**ENVisual前端设计文档**

V0.0.1

陈兴林 @ Enlink 数据中心

**目录**

[1. 概述 3](#_Toc534286540)

[2. 数据中心整体结构 3](#_Toc534286541)

[2.1 server服务端数据 3](#_Toc534286542)

[2.2 api管理 3](#_Toc534286543)

[2.3 store数据仓库 4](#_Toc534286544)

[2.4 前端组件 4](#_Toc534286545)

[3. 前端可视化总体结构描述 4](#_Toc534286546)

[3.1. Api管理 4](#_Toc534286547)

[3.1.1. url管理 5](#_Toc534286548)

[3.1.2. server服务端请求 5](#_Toc534286549)

[3.2. store数据仓库 5](#_Toc534286550)

[3.3. 前端组件 6](#_Toc534286551)

[4. 前端可视化功能点说明 8](#_Toc534286552)

[4.1. Login登录页 8](#_Toc534286553)

[4.2. 中英文 8](#_Toc534286554)

[4.3. 模块 9](#_Toc534286555)

[4.4. 面板 10](#_Toc534286556)

[4.5. 图表 13](#_Toc534286557)

[4.6. 数据源 14](#_Toc534286558)

[5. 组件关系图 15](#_Toc534286559)

# 概述

本文档描述ENVisual（Enlink数据可视化）产品的基本流程以及各模块需要设计的组件，依赖以及基本的页面交互。

交互设计详细说明本文档不做阐述，不涉及详细的每一个过程的说明。另外，UID设计也不在该文档范围内，本文档不对UI界面设计做任何描述。

前端架构请查阅《ENVisual-前端架构设计文档-V0.0.1》

本文档依赖《ENVisual-功能需求文档-V0.0.1》文档进行设计，对需求不了解，请查阅次文档。

整个项目基于vue技术站，使用前后端分离式开发模式。

# 架构设计

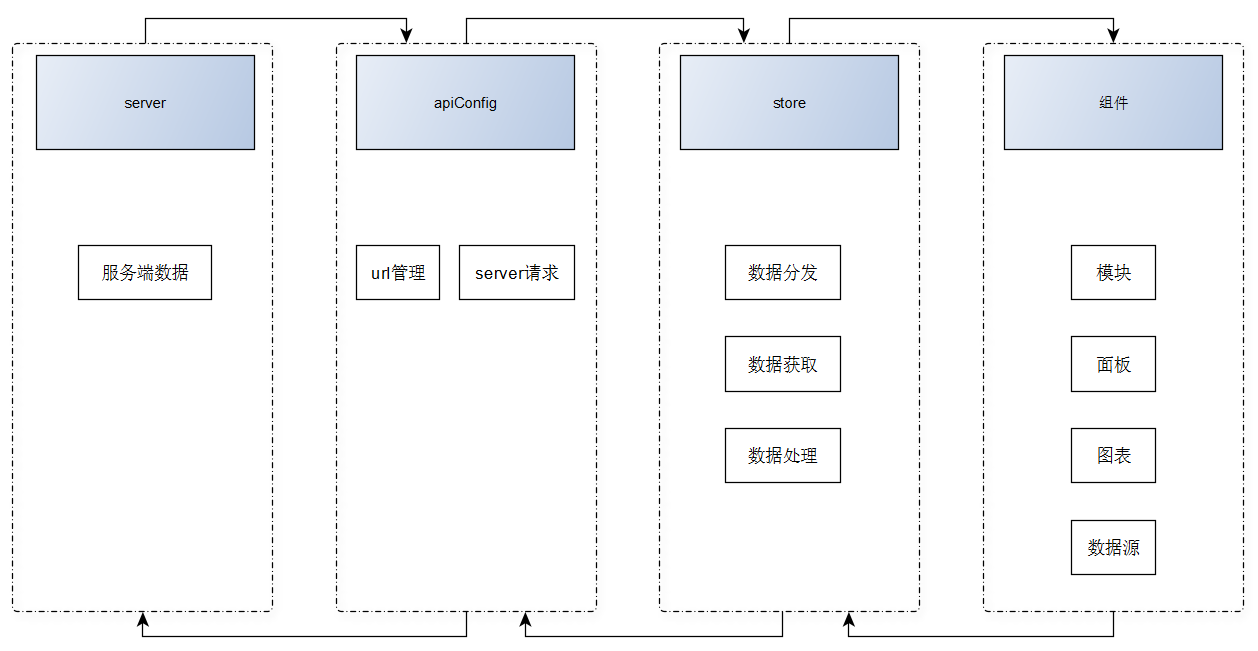


