快应用与Kaldi 答辩

姓名：王旭

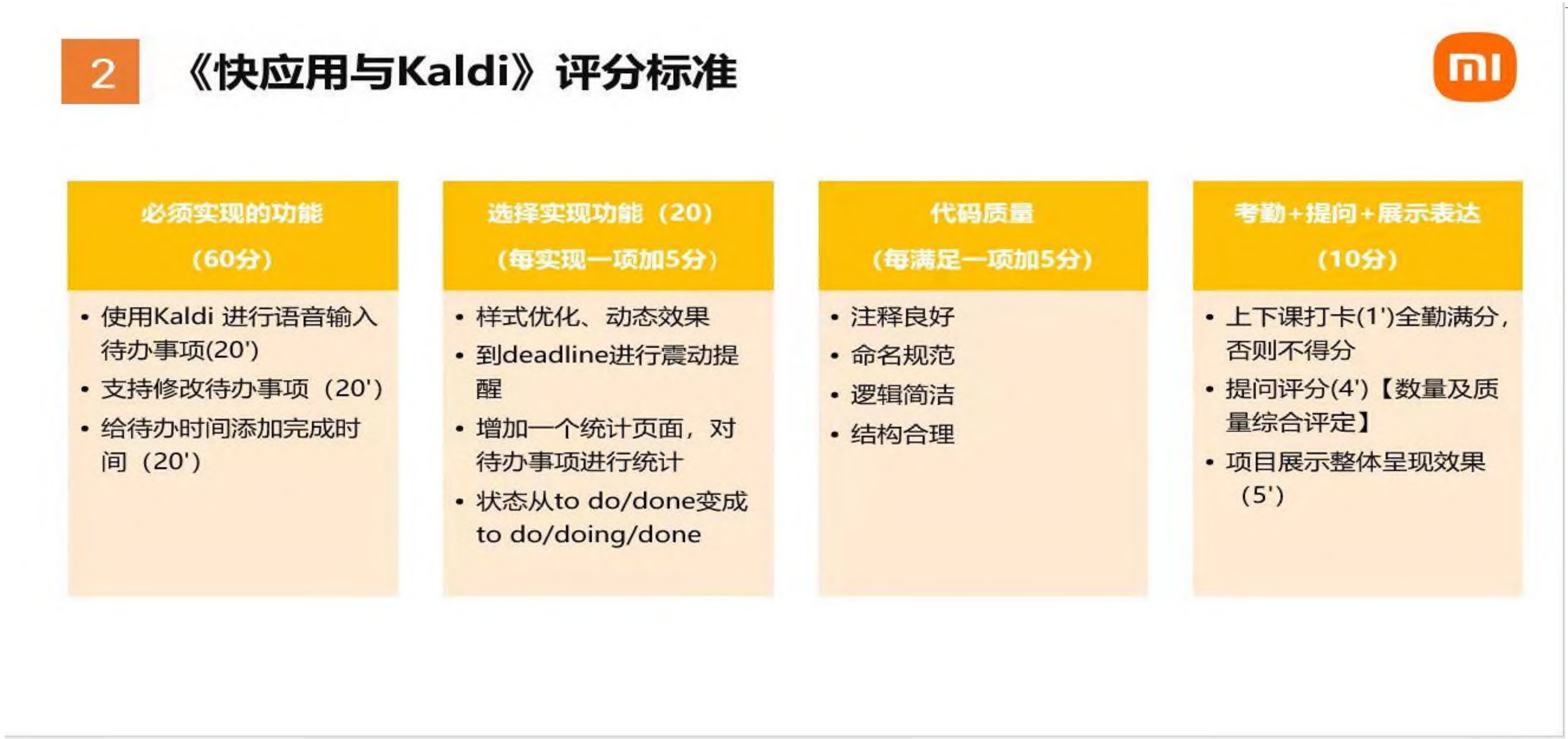
学号：3020001267

答辩序号：02

项目名称：todolist

应用包名：da

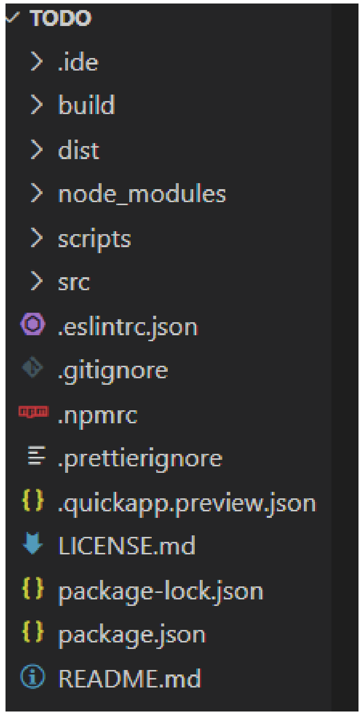
项目要求



2 《快应用与Kaldi》评分标准

必须实现的功能（60分）	选择实现功能（20）（每实现一项加5分）	代码质量（每满足一项加5分）	考勤+提问+展示表达（10分）
• 使用Kaldi 进行语音输入待办事项(20') • 支持修改待办事项（20'） • 给待办时间添加完成时间（20'）	• 样式优化、动态效果 • 到deadline进行震动提醒 • 增加一个统计页面，对待办事项进行统计 • 状态从to do/done变成to do/doing/done	• 注释良好 • 命名规范 • 逻辑简洁 • 结构合理	• 上下课打卡(1')全勤满分，否则不得分 • 提问评分(4')【数量及质量综合评定】 • 项目展示整体呈现效果（5'）

项目基本展示

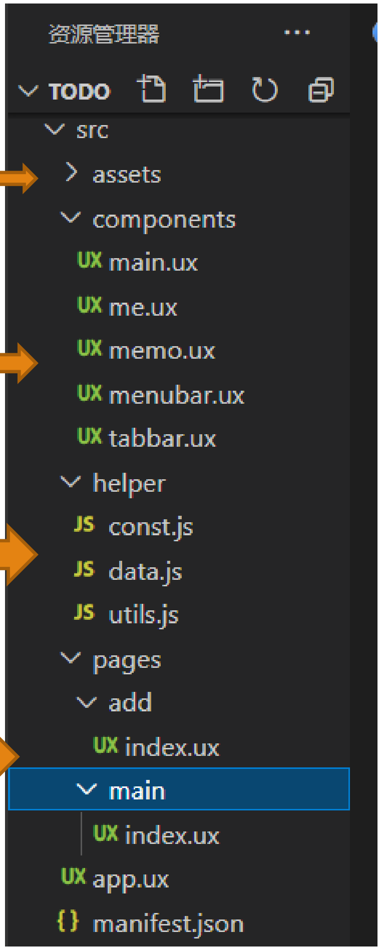

TODO
- .ide
- build
- dist
- node_modules
- scripts
- src
- .eslintrc.json
- .gitignore
- .npmrc
- .prettierignore
- .quickapp.preview.json
- LICENSE.md
- package-lock.json
- package.json
- README.md

图片等资源文件

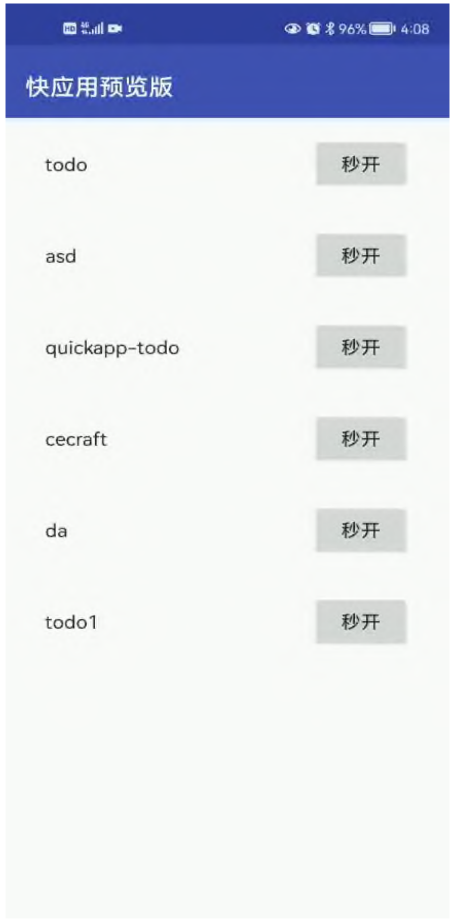
各种子组件

Javascript文件

主页面和添加页面


资源管理器

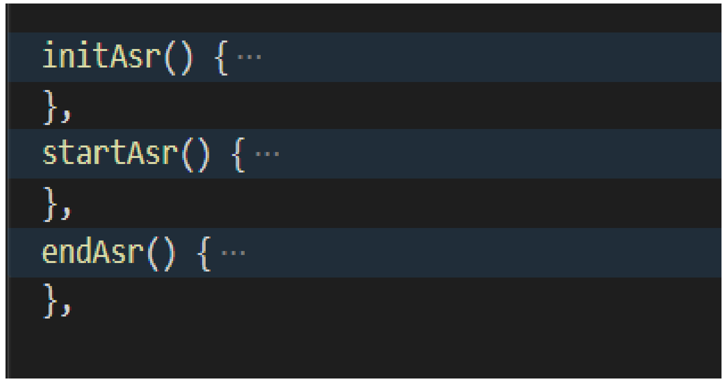
TODO
- src
  - assets
  - components
    - main.ux
    - me.ux
    - memo.ux
    - menubar.ux
    - tabbar.ux
  - helper
    - const.js
    - data.js
    - utils.js
  - pages
    - add
      - index.ux
    - main
      - index.ux
  - app.ux
  - manifest.json


快应用预览版

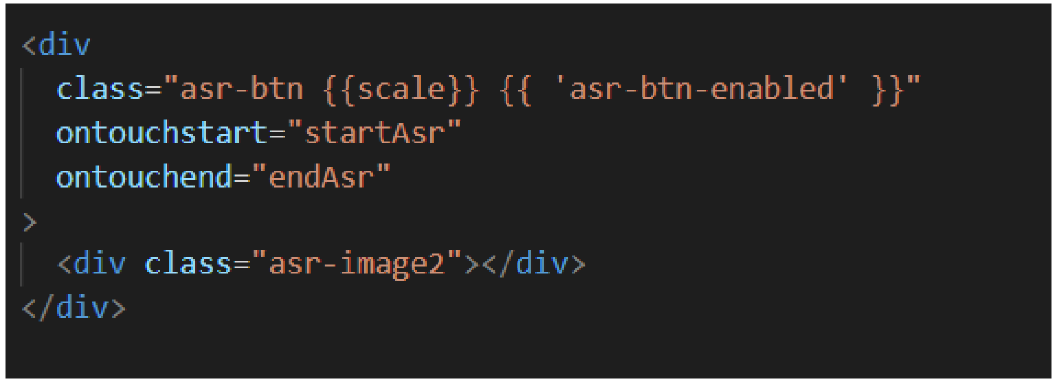
todo	秒开
asd	秒开
quickapp-todo	秒开
cecraft	秒开
da	秒开
todo1	秒开

