

页面可视化搭建框架 pipeline

陈韩杰 [@CntChen](#)

自我介绍

开源页面可视化搭建框架 [page-pipeline](#) 作者

在前端早读课发表过 [页面可视化搭建工具前生今世](#)(1282期)

在招联金融工作期间负责运营页面搭建工具落地

现为腾讯 AlloyTeam 成员

目录

1. 活动页面开发痛点(10%)
2. 可视化搭建工具前世今生(25%)
3. 可视化搭建工具技术要点(35%)
4. 理想的运营页面搭建工具(10%)
5. 开源搭建框架 pipeline(20%)

活动页面开发痛点

活动页面

锦瑟 -- 李商隐



锦瑟无端五十弦，一弦一柱思华年。
庄生晓梦迷蝴蝶，望帝春心托杜鹃。
沧海月明珠有泪，蓝田日暖玉生烟。
此情可待成追忆，只是当时已惘然。

锦瑟 -- 李商隐



锦瑟无端五十弦，一弦一柱思华年。
庄生晓梦迷蝴蝶，望帝春心托杜鹃。
沧海月明珠有泪，蓝田日暖玉生烟。
此情可待成追忆，只是当时已惘然。

送杜少府之任蜀州



城阙辅三秦，风烟望五津。
与君离别意，同是宦游人。
海内存知己，天涯若比邻。
无为在歧路，儿女共沾巾。

送杜少府之任蜀州 -- 王勃



城阙辅三秦，风烟望五津。
与君离别意，同是宦游人。
海内存知己，天涯若比邻。
无为在歧路，儿女共沾巾。

活动页面特点

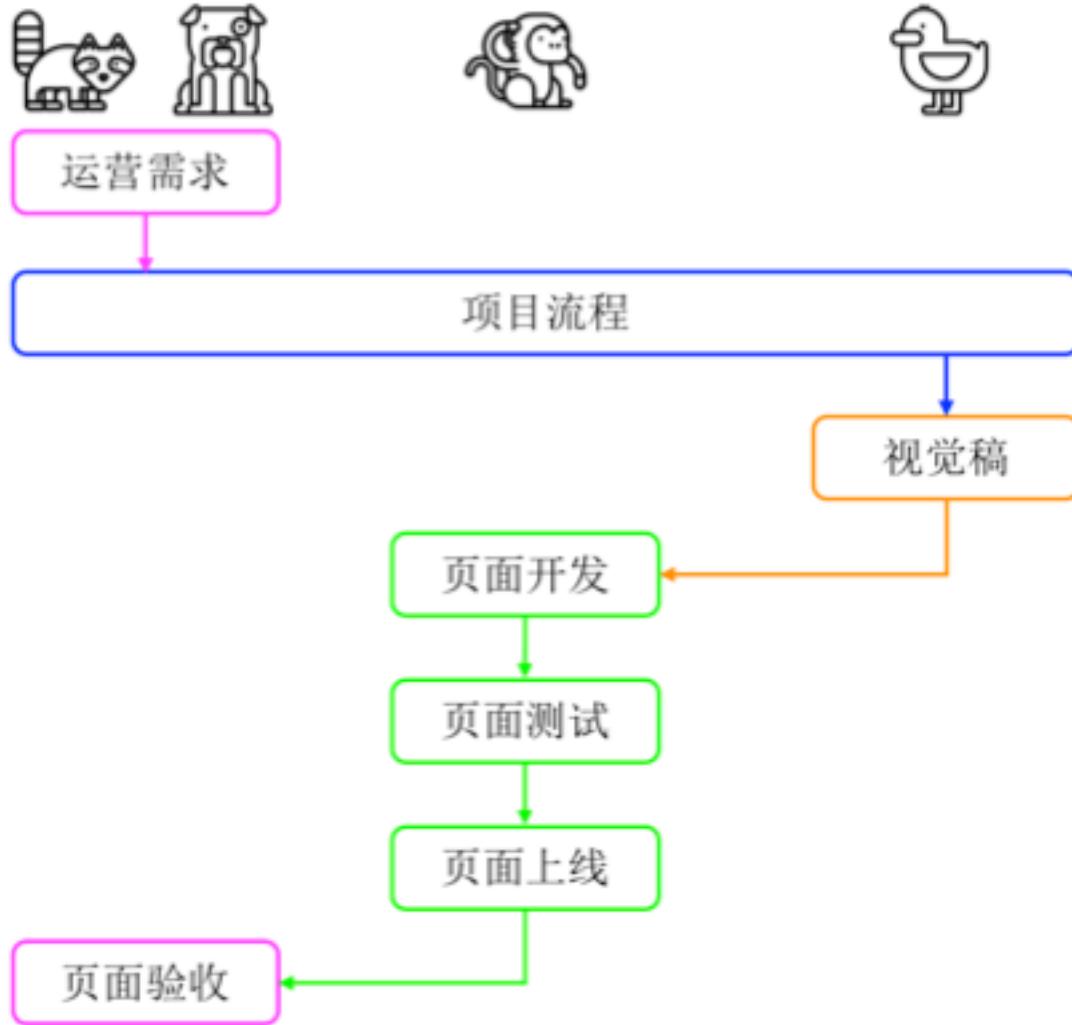
页面类似: 页面布局和业务逻辑较固定

需求高频: 每周甚至每天有多个这种需求

迭代快速: 开发时间短, 上线时间紧

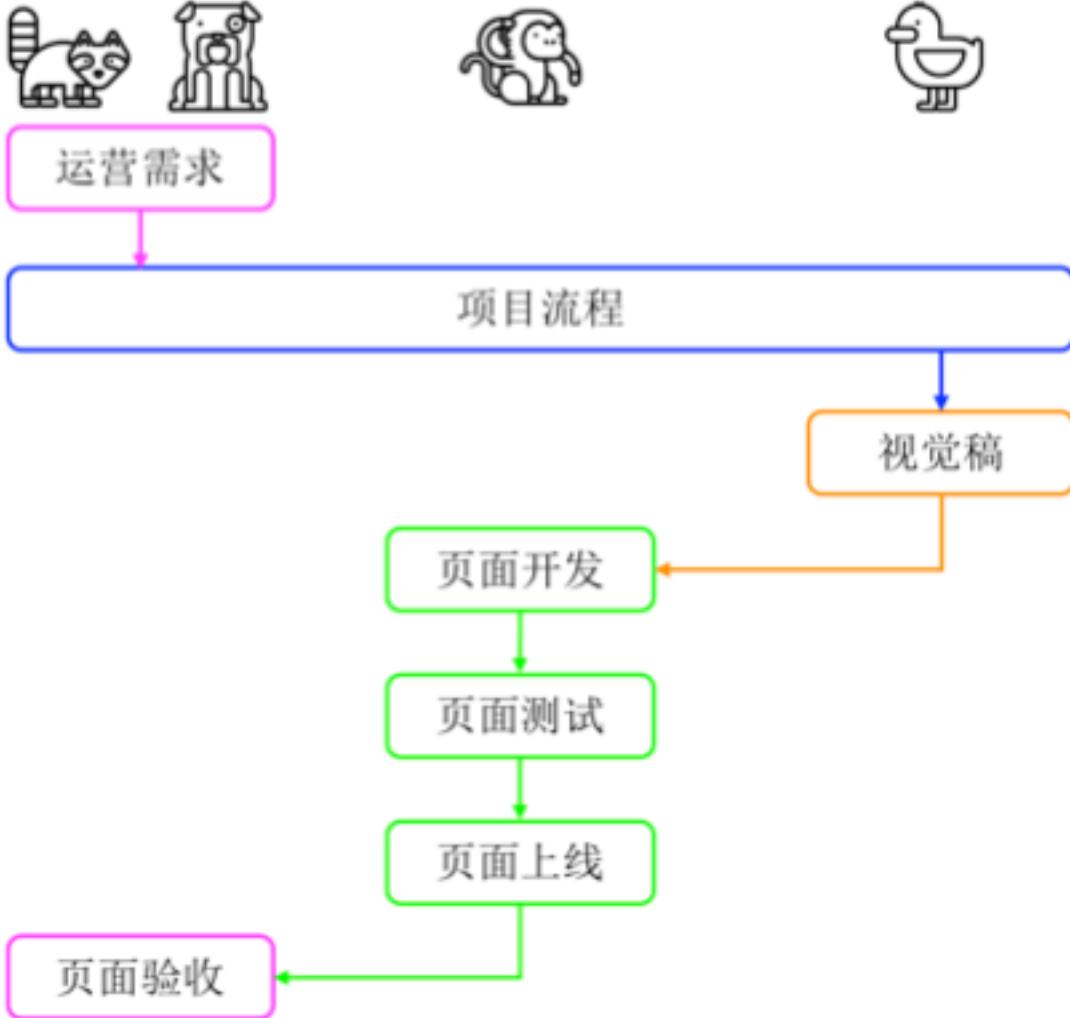
开发性价比低: 开发任务重复, 消耗时间和人力

常规开发流程