图2-1 数据中心可视化整体结构图

如上图2-1所示，整个项目分3个大功能块----api管理、store数据仓库和前端可视化组件。三个大功能块又依赖于服务端数据，形成项目完整的整体。

## 2.1 server服务端数据

Server服务端数据，由后端或第三方提供，不在前端项目实现范围内。在此不做服务端数据的过多说明，前端只需请求后端数据接口，进行数据交互即可。

## 2.2 api管理

如图2-2-1所示，api管理是基于凌乱的后端api接口url进行管理。同时，统一封装对后端数据的交互。

## 2.3 store数据仓库

数据仓库主要是copy服务端数据，然后统一处理后分发到前端组件。实现数据统一调度，方便数据全局共享。

## 2.4 前端组件

前端组件主要实现数据中心可视化相关的功能，也是整个项目呈现给客户的部分。将是前端项目最为重要的部分。

# 架构说明

## Api管理

Api管理分两个不分，其结构如下图3-1-1所示：

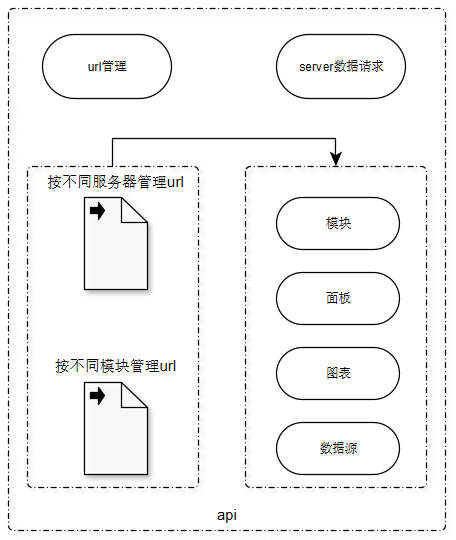


图 3-1-1 api管理

### url管理

url管理选择按模块进行划分管理，其实现原理是通过导出一个统一的模块变量，变量绑定相关属性，并储存响应的url地址。供server数据请求方法使用。

url管理设计方便前端协作开发人员统一添加url，实现高效率的协作开发。

### server服务端请求

server数据请求，跟url管理选择同一模式，按不同功能模块进行划分。每个功能模单独对数据进行请求。

其实现原理为：封装一个fetch函数，函数内调用axios并返回一个promise。每个与服务器交互的行为都通过该函数，并引入统一url管理文件，拿到相应模块的url传递给fetch函数，并导出返回结果的变量供数据仓库导入。

## store数据仓库

如图3-2-1所示：

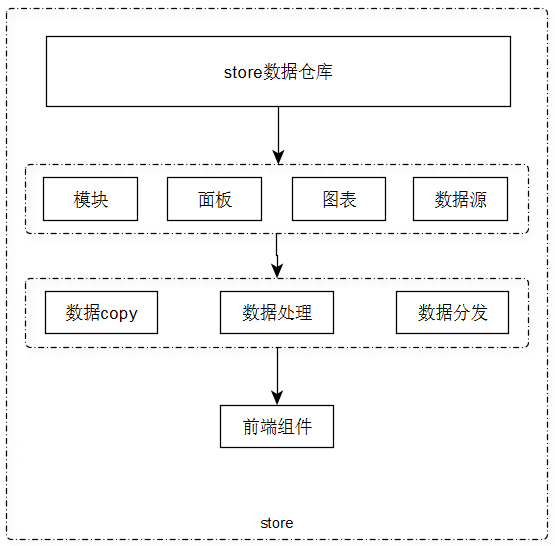


图3-2-1 store数据仓库

数据仓库结合vuex插件，划分为4个大模块，分别为模块（指可视化的一个功能）、面板、图表、数据源。4个模块分别实现响应的state、mutations、actions和getters。