必做任务一：kaldi语音输入

支撑代码

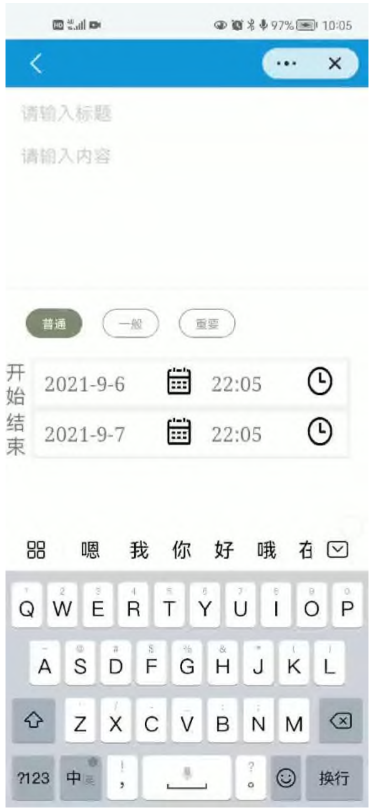


```
initAsr() {…
},
startAsr() {…
},
endAsr() {…
},
```



```
<div
  class="asr-btn {{scale}} {{ 'asr-btn-enabled' }}"
  ontouchstart="startAsr"
  ontouchend="endAsr"
>
  <div class="asr-image2"></div>
</div>
```

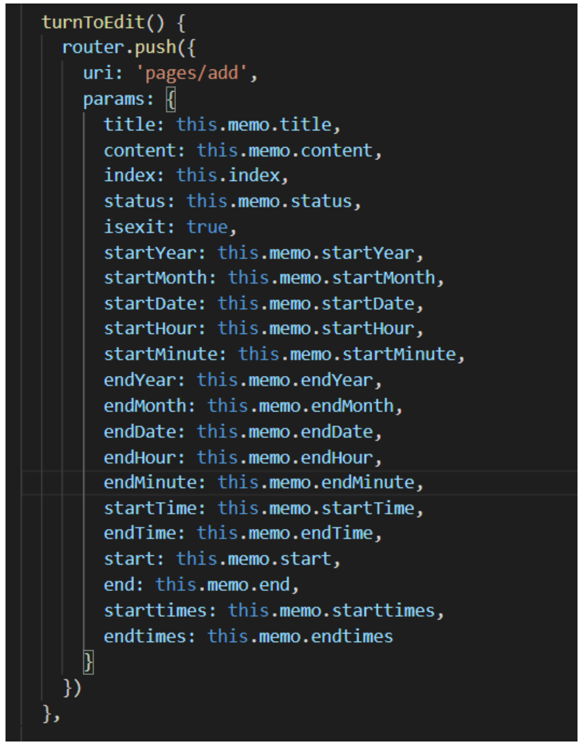
效果



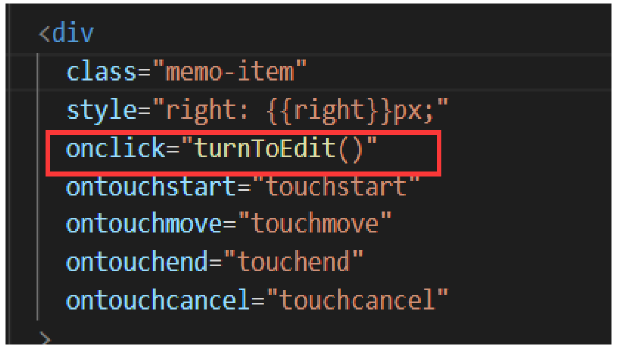
可以通过按住麦克风图标实现对于内容的语音输入

必做任务2：修改事件

支撑代码

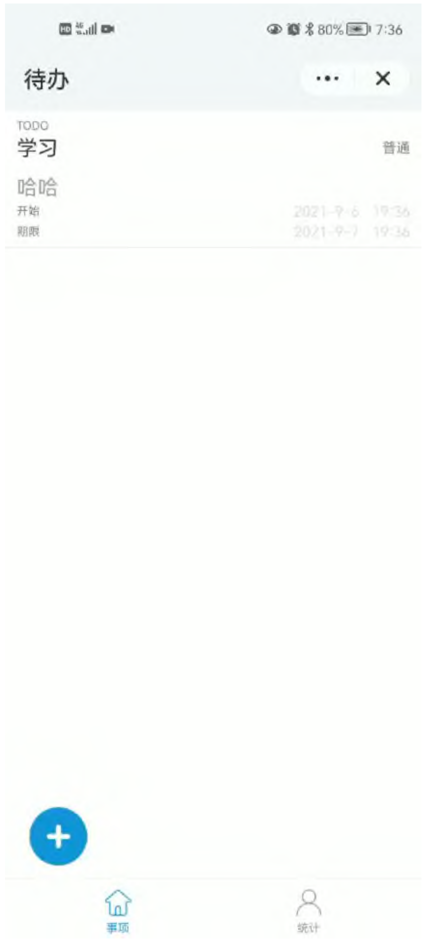


```
turnToEdit() {
  router.push({
    uri: 'pages/add',
    params: {
      title: this.memo.title,
      content: this.memo.content,
      index: this.index,
      status: this.memo.status,
      isexit: true,
      startYear: this.memo.startYear,
      startMonth: this.memo.startMonth,
      startDate: this.memo.startDate,
      startHour: this.memo.startHour,
      startMinute: this.memo.startMinute,
      endYear: this.memo.endYear,
      endMonth: this.memo.endMonth,
      endDate: this.memo.endDate,
      endHour: this.memo.endHour,
      endMinute: this.memo.endMinute,
      startTime: this.memo.startTime,
      endTime: this.memo.endTime,
      start: this.memo.start,
      end: this.memo.end,
      starttimes: this.memo.starttimes,
      endtimes: this.memo.endtimes
    }
  })
},
```



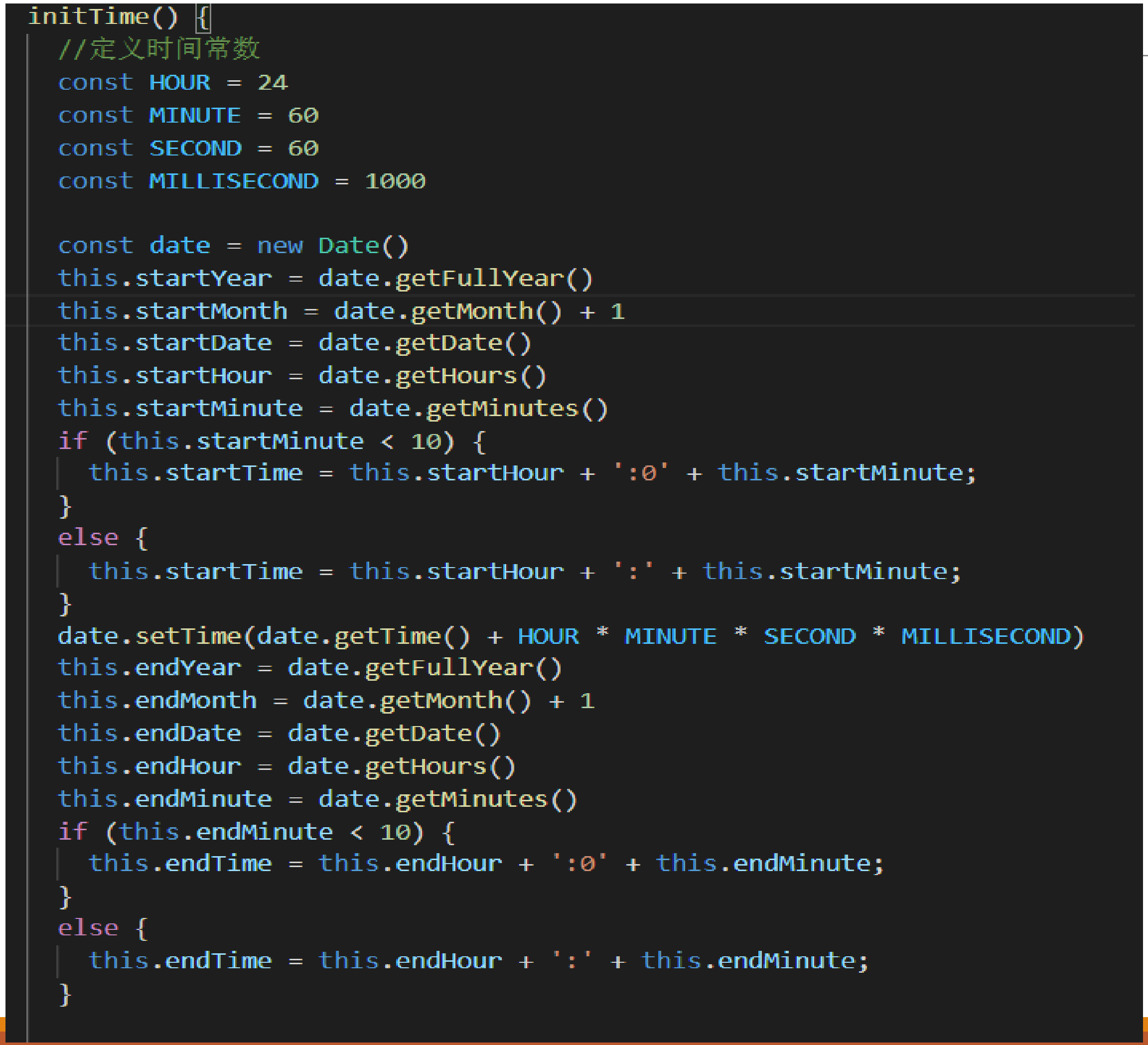
```
<div
  class="memo-item"
  style="right: {{right}}px;"
  onclick="turnToEdit()"
  ontouchstart="touchstart"
  ontouchmove="touchmove"
  ontouchend="touchend"
  ontouchcancel="touchcancel"
```

