

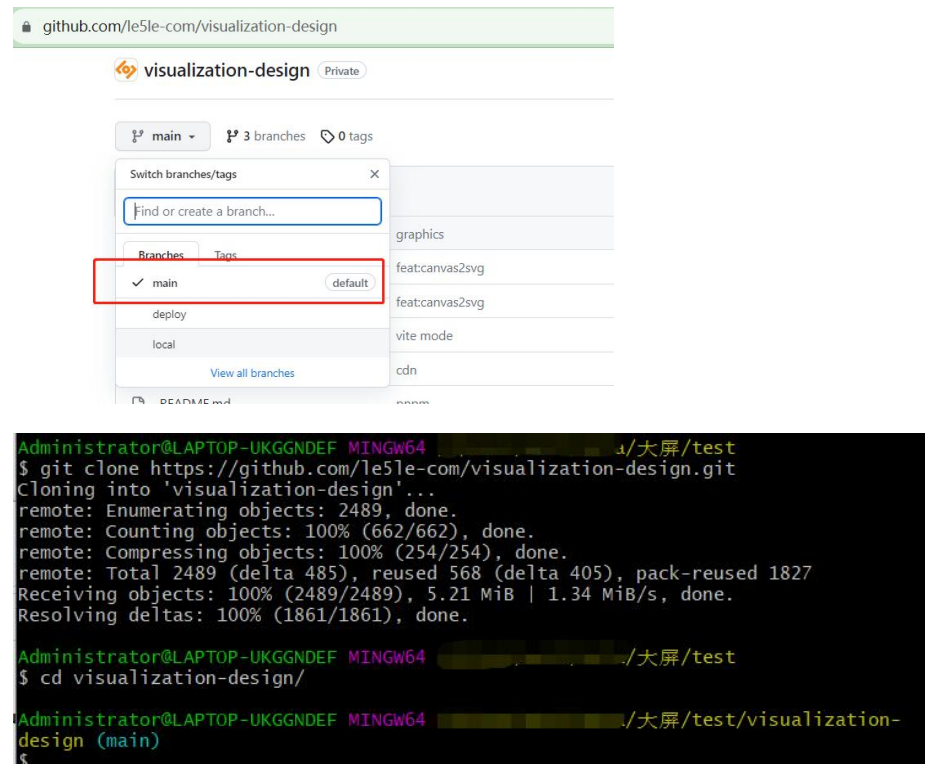
visualization-design 源码使用手册

visualization-design 源码使用手册	1
一 下载代码	2
二 安装依赖包（快速运行）	2
三 加载图形库	3
3.1 方案	3
3.2 模版	3
3.3 组件	3
3.4 图表	4
3.5 素材	5
3.6 图元	5
3.7 控件	6
3.8 图形	7
3.9 我的资源	8
3.9.1 方案	8
3.9.2 模版	9
3.9.3 组件	9
3.9.4 图片	9
3.9.5 3D	10
四 编译打包	10
五 源码结构说明	10
5.1 编辑器页面	11
5.1.1 头部菜单	11
5.1.2 左侧组件库	12
5.1.3 中间画布	16
5.1.3 右侧属性面板	20
5.1.3.1 不选中图元	20
5.1.3.2 单选图元	23
5.1.3.3 多选图元	28
5.2 预览页面	30
六 目录介绍	31
6.1 Public 公共静态资源目录	31
6.2 Src 开发目录	31
6.3 其他	31
七 运行流程	31
八 代码中实现登录链接	32
8.1 完全自己实现后端	32
8.2 购买了乐吾乐后端	32
九 部署集成	32

一 下载代码

项目地址: <https://github.com/le5le-com/visualization-design>

1. 拉取代码: `git clone https://github.com/le5le-com/visualization-design.git`
2. 进入项目文件: `cd visualization-design/`
3. 确认当前是 **main 分支**:



说明:

- ① **main 分支**的核心库是通过引入最新发布的稳定的 **npm 包(meta2d 版本)**: **【推荐】**

[不推荐使用下面方式]

- ② **local 分支**引用的是非稳定状态下的源码包。需要拉取核心库并将其放到该项目的同级目录下, 核心库地址: <https://github.com/le5le-com/meta2d.js>
(若使用 **local 分支**, 需全局搜索 **2d-components**, 并注释)

二 安装依赖包 (快速运行)

进入到 **visualization-design** 项目, 终端运行命令

`pnpm install` //安装依赖

`pnpm start` //启动项目

pnpm 下载地址: <https://pnpm.io/installation> (中文: <https://www.pnpm.cn/installation>)

三 加载图形库

3.1 方案

方案即图纸，通过/api/data/v/list 接口请求，通过 systemFlag=1 区分于非系统图纸。

```
const getCaseProjects = async (name: string, systemFlag = 1, current = 1, pageSize = 1000) => {
  const query: any = { tags: name };
  let collection = name == '系统组件' ? 'v.component' : 'v';
  const ret: any = await axios.post(
    `/api/data/${collection}/list`,
    {
      systemFlag
    },
    {
      params: {
        current,
        pageSize,
      },
    }
  );
  if (!ret) {
    return [];
  }
  for (const item of ret.list) { ...
  }
  return ret.list;
};
```

3.2 模版

模板同场景，通过 systemFlag=2 区分于非系统图纸。

乐吾乐提供场景和模板的后台管理平台，需要可以单独购买。

3.3 组件

通过/api/data/v.component/list 接口请求，通过 systemFlag=1 区分于非系统组件。

```
case '组件':
  if (activeAssets.value === 'system') {
    if (!componentCaches.length) {
      loading.value = true;
      componentCaches.push(...(await getCaseProjects('系统组件', 1)));
      loading.value = false;
      for (const component of componentCaches) {
        if (component.case) {
          let group = components.filter((item) => { item.name === component.case });
          if (group && group.length) {
            group[0].list.push(component);
          } else {
            components.push({
              name: component.case,
              list: [component]
            });
          }
        } else {
          components[0].list.push(component);
        }
      }
    }
  }
```

3.4 图表

图表主要包括 echarts 图表和乐吾乐图表

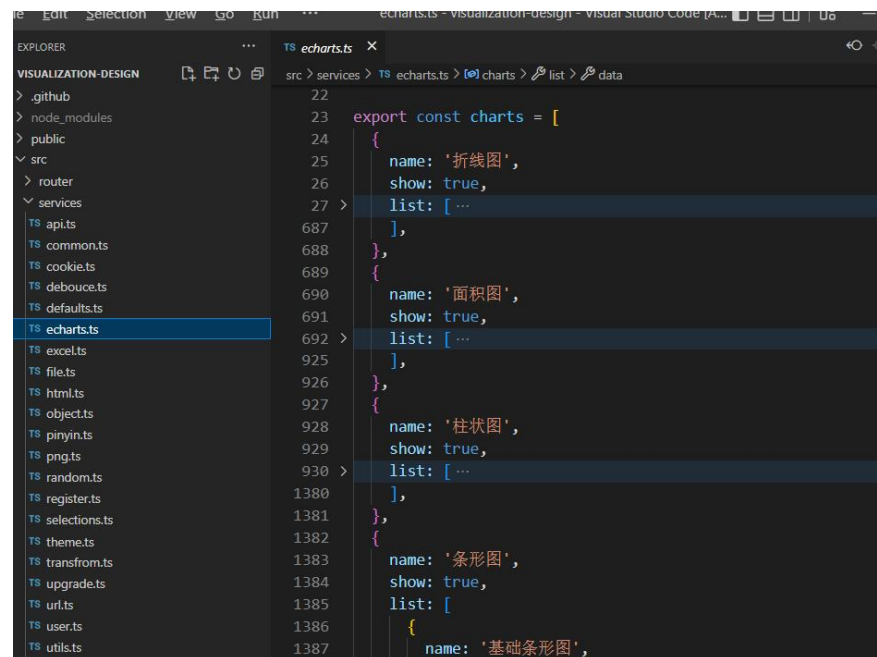
图形库使用对应的文档介绍：

Echarts 图表：

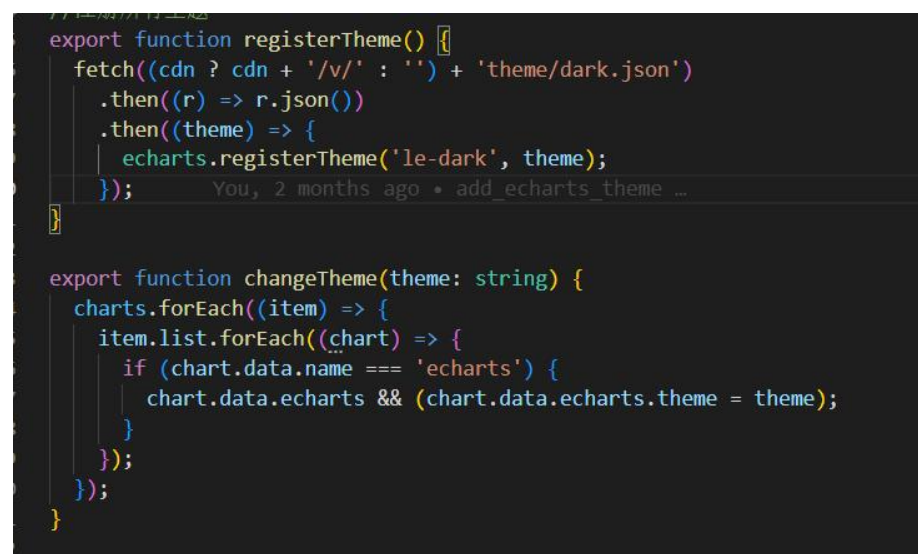
<https://doc.le5le.com/document/119754049#Echarts%E5%9B%BE%E8%A1%A8>

乐吾乐图表：<https://doc.le5le.com/document/119826189>

图表配置文件在 src/services/echarts.ts 文件里面。



这里有 echarts 图形库主题的注册及使用



3.5 素材

素材主要是图片资源，包括一些图片图标、装饰、标题和面板等。通过 `/api/assets/folders` 和 `/api/assets/files` 接口分别请求对应文件夹和文件夹下的文件。传入参数 `path:'v/material'`



```
case '素材':
  groupType.value = 1;
  if (!materials.length) {
    loading.value = true;
    materials.push(...(await getFolders('v/material')));
    loading.value = false;
  }
  subGroups.value = materials;
  break;
```

```
export async function getFolders(name: string, isSvg?: boolean) {
  const path = name;
  const folders: any = await axios.post('/api/assets/folders', {
    path,
  });
  if (!folders || !folders.list) {
    return [];
  }
  const files: any = await axios.post('/api/assets/files', {
    path,
  });
}
```

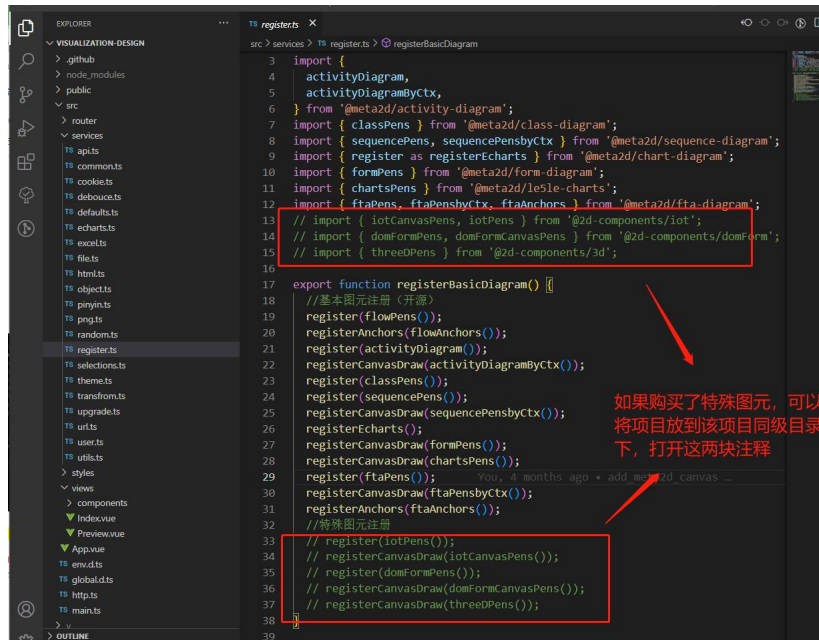
3.6 图元

图元主要是指不同行业/场景下的 `svg/png` 组件库，接口请求同素材，传入参数 `path` 分别是 `"svg"` / `"png"`

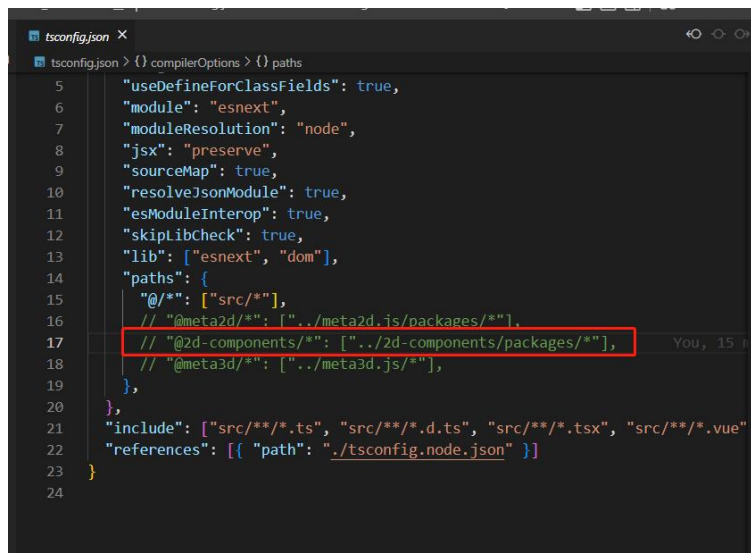
```
case '图元':
  if (!pngs.length) {
    loading.value = true;
    pngs.push(...(await getFolders('png')));
    pngs.push(...(await getFolders('svg', true)));
    loading.value = false;
  }
  subGroups.value = pngs;
  break;
```