统一在actions中对数据做异步请求，统一在mutations中对数据进行copy和缓存，统一通过getters对数据进行处理和分发。

前端通过触发响应的map对数据进行交互和共享。

## 前端功能点

前端组件是本项目最重要的部分，也是最难的部分，前端可视化的部分设计将直接影响开发进度。需要对整个项目需求进行透彻的理解，以及对整个项目实现逻辑有一个准确的把控。以下是前端功能点示意图：

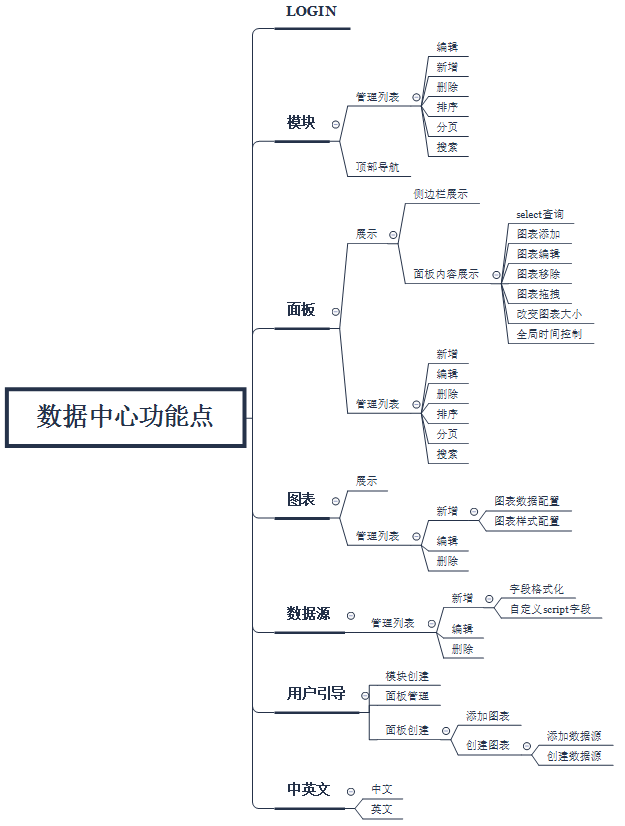


图 3-3-1 数据中心可视化功能点 示意图

如上图：3-3-1所示

数据中心主要分为以下几个大的功能点：

1. Login登录页，是整个可视化系统的入口

2. 中英文，实现中英文切换功能

3. 模块，对模块进行管理

4. 面板，管理响应的面板，展示响应的数据给客户

5. 图表，对面板要使用的图表进行管理

6. 数据源，管理图表所需的相关数据源

# 功能点说明

## Login登录页

登录页实现用户登录功能，用户登录验证用户信息是否合法。

## 中英文

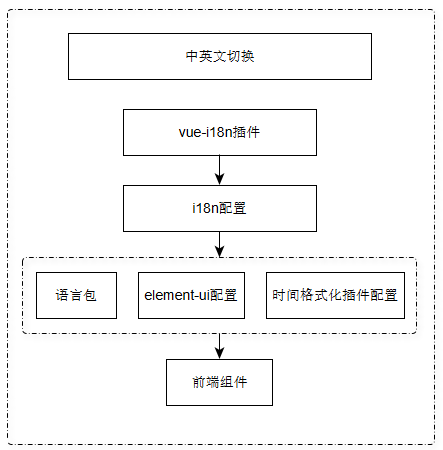


图4-2-1中英文切换

如图：4-2-1所示

中英文切换功能结合vue-i18n插件，实现整个项目中英文切换。

中英文切换功能，封装i18n配置，并配置element-ui中英文切换、moment时间格式化中英文切换。通过注入配置文件到项目可视化前端组件，统一对实现项目所有页面的中英文切换功能。