实现

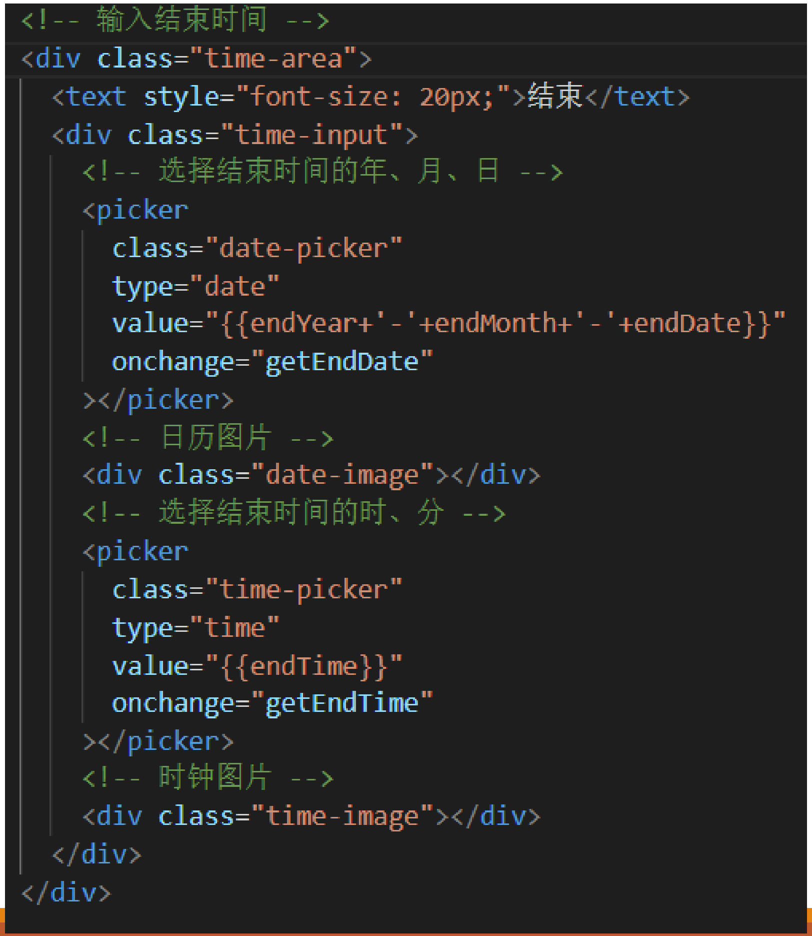


点击一下组件memo即可重新进入添加页面进行修改

必做任务3：添加时间



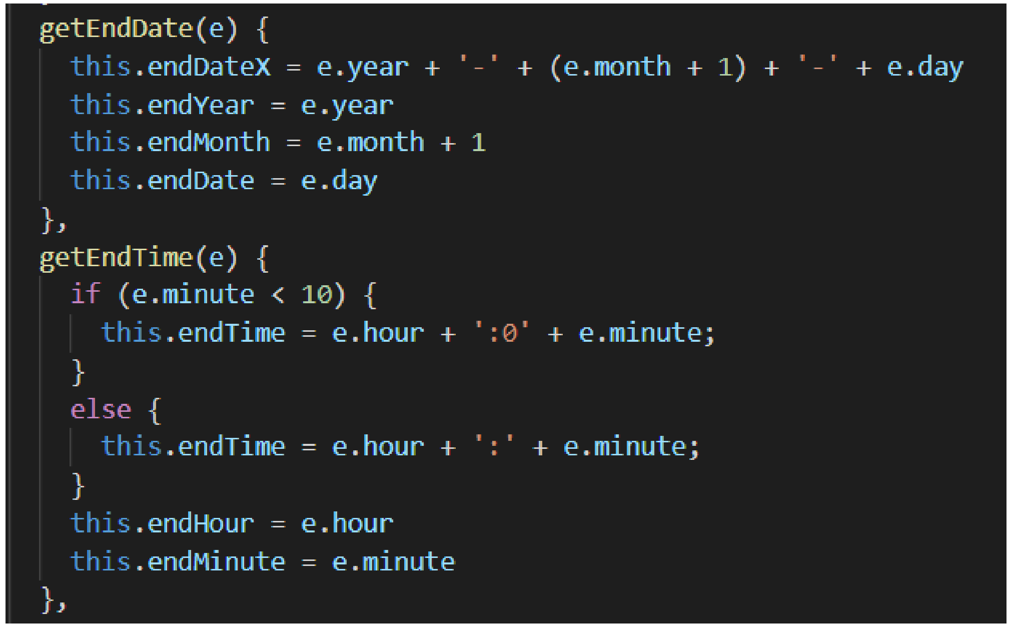
```
initTime() {
  //定义时间常数
  const HOUR = 24
  const MINUTE = 60
  const SECOND = 60
  const MILLISECOND = 1000

  const date = new Date()
  this.startYear = date.getFullYear()
  this.startMonth = date.getMonth() + 1
  this.startDate = date.getDate()
  this.startHour = date.getHours()
  this.startMinute = date.getMinutes()
  if (this.startMinute < 10) {
    this.startTime = this.startHour + ':0' + this.startMinute;
  }
  else {
    this.startTime = this.startHour + ':' + this.startMinute;
  }
  date.setTime(date.getTime() + HOUR * MINUTE * SECOND * MILLISECOND)
  this.endYear = date.getFullYear()
  this.endMonth = date.getMonth() + 1
  this.endDate = date.getDate()
  this.endHour = date.getHours()
  this.endMinute = date.getMinutes()
  if (this.endMinute < 10) {
    this.endTime = this.endHour + ':0' + this.endMinute;
  }
  else {
    this.endTime = this.endHour + ':' + this.endMinute;
  }
```



```
<!-- 输入结束时间 -->
<div class="time-area">
  <text style="font-size: 20px;">结束</text>
  <div class="time-input">
    <!-- 选择结束时间的年、月、日 -->
    <picker
      class="date-picker"
      type="date"
      value="{{endYear+'-'+endMonth+'-'+endDate}}"
      onchange="getEndDate"
    ></picker>
    <!-- 日历图片 -->
    <div class="date-image"></div>
    <!-- 选择结束时间的时、分 -->
    <picker
      class="time-picker"
      type="time"
      value="{{endTime}}"
      onchange="getEndTime"
    ></picker>
    <!-- 时钟图片 -->
    <div class="time-image"></div>
  </div>
</div>
```

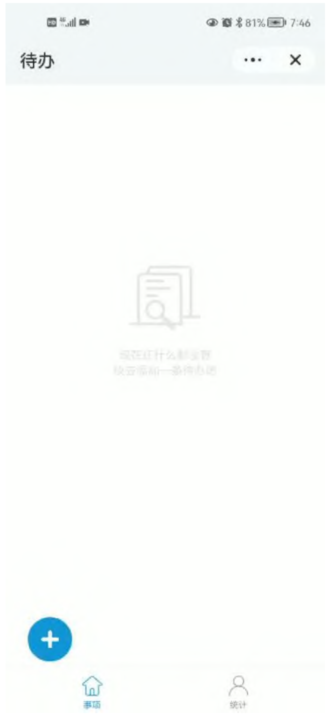
必做任务3：添加时间

支撑代码

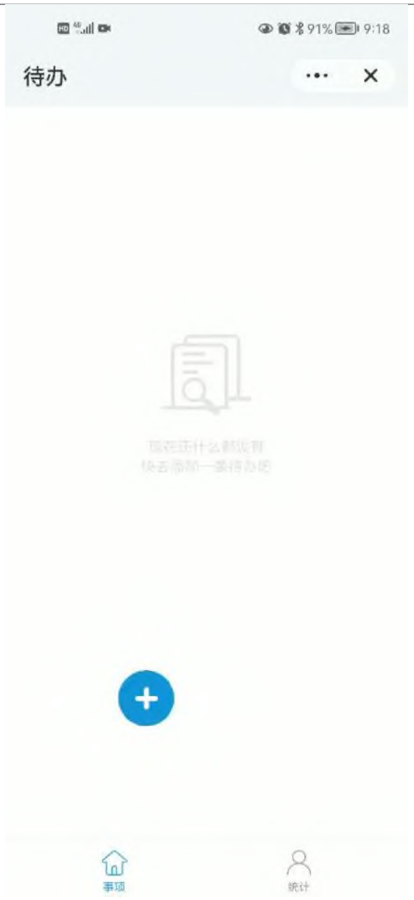


```
getEndDate(e) {
  this.endDateX = e.year + '-' + (e.month + 1) + '-' + e.day
  this.endYear = e.year
  this.endMonth = e.month + 1
  this.endDate = e.day
},
getEndTime(e) {
  if (e.minute < 10) {
    this.endTime = e.hour + ':0' + e.minute;
  }
  else {
    this.endTime = e.hour + ':' + e.minute;
  }
  this.endHour = e.hour
  this.endMinute = e.minute
},
```

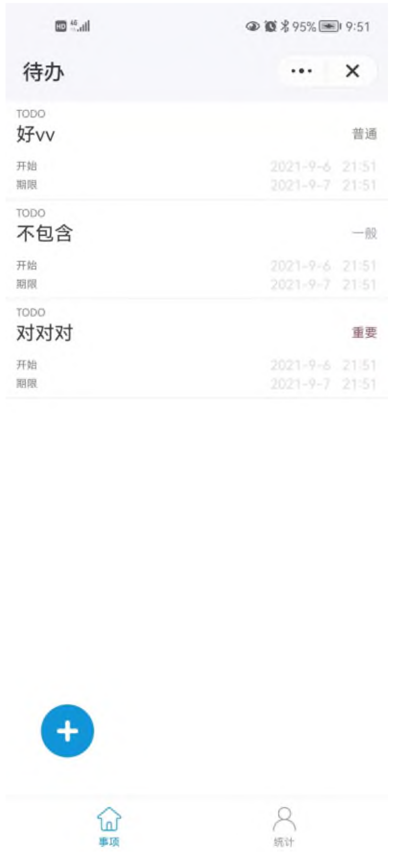
效果



数据存储演示



标签演示



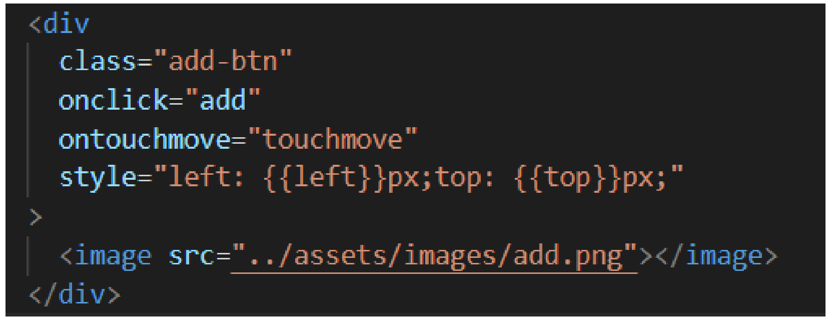
TODO
好vv 普通
开始 2021-9-6 21:51
期限 2021-9-7 21:51

TODO
不包含 一般
开始 2021-9-6 21:51
期限 2021-9-7 21:51

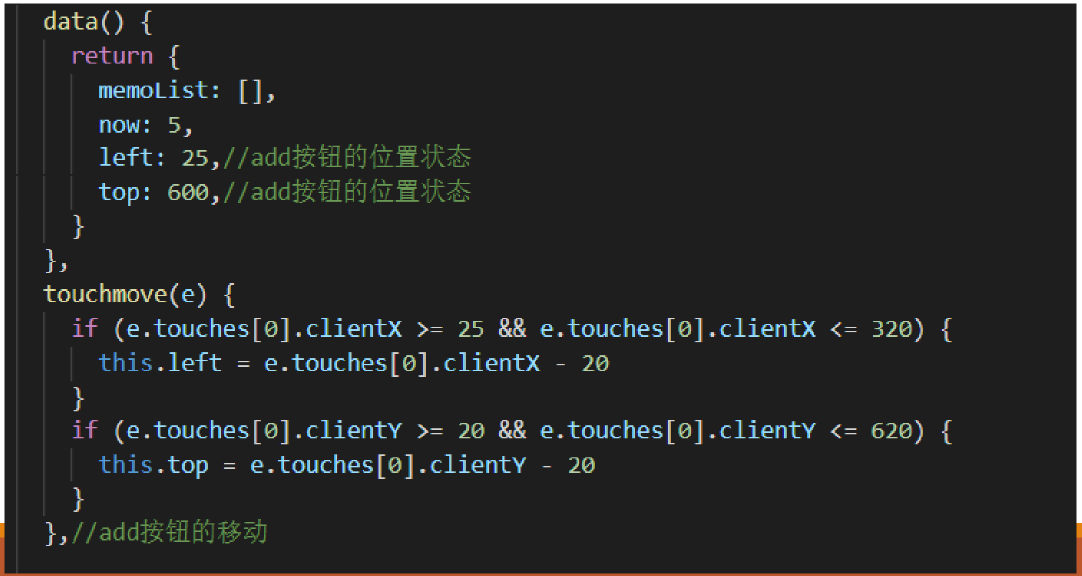
TODO
对对对 重要
开始 2021-9-6 21:51
期限 2021-9-7 21:51