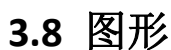
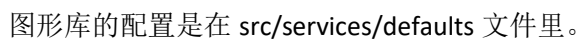
3.7 控件

控件主要是指一些控件图元，包含基础图元和特殊图元（需要单独购买）。图形库属性的详细介绍可以查看文档：<https://doc.le5le.com/document/119754049>
图元的注册是在 `src/services/register.ts` 文件里



如果购买了特殊图元，需要将特殊图元项目放到此项目同级目录下，并取消下面框选位置的注释：





<https://doc.le5le.com/document/119754049>

7 / 32


```

9 export const shapes = [
10   {
11     name: '基本形状',
12     show: true,
13     list: [ ...
14   ],
15   },
16   {
17     name: '脑图',
18     show: true,
19     list: [ ...
20   ],
21   },
22   {
23     name: '流程图',
24     show: true,
25     list: [ ...
26   ],
27   },
28   {
29     name: '活动图',
30     show: true,
31     list: [
32       {
33         name: '活动图',
34         show: true,
35         list: [
36           {
37             name: '活动图',
38             show: true,
39             list: [
40               {
41                 name: '活动图',
42                 show: true,
43                 list: [
44                 ]
45               }
46             ]
47           }
48         ]
49       }
50     ]
51   }
52 ]

```

3.9 我的资源

3.9.1 方案

```

//用户方案
subGroups.value = await getCollectionImageList('方案', 'v',1);
groupType.value = 1;
userLastName = name;

```

1. /api/directory/list 获取文件夹列表
2. /api/data/v/list 获取所有图纸数据
3. 再将图纸数据放入对应到文件夹

```

const getCollectionImageList = async (name?: string, collection?: string, userFlag?: number) => {
  //1. 获取网盘文件夹
  const fullpath = `/大屏/${name}`;
  let ret: { list: any[] } = await axios.post('/api/directory/list', {
    fullpath,
  });
  if (!ret) {
    // You, 4 months ago • feat:我的资源请求方式规范 ...
  }
  let list = [];
  for (let i of ret.list) { ...
  }
  const data = { ...
  };
  const config = { ...
  };
  //2. 请求所有图纸/组件数据
  const res: any = await getCollectionList(collection, data, config);
  //3. 将数据对应到网盘文件夹
  const results = [];
  const resultsMap = { ...
  };
  for (const item of list) { ...
  }
}

```


3.9.2 模版

```
subGroups.value = await getCollectionImageList('模板', 'v', 2);
groupType.value = 1;
userLastName = name;
```

同方案，通过 userFlag=1/2 区分是方案还是模版

3.9.3 组件

```
subGroups.value = await getCollectionImageList(
  '组件',
  'v.component',
  1
);
groupType.value = 1;
userLastName = name;
}
```

同方案, collection 对应 'v.component'。

3.9.4 图片

```
case '图片':
  loading.value = true;
  subGroups.value = await getImageList();
  loading.value = false;
  userLastName = name;
  break;
```

```
const getImageList = async () => {
  let ret: { list: any[] } = await axios.post('/api/directory/list', {
    fullpath: '/大屏/图片',
  });
  if (!ret) {
    return [];
  }
  let list = [];
  for (let i of ret.list) {
    if (i.fullpath.split('/').length === 4) {
      //不取当前文件夹
      list.push(i);
    }
  }
  return await Promise.all(...);
};
```

获取云盘下所有图片资源。

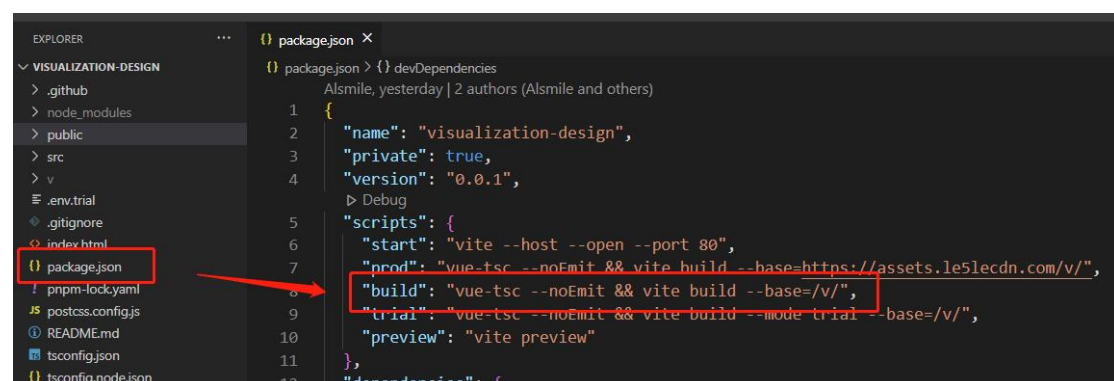
3.9.5 3D

自定义的 3d 场景，通过/api/data/3d/list 接口请求 3d 场景

```
case '3D':
  subGroups.value = [
    {
      name: '3D',
      list: [],
    },
  ];
  groupType.value = 1;
  await getPrivateGraphics();
  userLastName = name;
  break;
```

四 编译打包

运行命令：pnpm run build



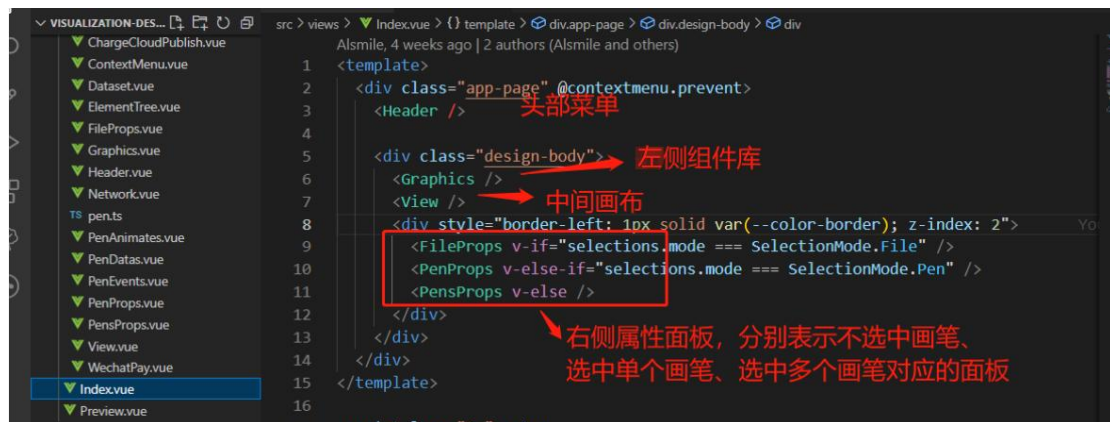
五 源码结构说明

整个项目分为两个页面：①Index.vue 大屏编辑页面 ②Preview.vue 大屏预览页面（运行页面）

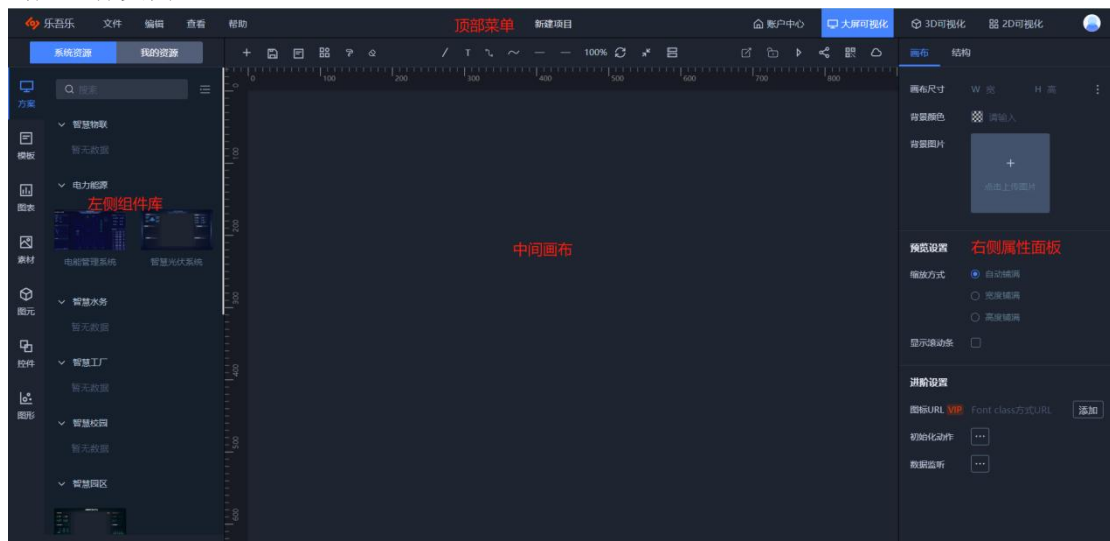
```
const routes = [
  { path: '/', component: () => import('@views/Index.vue') },
  { path: '/preview', component: () => import('@views/Preview.vue') },
];
```

5.1 编辑器页面

编辑器页面整体结构如下：



对应运行页面：



5.1.1 头部菜单

1. 目录：components/Header.vue
2. 源码说明：



- ① 右侧包含 logo 及公司名、文件、编辑、查看、帮助,都是通过下拉组件实现以文件为例,可以在每个选项的 click 事件定位到对应执行的代码内容。

```

<t-dropdown
  :minColumnWidth="200"
  :maxHeight="560"
  :delay2="[10, 150]"
  overlayClassName="header-dropdown"
  trigger1="click"
>
  <a> 文件 </a>
  <t-dropdown-menu>
    <t-dropdown-item @click="newFile">
      <a>新建文件</a>
    </t-dropdown-item>
    <t-dropdown-item @click="load(true)">
      <a>打开文件</a>
    </t-dropdown-item>
    <t-dropdown-item divider="true" @click="load">
      <a>导入文件</a>
    </t-dropdown-item>
    <t-dropdown-item>
      <a @click="save()">保存</a>
    </t-dropdown-item>
    <t-dropdown-item>
      <a @click="save(SaveType.SaveAs)">另保存</a>
    </t-dropdown-item>
    <t-dropdown-item divider="true">
      <a @click="downloadJson">下载JSON文件</a>
    </t-dropdown-item>
    <t-dropdown-item>

```

方法中可能用到了一些开源库（例如：下载文件用到了 `file-saver` 库）、调用了核心库方法，具体开发者自行阅读执行逻辑。

② 中间输入大屏图纸的名称

```

<div style= width: 148px; flex-shrink: 0 ></div>
<input v-model="data.name" @input="onInputName" />
Alsmile, 4 months ago • init
<a href= "assets/account" target= "blank">

```

③ 右侧是一些导航链接，包括账户中心、乐吾乐其他产品、登录/用户菜单

5.1.2 左侧组件库

1. 目录：components/Graphics.vue
2. 源码说明：

```

<div class="input-search">
  <div class="btn">
    <t-icon name="search" />
  </div>
  <t-input
    v-model="search"
    @change="onSearch"
    @enter="onSearch"
    placeholder="搜索"
  />

  <div class="ml-16">
    <t-tooltip content="展开/折叠">
      <t-icon
        name="menu-fold"
        class="hover"
        style="font-size: 16px"
        @click="onFold"
      />
    </t-tooltip>
  </div>
</div>

```

① 顶部的左侧搜索框用于从下面选中的组件中搜索内容，右侧按钮控制下面选中的组件整体的展开/折叠



搜索框执行代码如图：主要通过 visible 属性控制组件的显示/隐藏

```

<t-input
  v-model="search"
  @change="onSearch"
  @enter="onSearch"
  placeholder="搜索"
/>

```

```

5  const onSearch = () => {      Alsmile, last month • search ing
6  |   debounce(searchGraphics, 300);
7  | };
8
9  const searchGraphics = async () => {
10 |   if (search.value) {
11 |     activePanels[activeGroup.value].splice(
12 |       0,
13 |       activePanels[activeGroup.value].length
14 |     );
15 |   }
16
17 |   for (const group of subGroups.value) {
18 |     for (const item of group.list) {
19 |       if (search.value) {
20 |         item.visible = searchObjectPinyin(item, 'name', search.value);
21 |       } else {
22 |         item.visible = true;
23 |       }
24 |     }
25
26 |     if (search.value) {
27 |       activePanels[activeGroup.value].push(group.name);
28 |     }
29 |   }
30 | };
31

```

右侧按钮通过控制当前活动面板 activePanels key 的内容控制下面面板的展开/折叠。

```

<t-tooltip content="展开/折叠">
  <t-icon
    name="menu-fold"
    class="hover"
    style="font-size: 16px"
    @click="onFold"
  />
</t-tooltip>

```



```
const onFold = () => {
  if (!activatedPanels[activatedGroup.value]) {
    return;
  }

  if (activatedPanels[activatedGroup.value].length) {
    activatedPanels[activatedGroup.value] = [];
  } else {
    activatedPanels[activatedGroup.value] = [];
    for (const item of subGroups.value) {
      activatedPanels[activatedGroup.value].push(item.name);
    }
  }
};
```

② 下面组件库，不同的类型对应不同的组件库/场景/模版内容。



通过监听点击选中，根据 name 去请求不同的数据，展示对应的内容。


```
const groupChange = async (name: string) => {
  activatedGroup.value = name;
  groupType.value = 0;
  switch (name) {
    > case '方案': ...
    > case '模板': ...
    > case '图表': ...
    > case '控件': ...
    > case '素材': ...
    > case '图元': ...
    > case '图形': Alsmile, 3 months ago • 场景
    > case '组件': ...
    > case '图片': ...
    > case '3D': ...
  }
}
```