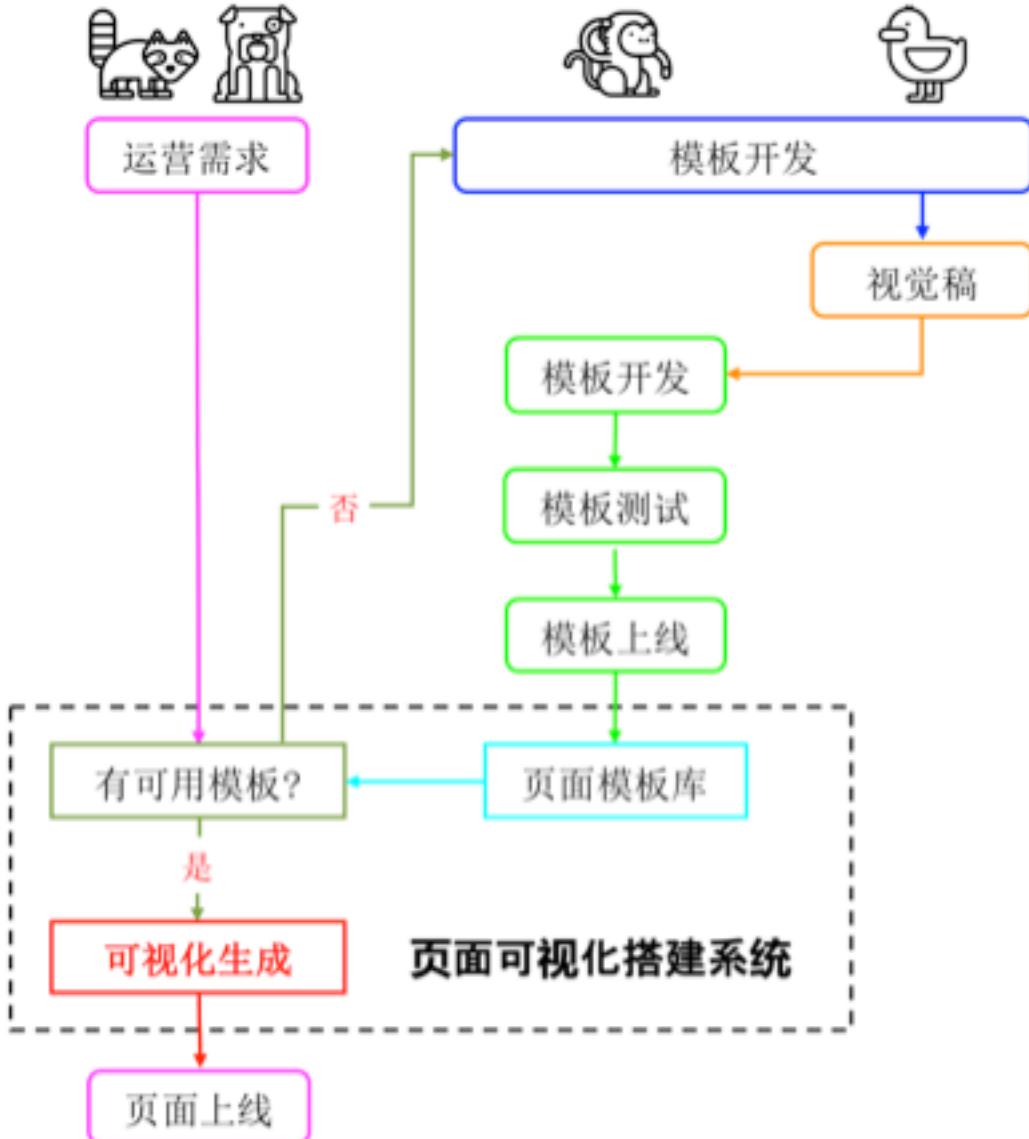
1. 运营/产品提出页面需求
2. 走项目流程进入开发环节
3. 开发根据设计稿完成页面开发
4. 测试进行页面测试
5. 运维进行页面上线
6. 运营/产品进行页面验收

常规开发流程的痛点



多方多参与, 反复沟通, **串行流程**
页面上线周期长, 无法快速响应活动需求
人力陷入重复工作泥潭, 忙碌而低效

更优的活动页面开发流程



1. 运营/产品提出页面需求
2. 运营/产品在**页面可视化搭建系统**中选取合适的页面模板进行页面搭建
3. 页面自动化发布上线, 流程完结
4. 如果运营/产品没有找到合适的模板
5. 开发进行页面模板开发, 并将页面模板添加到**页面可视化搭建系统**
6. 运营/产品继续流程2

页面可视化搭建工具

通过填写配置表单, 拖拉页面组件等可视化的方式, 实现页面的生成或修改.

目录

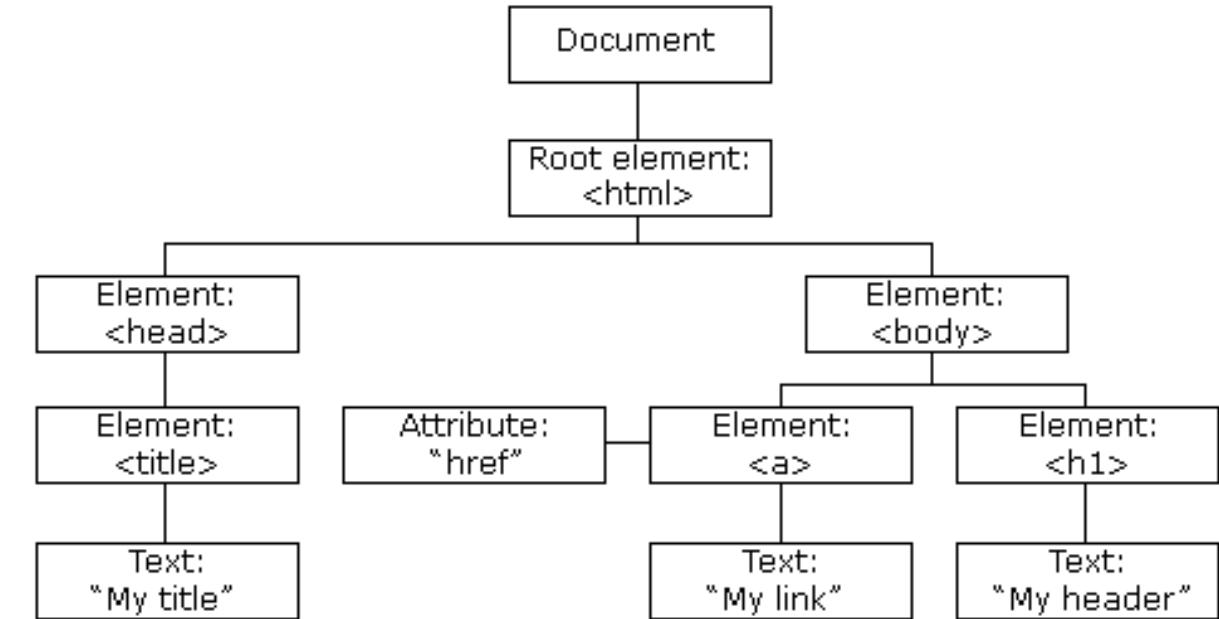
1. 活动页面开发痛点(10%)
2. 可视化搭建工具前世今生(25%)
3. 可视化搭建工具技术要点(35%)
4. 理想的运营页面搭建工具(10%)
5. 开源搭建框架 pipeline(20%)

页面是什么

```
HTML ▾
<!DOCTYPE html>
<html>
<head>
  <title>My title</title>
</head>
<body>
  <h1>My Header</h1>
  <a href="#">My link</a>
</body>
</html>
```

My Header

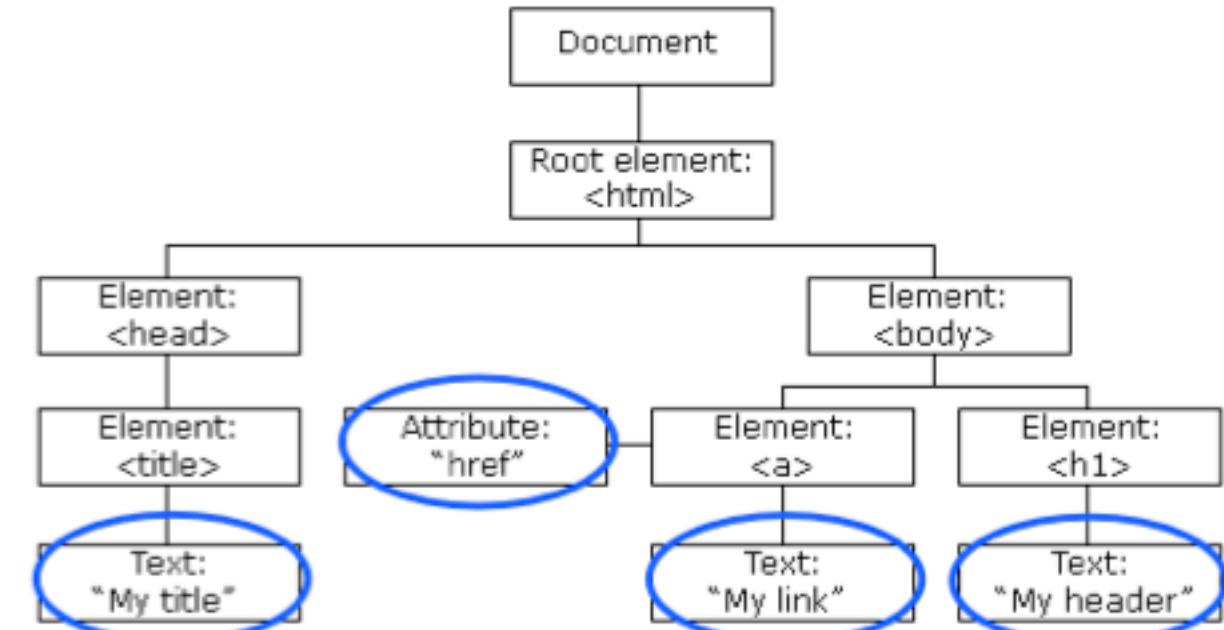
[My link](#)