选做任务一：动态效果（样式优化）

支撑代码

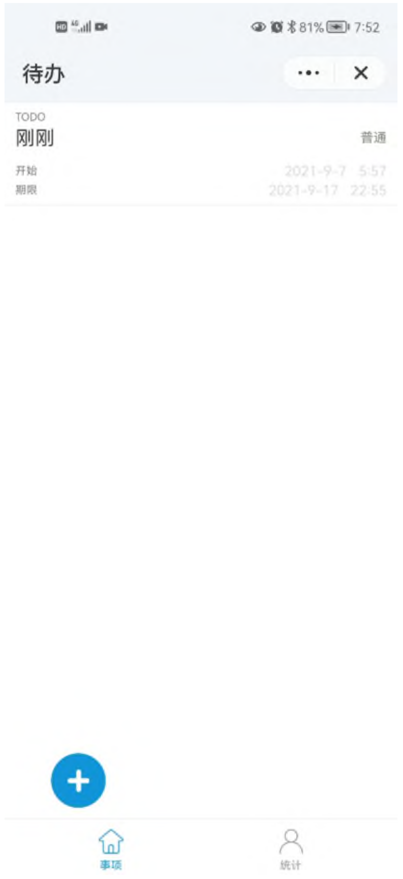


```
<div
  class="add-btn"
  onclick="add"
  ontouchmove="touchmove"
  style="left: {{left}}px;top: {{top}}px;"
>
  <image src="../assets/images/add.png"></image>
</div>
```



```
data() {
  return {
    memoList: [],
    now: 5,
    left: 25,//add按钮的位置状态
    top: 600,//add按钮的位置状态
  }
},
touchmove(e) {
  if (e.touches[0].clientX >= 25 && e.touches[0].clientX <= 320) {
    this.left = e.touches[0].clientX - 20
  }
  if (e.touches[0].clientY >= 20 && e.touches[0].clientY <= 620) {
    this.top = e.touches[0].clientY - 20
  }
},//add按钮的移动
```

效果

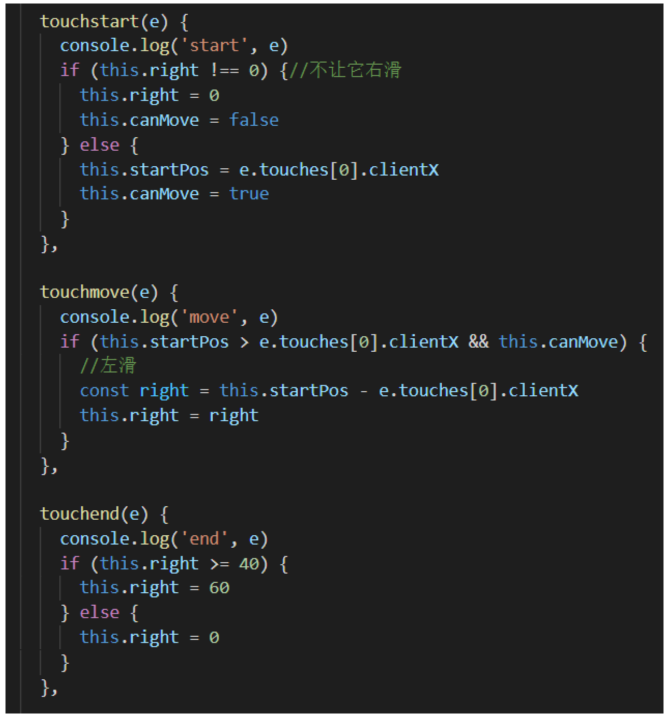


Add按钮的移动

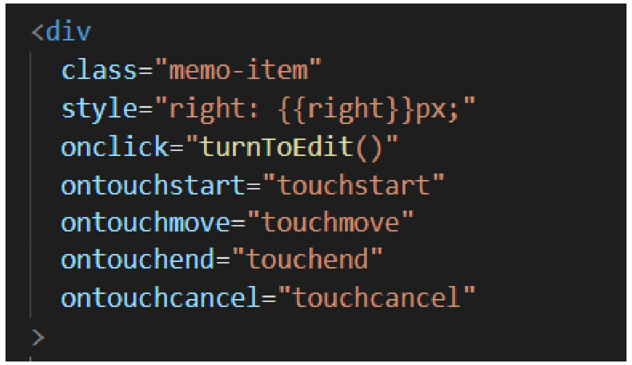
选做任务一：动态效果（样式优化）

代码实现

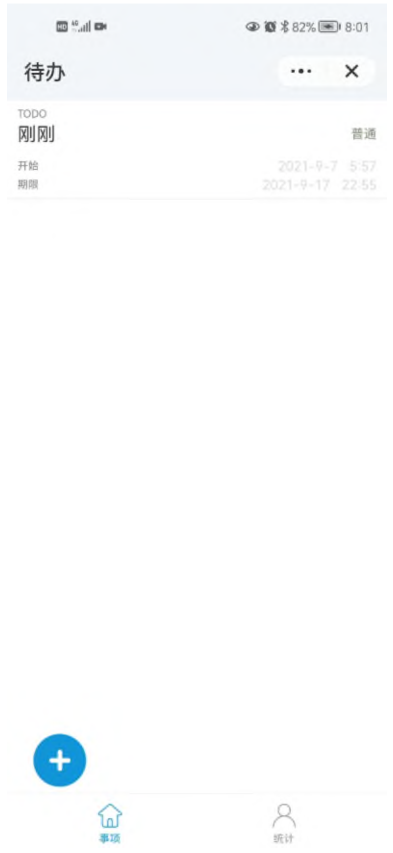
效果



```
touchstart(e) {
  console.log('start', e)
  if (this.right !== 0) {//不让它右滑
    this.right = 0
    this.canMove = false
  } else {
    this.startPos = e.touches[0].clientX
    this.canMove = true
  }
},

touchmove(e) {
  console.log('move', e)
  if (this.startPos > e.touches[0].clientX && this.canMove) {
    //左滑
    const right = this.startPos - e.touches[0].clientX
    this.right = right
  }
},

touchend(e) {
  console.log('end', e)
  if (this.right >= 40) {
    this.right = 60
  } else {
    this.right = 0
  }
},
```



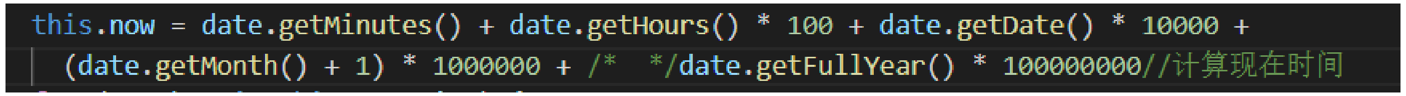
```
<div
  class="memo-item"
  style="right: {{right}}px;"
  onclick="turnToEdit()"
  ontouchstart="touchstart"
  ontouchmove="touchmove"
  ontouchend="touchend"
  ontouchcancel="touchcancel"
>
```



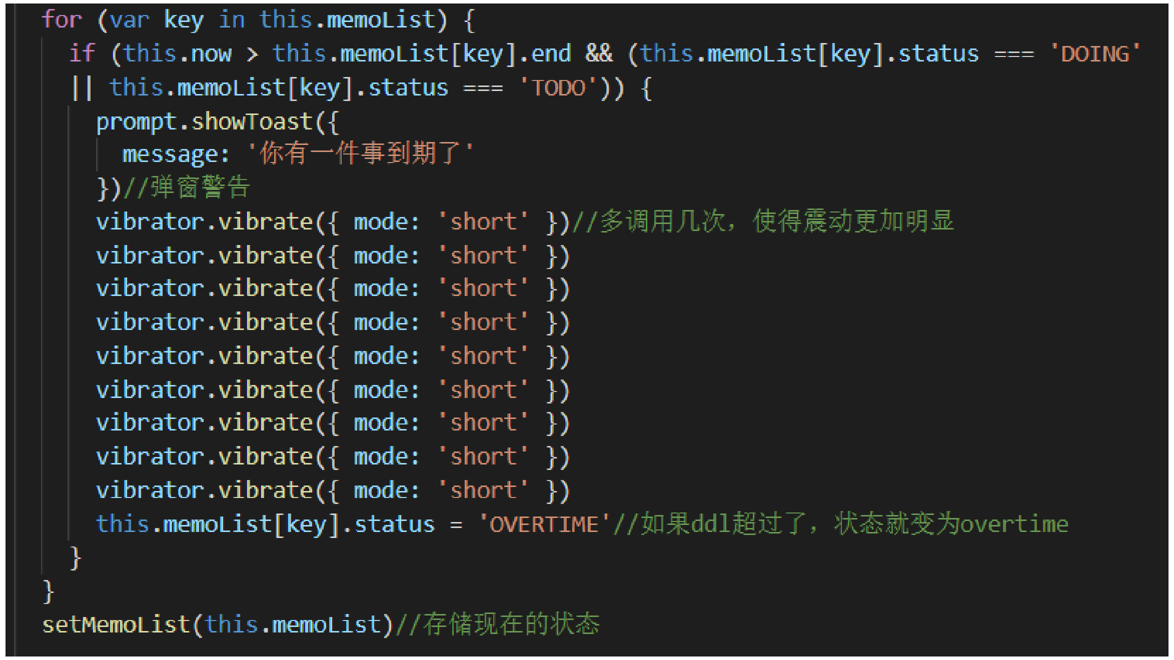
Memo组件的左移来实现状态的改变

选做任务二：DDL震动提醒

代码实现



```
this.now = date.getMinutes() + date.getHours() * 100 + date.getDate() * 10000 +
  (date.getMonth() + 1) * 1000000 + /*  */date.getFullYear() * 100000000//计算现在时间
```

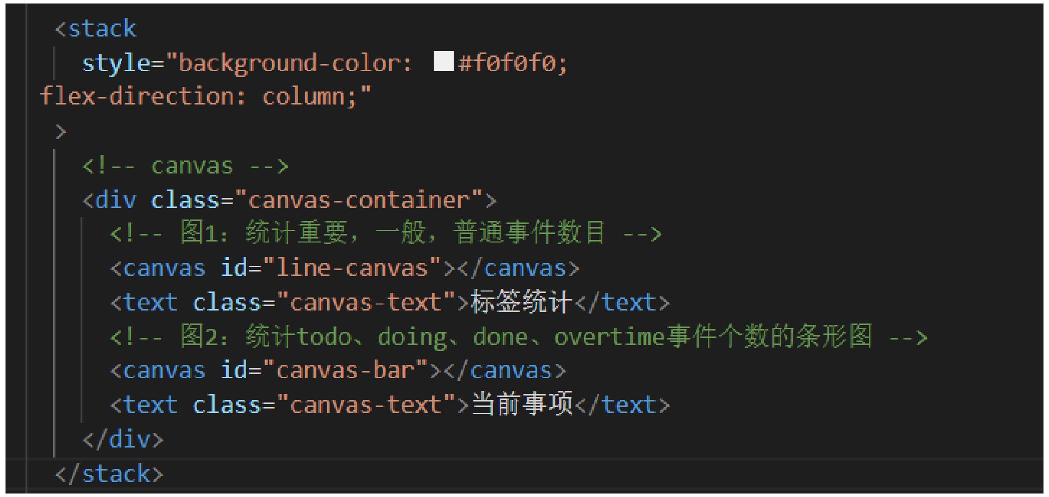


```
for (var key in this.memoList) {
  if (this.now > this.memoList[key].end && (this.memoList[key].status === 'DOING'
  || this.memoList[key].status === 'TODO')) {
    prompt.showToast({
      message: '你有一件事到期了'
    })//弹窗警告
    vibrator.vibrate({ mode: 'short' })//多调用几次，使得震动更加明显
    vibrator.vibrate({ mode: 'short' })
    vibrator.vibrate({ mode: 'short' })
    vibrator.vibrate({ mode: 'short' })
    vibrator.vibrate({ mode: 'short' })
    vibrator.vibrate({ mode: 'short' })
    vibrator.vibrate({ mode: 'short' })
    vibrator.vibrate({ mode: 'short' })
    vibrator.vibrate({ mode: 'short' })
    this.memoList[key].status = 'OVERTIME'//如果ddl超过了，状态就变为overtime
  }
}
setMemoList(this.memoList)//存储现在的状态
```