5.1.3 中间画布

① 顶部二级菜单



右侧是快捷按钮 新建、保存为大屏、保存为我的组件、格式刷和清除格式。

可以通过点击事件跳转到对应方法的执行，例如格式刷主要调用了核心库方法。

```
const oneFormat = () => {
  if (one.value) {
    one.value = false;
  } else {
    one.value = true;
    meta2d.setFormatPainter();
  }
  if (always.value) {
    always.value = false;
    one.value = false;
  }
};

const alwaysFormat = () => {
  always.value = true;
};

const clearFormat = () => {
  always.value = false;
  one.value = false;
  meta2d.clearFormatPainter();
};
```

中间是连线、视图、数据源管理相关操作

可以通过调用核心库方法将图纸切换到连线状态，下图代码表示控制关闭/打开连线状态

```

const oneDraw = () => {
  if (oneD.value) {
    oneD.value = false;
    if (!alwaysD.value) {
      meta2d.finishDrawLine();
      meta2d.drawLine();
      meta2d.store.options.disableAnchor = true;
    }
  } else {
    oneD.value = true;
    meta2d.drawLine(meta2d.store.options.drawingLineName);
    meta2d.store.options.disableAnchor = false;
  }
  if (alwaysD.value) {
    meta2d.finishDrawLine();
    meta2d.drawLine();
    oneD.value = false;
    alwaysD.value = false;
    meta2d.store.options.disableAnchor = true;
  }
};

```

通过修改 options 配置默认连线样式，下图代码表示修改连线类型。

```

const changeLineType = (value: string) => {
  currentLineType.value = value;
  if (meta2d) {
    meta2d.store.options.drawingLineName = value;
    meta2d.canvas.drawingLineName && (meta2d.canvas.drawingLineName
    meta2d.store.active?.forEach((pen) => {
      meta2d.updateLineType(pen, value);
    });
  }
};

```

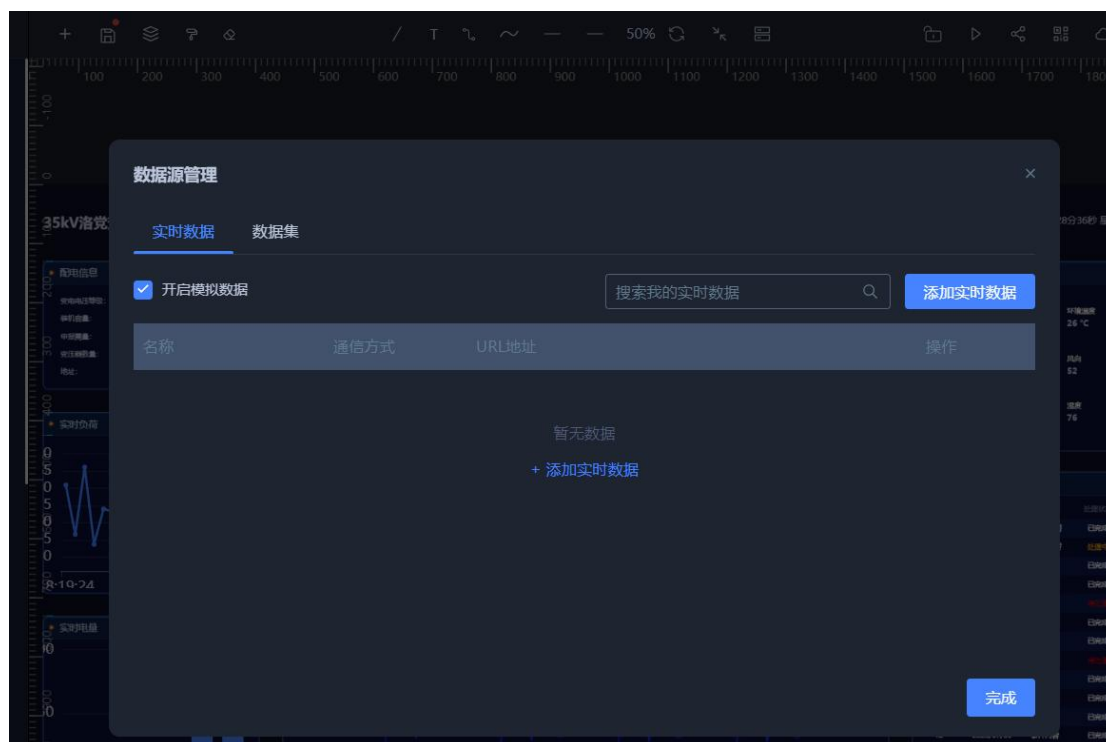
通过调用核心库方法改变当前视图大小

```
export const onScaleWindow = () => {
  // meta2d.fitView();
  meta2d.fitSizeView(true, 32);
};

export const onScaleFull = () => {
  meta2d.scale(1);
  // meta2d.centerView();
  const { x, y, origin, center } = meta2d.store.data;

  meta2d.translate(-x - origin.x, -y - origin.y);
  meta2d.translate(meta2d.store.options.x || 0, meta2d.store.
};
```

数据源管理主要包括实时数据和数据集管理



右侧包含锁定画布、运行、分享、云发布等。

主要是项目业务内容，涉及到大屏的正式发布，需要结合部署人员一起探讨。

② 核心画布

```
<div id="meta2d"></div>
```

```

onMounted(() => {
  meta2d = new Meta2d('meta2d', meta2dOptions);
  registerBasicDiagram();
  open(true);
  meta2d.on('active', active);
  meta2d.on('inactive', inactive);
  meta2d.on('scale', scaleSubscriber);
  meta2d.on('add', lineAdd);
  meta2d.on('opened', openedListener);

  meta2d.on('undo', patchFlag);
  meta2d.on('redo', patchFlag);
  meta2d.on('add', patchFlag);
  meta2d.on('delete', patchFlag);
  meta2d.on('rotatePens', patchFlag);
  meta2d.on('translatePens', patchFlag);

  // 所有编辑栏所做修改
  meta2d.on('components-update-value', patchFlag);
  meta2d.on('contextmenu', onContextMenu);
  meta2d.on('click', canvasClick);

  timer = setInterval(autoSave, 60000);

  window.onbeforeunload = () => {
    autoSave();
  };
});

```

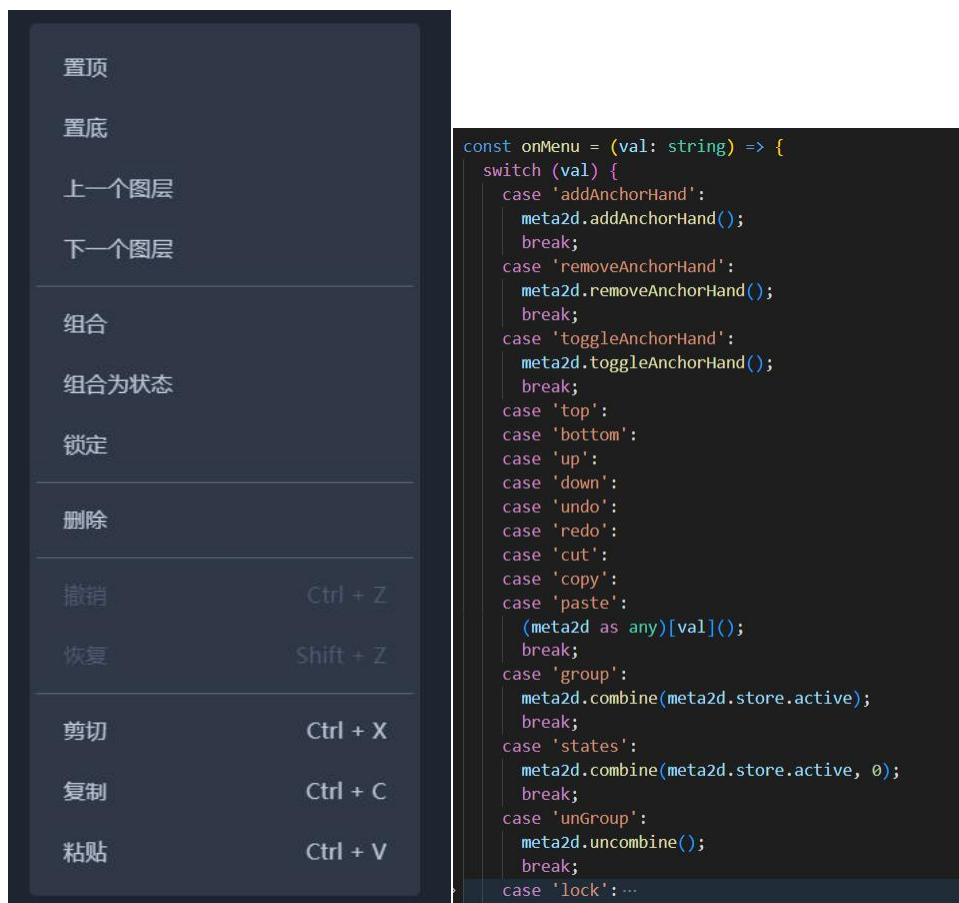
创建meta2d实例

注册图形库

监听画布消息

自动保存图纸

③ 右键菜单 (components/ ContextMenu.vue)



主要是通过调用核心库方法，具体方法说明可查看官方文档：

<https://doc.le5le.com/document/119882449#%E5%87%BD%E6%95%B0>

5.1.3 右侧属性面板

5.1.3.1 不选中图元

① 目录 components/FileProps.vue

② 源码说明：

1. 画布板块

画布板块主要分为，基本设置（包括大屏尺寸、背景颜色和背景图片）、预览设置（设置预览时/运行时的缩放方式、是否显示滚动条）、进阶设置（配置图片库、初始化和数据监听）

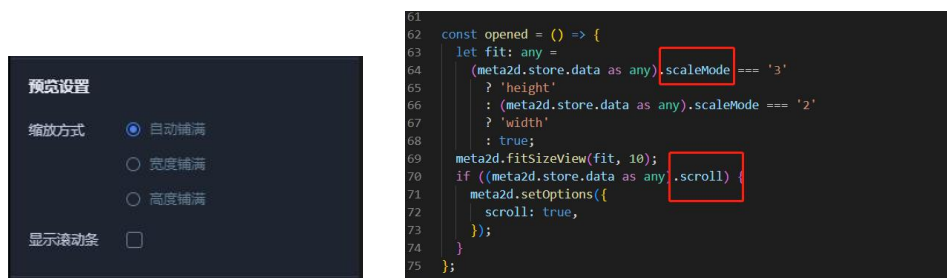


基本设置这一块，主要是通过修改 `meta2d.store.data` 下面的属性，具体说明可查看文档：
<https://doc.le5le.com/document/119882449#%E6%95%B0%E6%8D%AE%E7%BB%93%E6%9E%84>

4
 一些特殊属性：例如背景图片的设置，调用核心库 `setBackgroundImage` 方法。



预览设置这一块主要是配置预览时，大屏页面的显示方式，属于业务属性，使用可以查看 `Preview.vue` 页面。



进阶设置首先可以配置图标地址，输入字体图标地址后，会加载对应的图标库，在选中画笔进行图标设置时可以在图标抽屉中展示请求到的图标库。

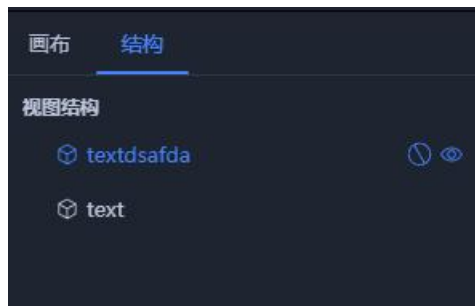


其次可以配置初始化动作，是指首次打开图纸后会执行的脚本。最后数据监听，即通信建立后获取数据的处理脚本，具体可查看文档：

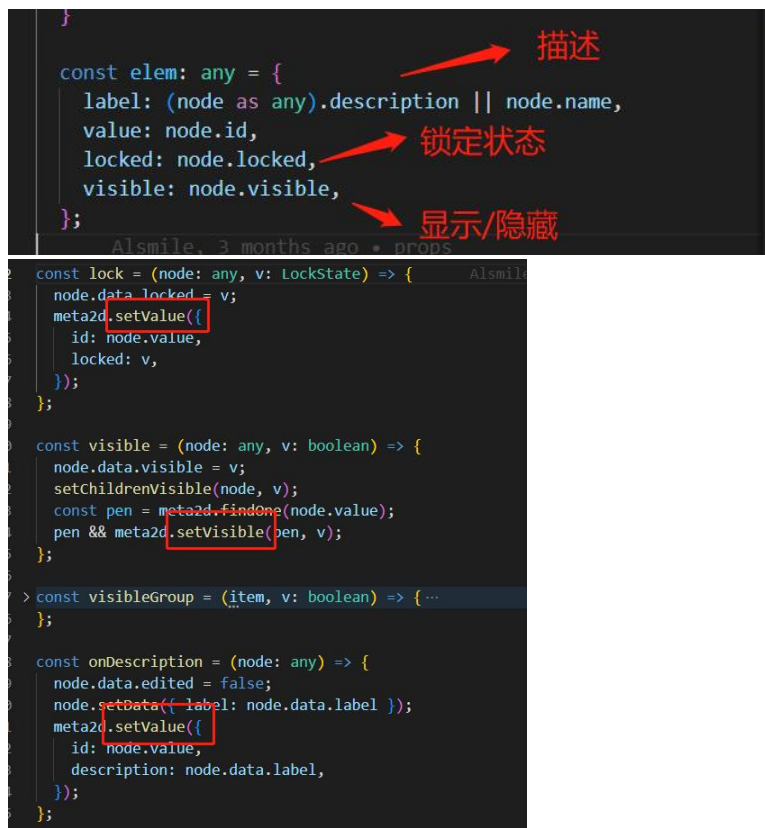
<https://doc.le5le.com/document/119620524#%E8%A7%A3%E6%9E%90%E8%87%AA%E5%AE%9A%E4%B9%89%E6%A0%BC%E5%BC%8F%E6%95%B0%E6%8D%AE>