页面是什么

```
HTML ▾
<!DOCTYPE html>
<html>
<head>
  <title>My title</title>
</head>
<body>
  <h1>My Header</h1>
  <a href="#">My link</a>
</body>
</html>
```

My Header

[My link](#)

Page

=

HTML Tree

+

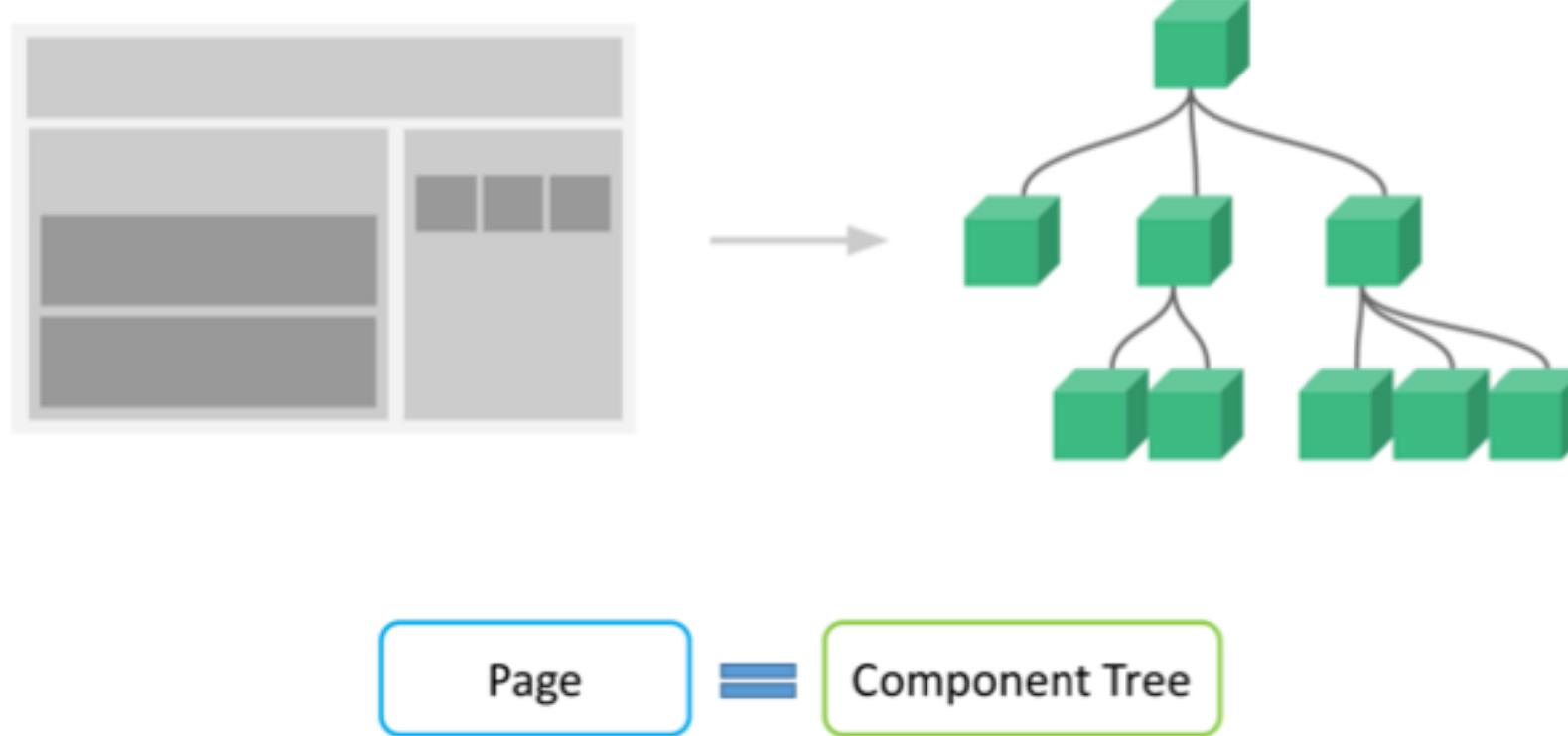
Data

静态页面与动态页面



页面组件化

页面组件化组合 HTML 元素, 实现了功能封装和可复用的页面组件



页面可视化搭建的定义

页面开发: 用编程工具(IDE)来编辑页面

页面可视化搭建: 用可视化交互的方式编辑页面

	编程开发页面	可视化搭建页面
技能要求	需要编程基础	可以没有编程基础
操作方式	在代码编辑器中编写代码	在可视化搭建工具中拖拉/填表/编写代码
使用人员	前端工程师	前端小白/ 运营和产品 /开发人员

页面可视化搭建, 是前端服务化的一种形式.