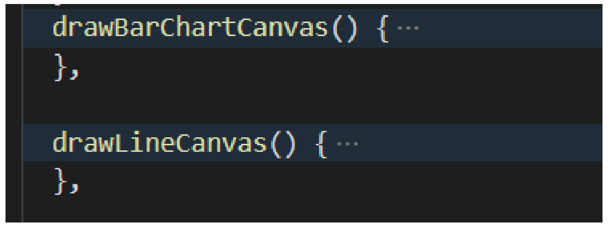
效果

选做任务三：统计页面

代码实现

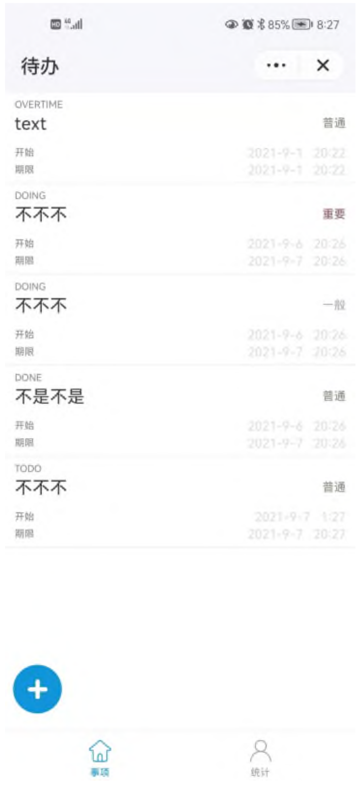


```
<stack
  style="background-color: ■#f0f0f0;
flex-direction: column;"
>
  <!-- canvas -->
  <div class="canvas-container">
    <!-- 图1：统计重要，一般，普通事件数目 -->
    <canvas id="line-canvas"></canvas>
    <text class="canvas-text">标签统计</text>
    <!-- 图2：统计todo、doing、done、overtime事件个数的条形图 -->
    <canvas id="canvas-bar"></canvas>
    <text class="canvas-text">当前事项</text>
  </div>
</stack>
```



```
drawBarChartCanvas() {…
},

drawLineCanvas() {…
},
```

效果



待办

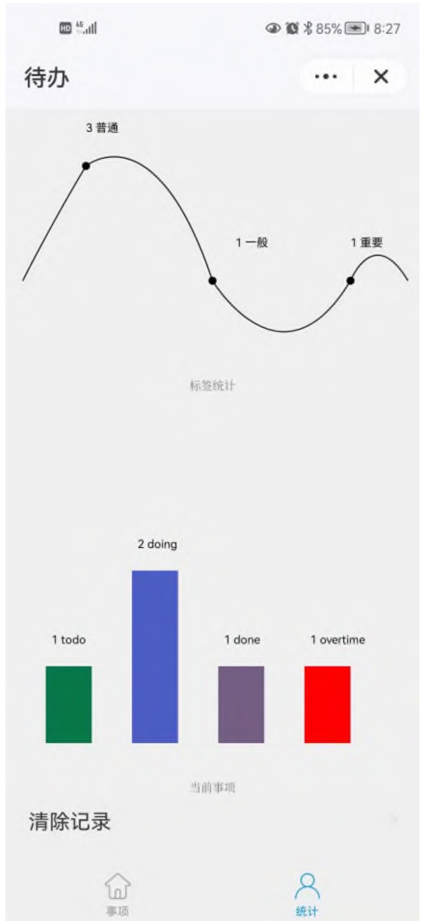
OVERTIME
text 普通
开始 2021-9-1 20:22
期限 2021-9-1 20:22

