**ENVisual后端设计文档**

V1.0.0

张立丹 @ Enlink 数据中心

**目录**

[1. 概述 3](#_Toc528693524)

[1.1. 说明 3](#_Toc528693525)

[1.2. 系统特点 3](#_Toc528693526)

[1.3. 技术选型 3](#_Toc528693527)

[1.4. 其他因素 4](#_Toc528693528)

[2. 系统功能分析 5](#_Toc528693529)

[2.1. 前端代理 5](#_Toc528693530)

[2.2. 用户控制 5](#_Toc528693531)

[2.3. 配置管理 5](#_Toc528693532)

[2.4. 数据查询 5](#_Toc528693533)

[3. 系统模块设计 6](#_Toc528693534)

[4. 系统API设计 7](#_Toc528693535)

[4.1. 静态资源：/static 7](#_Toc528693536)

[4.2. 用户管理：/user 7](#_Toc528693537)

[4.3. 配置管理：/cfg 7](#_Toc528693538)

[5. 目录页 8](#_Toc528693539)

[5.1. 格式 8](#_Toc528693540)

[5.2. 内容 8](#_Toc528693541)

[6. 正文 9](#_Toc528693542)

[6.1. 概述 9](#_Toc528693543)

[6.2. 标题 9](#_Toc528693544)

[6.3. 内容 9](#_Toc528693545)

[6.3.1. 表格 9](#_Toc528693546)

[6.3.2. 插图 10](#_Toc528693547)

[6.3.3. 代码及命令 10](#_Toc528693548)

# 概述

本文档描述ENVisual（Enlink 数据可视化）模块的后端设计部分，旨在描述后端产品的概貌，后端产品的实现不在本文档讨论范围之内，请参阅相关实现文档或阅读源代码。

## 说明

ENVisual是数据中心产品中与数据可视化相关的子产品（或模块），命名为ENVisual-Server。本文档依照《数据中心-功能需求文档-V0.0.1》编写，数据中心产品的需求及总体架构请参阅相关文档。

数据可视化重点关注数据的展示，与数据内容无关。被展示的数据存储于ES（ElasticSearch）中，而ENVisual-Server中仅保存面板和图表的配置、数据查询方式以及图表钻关系等内容，本文档后续章节将对此进行详细描述。

## 系统特点

根据功能需求文档，ENVisaul-Server有如下特点：

* **数据量少**  
  仅保存一些配置性的内容；
* **用户群小**  
  企业内部进行日志分析使用；
* **计算量小**  
  数据计算均由ES来完成；
* **业务逻辑简单**  
  无复杂的应用场景。

所以ENVisual-Server 以轻巧、实用、简单为设计原则 。实现时亦避免出现复杂的实现逻辑。

## 技术选型

根据ENVisual的功能需求、ENVisual-Server的系统特点，并结合公司内部的技术积累，对ENVisual-Server的技术选型如表1-1所示：

表格 1-1 ENVisual-Server技术选型

|  |  |  |
| --- | --- | --- |
| **功能** | **选型** | **说明** |
| 开发语言 | NodeJS |  |
| HTTP服务框架 | Express | 小巧、轻便。 |
| 数据存储 | Sqlite3 | 便于部署。  驱动引擎使用node-sqlite3。 |
| 开发环境 | WebStorm |  |

由于需求文档中未提及用户及权限控制，所以本设计暂不涉及用户管理部分，但在设计时有必要考虑该功能。

本系统中使用express-session库来实现用户的“会话（session）”控制功能，session的内容直接存放在系统内存中，如果ENVisual后续需求中需要与其他系统共享用户，则可以通过配置express-session轻松将session以其他方式（如Redis等）存储。

## 其他因素

根据数据中心产品和ENVisual模块的定位，以及ENVisual-Server的系统特点，暂时不需要考虑系统的性能、扩展性等其他因素。

# 系统功能分析

本章节描述ENVisual-Server详细的系统功能。本章节参考功能需求文档编写，并对需求文档进行了详细的补充。

## 前端代理

ENVisual产品设计时使用“前后端分离”的方式，即产品分为ENVisual-Server和ENVisual-Client两个模块。

为了简化部署，ENVisual-Client不提供Server部分，其静态资源交由ENVisual-Server来代理。产品发布时，将前端的静态资源放置于服务端某个的静态目录即可。

## 用户控制

根据需求文档，ENVisual系统不支持用户管理及权限控制。但是出于系统的完整性考虑，ENVisual-Server提供最基本的用户控制功能，即用户登录和密码修改功能。

## 配置管理

ENVisual是一个高可配置的可视化产品，包括菜单在内的几乎所有内容均可配置。所以，ENVisual-Server可简单的理解为对可视化功能配置的存储模块。

客户端需要的所有类型的配置，均通过URL进行区分。URL使用统一的格式进行设计，方便后续需求中，支持通过URL对权限进行控制，详细内容请参阅本文档URL设计章节。

## 数据查询

ENVisual产品中，前端（ENVisual-Client）对ES数据的查询的请求，由服务端进行代理，即ENVisual-Client将查询请求发送给ENVisual-Server，然后由ENVisual-Server向ES发起请求（实际上是将请求发送给ESAgent，参阅数据中心架构设计文档）。

# 系统模块设计

本章节描述ENVisual-Server系统中所涉及的各个模块，以及各模块之间的关系。模块的设计依据[章节2. 系统功能分析](#_系统功能分析)所描述的内容，如图3-1所示：

