

微信小程序 setData 最佳实践



本文用于记录微信小程序运行时性能问题及提升方法

推荐优先阅读：[小程序运行时性能](#)

一、背景

先来看官方文档中提及的性能相关说明：

`setData` 是小程序开发中使用最频繁的接口，也是最容易引发性能问题的接口。小程序的视图层目前使用 WebView 作为渲染载体，而逻辑层是由独立的 JavascriptCore 作为运行环境。在架构上，WebView 和 JavascriptCore 都是独立的模块，并不具备数据直接共享的通道。

简单的说，`setData` 的调用会把数据从逻辑层传到渲染层，**数据太大**会增加通信时间

常见的 `setData` 错误操作：

- 频繁得去 `setData`
- 每次 `setData` 都传递大量新数据 (或存在冗余数据)
- 后台态页面进行 `setData` (`setInterval` 未及时清理)

现有项目存在错误使用 `setData` 的场景，下面就围绕这些问题展开说明，最终形成最佳实践。

二、现存问题与方案

问题一 (临时变量)

问题描述：

```
home.vue U x
src > pages > home > home.vue > {} "home.vue"
1  <template>
2    <view class="wrap">
3      <view> {{ title }} </view>
4    </view>
5  </template>
6
7  <script>
8    export default {
9      data() {
10        return {
11          title: '趣小面',
12          timer: null,
13        };
14      },
15      created() {
16        this.timer = setTimeout(() => {
17          this.title = '趣小面 +';
18        }, 1000);
19      },
20      destroyed() {
21        this.timer = null;
22      },
23    };
24  </script>
25
26  <style>
27    .wrap {
28      padding: 24px;
29    }
30  </style>
31
```

此处 timer 与渲染无关

此处 timer 与视图渲染无关，不应挂载在 data 上

解决方式：



这里仅为了说明临时变量的使用会引起 data 冗余，关于定时器的最佳实践，参阅 [微信小程序定时器最佳实践](#)

HTMLBars

```
1 <template>
2   <view class="wrap">
3     <view> {{ title }} </view>
4   </view>
5 </template>
6
7
8 <script>
9   export default {
10     data() {
11       return {
12         title: '趣小面',
13       };
14     },
15     created() {
16       this.timer = null;
17     },
18     mounted() {
19       this.timer = setTimeout(() => {
20         this.title = '趣小面 +';
21       }, 1000);
22     },
23     destroyed() {
24       this.timer = null;
25     }
26   };
27 </script>
28
29 <style>
30 .wrap {
31   padding: 24px;
32 }
33 </style>
```

步骤：

1. 移除 data 中的 timer 属性 (**需要特别注意 临时变量 初始值，以及后续的值类型判断**)
2. 在 created 钩子里面设置 临时变量的初始值，如 `this.timer = null`

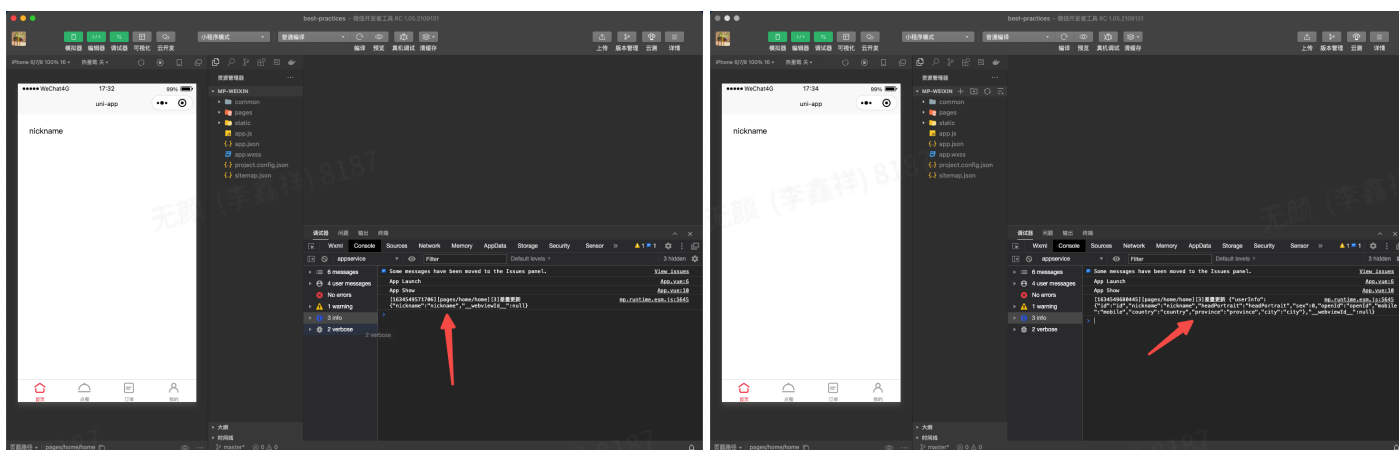
问题二 (mapState)