DOING
不不不 重要
开始 2021-9-6 20:26
期限 2021-9-7 20:26

DOING
不不不 一般
开始 2021-9-6 20:26
期限 2021-9-7 20:26

DONE
不是不是 普通
开始 2021-9-6 20:26
期限 2021-9-7 20:26

TODO
不不不 普通
开始 2021-9-7 1:27
期限 2021-9-7 20:27

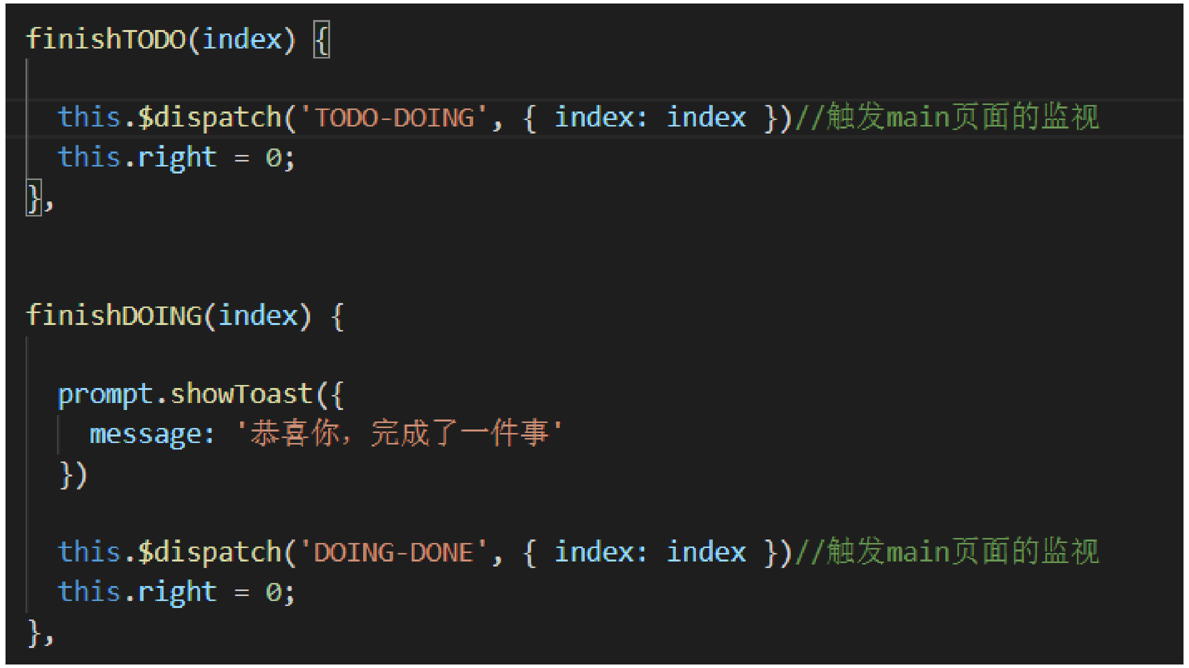


曲线统计标签（事件重要性）

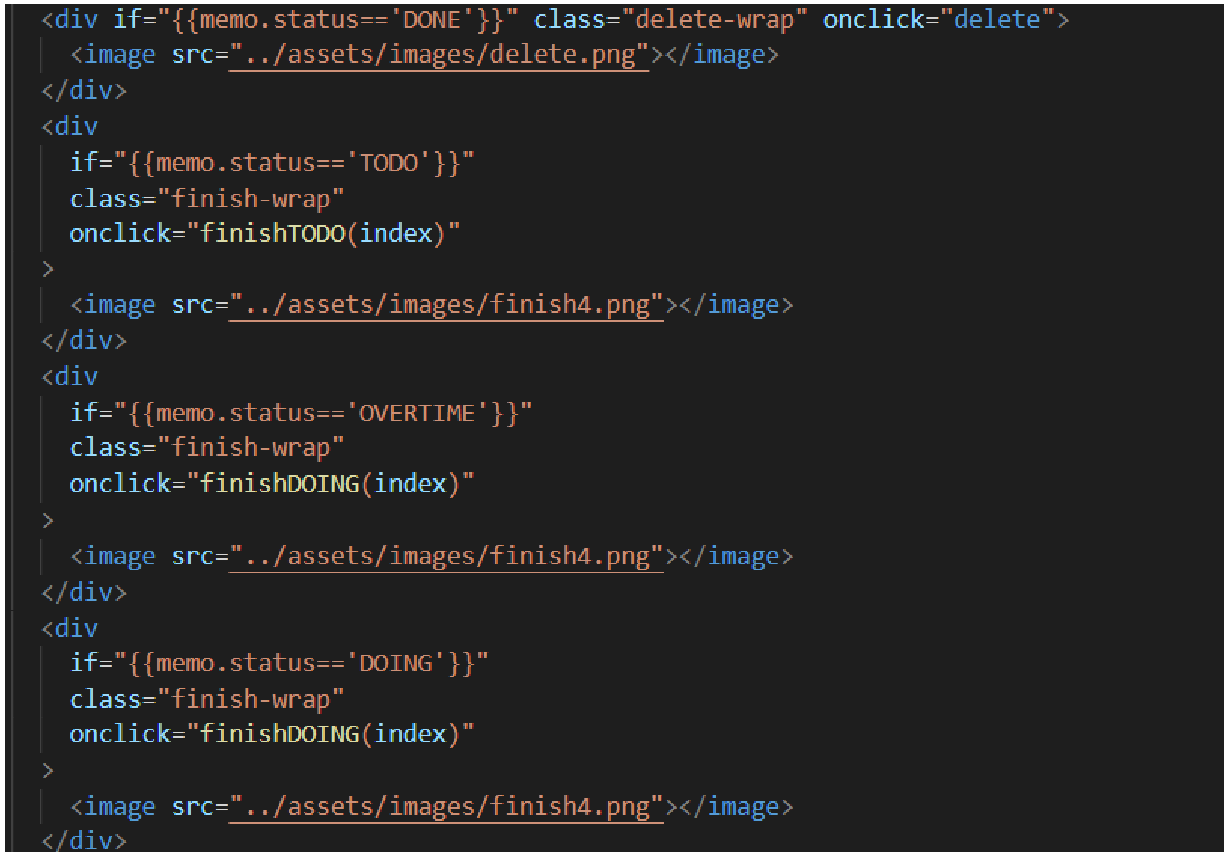
条形图统计事件状态数量

选做任务四：todo/doing/done(/overtime)状态

支撑代码

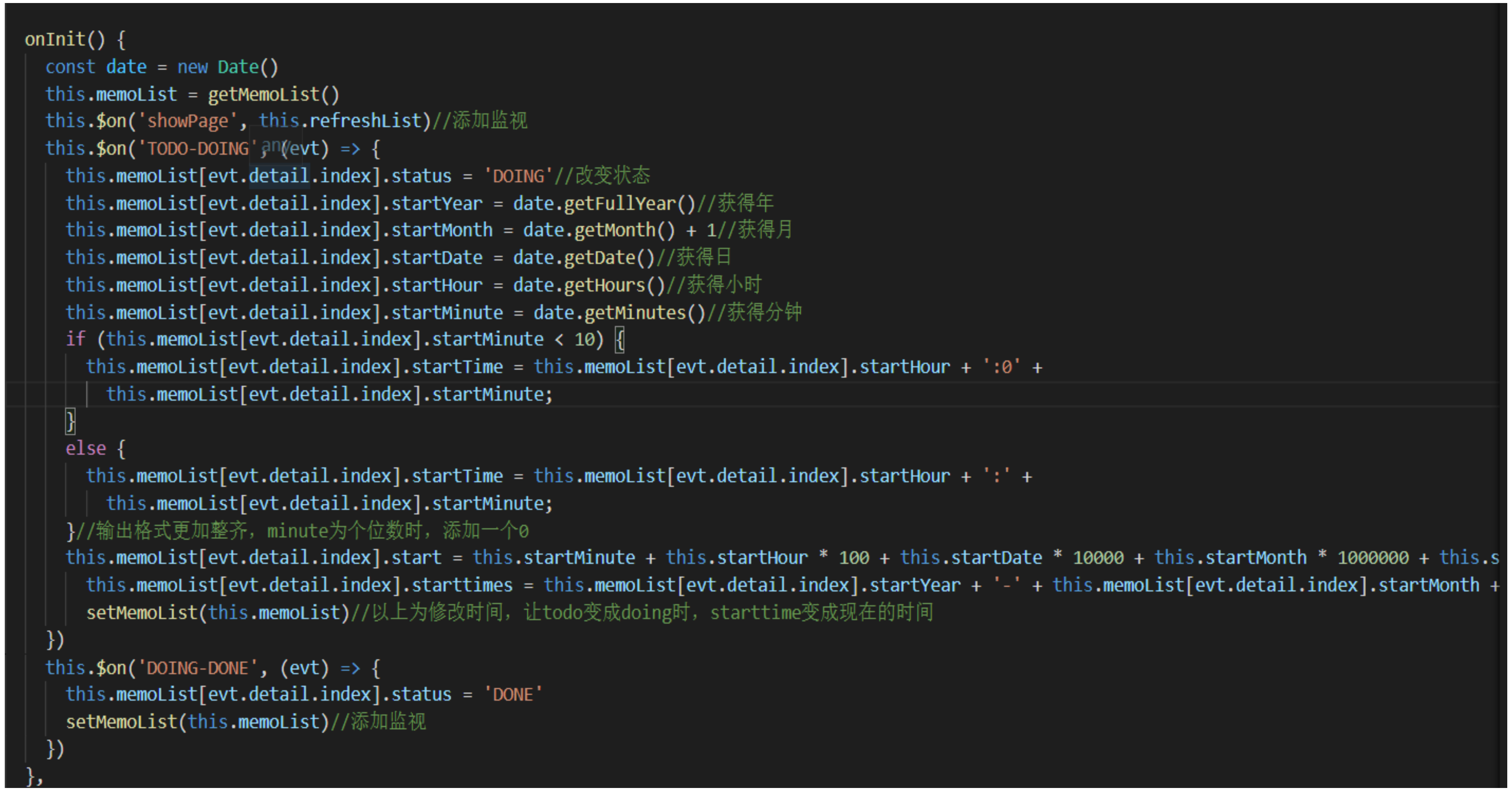


```
finishTODO(index) {

  this.$dispatch('TODO-DOING', { index: index })//触发main页面的监视
  this.right = 0;
},


finishDOING(index) {

  prompt.showToast({
    message: '恭喜你，完成了一件事'
  })

  this.$dispatch('DOING-DONE', { index: index })//触发main页面的监视
  this.right = 0;
},
```



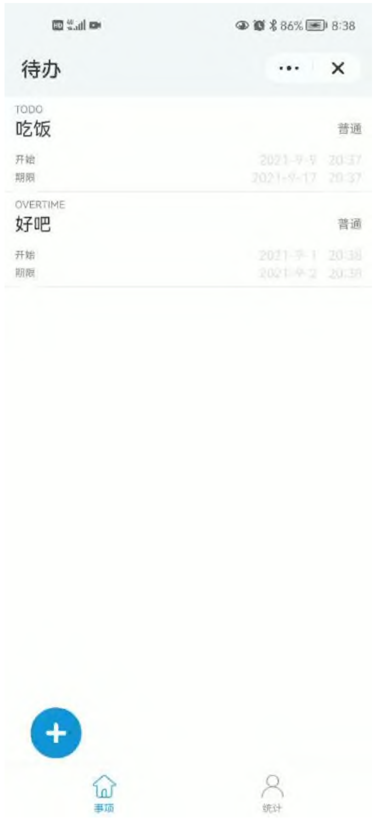
```
<div if="{{memo.status=='DONE'}}" class="delete-wrap" onclick="delete">
  <image src="../assets/images/delete.png"></image>
</div>
<div
  if="{{memo.status=='TODO'}}"
  class="finish-wrap"
  onclick="finishTODO(index)"
>
  <image src="../assets/images/finish4.png"></image>
</div>
<div
  if="{{memo.status=='OVERTIME'}}"
  class="finish-wrap"
  onclick="finishDOING(index)"
>
  <image src="../assets/images/finish4.png"></image>
</div>
<div
  if="{{memo.status=='DOING'}}"
  class="finish-wrap"
  onclick="finishDOING(index)"
>
  <image src="../assets/images/finish4.png"></image>
</div>
```

选做任务四：todo/doing/done状态



```
onInit() {
  const date = new Date()
  this.memoList = getMemoList()
  this.$on('showPage', this.refreshList)//添加监视
  this.$on('TODO-DOING', (evt) => {
    this.memoList[evt.detail.index].status = 'DOING'//改变状态
    this.memoList[evt.detail.index].startYear = date.getFullYear()//获得年
    this.memoList[evt.detail.index].startMonth = date.getMonth() + 1//获得月
    this.memoList[evt.detail.index].startDate = date.getDate()//获得日
    this.memoList[evt.detail.index].startHour = date.getHours()//获得小时
    this.memoList[evt.detail.index].startMinute = date.getMinutes()//获得分钟
    if (this.memoList[evt.detail.index].startMinute < 10) {
      this.memoList[evt.detail.index].startTime = this.memoList[evt.detail.index].startHour + ':0' +
        this.memoList[evt.detail.index].startMinute;
    }
    else {
      this.memoList[evt.detail.index].startTime = this.memoList[evt.detail.index].startHour + ':' +
        this.memoList[evt.detail.index].startMinute;
    }//输出格式更加整齐，minute为个位数时，添加一个0
    this.memoList[evt.detail.index].start = this.startMinute + this.startHour * 100 + this.startDate * 10000 + this.startMonth * 1000000 + this.s
      this.memoList[evt.detail.index].starttimes = this.memoList[evt.detail.index].startYear + '-' + this.memoList[evt.detail.index].startMonth +
      setMemoList(this.memoList)//以上为修改时间，让todo变成doing时，starttime变成现在的时间
  })
  this.$on('DOING-DONE', (evt) => {
    this.memoList[evt.detail.index].status = 'DONE'
    setMemoList(this.memoList)//添加监视
  })
},
```

效果



8:38

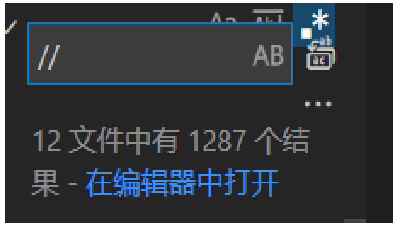
待办

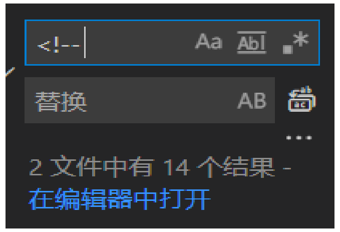
TODO
吃饭 普通
开始 2021-9-9 20:37
期限 2021-9-17 20:37

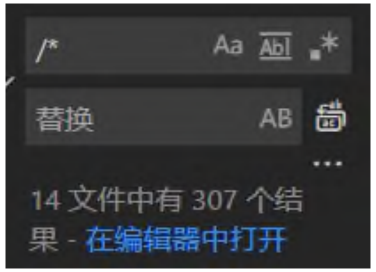
OVERTIME
好吧 普通
开始 2021-9-1 20:38
期限 2021-9-2 20:38