2. 结构板块（components/ElementTree.vue）

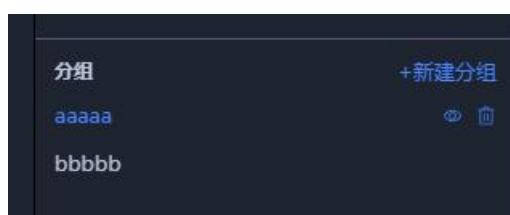
结构主要包括视图结构和分组，在视图结构可以设置画布中每个图元的描述名称、控制图元锁定状态和控制视图显示/隐藏，



主要是通过调用核心库的 setValue 和 setVisible 方法，对应的属性如下：



在分组中，可以新增/删除自定义分组名称，可以批量控制该分组对应图元的显示/隐藏。

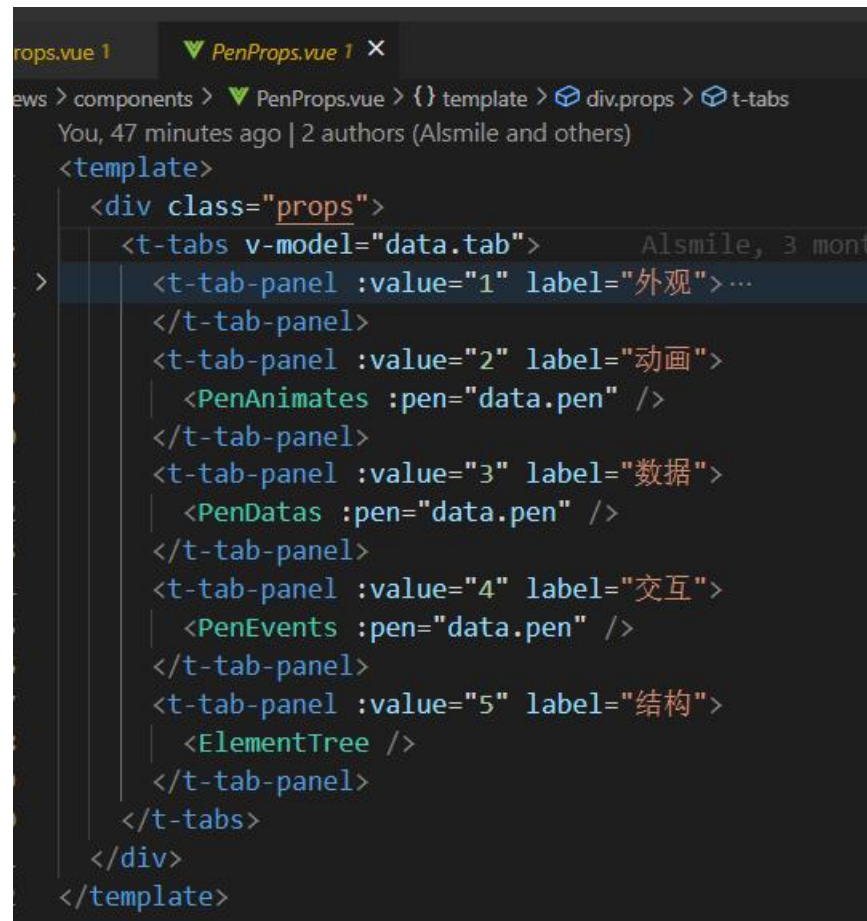


5.1.3.2 单选图元

① 目录 components/PenProps.vue

② 源码说明：

单选图元板块主要包括外观、动画、数据、交互和结构。



```
<template>
  <div class="props">
    <t-tabs v-model="data.tab">
      <t-tab-panel :value="1" label="外观">...
    </t-tab-panel>
    <t-tab-panel :value="2" label="动画">
      <PenAnimates :pen="data.pen" />
    </t-tab-panel>
    <t-tab-panel :value="3" label="数据">
      <PenDatas :pen="data.pen" />
    </t-tab-panel>
    <t-tab-panel :value="4" label="交互">
      <PenEvents :pen="data.pen" />
    </t-tab-panel>
    <t-tab-panel :value="5" label="结构">
      <ElementTree />
    </t-tab-panel>
    </t-tabs>
  </div>
</template>
```