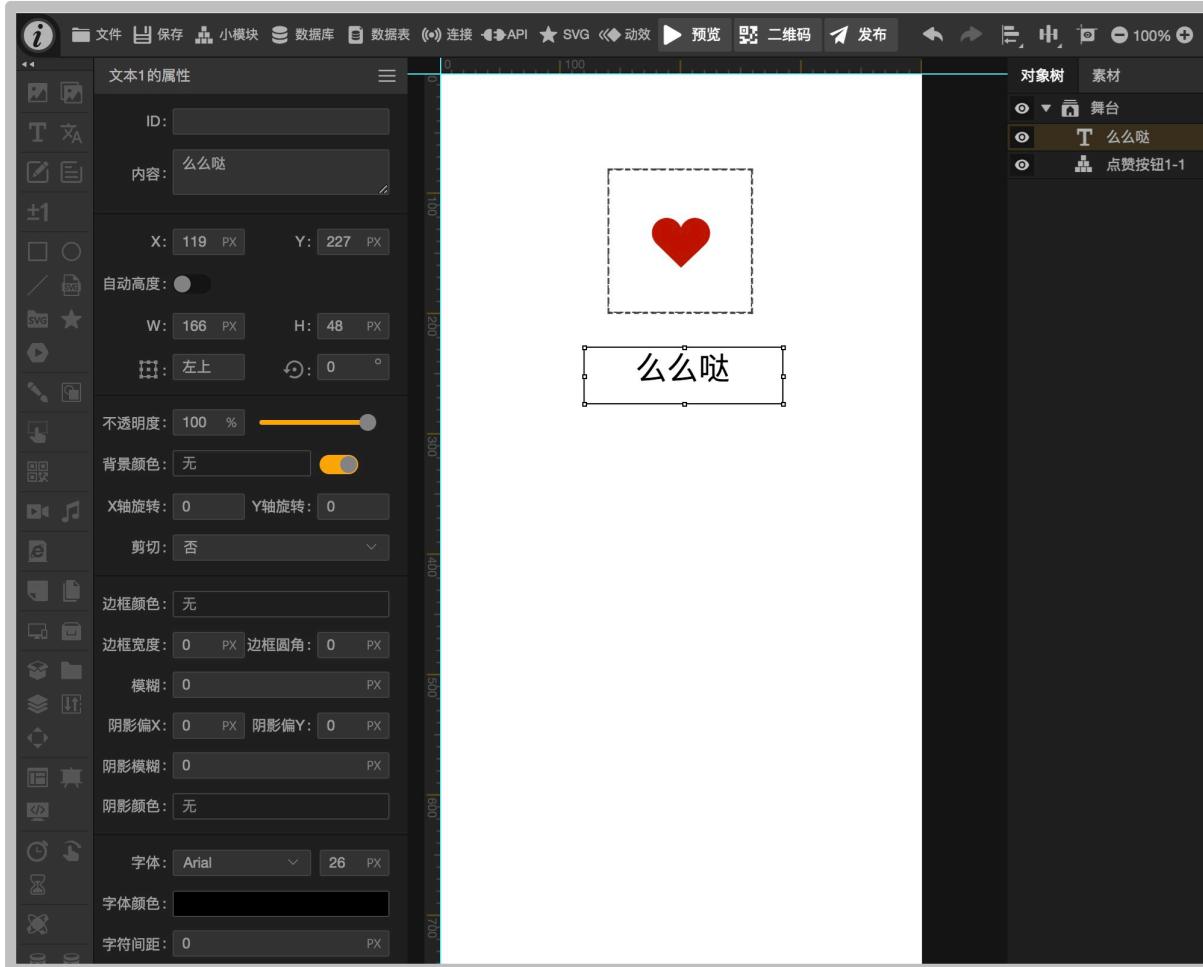
工具区分维度

当我们讨论页面可视化搭建工具时，怎么进行描述和讨论？

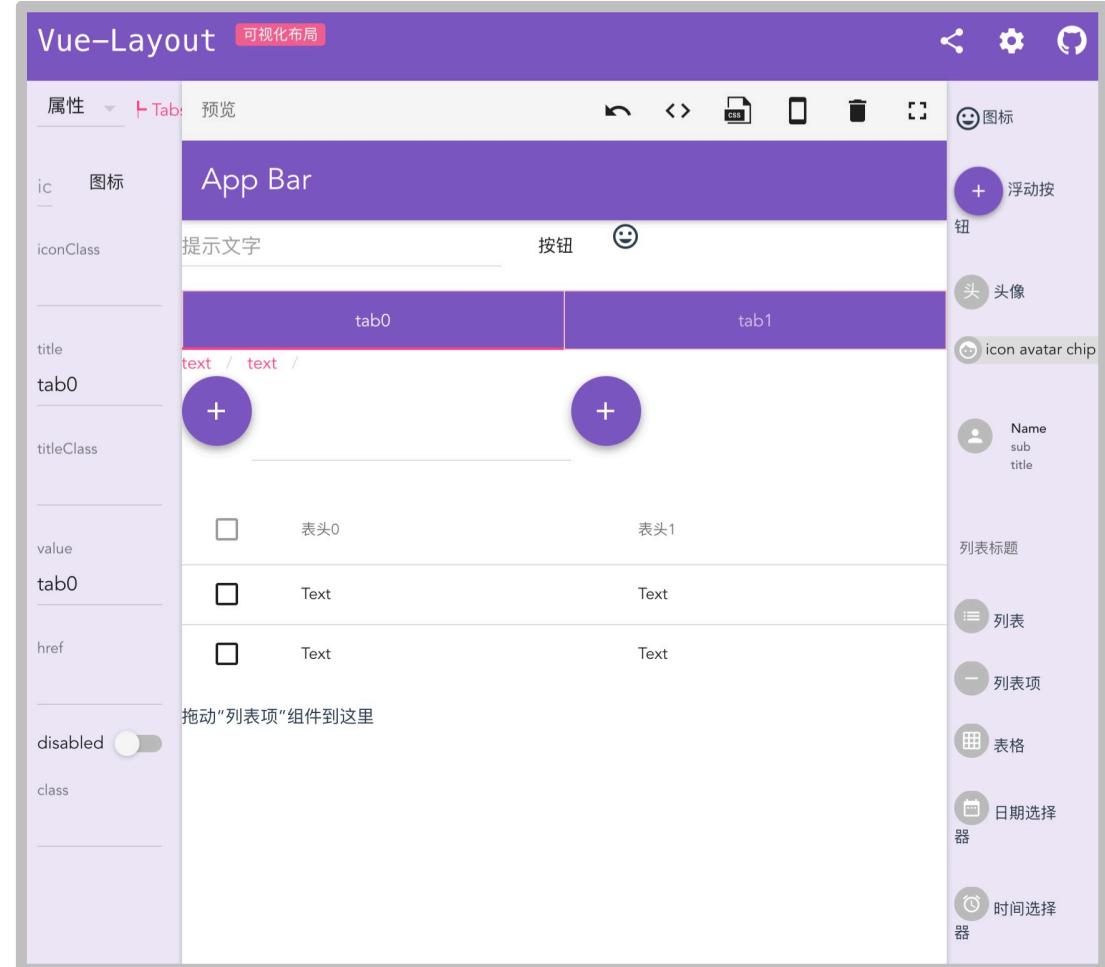
区分维度：

- **系统功能**: 工具提供的核心能力
- **面向客群**: 工具的主要服务人群
- **编辑自由度**: 页面可编辑单元的粒度

区分维度 -- 系统功能



HTML Tree 编辑



Component Tree 编辑

区分维度 -- 系统功能

This screenshot shows a card list component editor. At the top, there are buttons for '保存' (Save) and '编辑完成' (Edit Complete). Below the buttons are tabs for '内容' (Content), '设置' (Settings), and '日志' (Logs). The main area displays three cards with placeholder text: '云凤蝶在巴黎' (Cloud butterfly in Paris), '云凤蝶在伦敦' (Cloud butterfly in London), and '云凤蝶在纽约' (Cloud butterfly in New York). Each card has a title input field ('未命名标题') and a large preview image. Below each card are sections for '小标题' (Subtitle) and '大标题' (Main Title). A file upload section for '图片, 建议尺寸 750x320' (Image, recommended size 750x320) includes a '选择图片' (Select Picture) button and a preview thumbnail. A '跳转链接' (Jump Link) input field contains a URL.

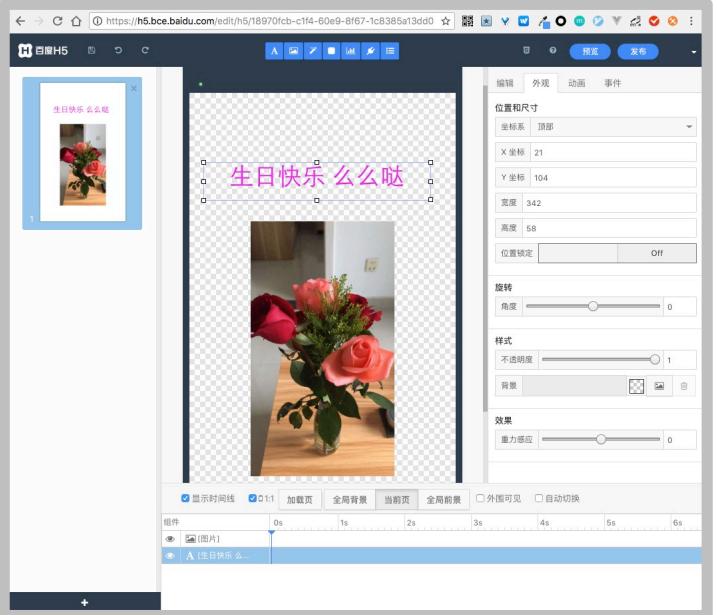
Data 编辑

This screenshot shows a dynamic logic editor interface. At the top, there are tabs for '脚本' (Script), '数据源' (Data Source), '页面预览' (Page Preview), and '全部展开' (Expand All). On the left, a tree view shows the page structure with nodes like 'Page(1)', 'Body(101)', 'Style(134)', etc. On the right, a code editor window displays a script block with a function definition:

```
$(function(){  
    // SomeThing additional toto Here.  
    alert('Hey, Guys !!');  
});
```

Dynamic Logic 编辑

区分维度 - 面向客群



前端小白

A screenshot of the 前端资源管理系统 (Front-end Resource Management System). It shows a "Page Preview" section with a green header bar and a "Page Configuration" section where a user is setting up the header area. The configuration includes a placeholder URL for the header image and a button to generate the page.

运营和产品

A screenshot of the 模板市场 (Template Market) interface. It displays a grid of templates categorized under React 物料源, Vue 物料源, and 自定义模板. Categories include 政府管理系统, 商务平台, 应用管理系统, CMS 内容管理系统, 语音对话系统, and 图书管理系统.

中后台开发人员

区分维度 -- 编辑自由度

This screenshot shows the Veeva web editor interface. On the left, there's a sidebar with various UI component icons like Button Toolbar, Heading, Image, Alert, Card, List Group, Horizontal Rule, Badge, Progress Bar, Nav Bar, Breadcrumbs, Pagination, Form, Text Input, and Text Area. The main workspace displays a Jumbotron component with a large heading "Jumbotron heading" and a subheading "Subheading". Below the heading is a paragraph of text: "Cras justo odio, dapibus ac facilisis in, egestas eget quam. Fusce dapibus, tellus ac cursus commodo, tortor mauris condimentum nibh, ut fermentum massa justo sit amet risus." At the bottom is a green button labeled "Sign up today". The top navigation bar shows a preview mode with "Pages" and a list of components: Jumbotron, Landio landing page, Album, Blog, Carousel, Offcanvas, Pricing, etc.

自由度为 HTML

This screenshot shows the Vue-Layout visual editor. It features a central preview area with a purple header labeled "App Bar" containing tabs "tab0" and "tab1". To the left is a properties panel for the "App Bar" component, showing fields for "icon", "iconClass", "title", "value", "href", "disabled", and "class". To the right is a sidebar with icons for various components: 图标 (Icon), 浮动按钮 (Floating Button), 头像 (Avatar), Name subtitle (Name subtitle), 表头 (Table Head), 列表项 (List Item), 表格 (Table), 日期选择器 (Date Picker), and 时间选择器 (Time Picker). The overall interface is designed for visual component composition.

自由度为 Component

This screenshot shows a visual editor interface with a light gray background. It contains three main sections: a top section with a large gray box labeled "Image", a middle section with a red border containing text "This is Title" and "This is Text" repeated four times, and a bottom section with another large gray box labeled "Image". Each section has a red border around it, indicating they are individual components or slots within a larger layout.

自由度为不嵌套的组件

运营页面可视化搭建工具

区分维度:

- **系统功能**: 工具提供的核心能力
- **面向客群**: 工具的主要服务人群
- **编辑自由度**: 页面可编辑单元的粒度

需要什么:

- Data 编辑功能
- 运营和产品
- 组件级别



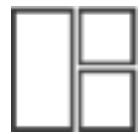
目录

1. 活动页面开发痛点(10%)
2. 可视化搭建工具前世今生(25%)
- 3. 可视化搭建工具技术要点(35%)**
4. 理想的运营页面搭建工具(10%)
5. 开源搭建框架 pipeline(20%)