问题描述：



有些时候，开发者由于偷懒会直接从全局数据获取全量数据。这么做，必然存在着冗余数据。

差别在于，首次执行的时候，computed 会全量执行 setData。



解决方式：

HTMLBars

```
1 <template>
2   <view class="wrap">
3     <view> {{ nickname }} </view>
4   </view>
5 </template>
6
7 <script>
8 import { mapState } from 'vuex';
9
10 export default {
11   computed: {
12     ...mapState({
13       nickname: state => state.userInfo.nickname,
14     }),
15   },
16 };
17 </script>
18
19 <style>
20 .wrap {
21   padding: 24px;
22 }
23 </style>
```

问题三 (watch)

```
home.vue x
src > pages > app > home.vue > {} "home.vue" > script > default
62   orderDynamicData: null,
63   orderDynamicTimer: null,
64   delivery: null,
65   // 上报
66   locationAuthorize: 0, // 1是 0否
67   showStore: 1, // 0是 1否
68   templateCodeTop: AD_CODE.HOME_BANNER_TOP,
69   templateCodeBottom: AD_CODE.HOME_BANNER_BOTTOM,
70   sceneInput: '',
71 };
72 },
73 computed: {
74   ...mapState(['isLogin', 'userInfo']),
75 },
76 watch: {
77   isLogin: {
78     handler(val) {
79       if (val) {
80         if (!this.isHide) {
81           this.isOnShowJudge = true;
82           this.orderDynamic(); // 默认先获取一次
83           this.orderDynamicFn();
84         }
85       } else {
86         clearInterval(this.orderDynamicTimer);
87         this.orderDynamicTimer = null;
88       }
89     },
90   },
91   userInfo: {
92     handler(val) {
93       if (val.openId) {
94         // 判断是否从公众号进入
95         if (this.$store.state.launchCode === '5') {
96           this.reportOfficial(val.openId);
97         }
98       }
99     },
100   },
101 },
```

对于未销毁的页面，此处会在后台执行 setData

解决方式：

1. 从业务逻辑分析，是否需要使用 watch
2. 移除响应 watch 逻辑

三、最佳实践



由于 setData 最佳实践修改涉及业务逻辑，所以需要每个业务负责人对应修改并让测试回归。

最佳实践 (包括但不限于)：

- data 数据 **只能** 挂与渲染相关的数据，其他数据直接挂在 this 上
- computed 中的数据 **只能** 挂载与渲染相关的数据
- 使用 mapState 获取数据时，**不能** 全量获取 **只能** 按需获取

- watch 数据时，**只能** 监听与渲染相关的数据

注意，渲染相关的数据是指 **template** 中绑定的数据。如： `<view> {{ title }} </view>`

示例代码：

HTMLBars

```
1 <template>
2   <view class="wrap">
3     <view> {{ title }} </view>
4     <view> {{ nickname }} </view>
5   </view>
6 </template>
7
8 <script>
9 import { mapState } from 'vuex';
10
11 export default {
12   data() {
13     return {
14       title: '趣小面' // 只能挂载与渲染有关的视图
15     };
16   },
17   created() {
18     this.timer = null; // timer 为临时变量, 且与渲染无关。无需挂在 data 上
19   },
20   mounted() {
21     this.timer = setTimeout(() => {
22       this.title = '趣小面 +';
23     }, 1000);
24   },
25   computed: {
26     ...mapState({
27       nickname: (state) => state.userInfo.nickname, // 只能按需获取
28     }),
29   },
30   watch: {
31     title() {
32       console.log('标题已变更') // 只能 watch 与渲染有关的数据
33     }
34   }
35   // ... 清除定时器相关逻辑
36 };
37 </script>
38
39 <style>
40 .wrap {
```