1. 外观

外观主要内容是设置图元的样式、包括 id 位置等基本设置、文字、图片/图标、自定义属性等。

外观

动画

数据

交互

结构

ID

116fa54

名称

text

分组

aaaaa

X

217.25

Y

77.25

0

W

160

H

30

圆角

不透明度

1

外观

前景颜色

悬停颜色

选中颜色

线条

1

末端样式

连接样式

背景

纯色

线性渐变

径向渐变

请输入

阴影

文字

属性

水平翻转

垂直翻转

锚点半径 4

禁止旋转

禁止缩放

禁用锚点

鼠标提示

```
1180 };
1181
1182 const changeValue = (prop: string) => {
1183   updatePen(data.pen, prop);
1184 };
1185
1186 const changeID = (value: any) => {
1187   if (!value) {
1188     initPenData();
1189     MessagePlugin.error('id 不能为空');
1190     return;
1191   }
1192   const oldID: string = data.pen.id;
1193   try {
1194     meta2d.changePenId(oldID, value);
1195     initPenData();
1196   } catch (error) {
1197     console.warn(error.message);
1198     MessagePlugin.error('id 修改失败, 请检查 id 是否');
1199     return;
1200   }
1201 };
1202
1203 const changeRectValue = (prop: string) => {
1204   data.rect.id = data.pen.id;
1205   data.rect.ratio = data.pen.ratio;
1206   updatePen(data.rect, prop);
1207 };
1208
1209 const onFontPopupVisible = (val: boolean) => {
1210   data.fontFamilyPopupVisible = val;
1211 };
1212
1213 const onFontFamily = (fontFamily: string) => {
1214   data.pen.fontFamily = fontFamily;
1215   data.fontFamilyPopupVisible = false;
1216   changeValue('fontFamily');
1217 };
1218
1219 export const updatePen = (pen: any, prop: string, render = true) => {
1220   const v: any = { id: pen.id };
1221   v[prop] = pen[prop];
1222   if (prop === 'width' && pen.ratio) {
1223     const rect = meta2d.findOne(pen.id);
1224     v.height = (pen.width / rect.width) * rect.height;
1225     pen.height = v.height;
1226   } else if (prop === 'height' && pen.ratio) {
1227     const rect = meta2d.findOne(pen.id);
1228     v.width = (pen.height / rect.height) * rect.width;
1229     pen.width = v.width;
1230   } else if (prop === 'shadow') {
1231     if (v[prop]) {
1232       lv.shadowOffsetX && (v.shadowOffsetX = 0);
1233       lv.shadowOffsetY && (v.shadowOffsetY = 0);
1234       lv.shadowBlur && (v.shadowBlur = 0);
1235     } else {
1236       v.shadowColor = '';
1237     }
1238   } else if (prop === 'lineGradientColors') {
1239     // @ts-ignore
1240     if (meta2d.store.active[0].name === 'line') {
1241       // @ts-ignore
1242       meta2d.store.active[0].calculative.gradientColorStop = null;
1243     } else {
1244       // @ts-ignore
1245       meta2d.store.active[0].calculative.lineGradient = null;
1246     }
1247     // 不同模式切换不同的系统配色
1248   } else if (prop === 'titleFnJs') {
1249     v.titleFn = null;
1250   } else if (prop === 'dash') {
1251     v.lineDash = lineDashObj[v[prop]];
1252   }
1253   meta2d.setValue(v, { render });
1254 };
1255
1256 export const updateRect = (rect: any, prop: string, render = true) => {
1257   const v: any = { id: rect.id };
1258   v[prop] = rect[prop];
1259   if (prop === 'width' && rect.ratio) {
1260     const pen = meta2d.findOne(rect.id);
1261     v.height = (rect.width / pen.width) * pen.height;
1262     rect.height = v.height;
1263   } else if (prop === 'height' && rect.ratio) {
1264     const pen = meta2d.findOne(rect.id);
1265     v.width = (rect.height / pen.height) * pen.width;
1266     rect.width = v.width;
1267   } else if (prop === 'shadow') {
1268     if (v[prop]) {
1269       lv.shadowOffsetX && (v.shadowOffsetX = 0);
1270       lv.shadowOffsetY && (v.shadowOffsetY = 0);
1271       lv.shadowBlur && (v.shadowBlur = 0);
1272     } else {
1273       v.shadowColor = '';
1274     }
1275   } else if (prop === 'lineGradientColors') {
1276     // @ts-ignore
1277     if (meta2d.store.active[0].name === 'line') {
1278       // @ts-ignore
1279       meta2d.store.active[0].calculative.gradientColorStop = null;
1280     } else {
1281       // @ts-ignore
1282       meta2d.store.active[0].calculative.lineGradient = null;
1283     }
1284     // 不同模式切换不同的系统配色
1285   } else if (prop === 'titleFnJs') {
1286     v.titleFn = null;
1287   } else if (prop === 'dash') {
1288     v.lineDash = lineDashObj[v[prop]];
1289   }
1290   meta2d.setValue(v, { render });
1291 };
1292
1293 export const updateText = (text: any, prop: string, render = true) => {
1294   const v: any = { id: text.id };
1295   v[prop] = text[prop];
1296   if (prop === 'fontSize') {
1297     v.fontSize = text.fontSize;
1298   } else if (prop === 'fontWeight') {
1299     v.fontWeight = text.fontWeight;
1300   } else if (prop === 'fontStyle') {
1301     v.fontStyle = text.fontStyle;
1302   } else if (prop === 'fontFamily') {
1303     v.fontFamily = text.fontFamily;
1304   } else if (prop === 'textColor') {
1305     v.textColor = text.textColor;
1306   } else if (prop === 'textAlign') {
1307     v.textAlign = text.textAlign;
1308   } else if (prop === 'textBaseline') {
1309     v.textBaseline = text.textBaseline;
1310   } else if (prop === 'textDirection') {
1311     v.textDirection = text.textDirection;
1312   } else if (prop === 'textOrientation') {
1313     v.textOrientation = text.textOrientation;
1314   } else if (prop === 'textOverflow') {
1315     v.textOverflow = text.textOverflow;
1316   } else if (prop === 'textWrap') {
1317     v.textWrap = text.textWrap;
1318   } else if (prop === 'textLineHeight') {
1319     v.textLineHeight = text.textLineHeight;
1320   } else if (prop === 'textLetterSpacing') {
1321     v.textLetterSpacing = text.textLetterSpacing;
1322   } else if (prop === 'textWordSpacing') {
1323     v.textWordSpacing = text.textWordSpacing;
1324   } else if (prop === 'textFontWeight') {
1325     v.textFontWeight = text.textFontWeight;
1326   } else if (prop === 'textFontStyle') {
1327     v.textFontStyle = text.textFontStyle;
1328   } else if (prop === 'textFontFamily') {
1329     v.textFontFamily = text.textFontFamily;
1330   } else if (prop === 'textColor') {
1331     v.textColor = text.textColor;
1332   } else if (prop === 'textAlign') {
1333     v.textAlign = text.textAlign;
1334   } else if (prop === 'textBaseline') {
1335     v.textBaseline = text.textBaseline;
1336   } else if (prop === 'textDirection') {
1337     v.textDirection = text.textDirection;
1338   } else if (prop === 'textOrientation') {
1339     v.textOrientation = text.textOrientation;
1340   } else if (prop === 'textOverflow') {
1341     v.textOverflow = text.textOverflow;
1342   } else if (prop === 'textWrap') {
1343     v.textWrap = text.textWrap;
1344   } else if (prop === 'textLineHeight') {
1345     v.textLineHeight = text.textLineHeight;
1346   } else if (prop === 'textLetterSpacing') {
1347     v.textLetterSpacing = text.textLetterSpacing;
1348   } else if (prop === 'textWordSpacing') {
1349     v.textWordSpacing = text.textWordSpacing;
1350   } else if (prop === 'textFontWeight') {
1351     v.textFontWeight = text.textFontWeight;
1352   } else if (prop === 'textFontStyle') {
1353     v.textFontStyle = text.textFontStyle;
1354   } else if (prop === 'textFontFamily') {
1355     v.textFontFamily = text.textFontFamily;
1356   }
1357   meta2d.setValue(v, { render });
1358 };
1359
1360 export const updateImage = (image: any, prop: string, render = true) => {
1361   const v: any = { id: image.id };
1362   v[prop] = image[prop];
1363   if (prop === 'x') {
1364     v.x = image.x;
1365   } else if (prop === 'y') {
1366     v.y = image.y;
1367   } else if (prop === 'width') {
1368     v.width = image.width;
1369   } else if (prop === 'height') {
1370     v.height = image.height;
1371   } else if (prop === 'angle') {
1372     v.angle = image.angle;
1373   } else if (prop === 'opacity') {
1374     v.opacity = image.opacity;
1375   } else if (prop === 'filter') {
1376     v.filter = image.filter;
1377   } else if (prop === 'src') {
1378     v.src = image.src;
1379   } else if (prop === 'alt') {
1380     v.alt = image.alt;
1381   } else if (prop === 'title') {
1382     v.title = image.title;
1383   } else if (prop === 'description') {
1384     v.description = image.description;
1385   } else if (prop === 'copyright') {
1386     v.copyright = image.copyright;
1387   } else if (prop === 'license') {
1388     v.license = image.license;
1389   } else if (prop === 'author') {
1390     v.author = image.author;
1391   } else if (prop === 'created') {
1392     v.created = image.created;
1393   } else if (prop === 'updated') {
1394     v.updated = image.updated;
1395   } else if (prop === 'deleted') {
1396     v.deleted = image.deleted;
1397   } else if (prop === 'status') {
1398     v.status = image.status;
1399   } else if (prop === 'type') {
1400     v.type = image.type;
1401   } else if (prop === 'category') {
1402     v.category = image.category;
1403   } else if (prop === 'tags') {
1404     v.tags = image.tags;
1405   } else if (prop === 'keywords') {
1406     v.keywords = image.keywords;
1407   } else if (prop === 'description') {
1408     v.description = image.description;
1409   } else if (prop === 'copyright') {
1410     v.copyright = image.copyright;
1411   } else if (prop === 'license') {
1412     v.license = image.license;
1413   } else if (prop === 'author') {
1414     v.author = image.author;
1415   } else if (prop === 'created') {
1416     v.created = image.created;
1417   } else if (prop === 'updated') {
1418     v.updated = image.updated;
1419   } else if (prop === 'deleted') {
1420     v.deleted = image.deleted;
1421   } else if (prop === 'status') {
1422     v.status = image.status;
1423   } else if (prop === 'type') {
1424     v.type = image.type;
1425   } else if (prop === 'category') {
1426     v.category = image.category;
1427   } else if (prop === 'tags') {
1428     v.tags = image.tags;
1429   } else if (prop === 'keywords') {
1430     v.keywords = image.keywords;
1431   }
1432   meta2d.setValue(v, { render });
1433 };
1434
1435 export const updateGroup = (group: any, prop: string, render = true) => {
1436   const v: any = { id: group.id };
1437   v[prop] = group[prop];
1438   if (prop === 'x') {
1439     v.x = group.x;
1440   } else if (prop === 'y') {
1441     v.y = group.y;
1442   } else if (prop === 'width') {
1443     v.width = group.width;
1444   } else if (prop === 'height') {
1445     v.height = group.height;
1446   } else if (prop === 'angle') {
1447     v.angle = group.angle;
1448   } else if (prop === 'opacity') {
1449     v.opacity = group.opacity;
1450   } else if (prop === 'filter') {
1451     v.filter = group.filter;
1452   } else if (prop === 'src') {
1453     v.src = group.src;
1454   } else if (prop === 'alt') {
1455     v.alt = group.alt;
1456   } else if (prop === 'title') {
1457     v.title = group.title;
1458   } else if (prop === 'description') {
1459     v.description = group.description;
1460   } else if (prop === 'copyright') {
1461     v.copyright = group.copyright;
1462   } else if (prop === 'license') {
1463     v.license = group.license;
1464   } else if (prop === 'author') {
1465     v.author = group.author;
1466   } else if (prop === 'created') {
1467     v.created = group.created;
1468   } else if (prop === 'updated') {
1469     v.updated = group.updated;
1470   } else if (prop === 'deleted') {
1471     v.deleted = group.deleted;
1472   } else if (prop === 'status') {
1473     v.status = group.status;
1474   } else if (prop === 'type') {
1475     v.type = group.type;
1476   } else if (prop === 'category') {
1477     v.category = group.category;
1478   } else if (prop === 'tags') {
1479     v.tags = group.tags;
1480   } else if (prop === 'keywords') {
1481     v.keywords = group.keywords;
1482   }
1483   meta2d.setValue(v, { render });
1484 };
1485
1486 export const updateCanvas = (canvas: any, prop: string, render = true) => {
1487   const v: any = { id: canvas.id };
1488   v[prop] = canvas[prop];
1489   if (prop === 'x') {
1490     v.x = canvas.x;
1491   } else if (prop === 'y') {
1492     v.y = canvas.y;
1493   } else if (prop === 'width') {
1494     v.width = canvas.width;
1495   } else if (prop === 'height') {
1496     v.height = canvas.height;
1497   } else if (prop === 'angle') {
1498     v.angle = canvas.angle;
1499   } else if (prop === 'opacity') {
1500     v.opacity = canvas.opacity;
1501   } else if (prop === 'filter') {
1502     v.filter = canvas.filter;
1503   } else if (prop === 'src') {
1504     v.src = canvas.src;
1505   } else if (prop === 'alt') {
1506     v.alt = canvas.alt;
1507   } else if (prop === 'title') {
1508     v.title = canvas.title;
1509   } else if (prop === 'description') {
1510     v.description = canvas.description;
1511   } else if (prop === 'copyright') {
1512     v.copyright = canvas.copyright;
1513   } else if (prop === 'license') {
1514     v.license = canvas.license;
1515   } else if (prop === 'author') {
1516     v.author = canvas.author;
1517   } else if (prop === 'created') {
1518     v.created = canvas.created;
1519   } else if (prop === 'updated') {
1520     v.updated = canvas.updated;
1521   } else if (prop === 'deleted') {
1522     v.deleted = canvas.deleted;
1523   } else if (prop === 'status') {
1524     v.status = canvas.status;
1525   } else if (prop === 'type') {
1526     v.type = canvas.type;
1527   } else if (prop === 'category') {
1528     v.category = canvas.category;
1529   } else if (prop === 'tags') {
1530     v.tags = canvas.tags;
1531   } else if (prop === 'keywords') {
1532     v.keywords = canvas.keywords;
1533   }
1534   meta2d.setValue(v, { render });
1535 };
1536
1537 export const updatePage = (page: any, prop: string, render = true) => {
1538   const v: any = { id: page.id };
1539   v[prop] = page[prop];
1540   if (prop === 'x') {
1541     v.x = page.x;
1542   } else if (prop === 'y') {
1543     v.y = page.y;
1544   } else if (prop === 'width') {
1545     v.width = page.width;
1546   } else if (prop === 'height') {
1547     v.height = page.height;
1548   } else if (prop === 'angle') {
1549     v.angle = page.angle;
1550   } else if (prop === 'opacity') {
1551     v.opacity = page.opacity;
1552   } else if (prop === 'filter') {
1553     v.filter = page.filter;
1554   } else if (prop === 'src') {
1555     v.src = page.src;
1556   } else if (prop === 'alt') {
1557     v.alt = page.alt;
1558   } else if (prop === 'title') {
1559     v.title = page.title;
1560   } else if (prop === 'description') {
1561     v.description = page.description;
1562   } else if (prop === 'copyright') {
1563     v.copyright = page.copyright;
1564   } else if (prop === 'license') {
1565     v.license = page.license;
1566   } else if (prop === 'author') {
1567     v.author = page.author;
1568   } else if (prop === 'created') {
1569     v.created = page.created;
1570   } else if (prop === 'updated') {
1571     v.updated = page.updated;
1572   } else if (prop === 'deleted') {
1573     v.deleted = page.deleted;
1574   } else if (prop === 'status') {
1575     v.status = page.status;
1576   } else if (prop === 'type') {
1577     v.type = page.type;
1578   } else if (prop === 'category') {
1579     v.category = page.category;
1580   } else if (prop === 'tags') {
1581     v.tags = page.tags;
1582   } else if (prop === 'keywords') {
1583     v.keywords = page.keywords;
1584   }
1585   meta2d.setValue(v, { render });
1586 };
1587
1588 export const updatePageGroup = (pageGroup: any, prop: string, render = true) => {
1589   const v: any = { id: pageGroup.id };
1590   v[prop] = pageGroup[prop];
1591   if (prop === 'x') {
1592     v.x = pageGroup.x;
1593   } else if (prop === 'y') {
1594     v.y = pageGroup.y;
1595   } else if (prop === 'width') {
1596     v.width = pageGroup.width;
1597   } else if (prop === 'height') {
1598     v.height = pageGroup.height;
1599   } else if (prop === 'angle') {
1600     v.angle = pageGroup.angle;
1601   } else if (prop === 'opacity') {
1602     v.opacity = pageGroup.opacity;
1603   } else if (prop === 'filter') {
1604     v.filter = pageGroup.filter;
1605   } else if (prop === 'src') {
1606     v.src = pageGroup.src;
1607   } else if (prop === 'alt') {
1608     v.alt = pageGroup.alt;
1609   } else if (prop === 'title') {
1610     v.title = pageGroup.title;
1611   } else if (prop === 'description') {
1612     v.description = pageGroup.description;
1613   } else if (prop === 'copyright') {
1614     v.copyright = pageGroup.copyright;
1615   } else if (prop === 'license') {
1616     v.license = pageGroup.license;
1617   } else if (prop === 'author') {
1618     v.author = pageGroup.author;
1619   } else if (prop === 'created') {
1620     v.created = pageGroup.created;
1621   } else if (prop === 'updated') {
1622     v.updated = pageGroup.updated;
1623   } else if (prop === 'deleted') {
1624     v.deleted = pageGroup.deleted;
1625   } else if (prop === 'status') {
1626     v.status = pageGroup.status;
1627   } else if (prop === 'type') {
1628     v.type = pageGroup.type;
1629   } else if (prop === 'category') {
1630     v.category = pageGroup.category;
1631   } else if (prop === 'tags') {
1632     v.tags = pageGroup.tags;
1633   } else if (prop === 'keywords') {
1634     v.keywords = pageGroup.keywords;
1635   }
1636   meta2d.setValue(v, { render });
1637 };
1638
1639 export const updatePageImage = (pageImage: any, prop: string, render = true) => {
1640   const v: any = { id: pageImage.id };
1641   v[prop] = pageImage[prop];
1642   if (prop === 'x') {
1643     v.x = pageImage.x;
1644   } else if (prop === 'y') {
1645     v.y = pageImage.y;
1646   } else if (prop === 'width') {
1647     v.width = pageImage.width;
1648   } else if (prop === 'height') {
1649     v.height = pageImage.height;
1650   } else if (prop === 'angle') {
1651     v.angle = pageImage.angle;
1652   } else if (prop === 'opacity') {
1653     v.opacity = pageImage.opacity;
1654   } else if (prop === 'filter') {
1655     v.filter = pageImage.filter;
1656   } else if (prop === 'src') {
1657     v.src = pageImage.src;
1658   } else if (prop === 'alt') {
1659     v.alt = pageImage.alt;
1660   } else if (prop === 'title') {
1661     v.title = pageImage.title;
1662   } else if (prop === 'description') {
1663     v.description = pageImage.description;
1664   } else if (prop === 'copyright') {
1665     v.copyright = pageImage.copyright;
1666   } else if (prop === 'license') {