技术要点



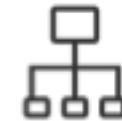
页面组件化



页面模板



页面编辑



组件层级关系



页面打包



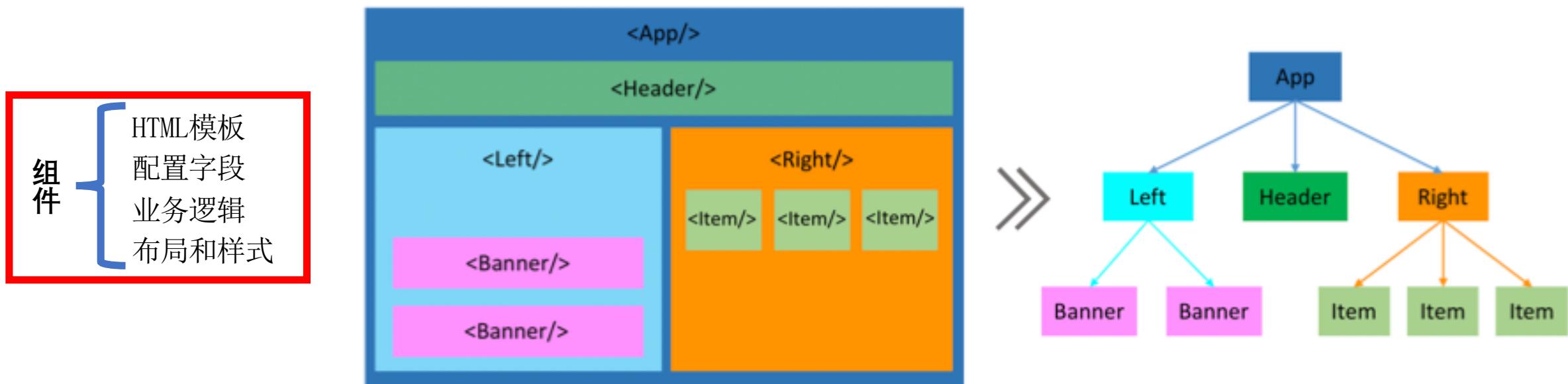
实时预览



组件开发

1. 页面组件化 – 组件化的优点

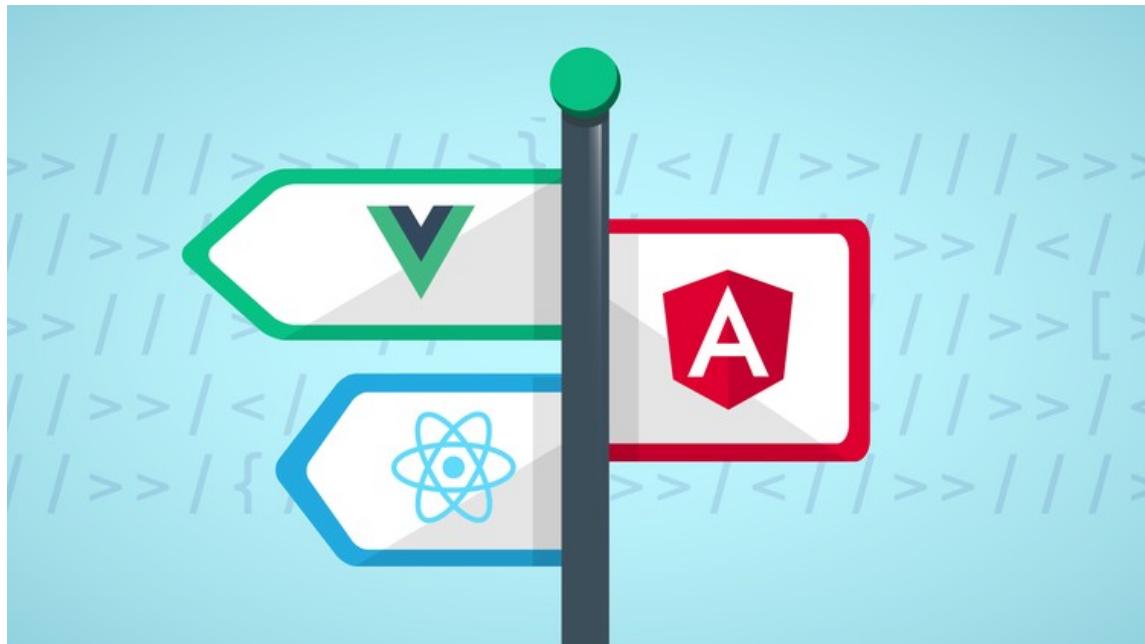
组件是对 HTML 元素、元素布局和样式、业务逻辑的封装



页面搭建: ○ 组件树组合 ○ 组件配置编辑

1. 页面组件化 – 页面前端框架

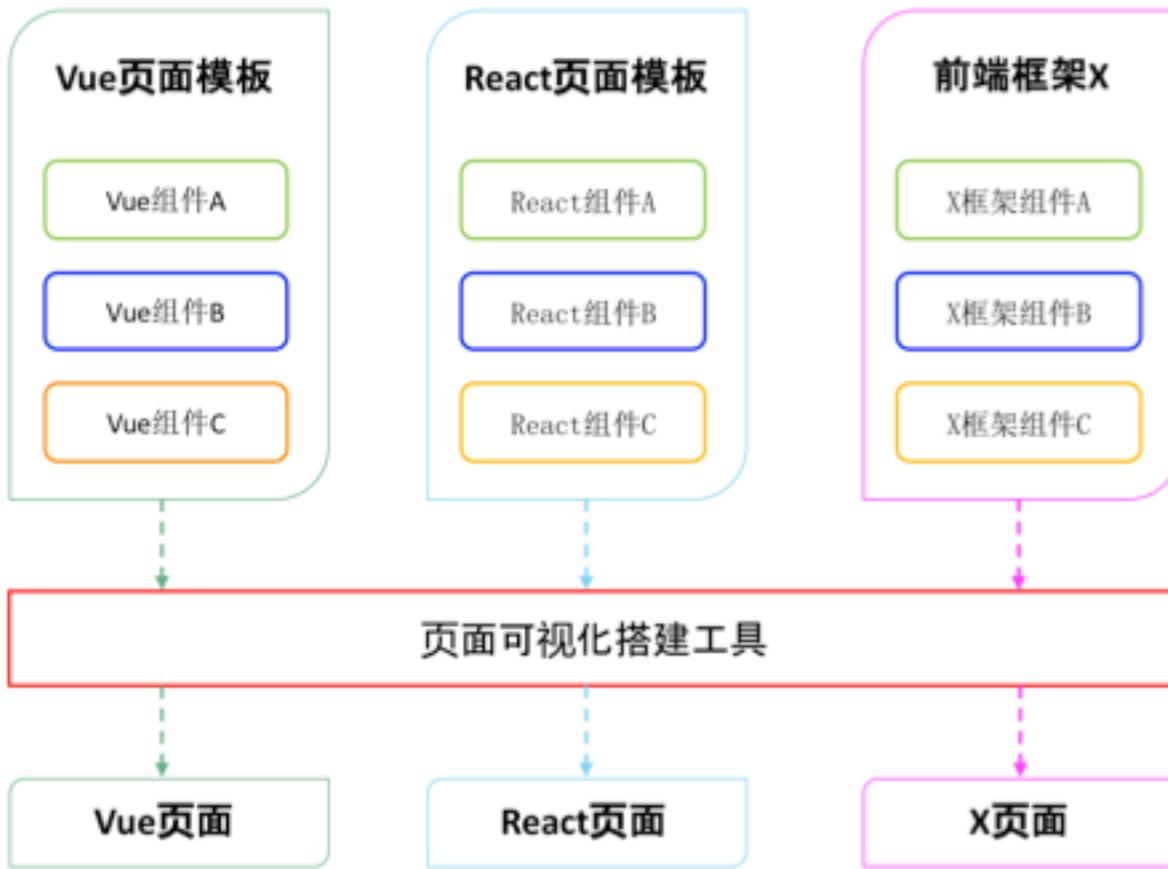
页面组件化依靠**前端框架**实现，可视化搭建工具需适配页面前端框架



业务前端框架：

- 沉淀了大量技术组件和业务组件
- 开发人员熟悉业务前端框架

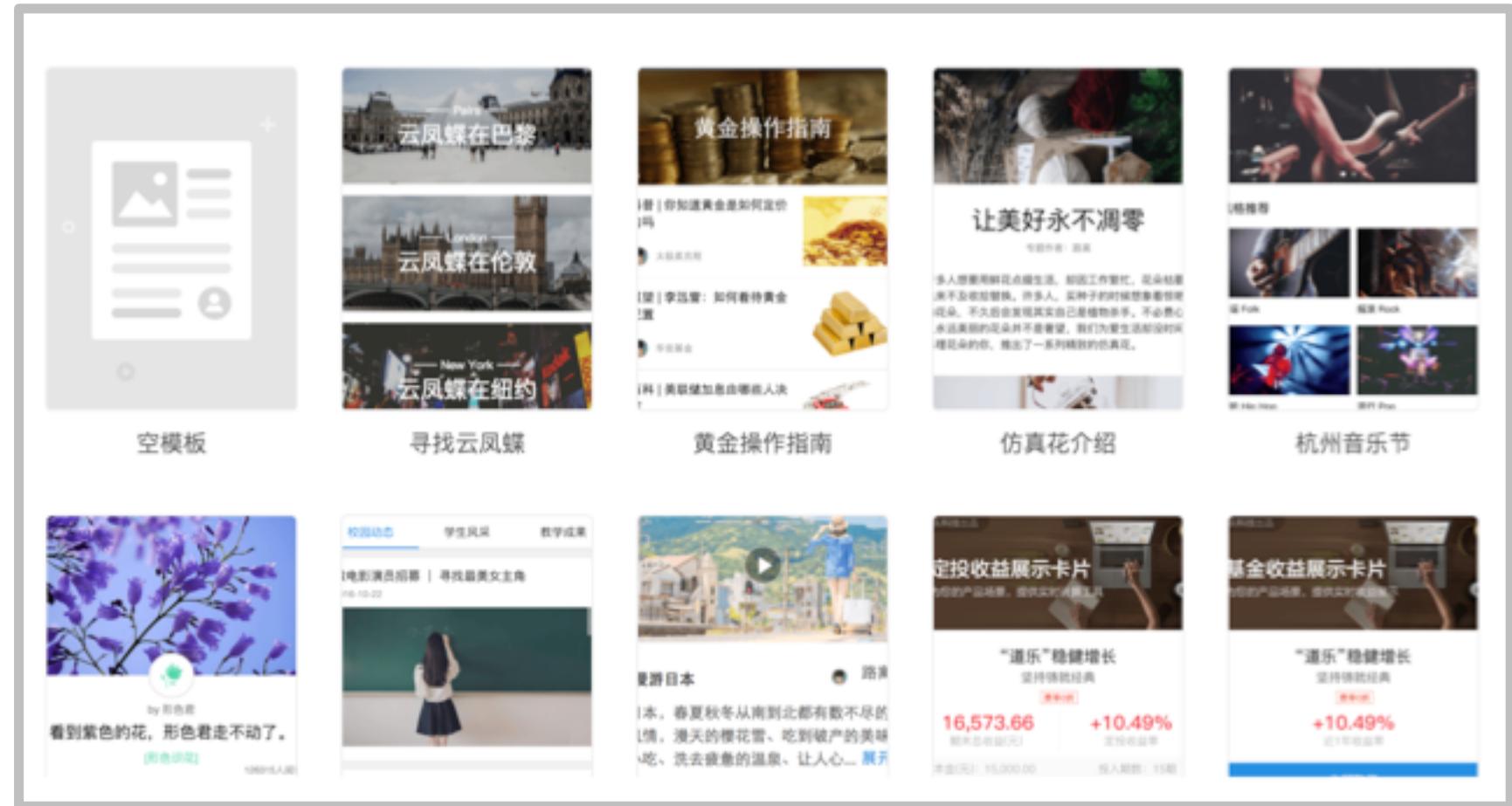
1. 页面组件化 - 页面前端框架



难点1: 页面可视化搭建工具与页面前端框架解偶

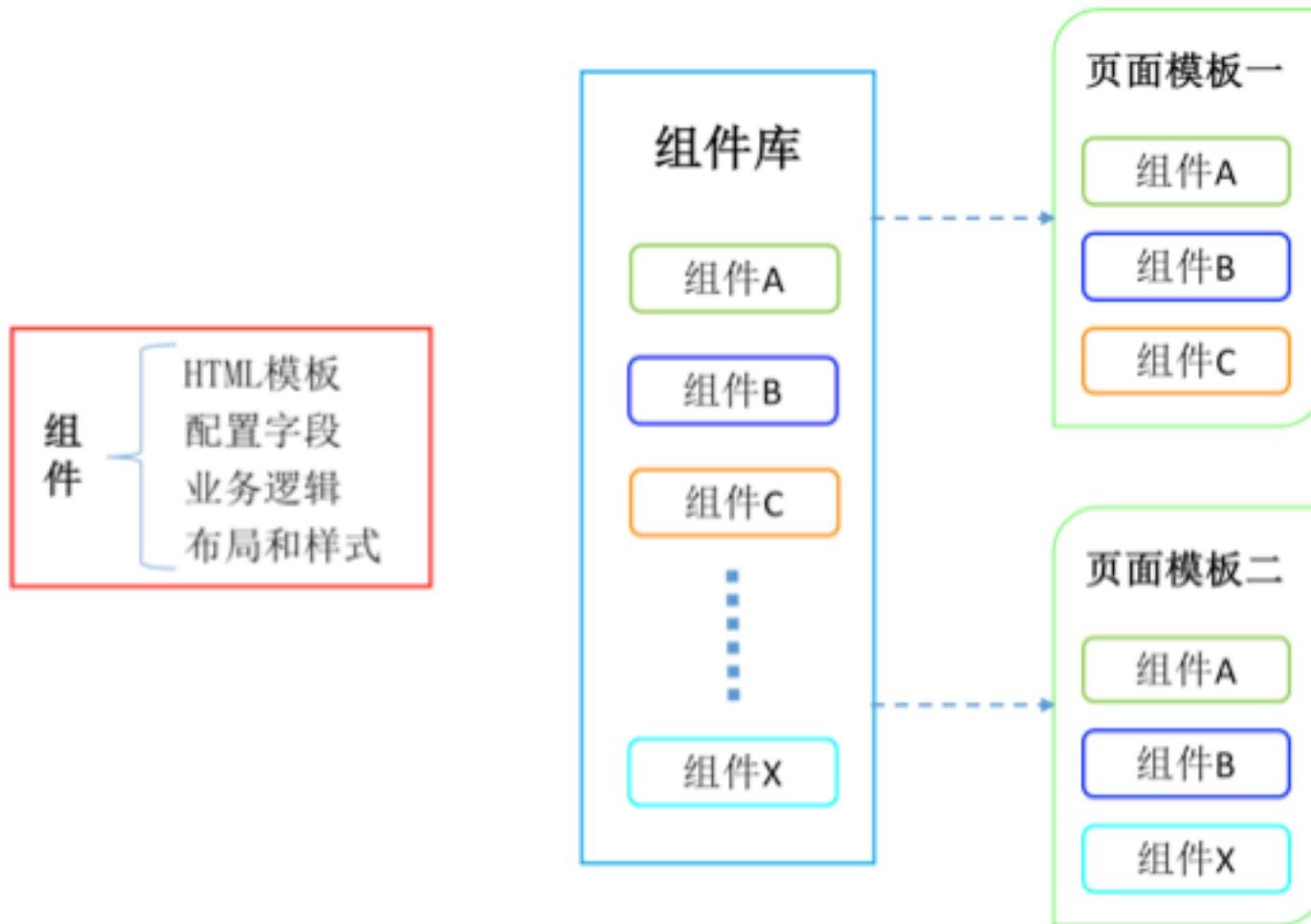
2. 页面模板

模板是带有默认数据的页面



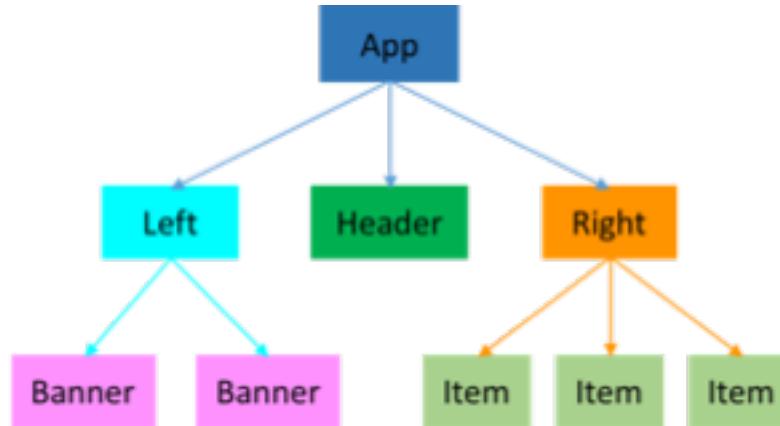
2. 页面模板

模板是从组件库中选取部分组件，并带有各个组件的默认配置数据



3. 页面编辑 -- 组件树

组件树声明方式



```
render() {
  return (
    <App>
      <Header></Header>
      <Left>
        <Banner type={a}></Banner>
        <Banner type={b}></Banner>
      </Left>
      <Right>
        {[1, 2, 3].map(i => <Item>i</Item>) }
      </Right>
    </App>
  );
}
```

React 组件树声明

```
<template>
  <App>
    <Header></Header>
    <Left>
      <Banner :type="a"></Banner>
      <Banner :type="b"></Banner>
    </Left>
    <Right>