## 模块

如图：4-3-1所示

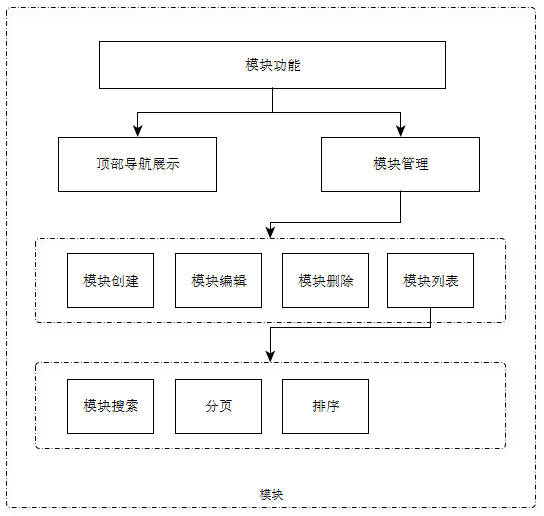


图4-3-1 模块功能 示意图

模块功能，主要分为两部分

1. 在顶部导航展示
2. 模块管理

模块管理又分为4个部分：

1. 模块创建，模块创建后将直接更新顶部模块导航，并更新模块列表。
2. 模块编辑更新，模块信息编辑更新成功后，更新模块在列表中的信息。
3. 模块删除
4. 模块列表，模块列表展示模块的基础信息，并提供单条模块数据编辑、查看入口。

模块列表实现三个次要功能：

1. 可在搜索框中搜索过滤模块数据
2. 模块数据可进行分页展示
3. 模块在列表中可进行简单的排序

## 面板

如图：

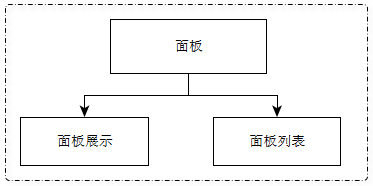


图 4-4-1 面板功能

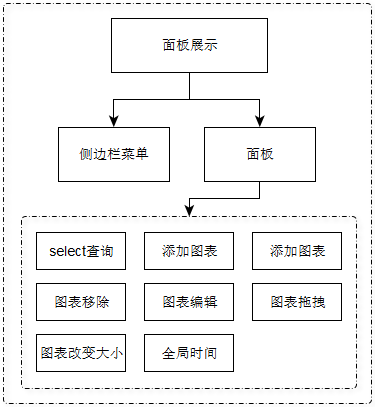


图 4-4-2 面板展示

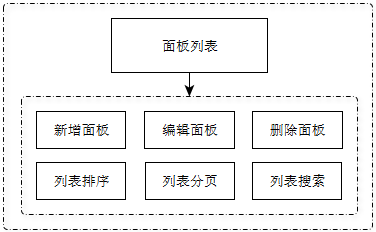


图 4-4-3 面板列表

**如图4-4-1，4-4-2，4-4-3所示：**

**面板功能模块，分面板展示和面板列表两个部分。**

面板展示又分侧边栏，面板名称导航展示和面板内容展示面板详情。

**面板详细信息展示里集成如下功能：**

1. 面板select语句查询功能；
2. 面板添加图表入口，进入图表添加页面
3. 面板编辑图表入口，进入图表编辑页面
4. 面板移除图表，移除当前面板拥有的图表
5. 面板图表编辑集成图表拖拽功能，拖拽改变位置的图表将重新排序
6. 面板图表编辑同时集成图表大小用户修改，保存后保存图表的新宽高样式

**面板列表有一下功能点：**

1. 新增面板入口，完成面板新增的所有功能
2. 编辑面板入口，完成面板所有编辑功能，并进行保存
3. 删除面板，删除当前面板在数据库的记录
4. 列表排序，对面板数据进行排序功能
5. 列表分页，对列表数据进行分页处理并分页显示
6. 列表搜索，根据输入的关键字实现列表数据过滤显示