1667     v.license = pageImage.license;
1668   } else if (prop === 'author') {
1669     v.author = pageImage.author;
1670   } else if (prop === 'created') {
1671     v.created = pageImage.created;
1672   } else if (prop === 'updated') {
1673     v.updated = pageImage.updated;
1674   } else if (prop === 'deleted') {
1675     v.deleted = pageImage.deleted;
1676   } else if (prop === 'status') {
1677     v.status = pageImage.status;
1678   } else if (prop === 'type') {
1679     v.type = pageImage.type;
1680   } else if (prop === 'category') {
1681     v.category = pageImage.category;
1682   } else if (prop === 'tags') {
1683     v.tags = pageImage.tags;
1684   } else if (prop === 'keywords') {
1685     v.keywords = pageImage.keywords;
1686   }
1687   meta2d.setValue(v, { render });
1688 };
1689
1690 export const updatePageText = (pageText: any, prop: string, render = true) => {
1691   const v: any = { id: pageText.id };
1692   v[prop] = pageText[prop];
1693   if (prop === 'x') {
1694     v.x = pageText.x;
1695   } else if (prop === 'y') {
1696     v.y = pageText.y;
1697   } else if (prop === 'width') {
1698     v.width = pageText.width;
1699   } else if (prop === 'height') {
1700     v.height = pageText.height;
1701   } else if (prop === 'angle') {
1702     v.angle = pageText.angle;
1703   } else if (prop === 'opacity') {
1704     v.opacity = pageText.opacity;
1705   } else if (prop === 'filter') {
1706     v.filter = pageText.filter;
1707   } else if (prop === 'src') {
1708     v.src = pageText.src;
1709   } else if (prop === 'alt') {
1710     v.alt = pageText.alt;
1711   } else if (prop === 'title') {
1712     v.title = pageText.title;
1713   } else if (prop === 'description') {
1714     v.description = pageText.description;
1715   } else if (prop === 'copyright') {
1716     v.copyright = pageText.copyright;
1717   } else if (prop === 'license') {
1718     v.license = pageText.license;
1719   } else if (prop === 'author') {
1720     v.author = pageText.author;
1721   } else if (prop === 'created') {
1722     v.created = pageText.created;
1723   } else if (prop === 'updated') {
1724     v.updated = pageText.updated;
1725   } else if (prop === 'deleted') {
1726     v.deleted = pageText.deleted;
1727   } else if (prop === 'status') {
1728     v.status = pageText.status;
1729   } else if (prop === 'type') {
1730     v.type = pageText.type;
1731   } else if (prop === 'category') {
1732     v.category = pageText.category;
1733   } else if (prop === 'tags') {
1734     v.tags = pageText.tags;
1735   } else if (prop === 'keywords') {
1736     v.keywords = pageText.keywords;
1737   }
1738   meta2d.setValue(v, { render });
1739 };
1740
1741 export const updatePageImageGroup = (pageImageGroup: any, prop: string, render = true) => {
1742   const v: any = { id: pageImageGroup.id };
1743   v[prop] = pageImageGroup[prop];
1744   if (prop === 'x') {
1745     v.x = pageImageGroup.x;
1746   } else if (prop === 'y') {
1747     v.y = pageImageGroup.y;
1748   } else if (prop === 'width') {
1749     v.width = pageImageGroup.width;
1750   } else if (prop === 'height') {
1751     v.height = pageImageGroup.height;
1752   } else if (prop === 'angle') {
1753     v.angle = pageImageGroup.angle;
1754   } else if (prop === 'opacity') {
1755     v.opacity = pageImageGroup.opacity;
1756   } else if (prop === 'filter') {
1757     v.filter = pageImageGroup.filter;
1758   } else if (prop === 'src') {
1759     v.src = pageImageGroup.src;
1760   } else if (prop === 'alt') {
1761     v.alt = pageImageGroup.alt;
1762   } else if (prop === 'title') {
1763     v.title = pageImageGroup.title;
1764   } else if (prop === 'description') {
1765     v.description = pageImageGroup.description;
1766   } else if (prop === 'copyright') {
1767     v.copyright = pageImageGroup.copyright;
1768   } else if (prop === 'license') {
1769     v.license = pageImageGroup.license;
1770   } else if (prop === 'author') {
1771     v.author = pageImageGroup.author;
1772   } else if (prop === 'created') {
1773     v.created = pageImageGroup.created;
1774   } else if (prop === 'updated') {
1775     v.updated = pageImageGroup.updated;
1776   } else if (prop === 'deleted') {
1777     v.deleted = pageImageGroup.deleted;
1778   } else if (prop === 'status') {
1779     v.status = pageImageGroup.status;
1780   } else if (prop === 'type') {
1781     v.type = pageImageGroup.type;
1782   } else if (prop === 'category') {
1783     v.category = pageImageGroup.category;
1784   } else if (prop === 'tags') {
1785     v.tags = pageImageGroup.tags;
1786   } else if (prop === 'keywords') {
1787     v.keywords = pageImageGroup.keywords;
1788   }
1789   meta2d.setValue(v, { render });
1790 };
1791
1792 export const updatePageTextGroup = (pageTextGroup: any, prop: string, render = true) => {
1793   const v: any = { id: pageTextGroup.id };
1794   v[prop] = pageTextGroup[prop];
1795   if (prop === 'x') {
1796     v.x = pageTextGroup.x;
1797   } else if (prop === 'y') {
1798     v.y = pageTextGroup.y;
1799   } else if (prop === 'width') {
1800     v.width = pageTextGroup.width;
1801   } else if (prop === 'height') {
1802     v.height = pageTextGroup.height;
1803   } else if (prop === 'angle') {
1804     v.angle = pageTextGroup.angle;
1805   } else if (prop === 'opacity') {
1806     v.opacity = pageTextGroup.opacity;
1807   } else if (prop === 'filter') {
1808     v.filter = pageTextGroup.filter;
1809   } else if (prop === 'src') {
1810     v.src = pageTextGroup.src;
1811   } else if (prop === 'alt') {
1812     v.alt = pageTextGroup.alt;
1813   } else if (prop === 'title') {
1814     v.title = pageTextGroup.title;
1815   } else if (prop === 'description') {
1816     v.description = pageTextGroup.description;
1817   } else if (prop === 'copyright') {
1818     v.copyright = pageTextGroup.copyright;
1819   } else if (prop === 'license') {
1820     v.license = pageTextGroup.license;
1821   } else if (prop === 'author') {
1822     v.author = pageTextGroup.author;
1823   } else if (prop === 'created') {
1824     v.created = pageTextGroup.created;
1825   } else if (prop === 'updated') {
1826     v.updated = pageTextGroup.updated;
1827   } else if (prop === 'deleted') {
1828     v.deleted = pageTextGroup.deleted;
1829   } else if (prop === 'status') {
1830     v.status = pageTextGroup.status;
1831   } else if (prop === 'type') {
1832     v.type = pageTextGroup.type;
1833   } else if (prop === 'category') {
1834     v.category = pageTextGroup.category;
1835   } else if (prop === 'tags') {
1836     v.tags = pageTextGroup.tags;
1837   } else if (prop === 'keywords') {
1838     v.keywords = pageTextGroup.keywords;
1839   }
1840   meta2d.setValue(v, { render });
1841 };
1842
1843 export const updatePageImageGroupImage = (pageImageGroupImage: any, prop: string, render = true) => {
1844   const v: any = { id: pageImageGroupImage.id };
1845   v[prop] = pageImageGroupImage[prop];
1846   if (prop === 'x') {
1847     v.x = pageImageGroupImage.x;
1848   } else if (prop === 'y') {
1849     v.y = pageImageGroupImage.y;
1850   } else if (prop === 'width') {
1851     v.width = pageImageGroupImage.width;
1852   } else if (prop === 'height') {
1853     v.height = pageImageGroupImage.height;
1854   } else if (prop === 'angle') {
1855     v.angle = pageImageGroupImage.angle;
1856   } else if (prop === 'opacity') {
1857     v.opacity = pageImageGroupImage.opacity;
1858   } else if (prop === 'filter') {
1859     v.filter = pageImageGroupImage.filter;
1860   } else if (prop === 'src') {
1861     v.src = pageImageGroupImage.src;
1862   } else if (prop === 'alt') {
1863     v.alt = pageImageGroupImage.alt;
1864   } else if (prop === 'title') {
1865     v.title = pageImageGroupImage.title;
1866   } else if (prop === 'description') {
1867     v.description = pageImageGroupImage.description;
1868   } else if (prop === 'copyright') {
1869     v.copyright = pageImageGroupImage.copyright;
1870   } else if (prop === 'license') {
1871     v.license = pageImageGroupImage.license;
1872   } else if (prop === 'author') {
1873     v.author = pageImageGroupImage.author;
1874   } else if (prop === 'created') {
1875     v.created = pageImageGroupImage.created;
1876   } else if (prop === 'updated') {
1877     v.updated = pageImageGroupImage.updated;
1878   } else if (prop === 'deleted') {
1879     v.deleted = pageImageGroupImage.deleted;
1880   } else if (prop === 'status') {
1881     v.status = pageImageGroupImage.status;
1882   } else if (prop === 'type') {
1883     v.type = pageImageGroupImage.type;
1884   } else if (prop === 'category') {
1885     v.category = pageImageGroupImage.category;
1886   } else if (prop === 'tags') {
1887     v.tags = pageImageGroupImage.tags;
1888   } else if (prop === 'keywords') {
1889     v.keywords = pageImageGroupImage.keywords;
1890   }
1891   meta2d.setValue(v, { render });
1892 };
1893
1894 export const updatePageTextGroupImage = (pageTextGroupImage: any, prop: string, render = true) => {
1895   const v: any = { id: pageTextGroupImage.id };
1896   v[prop] = pageTextGroupImage[prop];
1897   if (prop === 'x') {
1898     v.x = pageTextGroupImage.x;
1899   } else if (prop === 'y') {
1900     v.y = pageTextGroupImage.y;
1901   } else if (prop === 'width') {
1902     v.width = pageTextGroupImage.width;
1903   } else if (prop === 'height') {
1904     v.height = pageTextGroupImage.height;
1905   } else if (prop === 'angle') {
1906     v.angle = pageTextGroupImage.angle;
1907   } else if (prop === 'opacity') {
1908     v.opacity = pageTextGroupImage.opacity;
1909   } else if (prop === 'filter') {
1910     v.filter = pageTextGroupImage.filter;
1911   } else if (prop === 'src') {
1912     v.src = pageTextGroupImage.src;
1913   } else if (prop === 'alt') {
1914     v.alt = pageTextGroupImage.alt;
1915   } else if (prop === 'title') {
1916     v.title = pageTextGroupImage.title;
1917   } else if (prop === 'description') {
1918     v.description = pageTextGroupImage.description;
1919   } else if (prop === 'copyright') {
1920     v.copyright = pageTextGroupImage.copyright;
1921   } else if (prop === 'license') {
1922     v.license = pageTextGroupImage.license;
1923   } else if (prop === 'author') {
1924     v.author = pageTextGroupImage.author;
1925   } else if (prop === 'created') {
1926     v.created = pageTextGroupImage.created;
1927   } else if (prop === 'updated') {
1928     v.updated = pageTextGroupImage.updated;
1929   } else if (prop === 'deleted') {
1930     v.deleted = pageTextGroupImage.deleted;
1931   } else if (prop === 'status') {
1932     v.status = pageTextGroupImage.status;
1933   } else if (prop === 'type') {
1934     v.type = pageTextGroupImage.type;
1935   } else if (prop === 'category') {
1936     v.category = pageTextGroupImage.category;
1937   } else if (prop === 'tags') {
1938     v.tags = pageTextGroupImage.tags;
1939   } else if (prop === 'keywords') {
1940     v.keywords = pageTextGroupImage.keywords;
1941   }
1942   meta2d.setValue(v, { render });
1943 };
1944
1945 export const updatePageImageGroupImageGroup = (pageImageGroupImageGroup: any, prop: string, render = true) => {
1946   const v: any = { id: pageImageGroupImageGroup.id };
1947   v[prop] = pageImageGroupImageGroup[prop];
1948   if (prop === 'x') {
1949     v.x = pageImageGroupImageGroup.x;
1950   } else if (prop === 'y') {
1951     v.y = pageImageGroupImageGroup.y;
1952   } else if (prop === 'width') {
1953     v.width = pageImageGroupImageGroup.width;
1954   } else if (prop === 'height') {
1955     v.height = pageImageGroupImageGroup.height;
1956   } else if (prop === 'angle') {
1957     v.angle = pageImageGroupImageGroup.angle;
1958   } else if (prop === 'opacity') {
1959     v.opacity = pageImageGroupImageGroup.opacity;
1960   } else if (prop === 'filter') {
1961     v.filter = pageImageGroupImageGroup.filter;
1962   } else if (prop === 'src') {
1963     v.src = pageImageGroupImageGroup.src;
1964   } else if (prop === 'alt') {
1965     v.alt = pageImageGroupImageGroup.alt;
1966   } else if (prop === 'title') {
1967     v.title = pageImageGroupImageGroup.title;
1968   } else if (prop === 'description') {
1969     v.description = pageImageGroupImageGroup.description;
1970   } else if (prop === 'copyright') {
1971     v.copyright = pageImageGroupImageGroup.copyright;
1972   } else if (prop === 'license') {
1973     v.license = pageImageGroupImageGroup.license;
1974   } else if (prop === 'author') {
1975     v.author = pageImageGroupImageGroup.author;
1976   } else if (prop === 'created') {
1977     v.created = pageImageGroupImageGroup.created;
1978   } else if (prop === 'updated') {
1979     v.updated = pageImageGroupImageGroup.updated;
1980   } else if (prop === 'deleted') {
1981     v.deleted = pageImageGroupImageGroup.deleted;
1982   } else if (prop === 'status') {
1983     v.status = pageImageGroupImageGroup.status;
1984   } else if (prop === 'type') {
1985     v.type = pageImageGroupImageGroup.type;
1986   } else if (prop === 'category') {
1987     v.category = pageImageGroupImageGroup.category;
1988   } else if (prop === 'tags') {
1989     v.tags = pageImageGroupImageGroup.tags;
1990   } else if (prop === 'keywords') {
1991     v.keywords = pageImageGroupImageGroup.keywords;
1992   }
1993   meta2d.setValue(v, { render });
1994 };
1995
1996 export const updatePageTextGroupImageGroup = (pageTextGroupImageGroup: any, prop: string, render = true) => {
1997   const v: any = { id: pageTextGroupImageGroup.id };
1998   v[prop] = pageTextGroupImageGroup[prop];
1999   if (prop === 'x') {
2000     v.x = pageTextGroupImageGroup.x;
2001   } else if (prop === 'y') {
2002     v.y = pageTextGroupImageGroup.y;
2003   } else if (prop === 'width') {
2004     v.width = pageTextGroupImageGroup.width;
2005   } else if (prop === 'height') {
2006     v.height = pageTextGroupImageGroup.height;
2007   } else if (prop === 'angle') {
2008     v.angle = pageTextGroupImageGroup.angle;
2009   } else if (prop === 'opacity') {
2010     v.opacity = pageTextGroupImageGroup.opacity;
2011   } else if (prop === 'filter') {
2012     v.filter = pageTextGroupImageGroup.filter;
2013   } else if (prop === 'src') {
2014     v.src = pageTextGroupImageGroup.src;
2015   } else if (prop === 'alt') {
2016     v.alt = pageTextGroupImageGroup.alt;
2017   } else if (prop === 'title') {
2018     v.title = pageTextGroupImageGroup.title;
```



```

4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100
101
102
103
104
105
106
107
108
109
110
111
112
113
114
115
116
117
118
119
120
121
122
123
124
125
126
127
128
129
130
131
132
133
134
135
136
137
138
139
140
141
142
143
144
145
146
147
148
149
150
151
152
153
154
155
156
157
158
159
160
161
162
163
164
165
166
167
168
169
170
171
172
173
174
175
176
177
178
179
180
181
182
183
184
185
186
187
188
189
190
191
192
193
194
195
196
197
198
199
200
201
202
203
204
205
206
207
208
209
210
211
212
213
214
215
216
217
218
219
220
221
222
223
224
225
226
227
228
229
230
231
232
233
234
235
236
237
238
239
240
241
242
243
244
245
246
247
248
249
250
251
252
253
254
255
256
257
258
259
260
261
262
263
264
265
266
267
268
269
270
271
272
273
274
275
276
277
278
279
280
281
282
283
284
285
286
287
288
289
290
291
292
293
294
295
296
297
298
299
300
301
302
303
304
305
306
307
308
309
310
311
312
313
314
315
316
317
318
319
320
321
322
323
324
325
326
327
328
329
330
331
332
333
334
335
336
337
338
339
340
341
342
343
344
345
346
347
348
349
350
351
352
353
354
355
356
357
358
359
360
361
362
363
364
365
366
367
368
369
370
371
372
373
374
375
376
377
378
379
380
381
382
383
384
385
386
387
388
389
390
391
392
393
394
395
396
397
398
399
400
401
402
403
404
405
406
407
408
409
410
411
412
413
414
415
416
417
418
419
420
421
422
423
424
425
426
427
428
429
430
431
432
433
434
435
436
437
438
439
440
441
442
443
444
445
446
447
448
449
450
451
452
453
454
455
456
457
458
459
460
461
462
463
464
465
466
467
468
469
470
471
472
473
474
475
476
477
478
479
480
481
482
483
484
485
486
487
488
489
490
491
492
493
494
495
496
497
498
499
500
501
502
503
504
505
506
507
508
509
510
511
512
513
514
515
516
517
518
519
520
521
522
523
524
525
526
527
528
529
530
531
532
533
534
535
536
537
538
539
540
541
542
543
544
545
546
547
548
549
550
551
552
553
554
555
556
557
558
559
560
561
562
563
564
565
566
567
568
569
570
571
572
573
574
575
576
577
578
579
580
581
582
583
584
585
586
587
588
589
590
591
592
593
594
595
596
597
598
599
600
601
602
603
604
605
606
607
608
609
610
611
612
613
614
615
616
617
618
619
620
621
622
623
624
625
626
627
628
629
630
631
632
633
634
635
636
637
638
639
640
641
642
643
644
645
646
647
648
649
650
651
652
653
654
655
656
657
658
659
660
661
662
663
664
665
666
667
668
669
670
671
672
673
674
675
676
677
678
679
680
681
682
683
684
685
686
687
688
689
690
691
692
693
694
695
696
697
698
699
700
701
702
703
704
705
706
707
708
709
710
711
712
713
714
715
716
717
718
719
720
721
722
723
724
725
726
727
728
729
730
731
732
733
734
735
736
737
738
739
740
741
742
743
744
745
746
747
748
749
750
751
752
753
754
755
756
757
758
759
760
761
762
763
764
765
766
767
768
769
770
771
772
773
774
775
776
777
778
779
780
781
782
783
784
785
786
787
788
789
790
791
792
793
794
795
796
797
798
799
800
801
802
803
804
805
806
807
808
809
810
811
812
813
814
815
816
817
818
819
820
821
822
823
824
825
826
827
828
829
830
831
832
833
834
835
836
837
838
839
840
841
842
843
844
845
846
847
848
849
850
851
852
853
854
855
856
857
858
859
860
861
862
863
864
865
866
867
868
869
870
871
872
873
874
875
876
877
878
879
880
881
882
883
884
885
886
887
888
889
890
891
892
893
894
895
896
897
898
899
900
901
902
903
904
905
906
907
908
909
910
911
912
913
914
915
916
917
918
919
920
921
922
923
924
925
926
927
928
929
930
931
932
933
934
935
936
937
938
939
940
941
942
943
944
945
946
947
948
949
950
951
952
953
954
955
956
957
958
959
960
961
962
963
964
965
966
967
968
969
970
971
972
973
974
975
976
977
978
979
980
981
982
983
984
985
986
987
988
989
990
991
992
993
994
995
996
997
998
999
1000

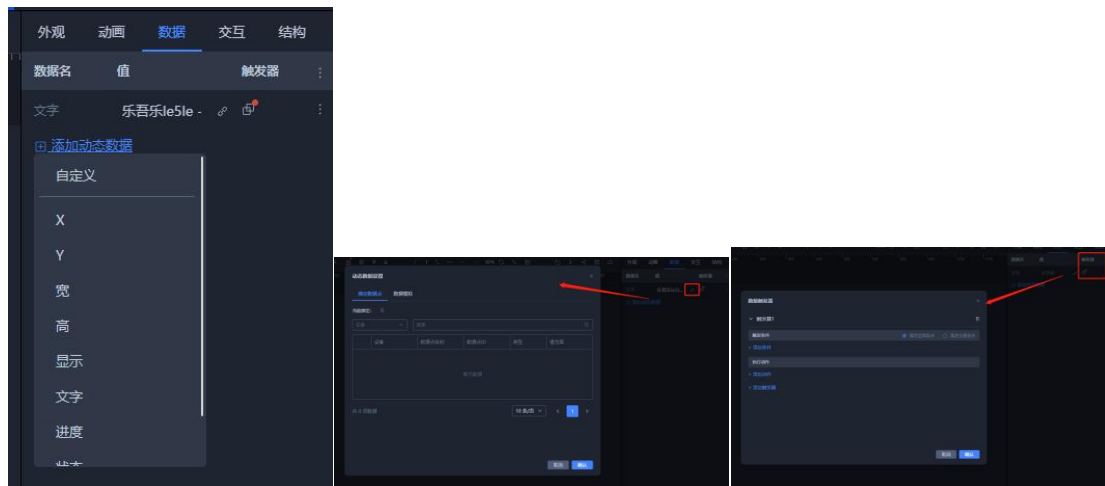
```