      <Item v-for="(i, item) in [1, 2, 3]">{{item}}</Item>
    </Right>
  </App>
</template>
```

Vue 组件树声明

3. 页面编辑 -- 组件树

组件树编辑, 修改后更新源码,
重新打包



```
68  render() {
69    ...
70    <App>
71      ...
72    <Left>
73    <Banner type={a}></Banner>
74    <Banner type={b}></Banner>
75  </Left>
76  <Right>
77  {[1, 2, 3].map(i => <Item>i</Item>)}
78  </Right>
79  </App>
80  );
81 }
```

```
68  render() {
69  ...
70  <App>
71  ...
72 + <NewLeft>
73 + </NewLeft>
74  ...
75  {[1, 2, 3].map(i => <Item>i</Item>)}
76  ...
77  </App>
78  );
79 }
```

3. 页面编辑 – 组件树

动态组件，编辑组件树后用新的组件声明渲染出页面

```
[  
  {  
    "id": "header-demo",  
    "name": "header-demo"  
  },  
  {  
    "id": "info-demo",  
    "name": "info-demo"  
  },  
  {  
    "id": "gap-demo",  
    "name": "gap-demo"  
  },  
  {  
    "id": "weather-demo",  
    "name": "weather-demo"  
  }  
]
```



```
<template>  
  <div class="vue-main">  
    <component  
      v-for="oneComponent in mycomponents"  
      :key="oneComponent.id"  
      :is="oneComponent.name">  
    </component>  
  </div>  
</template>
```