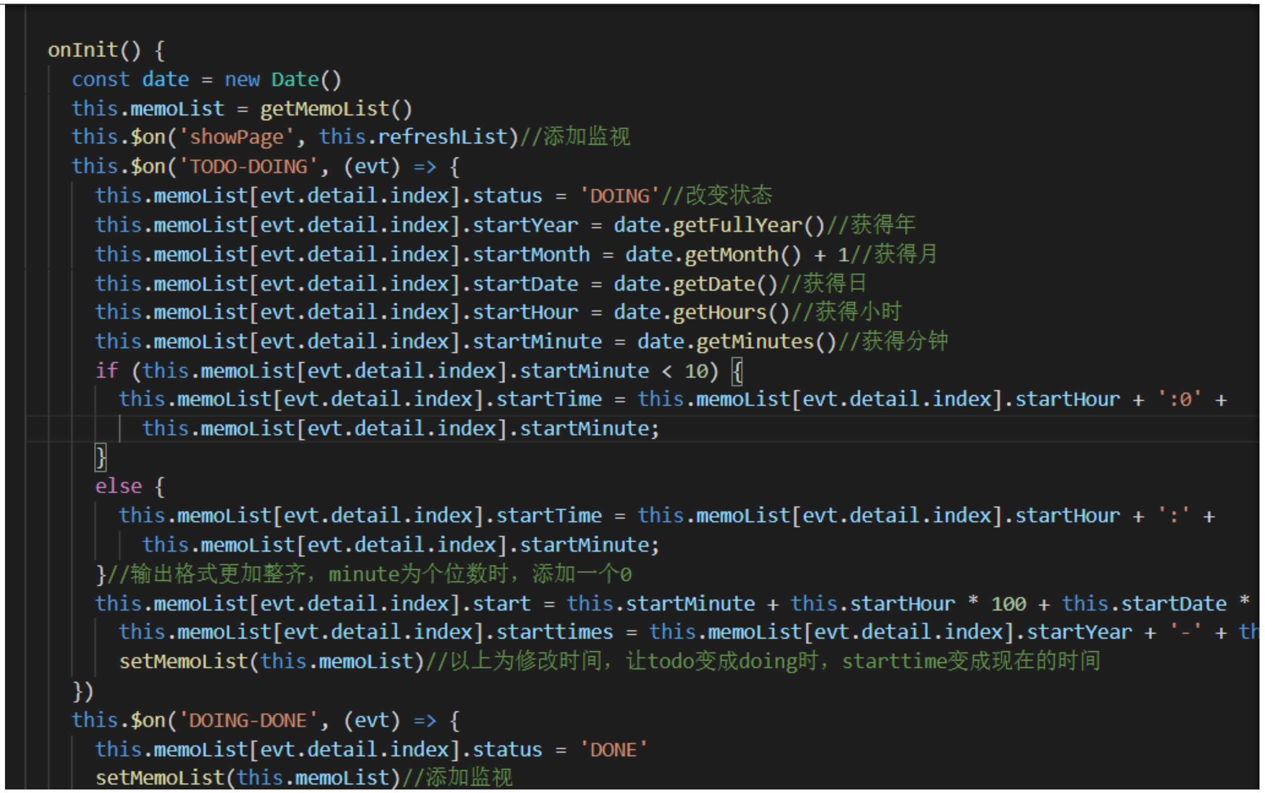
代码质量

注释良好


```
//
12 文件中有 1287 个结果 - 在编辑器中打开
```

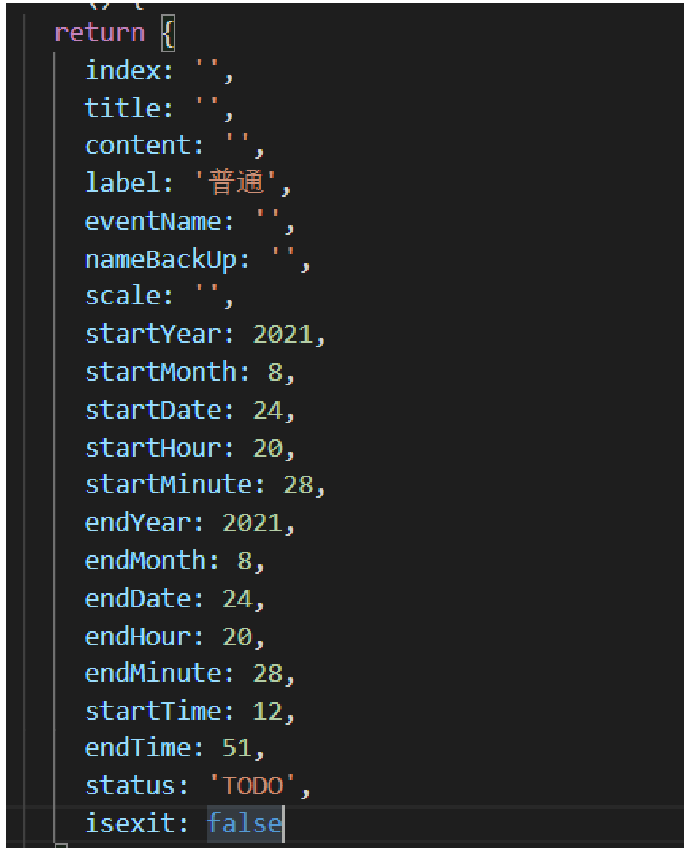

```
<!--
替换
2 文件中有 14 个结果 - 在编辑器中打开
```


```
/*
替换
14 文件中有 307 个结果 - 在编辑器中打开
```


```
onInit() {
  const date = new Date()
  this.memoList = getMemoList()
  this.$on('showPage', this.refreshList)//添加监视
  this.$on('TODO-DOING', (evt) => {
    this.memoList[evt.detail.index].status = 'DOING'//改变状态
    this.memoList[evt.detail.index].startYear = date.getFullYear()//获得年
    this.memoList[evt.detail.index].startMonth = date.getMonth() + 1//获得月
    this.memoList[evt.detail.index].startDate = date.getDate()//获得日
    this.memoList[evt.detail.index].startHour = date.getHours()//获得小时
    this.memoList[evt.detail.index].startMinute = date.getMinutes()//获得分钟
    if (this.memoList[evt.detail.index].startMinute < 10) {
      this.memoList[evt.detail.index].startTime = this.memoList[evt.detail.index].startHour + ':0' +
        this.memoList[evt.detail.index].startMinute;
    }
    else {
      this.memoList[evt.detail.index].startTime = this.memoList[evt.detail.index].startHour + ':' +
        this.memoList[evt.detail.index].startMinute;
    }//输出格式更加整齐，minute为个位数时，添加一个0
    this.memoList[evt.detail.index].start = this.startMinute + this.startHour * 100 + this.startDate *
      this.memoList[evt.detail.index].starttimes = this.memoList[evt.detail.index].startYear + '-' + th
      setMemoList(this.memoList)//以上为修改时间，让todo变成doing时，starttime变成现在的时间
  })
  this.$on('DOING-DONE', (evt) => {
    this.memoList[evt.detail.index].status = 'DONE'
    setMemoList(this.memoList)//添加监视
```

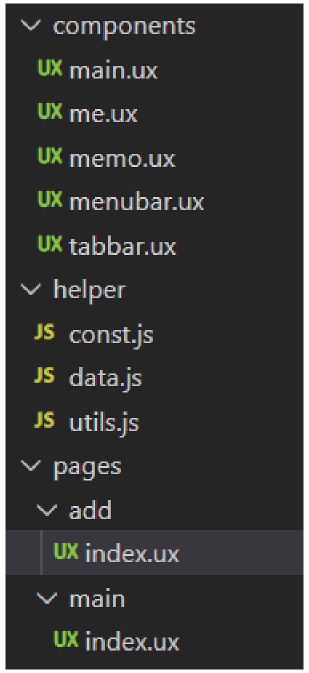
代码质量

命名规范：
（驼峰式命名）
变量含义一目了然



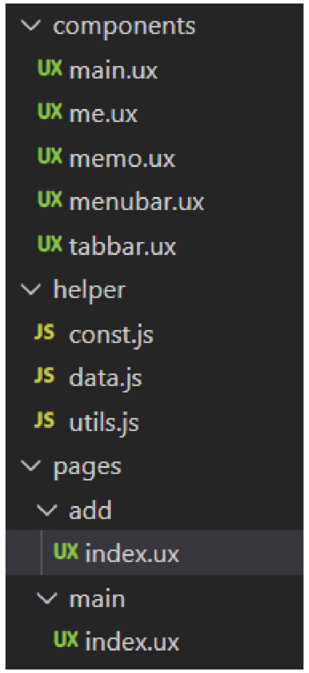
```
return {
  index: '',
  title: '',
  content: '',
  label: '普通',
  eventName: '',
  nameBackUp: '',
  scale: '',
  startYear: 2021,
  startMonth: 8,
  startDate: 24,
  startHour: 20,
  startMinute: 28,
  endYear: 2021,
  endMonth: 8,
  endDate: 24,
  endHour: 20,
  endMinute: 28,
  startTime: 12,
  endTime: 51,
  status: 'TODO',
  isexit: false
```

结构合理
（大量运
用自定义
组件）

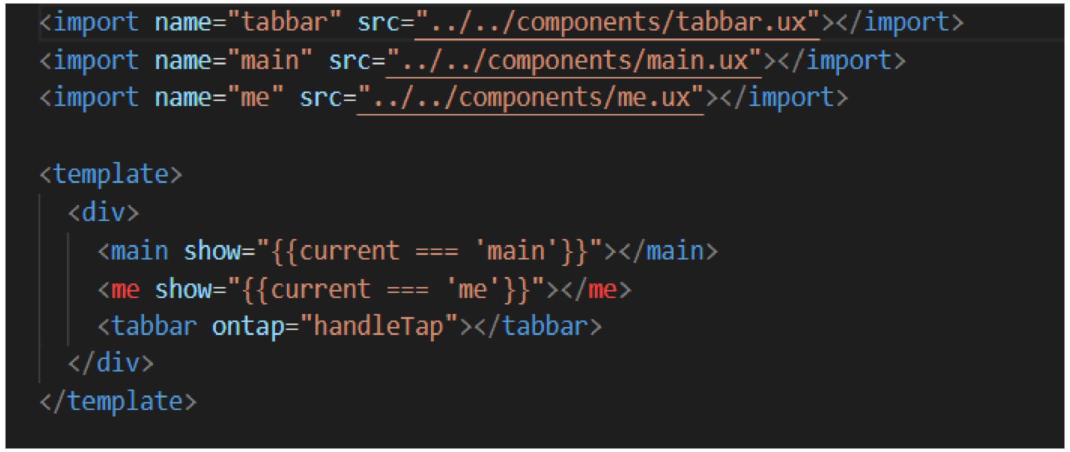


```
components
  main.ux
  me.ux
  memo.ux
  menubar.ux
  tabbar.ux
helper
  const.js
  data.js
  utils.js
pages
  add
    index.ux
  main
    index.ux
```

结构合理



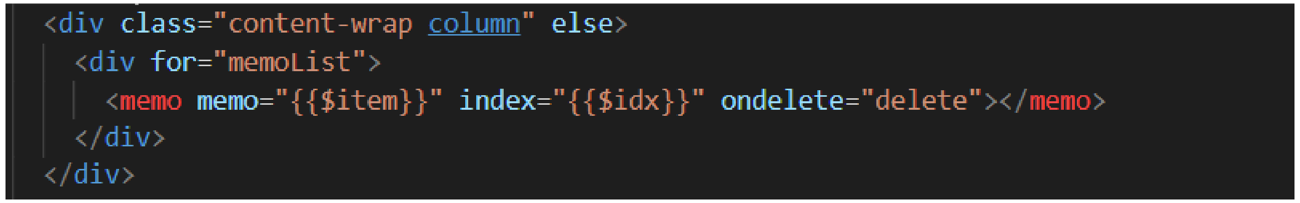
```
∨ components
  UX main.ux
  UX me.ux
  UX memo.ux
  UX menubar.ux
  UX tabbar.ux
∨ helper
  JS const.js
  JS data.js
  JS utils.js
∨ pages
  ∨ add
    UX index.ux
  ∨ main
    UX index.ux
```



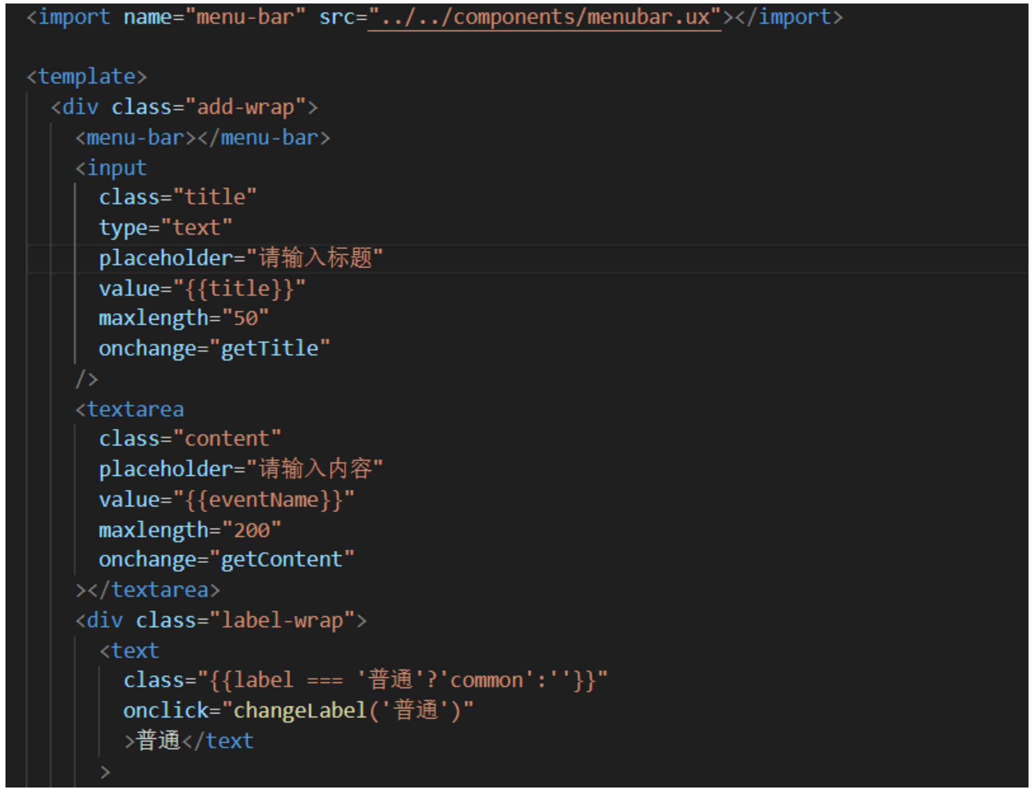
```
<import name="tabbar" src="../../components/tabbar.ux"></import>
<import name="main" src="../../components/main.ux"></import>
<import name="me" src="../../components/me.ux"></import>

<template>
  <div>
    <main show="{{current === 'main'}}"></main>
    <me show="{{current === 'me'}}"></me>
    <tabbar ontap="handleTap"></tabbar>
  </div>
</template>
```