animations 属性用于存放该节点配置的多个动画，最终动画的执行是由 frames 属性控制的，通过 startAnimate/stopAnimate 方法控制动画的执行/停止。动画相关属性介绍可查看文档：

<https://doc.le5le.com/document/119895613>

3. 数据（components/PenDatas.vue）

数据主要是绑定数据点和设置（值变化）触发器，主要是修改 pen 的 realTimes 属性，具体可以查看文档：<https://doc.le5le.com/document/135786389>



4. 交互（components/PenEvents.vue）

交互主要是配置节点的交互事件，主要修改的是 pen 的 events 属性，具体可以查看文档：<https://doc.le5le.com/document/119627190>



5. 结构

同 不选中图元 结构板块

5.1.3.3 多选图元

① 目录 components/PensProps.vue

② 源码说明：



外观主要包括对多个图元的统一锁定、显示状态的修改，对齐操作以及外观、文字等一些公共样式的统一修改。

锁定/显示主要通过调用核心库 `setValue`、`setVisible` 方法。

```
const lock = (v: LockState) => {
  data.locked = v;
  for (const item of selections.pens) {
    meta2d.setValue({
      id: item.id,
      locked: v,
    });
  }
};

const visible = (v: boolean) => {
  data.visible = v;
  for (const item of selections.pens) {
    meta2d.setVisible(item as any, v);
  }
};
```

对齐操作也是直接调用核心库开源方法，具体方法说明可见文档：

<https://doc.le5le.com/document/119882449#%E5%87%BD%E6%95%B0>

```
const align = (align: string) => {
  if (align === 'h-distribute') {
    meta2d.spaceBetween(meta2d.store.active);
  } else if (align === 'v-distribute') {
    meta2d.spaceBetweenColumn(meta2d.store.active);
  } else {
    meta2d.alignNodes(align, meta2d.store.active);
  }
};

const align2 = (align: string) => {
  if (align === 'same-size') {
    meta2d.beSameByLast(meta2d.store.active);
  } else {
    meta2d.alignNodesByLast(align, meta2d.store.active);
  }
};
```

修改公共样式和上面修改单个图元样式一样，调用核心库 `setValue` 方法，特殊属性提前处理。注意这里遍历所有 `pen` 进行 `setValue` 是没有直接 `render` 的，最后再统一 `render`。具体可以参考文档：