Vue动态组件

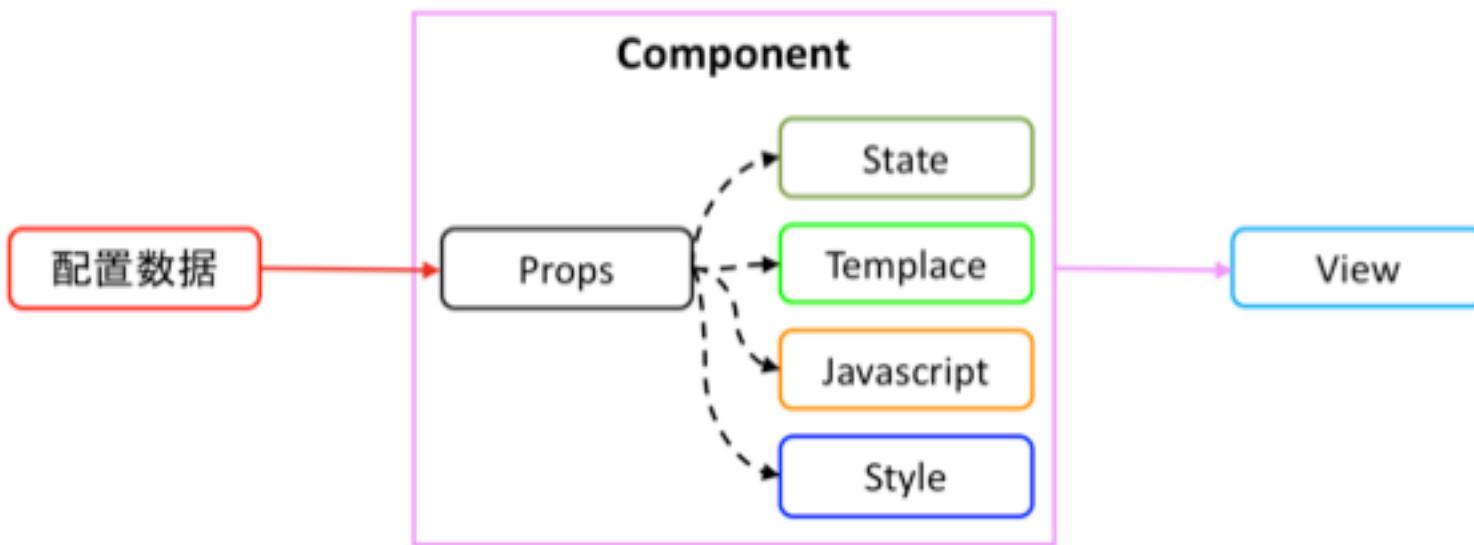


动态组件渲染成页面

3. 页面编辑 – 页面内容

组件化页面的内容编辑, 其实是对页面中各组件的组件属性进行配置

组件包含组件属性(Props), 组件状态(State), 组件HTML模板(Template),
组件业务逻辑(Javascript), 组件样式布局(Style)等



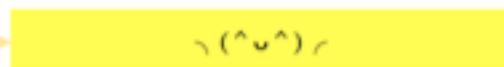
3. 页面编辑 - 页面内容

组件是差异化的，组件的配置属性也是差异化的
理想的数据格式为 JSON

```
{  
  "title": "页面可视化搭建框架",  
  "src": "https://github.com/u/38666040",  
  "link": "https://github.com/page-pipeline"  
}
```



```
{  
  "backgroundColor": "#FFFF00",  
  "textColor": "#000000",  
  "text": "\u0329(^o^)\u0329",  
  "height": 48  
}
```

