web 组件化设计实践

目前方案的问题，jsp的include和jquery的load —— 无法传递数据

组件的拆分

遵循单一职责原则

basic组件

如按钮、容器等与业务无关，且无法分割的组件。建议将容器类、锚和按钮等组件也进行简单的封装，以便于多项目复用。

module组件

由basic组件组成，如金盾中的handle，由一个容器、新建等按钮以及搜索框组成。

page

功能模块，由多饿basic或module组件组成，建议将业务数据操作在此类组件中进行。

组件状态管理

项目结构

单个组件

- mdm-button /
  - mdm-button.vue
  - index.js
  - readme.md

```
index.js
```

```
import mdmButton from './mdm-button';

export default mdmButton;
```

index.js的作用是，import 到文件mdm-button文件目录的位置就可以导入mdm-button模块，具体参考node.js文档。

readme.md 主要记录一下组件的用法等信息，建议整理。

组件文件夹

- components
  - component_1 /
    - src /
    - compoent_1.vue
    - index.js
    - readme.md
  - component_2 /
  - component_3 /
  - index.js
- page.vue

其中index.js用于集合组件

```
index.js

import component_1 from './component_1';
import component_2 from './component_2';
import component_3 from './component_3';

export const {
    component_1,
    component_2,
    component_3
};
```

注意：需要异步加载的组件不要这样通过 *index.js* 打包导出，否则会导致 *webpack code split* 失败！

在page.vue中以如下方式引入组件：

```
<script>
    import {
        component_1,
        component_2,
        component_3
    } from 'components' // webpack配置中将compoents字段映射到src/components/文件
```

```
夹上

    export default {
        ...
    }
</script>
```

约定

样式

1. 每个组件必须为块级元素，且没有脱离文档流，使之易于控制布局样式
2. 组件内部最外层不设margin，组件间间距在父组件中设置
3. 总的来说，组件只负责自身内部的样式，不应对外部及其他组件造成影响

数据

1. 组件内部尽量只存储跟ui有关的数据变量，跟业务有关的由vuex处理
2. 数据请求单独抽离成api模块，如若遇到类似ajax重构fetch这种情况，可以只改api层的代码，而不用更改业务层代码。

代码

尽量添加props验证，便于调试和code review