<https://doc.le5le.com/document/119882449#setValue>


```
src > views > components > PenPop.vue > {} script setup > @ changeValue
864
865
866 const align = (align: string) => {
867   if (align === 'h-distribute') {
868     meta2d.spaceBetween(meta2d.store.active);
869   } else if (align === 'v-distribute') {
870     meta2d.spacebetweenColumn(meta2d.store.active);
871   } else {
872     meta2d.alignNodes(align, meta2d.store.active);
873   }
874 };
875
876 const align2 = (align: string) => {
877   if (align === 'same-size') {
878     meta2d.besameByLast(meta2d.store.active);
879   } else {
880     meta2d.alignNodesByLast(align, meta2d.store.active);
881   }
882 };
883
884 const changeValue = (prop: string) => {
885   for (const item of selections.pens) {
886     data.id = item.id;
887     updatePen(data, prop, false);
888   }
889   meta2d.render();
890 };
891
892 const onFontFamily = (fontfamily: string) => {
893   data.fontfamily = fontfamily;
894   data.fontfamilypopupVisible = false;
895   changeValue('fontfamily');
896 };
897
898 const onFontPopupVisible = (val: boolean) => {
899   data.fontfamilypopupVisible = val;
900 };
901
902
903
904
905
906
907
908
909
910
911
912
913
914
915
916
917
918
919
920
921
922
923
924
925
926
927
928
929
930
931
932
933
934
935
936
937
938
939
940
941
942
943
944
945
946
947
948
949
950
951
952
953
954
955
956
957
958
959
960
961
962
963
964
965
966
967
968
969
970
971
972
973
974
975
976
977
978
979
980
981
982
983
984
985
986
987
988
989
990
991
992
993
994
995
996
997
998
999
1000
1001
1002
1003
1004
1005
1006
1007
1008
1009
1010
1011
1012
1013
1014
1015
1016
1017
1018
1019
1020
1021
1022
1023
1024
1025
1026
1027
1028
1029
1030
1031
1032
1033
1034
1035
1036
1037
1038
1039
1040
1041
1042
1043
1044
1045
1046
1047
1048
1049
1050
1051
1052
1053
1054
1055
1056
1057
1058
1059
1060
1061
1062
1063
1064
1065
1066
1067
1068
1069
1070
1071
1072
1073
1074
1075
1076
1077
1078
1079
1080
1081
1082
1083
1084
1085
1086
1087
1088
1089
1090
1091
1092
1093
1094
1095
1096
1097
1098
1099
1100
1101
1102
1103
1104
1105
1106
1107
1108
1109
1110
1111
1112
1113
1114
1115
1116
1117
1118
1119
1120
1121
1122
1123
1124
1125
1126
1127
1128
1129
1130
1131
1132
1133
1134
1135
1136
1137
1138
1139
1140
1141
1142
1143
1144
1145
1146
1147
1148
1149
1150
1151
1152
1153
1154
1155
1156
1157
1158
1159
1160
1161
1162
1163
1164
1165
1166
1167
1168
1169
1170
1171
1172
1173
1174
1175
1176
1177
1178
1179
1180
1181
1182
1183
1184
1185
1186
1187
1188
1189
1190
1191
1192
1193
1194
1195
1196
1197
1198
1199
1200
1201
1202
1203
1204
1205
1206
1207
1208
1209
1210
1211
1212
1213
1214
1215
1216
1217
1218
1219
1220
1221
1222
1223
1224
1225
1226
1227
1228
1229
1230
1231
1232
1233
1234
1235
1236
1237
1238
1239
1240
1241
1242
1243
1244
1245
1246
1247
1248
1249
1250
1251
1252
1253
1254
1255
1256
1257
1258
1259
1260
1261
1262
1263
1264
1265
1266
1267
1268
1269
1270
1271
1272
1273
1274
1275
1276
1277
1278
1279
1280
1281
1282
1283
1284
1285
1286
1287
1288
1289
1290
1291
1292
1293
1294
1295
1296
1297
1298
1299
1300
1301
1302
1303
1304
1305
1306
1307
1308
1309
1310
1311
1312
1313
1314
1315
1316
1317
1318
1319
1320
1321
1322
1323
1324
1325
1326
1327
1328
1329
1330
1331
1332
1333
1334
1335
1336
1337
1338
1339
1340
1341
1342
1343
1344
1345
1346
1347
1348
1349
1350
1351
1352
1353
1354
1355
1356
1357
1358
1359
1360
1361
1362
1363
1364
1365
1366
1367
1368
1369
1370
1371
1372
1373
1374
1375
1376
1377
1378
1379
1380
1381
1382
1383
1384
1385
1386
1387
1388
1389
1390
1391
1392
1393
1394
1395
1396
1397
1398
1399
1400
1401
1402
1403
1404
1405
1406
1407
1408
1409
1410
1411
1412
1413
1414
1415
1416
1417
1418
1419
1420
1421
1422
1423
1424
1425
1426
1427
1428
1429
1430
1431
1432
1433
1434
1435
1436
1437
1438
1439
1440
1441
1442
1443
1444
1445
1446
1447
1448
1449
1450
1451
1452
1453
1454
1455
1456
1457
1458
1459
1460
1461
1462
1463
1464
1465
1466
1467
1468
1469
1470
1471
1472
1473
1474
1475
1476
1477
1478
1479
1480
1481
1482
1483
1484
1485
1486
1487
1488
1489
1490
1491
1492
1493
1494
1495
1496
1497
1498
1499
1500
1501
1502
1503
1504
1505
1506
1507
1508
1509
1510
1511
1512
1513
1514
1515
1516
1517
1518
1519
1520
1521
1522
1523
1524
1525
1526
1527
1528
1529
1530
1531
1532
1533
1534
1535
1536
1537
1538
1539
1540
1541
1542
1543
1544
1545
1546
1547
1548
1549
1550
1551
1552
1553
1554
1555
1556
1557
1558
1559
1560
1561
1562
1563
1564
1565
1566
1567
1568
1569
1570
1571
1572
1573
1574
1575
1576
1577
1578
1579
1580
1581
1582
1583
1584
1585
1586
1587
1588
1589
1590
1591
1592
1593
1594
1595
1596
1597
1598
1599
1600
1601
1602
1603
1604
1605
1606
1607
1608
1609
1610
1611
1612
1613
1614
1615
1616
1617
1618
1619
1620
1621
1622
1623
1624
1625
1626
1627
1628
1629
1630
1631
1632
1633
1634
1635
1636
1637
1638
1639
1640
1641
1642
1643
1644
1645
1646
1647
1648
1649
1650
1651
1652
1653
1654
1655
1656
1657
1658
1659
1660
1661
1662
1663
1664
1665
1666
1667
1668
1669
1670
1671
1672
1673
1674
1675
1676
1677
1678
1679
1680
1681
1682
1683
1684
1685
1686
1687
1688
1689
1690
1691
1692
1693
1694
1695
1696
1697
1698
1699
1700
1701
1702
1703
1704
1705
1706
1707
1708
1709
1710
1711
1712
1713
1714
1715
1716
1717
1718
1719
1720
1721
1722
1723
1724
1725
1726
1727
1728
1729
1730
1731
1732
1733
1734
1735
1736
1737
1738
1739
1740
1741
1742
1743
1744
1745
1746
1747
1748
1749
1750
1751
1752
1753
1754
1755
1756
1757
1758
1759
1760
1761
1762
1763
1764
1765
1766
1767
1768
1769
1770
1771
1772
1773
1774
1775
1776
1777
1778
1779
1780
1781
1782
1783
1784
1785
1786
1787
1788
1789
1790
1791
1792
1793
1794
1795
1796
1797
1798
1799
1800
1801
1802
1803
1804
1805
1806
1807
1808
1809
1810
1811
1812
1813
1814
1815
1816
1817
1818
1819
1820
1821
1822
1823
1824
1825
1826
1827
1828
1829
1830
1831
1832
1833
1834
1835
1836
1837
1838
1839
1840
1841
1842
1843
1844
1845
1846
1847
1848
1849
1850
1851
1852
1853
1854
1855
1856
1857
1858
1859
1860
1861
1862
1863
1864
1865
1866
1867
1868
1869
1870
1871
1872
1873
1874
1875
1876
1877
1878
1879
1880
1881
1882
1883
1884
1885
1886
1887
1888
1889
1890
1891
1892
1893
1894
1895
1896
1897
1898
1899
1900
1901
1902
1903
1904
1905
1906
1907
1908
1909
1910
1911
1912
1913
1914
1915
1916
1917
1918
1919
1920
1921
1922
1923
1924
1925
1926
1927
1928
1929
1930
1931
1932
1933
1934
1935
1936
1937
1938
1939
1940
1941
1942
1943
1944
1945
1946
1947
1948
1949
1950
1951
1952
1953
1954
1955
1956
1957
1958
1959
1960
1961
1962
1963
1964
1965
1966
1967
1968
1969
1970
1971
1972
1973
1974
1975
1976
1977
1978
1979
1980
1981
1982
1983
1984
1985
1986
1987
1988
1989
1990
1991
1992
1993
1994
1995
1996
1997
1998
1999
2000
2001
2002
2003
2004
2005
2006
2007
2008
2009
2010
2011
2012
2013
2014
2015
2016
2017
2018
2019
2020
2021
2022
2023
2024
2025
2026
2027
2028
2029
2030
2031
2032
2033
2034
2035
2036
2037
2038
2039
2040
2041
2042
2043
2044
2045
2046
2047
2048
2049
2050
2051
2052
2053
2054
2055
2056
2057
2058
2059
2060
2061
2062
2063
2064
2065
2066
2067
2068
2069
2070
2071
2072
2073
2074
2075
2076
2077
2078
2079
2080
2081
2082
2083
2084
2085
2086
2087
2088
2089
2090
2091
2092
2093
2094
2095
2096
2097
2098
2099
2100
2101
2102
2103
2104
2105
2106
2107
2108
2109
2110
2111
2112
2113
2114
2115
2116
2117
2118
2119
2120
2121
2122
2123
2124
2125
2126
2127
2128
2129
2130
2131
2132
2133
2134
2135
2136
2137
2138
2139
2140
2141
2142
2143
2144
2145
2146
2147
2148
2149
2150
2151
2152
2153
2154
2155
2156
2157
2158
2159
2160
2161
2162
2163
2164
2165
2166
2167
2168
2169
2170
2171
2172
2173
2174
2175
2176
2177
2178
2179
2180
2181
2182
2183
2184
2185
2186
2187
2188
2189
2190
2191
2192
2193
2194
2195
2196
2197
2198
2199
2200
2201
2202
2203
2204
2205
2206
2207
2208
2209
2210
2211
2212
2213
2214
2215
2216
2217
2218
2219
2220
2221
2222
2223
2224
2225
2226
2227
2228
2229
2230
2231
2232
2233
2234
2235
2236
2237
2238
2239
2240
2241
2242
2243
2244
2245
2246
2247
2248
2249
2250
2251
2252
2253
2254
2255
2256
2257
2258
2259
2260
2261
2262
2263
2264
2265
2266
2267
2268
2269
2270
2271
2272
2273
2274
2275
2276
2277
2278
2279
2280
2281
2282
2283
2284
2285
2286
2287
2288
2289
2290
2291
2292
2293
2294
2295
2296
2297
2298
2299
2300
2301
2302
2303
2304
2305
2306
2307
2308
2309
2310
2311
2312
2313
2314
2315
2316
2317
2318
2319
2320
2321
2322
2323
2324
2325
2326
2327
2328
2329
2330
2331
2332
2333
2334
2335
2336
2337
2338
2339
2340
2341
2342
2343
2344
2345
2346
2347
2348
2349
2350
2351
2352
2353
2354
2355
2356
2357
2358
2359
2360
2361
2362
2363
2364
2365
2366
2367
2368
2369
2370
2371
2372
2373
2374
2375
2376
2377
2378
2379
2380
2381
2382
2383
2384
2385
2386
2387
2388
2389
2390
2391
2392
2393
2394
2395
2396
2397
2398
2399
2400
2401
2402
2403
2404
2405
2406
2407
2408
2409
2410
2411
2412
2413
2414
2415
2416
2417
2418
2419
2420
2421
2422
2423
2424
2425
2426
2427
2428
2429
2430
2431
2432
2433
2434
2435
2436
2437
2438
2439
2440
2441
2442
2443
2444
2445
2446
2447
2448
2449
2450
2451
2452
2453
2454
2455
2456
2457
2458
2459
2460
2461
2462
2463
2464
2465
2466
2467
2468
2469
2470
2471
2472
2473
2474
2475
2476
2477
2478
2479
2480
2481
2482
2483
2484
2485
2486
2487
2488
2489
2490
2491
2492
2493
2494
2495
2496
2497
2498
2499
2500
2501
2502
2503
2504
2505
2506
2507
2508
2509
2510
2511
2512
2513
2514
2515
2516
2517
2518
2519
2520
2521
2522
2523
2524
2525
2526
2527
2528
2529
2530
2531
2532
2533
2534
2535
2536
2537
2538
2539
2540
2541
2542
2543
2544
2545
2546
2547
2548
2549
2550
2551
2552
2553
2554
2555
2556
2557
2558
2559
2560
2561
2562
2563
2564
2565
2566
2567
2568
2569
2570
2571
2572
2573
2574
2575
2576
2577
2578
2579
2580
2581
2582
2583
2584
2585
2586
2587
2588
2589
2590
2591
2592
2593
2594
2595
2596
2597
2598
2599
2600
2601
2602
2603
2604
2605
2606
2607
2608
2609
2610
2611
2612
2613
2614
2615
2616
2617
2618
2619
2620
2621
2622
2623
2624
2625
2626
2627
2628
2629
2630
2631
2632
2633
2634
2635
2636
2637
2638
2639
2640
2641
2642
2643
2644
2645
2646
2647
2648
2649
2650
2651
2652
2653
2654
2655
2656
2657
2658
2659
2660
2661
2662
2663
2664
2665
2666
2667
2668
2669
2670
2671
2672
2673
2674
2675
2676
2677
2678
2679
2680
2681
2682
2683
2684
2685
2686
2687
2688
2689
2690
2691
2692
2693
2694
2695
2696
2697
2698
2699
2700
2701
2702
2703
2704
2705
2706
2707
2708
2709
2710
2711
2712
2713
2714
2715
2716
2717
2718
2719
2720
2721
2722
2723
2724
2725
2726
2727
2728
2729
2730
2731
2732
2733
2734
2735
2736
2737
2738
2739
2740
2741
2742
2743
2744
2745
2746
2747
2748
2749
2750
2751
2752
2753
2754
2755
2756
2757
2758
2759
2760
2761
2762
2763
2764
2765
2766
2767
2768
2769
2770
2771
2772
2773
2774
2775
2776
2777
2778
2779
2780
2781
2782
2783
2784
2785
2786
2787
2788
2789
2790
2791
2792
2793
2794
2795
2796
2797
2798
2799
2800
2801
2802
2803
2804
2805
2806
2807
2808
2809
2810
2811
2812
2813
2814
2815
2816
2817
2818
2819
2820
2821
2822
2823
2824
2825
2826
2827
2828
2829
2830
2831
2832
2833
2834
2835
2836
2837
2838
2839
2840
2841
2842
2843
2844
2845
2846
2847
2848
2849
2850
2851
2852
2853
2854
2855
2856
2857
2858
2859
2860
2861
2862
2863
2864
2865
2866
2867
2868
2869
2870
2871
2872
2873
2874
2875
2876
2877
2878
2879
2880
2881
2882
2883
2884
2885
2886
2887
2888
2889
2890
2891
2892
2893
2894
2895
2896
2897
2898
2899
2900
2901
2902
2903
2904
2905
2906
2907
2908
2909
2910
2911
2912
2913
2914
2915
2916
2917
2918
2919
2920
2921
2922
2923
2924
2925
2926
2927
2928
2929
2930
2931
2932
2933
2934
2935
2936
2937
2938
2939
2940
2941
2942
2943
2944
2945
2946
2947
2948
2949
2950
2951
2952
2953
2954
2955
2956
2957
2958
2959
2960
2961
2962
2963
2964
2965
2966
2967
2968
2969
2970
2971
2972
2973
2974
2975
2976
2977
2978
2979
2980
2981
2982
2983
2984
2985
2986
2987
2988
2989
2990
2991
2992
2993
2994
2995
2996
2997
2998
2999
3000
3001
3002
3003
3004
3005
3006
3007
3008
3009
3010
3011
3012
3013
3014
3015
3016
3017
3018
3019
3020
3021
3022
3023
3024
3025
3026
3027
3028
3029
3030
3031
3032
3033
3034
3035
3036
3037
3038
3039
3040
3041
3042
3043
3044
3045
3046
3047
3048
3049
3050
3051
3052
3053
3054
3055
3056
3057
3058
3059
3060
3061
3062
3063
3064
3065
3066
3067
3068
3069
3070
3071
3072
3073
3074
3075
3076
3077
3078
3079
3080
3081
3082
3083
3084
3085
3086
3087
3088
3089
3090
3091
3092
3093
3094
3095
3096
3097
3098
3099
3100
3101
3102
3103
3104
3105
3106
3107
3108
3109
3110
3111
3112
3113
3114
3115
3116
3117
3118
3119
3120
3121
3122
3123
3124
3125
3126
3127
3128
3129
3130
3131
3132
3133
3134
3135
3136
3137
3138
3139
3140
3141
3142
3143
3144
3145
3146
3147
3148
3149
3150
3151
3152
3153
3154
3155
3156
3157
3158
3159
3160
3161
3162
3163
3164
3165
3166
3167
3168
3169
3170
3171
3172
3173
3174
3175
3176
3177
3178
3179
3180
3181
3182
3183
3184
3185
3186
3187
3188
3189
3190
3191
3192
3193
3194
3195
3196
3197
3198
3199
3200
3201
3202
3203
3204
3205
3206
3207
3208
3209
3210
3211
3212
3213
3214
3215
3216
3217
3218
3219
3220
3221
3222
3223
3224
3225
3226
3227
3228
3229
3230
3231
3232
3233
3234
3235
3236
3237
3238
3239
3240
3241
3242
3243
3244
3245
3246
3247
3248
3249
3250
3251
3252
3253
3254
3255
3256
3257
3258
3259
3260
3261
3262
3263
3264
3265
3266
3267
3268
3269
3270
3271
3272
3273
3274
3275
3276
3277
3278
3279
3280
3281
3282
3283
3284
3285
3286
3287
3288
3289
3290
3291
3292
3293
3294
3295
3296
3297
3298
3299
3300
3301
3302
3303
3304

```

六 目录介绍

6.1 Public 公共静态资源目录

public/icon 项目图标库

public/img 项目图片资源存放

public/js 项目需要的一些离线资源包，例如 echarts 包（echarts.min.js）

public/theme echarts 图形库主题文件存放位置，具体可以查看：

<https://echarts.apache.org/zh/theme-builder.html>

public/view 编辑器“下载离线部署包”/“组件”等功能所需要的运行环境文件

Public/data.xlsx 数据集 Excel 示例

Public/favicon.ico 左上角 logo

Public/rotate.cur 图形节点旋转时鼠标样式

6.2 Src 开发目录

Src/assets 静态资源，fonts 下是系统字体

Src/router 路由配置

Src/services 公用服务（公用方法）

Src/styles 公用样式文件

Src/view 主要的页面，首页和预览页面

Src/view/components 组件

Src/App.vue 主组件

Src/global.d.ts 可以从全局范围访问的库

Src/http.ts axios 配置和网络请求/响应拦截器

Src/main.ts 入口 ts 文件

6.3 其他

Index.html 入口 html 文件

Package.json 项目依赖描述

postcss.config.js postcss 配置文件

Tsconfig.json ts 配置文件

Vite.config.ts vite 配置文件

七 运行流程

Vue 项目运行流程：index.html > App.vue 的 export 外的 js 代码 > main.js > App.vue 的 export 里面的 js 代码

想了解更多请学习 vue: <https://v3.cn.vuejs.org/>

快速修改代码：用 VS code 搜索关键字

八 代码中实现登录链接

8.1 完全自己实现后端

可以参考后端 API 接口文档：<https://doc.le5le.com/document/7>

8.2 购买了乐吾乐后端

参考后端使用手册运行部署后端。然后通过前后端分离模式部署即可

九 部署集成

因为大屏编辑器代码比较重，推荐直接独立部署，通过域名访问或者 iframe 嵌入。