## 图表

如图4-5-1所示：

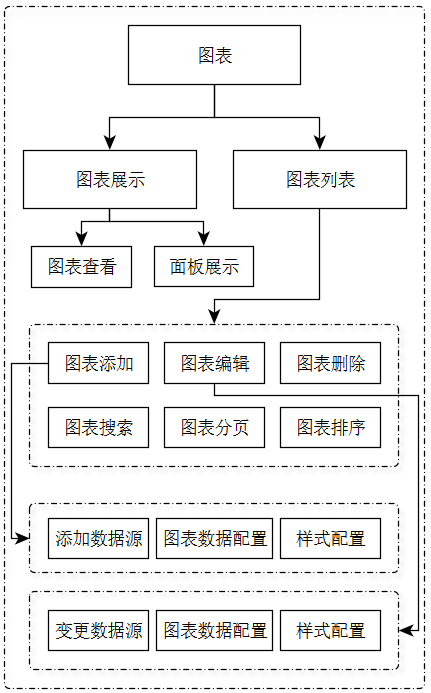


图 4-5-1 图表功能

图表功能同样分图表列表和图表展示两个功能点入口：

**图表展示：**

图表展示分图表查看时展示和添加到面板中进行展示，两种展示方式

**图表列表有以下功能点：**

1. 添加图表，添加图表时提供数据源选择和数据源添加入口，同时添加图表时要实现图表样式的配置，图表数据的配置等。
2. 编辑图表，编辑图表时图表数据源变更、图表数据配置更改、图表样式更改，运行编辑结果后，应更新当前被更新的图表展示。保存后，数据将同步到服务端进行保存。
3. 删除图表，图表删除将永久删除该图表的所有信息
4. 图表列表搜索，根据输入的关键字过滤图表数据，进行展示
5. 图表列表排序，对图表数据进行简单的倒叙、正序排序功能处理
6. 图表数据分页，将图表数据分页后再展示给用户

