-button">保 存</button>
```

会员中心 - 更新Store信息

修改头像 → 更新Store头像

修改昵称 → 更新Store昵称 → 返回上一页


```
// 获取Store
const memberStore = useMemberStore()

// 文件上传
uni.uploadFile({
  url: '/member/profile/avatar',
  name: 'file',
  filePath: tempFilePath,
  success: (res) => {
    if (res.statusCode === 200) {
      const avatar = JSON.parse(res.data).result.avatar
      profile.value!.avatar = avatar
      // 更新Store头像
      memberStore.profile!.avatar = avatar
      uni.showToast({ icon: 'success', title: '更新成功' })
    } else {
      uni.showToast({ icon: 'error', title: '出现错误' })
    }
  },
})
```

```
// 点击保存提交表单
const onSubmit = async () => {
  const res = await putMemberProfileAPI({
    nickname: profile.value.nickname,
  })
  // 更新Store昵称
  memberStore.profile!.nickname = res.result.nickname
  uni.showToast({ icon: 'success', title: '保存成功' })
  // 返回上一页
  setTimeout(() => {
    uni.navigateBack()
  }, 400)
}
```

温馨提示：如果没有历史记录，就不能返回上一页。

会员中心 - 修改性别

单选事件 → 获取性别 → 提交更新

```
// 性别修改
const onGenderChange: UniHelper.RadioGroupOnChange = (ev) => {
  profile.value.gender = ev.detail.value as Gender
}


// 点击保存提交表单
const onSubmit = async () => {
  const res = await putMemberProfileAPI({
    nickname: profile.value.nickname,
    gender: profile.value.gender,
  })
  // …省略
}
```

```
<radio-group @change="onGenderChange">
  <label class="radio">
    <radio value="男" color="#27ba9b" :checked="profile?.gender === '男'" />
    男
  </label>
  <label class="radio">
    <radio value="女" color="#27ba9b" :checked="profile?.gender === '女'" />
    女
  </label>
</radio-group>
```

会员中心 - 修改生日

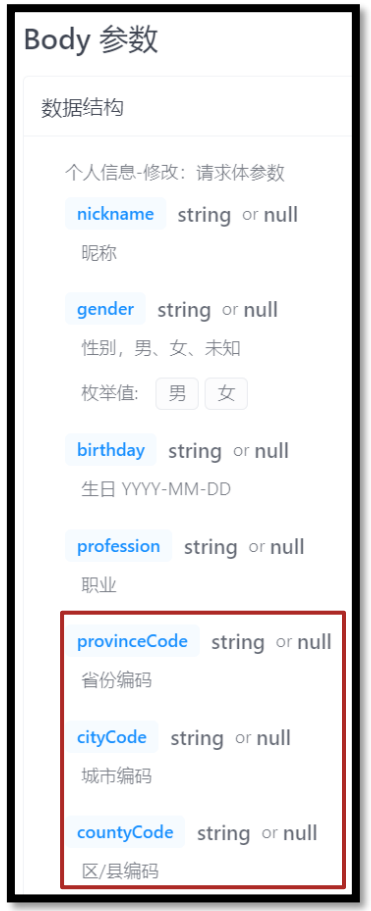
picker事件 → 获取日期 → 提交更新

```
// 修改生日
const onBirthdayChange: UniHelper.DatePickerOnChange = (ev) => {
  profile.value.birthday = ev.detail.value
}

// 点击保存提交表单
const onSubmit = async () => {
  const res = await putMemberProfileAPI({
    nickname: profile.value.nickname,
    gender: profile.value.gender,
    birthday: profile.value.birthday,
  })
  // …省略
}

<picker
  @change="onBirthdayChange"
  mode="date"
  :value="profile?.birthday"
  start="1900-01-01"
  :end="new Date()"
>
  <view v-if="profile?.birthday">{{ profile?.birthday }}</view>
  <view class="placeholder" v-else>请选择日期</view>
</picker>
```

会员中心 - 修改城市

picker事件 → 获取城市名字 → 界面更新(前端)
picker事件 → 获取城市编码 → 提交更新(后端)

```
// 修改城市
let fullLocationCode: [string, string, string] = ['', '', '']
const onFullLocationChange: UniHelper.RegionPickerOnChange = (ev) => {
  profile.value.fullLocation = ev.detail.value.join(' ')  // 前端界面更新
  fullLocationCode = ev.detail.code!                      // 后端数据更新

}
// 点击保存提交表单
const onSubmit = async () => {
  const { nickname, gender, birthday } = profile.value
  const [provinceCode, cityCode, countyCode] = fullLocationCode    // 省市区编码
  const res = await putMemberProfileAPI({
    nickname, gender, birthday,
    provinceCode, cityCode, countyCode,
  })
  // ...省略
}
<picker
  @change="onFullLocationChange"
  mode="region"
  :value="profile.fullLocation?.split(' ')"
>
  <view v-if="profile?.fullLocation">{{ profile?.fullLocation }}</view>
  <view class="placeholder" v-else>请选择城市</view>
</picker>
```



会员中心 - 个人信息页总结

- 静态结构
  - 分包
  - 自定义导航
- 上传头像
  - 拍照/选择图片
  - 上传文件
- 表单
  - input 双向绑定
  - radio 单选按钮
  - picker 选择器(日期/城市)
- 头像昵称信息同步
  - pinia 状态管理

传智教育旗下高端IT教育品牌