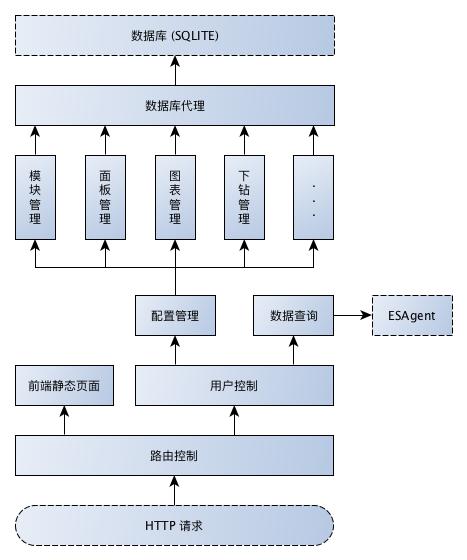


图 3‑1 ENVisual-Server系统模块

其中“HTTP请求”与“ESAgent”两个模块不在ENVisual-Server模块设计范围之内。图中的“箭头”表示请求传递的方向。

# 系统API设计

本小节描述ENVisual-Server对外提供的API设计原则 ，详细的API说明，请参阅API说明文档。根据图3-1所示的“ENVisual-Server系统模块”，将URL按层级划分，方便根据URL进行路由以及用户控制。

## 静态资源：/static

静态资源API使用static进行标识，其中静态资源包括客户端实现的html、CSS、JS等资源。该类型API二级以后的URL部分，由客户端自主定义，不在本文档描述范围之内。

## 用户管理：/user

用户管理（或用户控制）API使用user进行标识，该类型API二级以后的URL部分根据功能进行详细划分。如表3-1所示：

表格 3-1 用户管理API

|  |  |
| --- | --- |
| **URL** | **说明** |
| /user/login | 用户登录。 |
| /user/logout | 用户登出。 |
| /user/modifyPassword | 修改密码。 |
| ... |  |

表中列出了三个API对应的URL，并分别介绍了API的功能。

用户管理模块详细的API设计，以及API格式，请阅读API说明文档。

## 配置管理：/cfg

配置管理API使用cfg进行标识，该类型API二级以后的URL部分根据根据配置对象的不同进行详细划分。

首页的内容必需包括：（1）文档标题；（2）文档版本号；（3）作者名称；（4）部门名称。具体格式请参阅文档首页内容。

# 目录页

本章节描述文档“目录页”格式及内容。

## 格式

目录页设置页眉和页脚（可以不包含页脚本），页眉、页脚内容设置需要与“正文”中的页眉、页脚内容设置一致。

目录页第一行的居中位置必需为“目录”，以标识当前页为目录页。后面紧跟着目录内容。目录内容根据正文标题的级别进行缩进。

## 内容

页眉内容包括两个部分：（1）左侧为文档标题及版本号；（2）右侧为公司LOGO。

页脚内容包括两个部分：（1）左侧为跳转到目录的链接；（2）右侧为页码。

目录域必需自动生成，不能够手动输入。在更新内容时，需要同步更新目录域。具体格式请参阅本文档目录页内容。

# 正文

本章节描述文档“正文”格式及内容。

## 概述

正文设置页眉和页脚，其内容、格式与目录页保持一致（参阅章节 [3. 目录页](#_目录页) 的内容）。

## 标题

正文页必需包含章节标题，一级标题居中对齐，表示“章节”的标题；其他（二级以下）标题均左对齐，表示“小节”的标题。

新的章节必需从新的页开始，例如，本文档中第2章与第3章分别从新的页开始。各章节中的小节，无需从新的页开始。

## 内容

本小节描述正文内容的相关格式。

### 表格

当文档中需要插入表格时，需包括如下信息：

* 表格编号；
* 表注说明（或题注）；
* 表头。

在需要引用到表格的地方使用“如表 xxx 所示”来指明引用的表格，不能出现“如上表所示”的字样。表格的格式示例，如表4-1所示：

表 -1表格示例

| **名称** | **内容** | **说明** |
| --- | --- | --- |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

**注意**：表格的表头一般使用居中、粗体样式，且设置浅色背景色。表格的题注位于表格的上方右对齐，使用斜体、灰色样式。

如果表格需要跨页，则将其拆分成两个或多个表格，拆分后的每个表格必需包含表头。

***[由于本页剩余的空间过少，下一章节从新的页开始，实际文档编写中，需要注意这些小细节]***

### 插图

当文档中需要插入图片时，需要包括如下信息：

* 图编号；
* 图注说明（或题注）。

在需要引用到插图的地方使用“如图 xxx 所示”来指明引用的图片，不能出现“如上图所示”的字样。插图的格式示例，如图4-1所示：

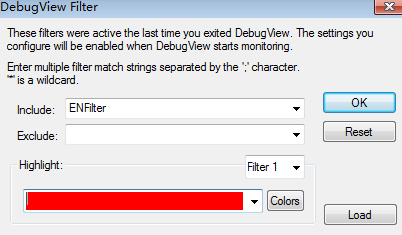


图 ‑1 插图示例（插图内容请忽略）

插图使用“嵌入型”环绕方式，并居中对齐，插图的题注也使用居中对齐的样式。

### 代码及命令

当文档中需要展示代码或命令时，使用表格的一个单元格来展示，根据内容的长度来选择性的设置题注。一般用于展示命令时，不设置题注；而用于展示代码时，需要设置题注。

示例一，用于展示命令。在linux操作系统下，可使用ps与grep命令结合，来查看系统中运行的nginx进程，如下表所示：

|  |
| --- |
| ps -ef | grep nginx |

示例二，用于展示代码。所有使用C语言的程序员，基本上都是从“hello world”程序开始学习C语言的，代码如表4-2所示：

表 -2 hello world示例代码

|  |
| --- |
| #include <stdio.h>  int main(int argc, char\*\* argv) {  printf("hello world\n");  return 0;  } |

表格展示代码时，可根据展示内容的多少来调整代码内容的样式（例如，使用小号字体）。