## 数据源

如图 4-6-1所示：

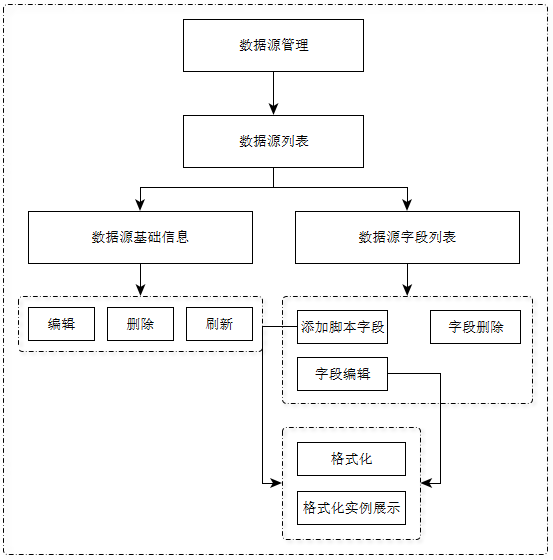


图 4-6-1 数据源

数据源功能主要功能都集中在数据源列表页面

数据源列表页面展示数据源列表、数据源基础信息、数据源字段列表等

数据源基础信息展示栏拥有对数据源的删除、编辑、更新功能。

数据源字段列表可编辑、删除字段，同时也可以添加脚本字段

字段编辑：

编辑字段的基本信息，格式化字段值，并给出实例结果进行展示。格式化规则保存到配置中更新到服务器，在展示时，调用格式化规则

# 组件关系图

概述：组件嵌套关系，遵循vue组件设和单页面应用设计思路，根据项目需求，按各个功能模块加载到同一个内容组件里，实现单页面加载。

如图5-1所示：

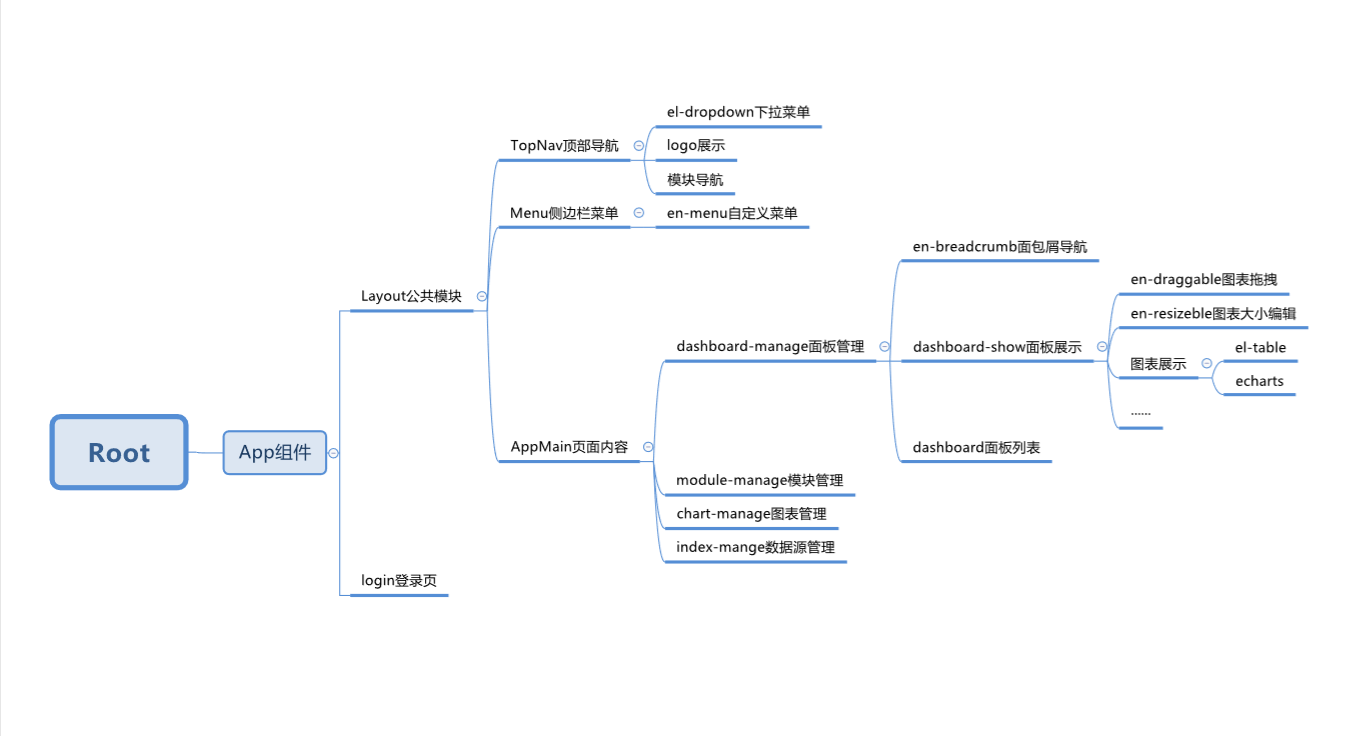


图5-1 组件关系结构图

所有组件都挂载在app.vue组件下，也就是说所有组件都是app组件的子组件

App组件下挂载直接组件layout组件和login组件。

Login组件呈现项目登录页面及相应的功能，登录成功后跳转到layout组件。

Layout组件设计三个直接组件topNav组件、menu组件、appMain组件，分别对应顶部导航，侧边栏菜单导航和页面内容。

TopNav设计直接子组件logo展示组件、模块导航组件和下拉菜单组件。

Menu展示侧边栏菜单，其子组件根据是否有子集递归调用自己。

topNav和menu为公共组件，提供给所有页面公用，appMain组件加载属于自己的子组件呈现不同的内容。

例如：

面板管理加载面板管理组件（dashboard-manage组件），再设计面板管理的直接子组件面包屑导航en-breachcrumb组件、面板展示dashboard-show组件、面板列表dashboard组件。

面板展示组件又设计直接子组件en-draggable组件、en-resizeable组件图表组件。

其他页面设计模式按相应功能类似面板组件，在此不做详细描述说明。