主页面main

主页面下单main组件中

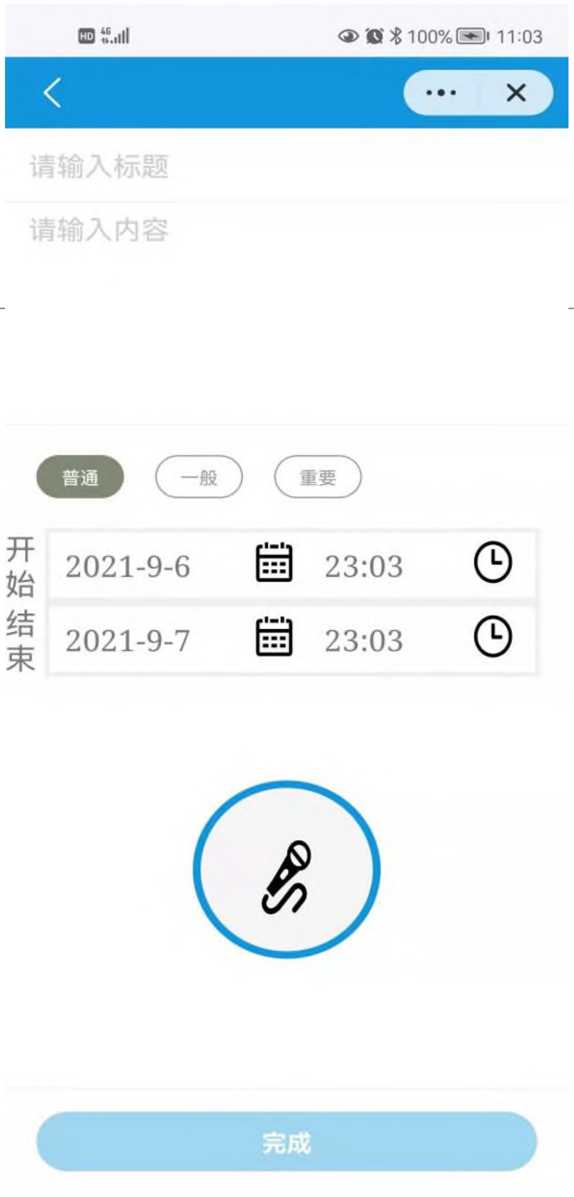


```
<div class="content-wrap column" else>
  <div for="memoList">
    <memo memo="{{$item}}" index="{{$idx}}" ondelete="delete"></memo>
  </div>
</div>
```



```
<import name="menu-bar" src="../../components/menubar.ux"></import>

<template>
  <div class="add-wrap">
    <menu-bar></menu-bar>
    <input
      class="title"
      type="text"
      placeholder="请输入标题"
      value="{{title}}"
      maxlength="50"
      onchange="getTitle"
    />
    <textarea
      class="content"
      placeholder="请输入内容"
      value="{{eventName}}"
      maxlength="200"
      onchange="getContent"
    ></textarea>
    <div class="label-wrap">
      <text
        class="{{label === '普通'?'common':''}}"
        onclick="changeLabel('普通')"
      >普通</text
      >
```

Add页面


100% 11:03

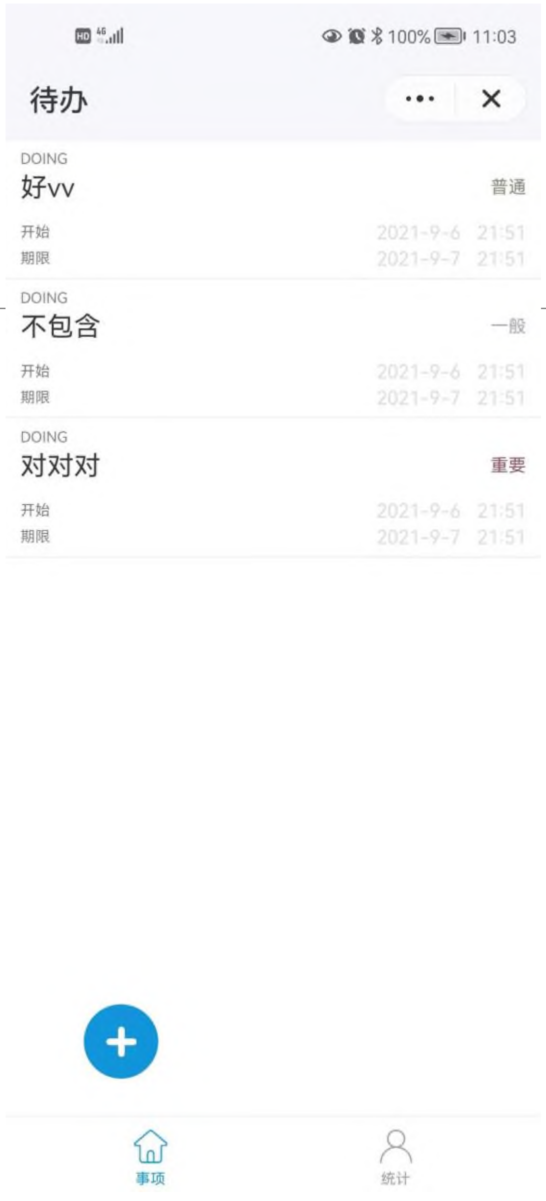
请输入标题

请输入内容

普通 一般 重要

开始	2021-9-6	23:03
结束	2021-9-7	23:03

完成


100% 11:03

待办

DOING
好vv 普通

开始	2021-9-6 21:51
期限	2021-9-7 21:51

DOING
不包含 一般

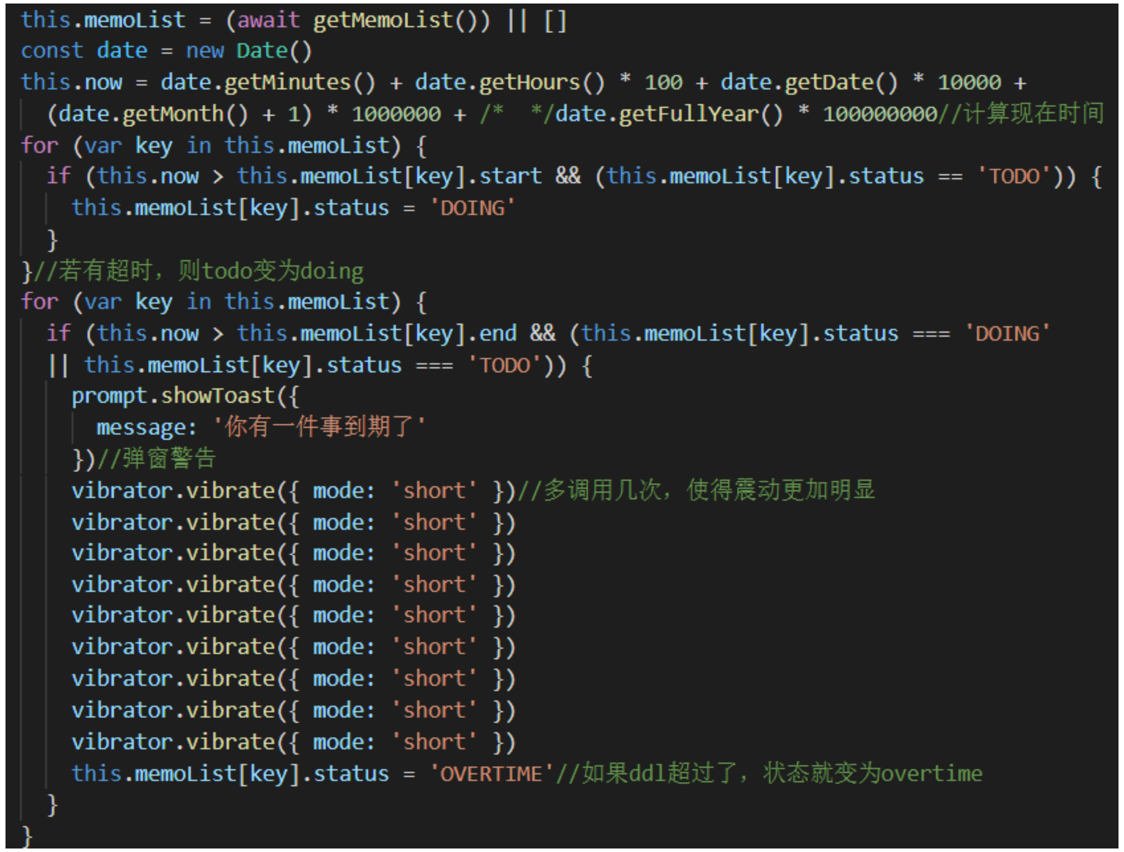
开始	2021-9-6 21:51
期限	2021-9-7 21:51

DOING
对对对 重要

开始	2021-9-6 21:51
期限	2021-9-7 21:51

事项 统计

逻辑简洁（以时间判断为例）



```
this.memoList = (await getMemoList()) || []
const date = new Date()
this.now = date.getMinutes() + date.getHours() * 100 + date.getDate() * 10000 +
  (date.getMonth() + 1) * 1000000 + /*  */date.getFullYear() * 100000000//计算现在时间
for (var key in this.memoList) {
  if (this.now > this.memoList[key].start && (this.memoList[key].status == 'TODO')) {
    this.memoList[key].status = 'DOING'
  }
}//若有超时，则todo变为doing
for (var key in this.memoList) {
  if (this.now > this.memoList[key].end && (this.memoList[key].status === 'DOING'
  || this.memoList[key].status === 'TODO')) {
    prompt.showToast({
      message: '你有一件事到期了'
    })//弹窗警告
    vibrator.vibrate({ mode: 'short' })//多调用几次，使得震动更加明显
    vibrator.vibrate({ mode: 'short' })
    vibrator.vibrate({ mode: 'short' })
    vibrator.vibrate({ mode: 'short' })
    vibrator.vibrate({ mode: 'short' })
    vibrator.vibrate({ mode: 'short' })
    vibrator.vibrate({ mode: 'short' })
    vibrator.vibrate({ mode: 'short' })
    vibrator.vibrate({ mode: 'short' })
    this.memoList[key].status = 'OVERTIME'//如果ddl超过了，状态就变为overtime
  }
}
```

this.start直接就等于年份乘极大的权重（100000000）+月份乘次小的权重（1000000）再以此类推

感谢聆听