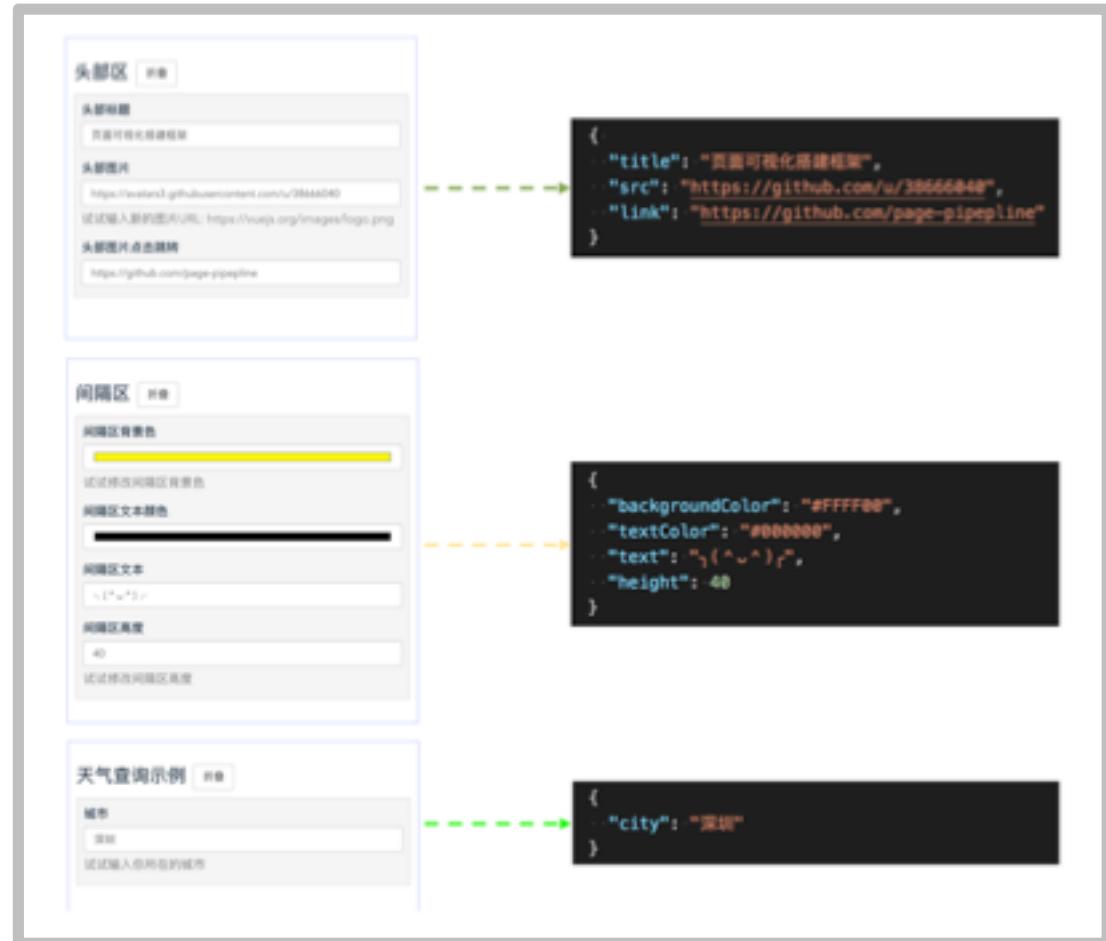


```
{  
  "city": "深圳"  
}
```



3. 页面编辑 – 页面内容 – 配置表单

提供可视化的编辑方式 -- 使用组件配置表单来填入配置数据



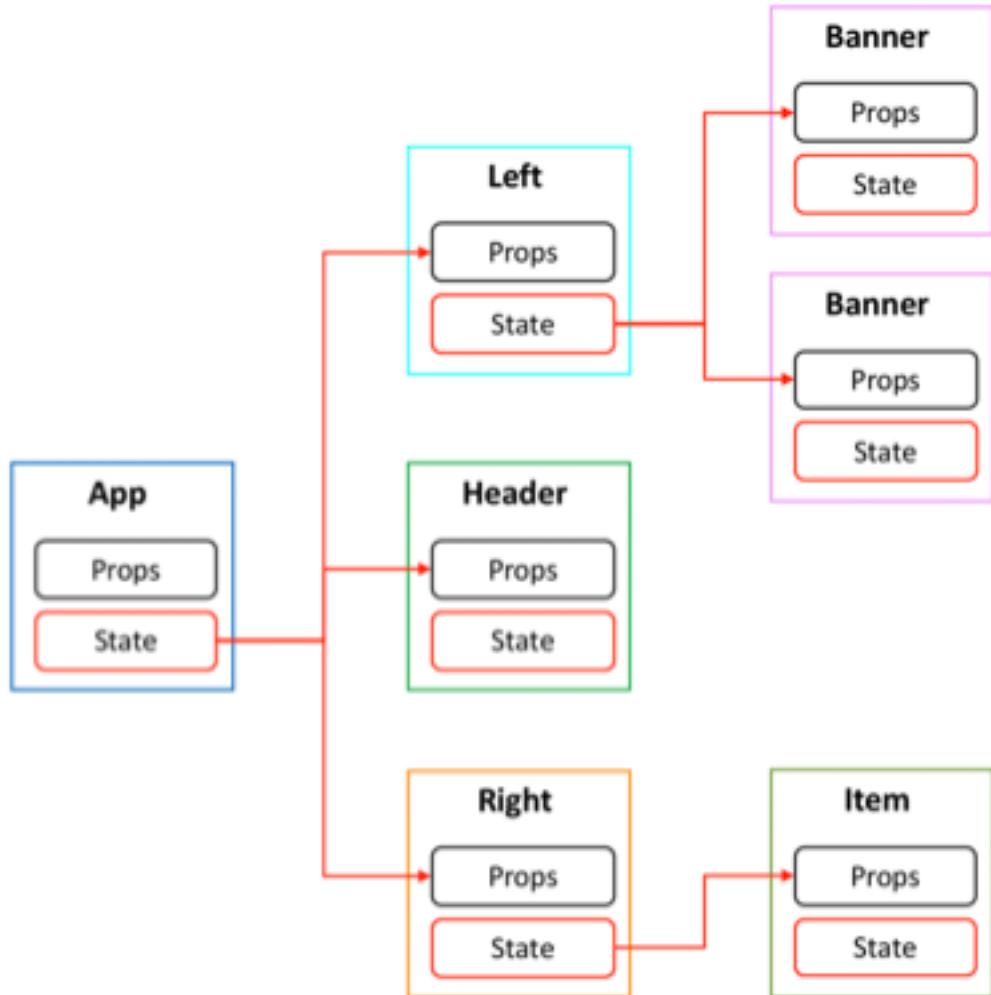
配置表单交互功能完善, 容易使用

配置表单可以内置类型限定, 避免填入错误的配置数据类型

难点2: 如何快速生成模板配置数据编辑表单

4.组件层级关系

组件树定义了组件层级关系，父子组件通过数据流和事件进行关联

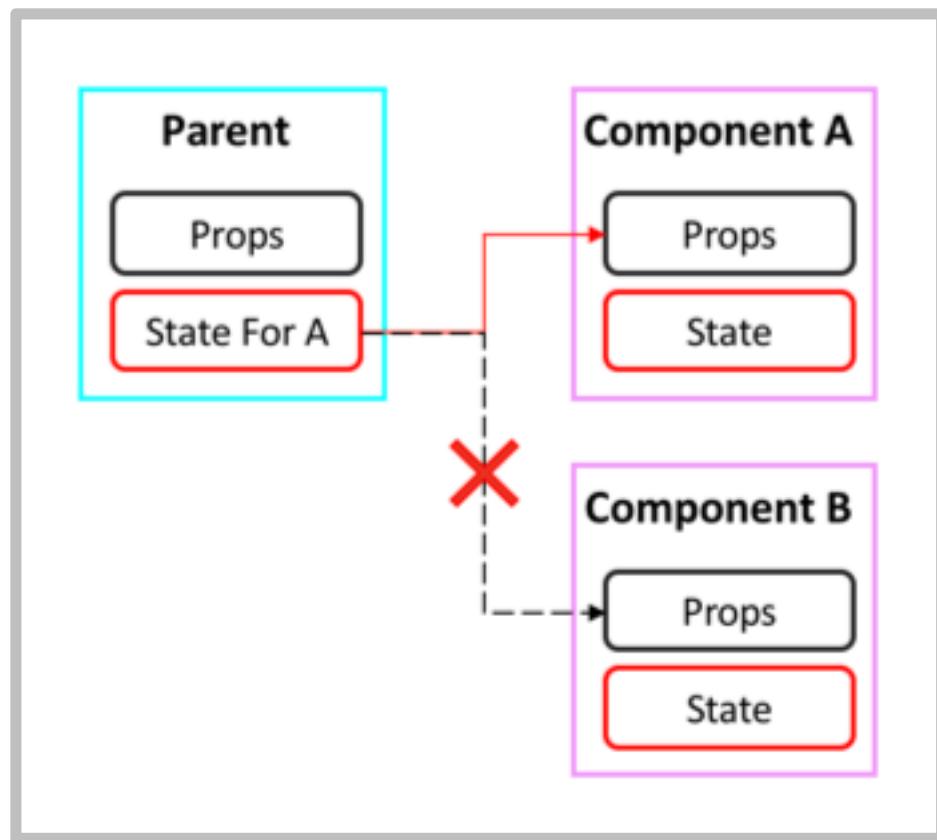


数据从父组件的 State 传递到子组件的 Props

子组件的变更触发 Event 通知父组件

4.组件层级关系

组件层级关系变化影响数据流和布局



HTML ▾

```
<!DOCTYPE html>
<html>
<head>
  <title>组件嵌套</title>
</head>
<body>
  <span>行内组件</span>

  <span>
    行内组件包含块级组件
    <p>块级组件</p>
  行内组件包含块级组件
  </span>
</body>
</html>
```

行内组件 行内组件包含块级组件
块级组件
行内组件包含块级组件

难点3: 如何组织页面组件的层级关系

5. 页面实时预览

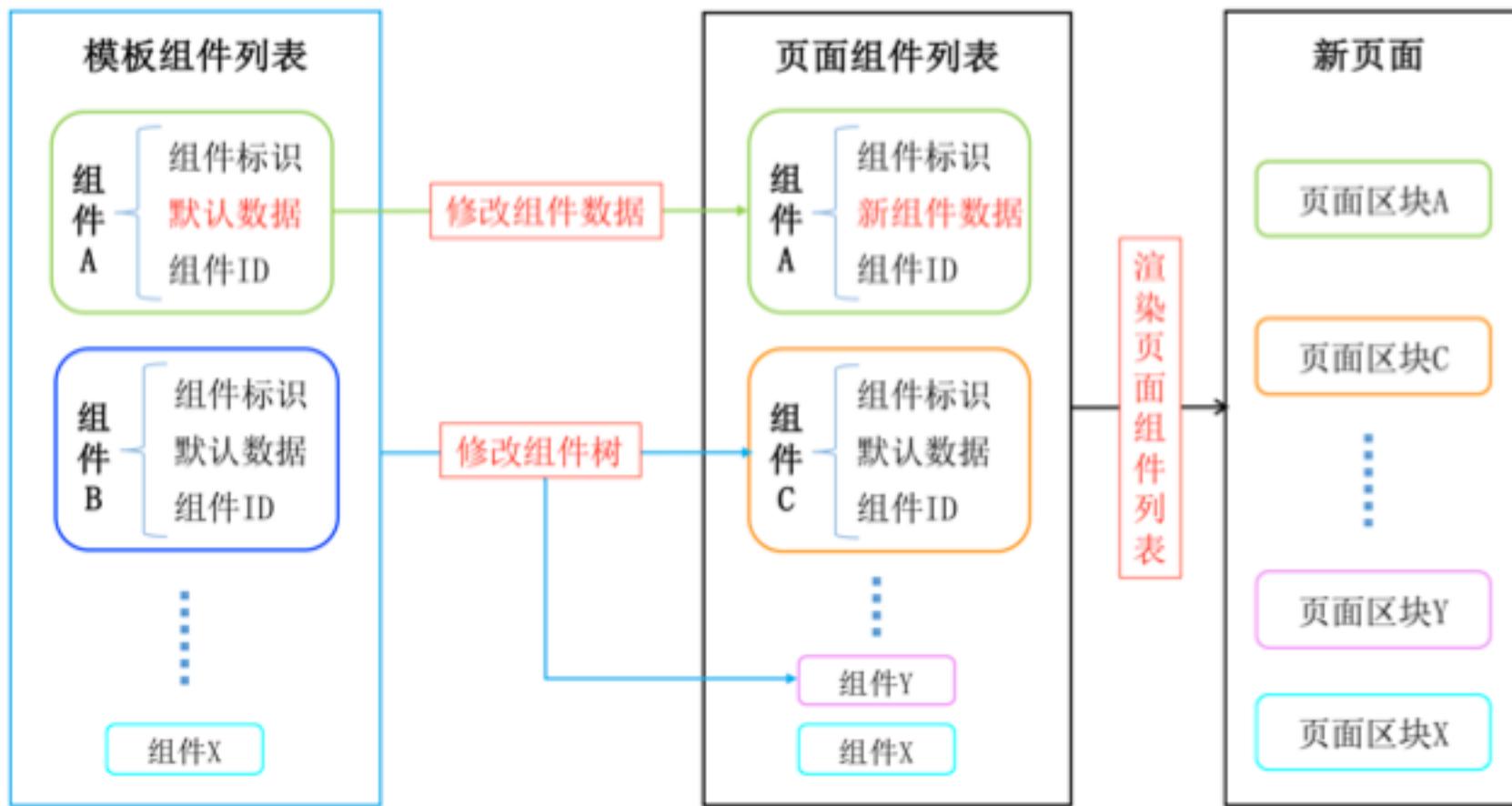
查看和验证可视化编辑的效果



5. 页面实时预览

修改组件树或修改组件数据需要触发重新渲染和预览

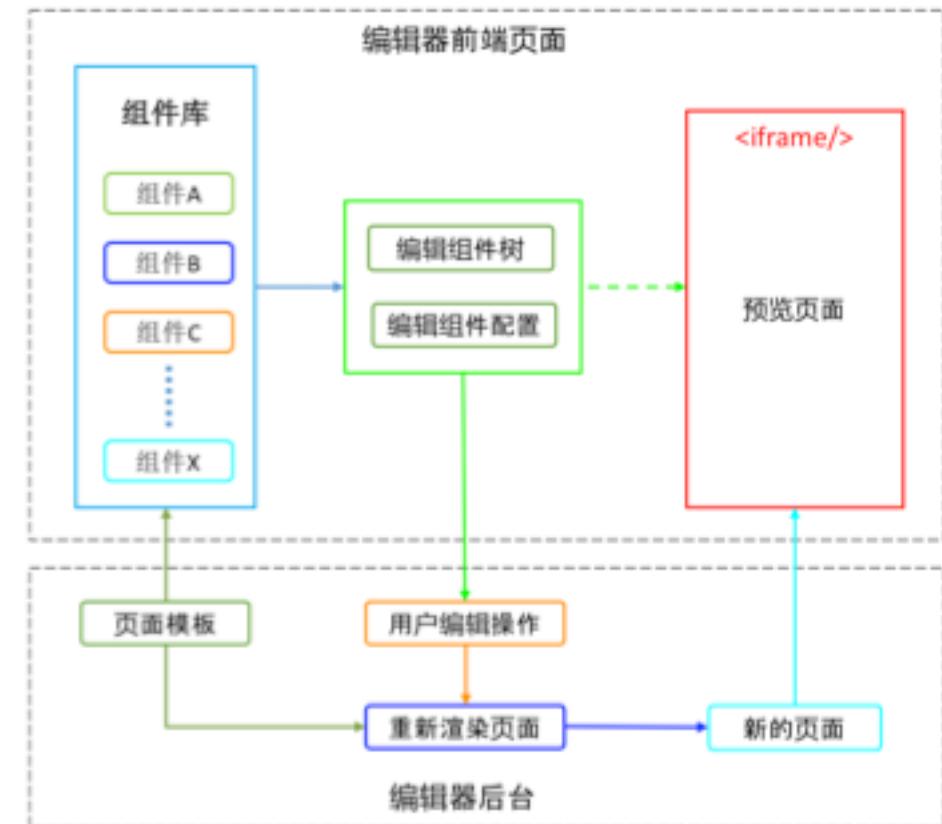
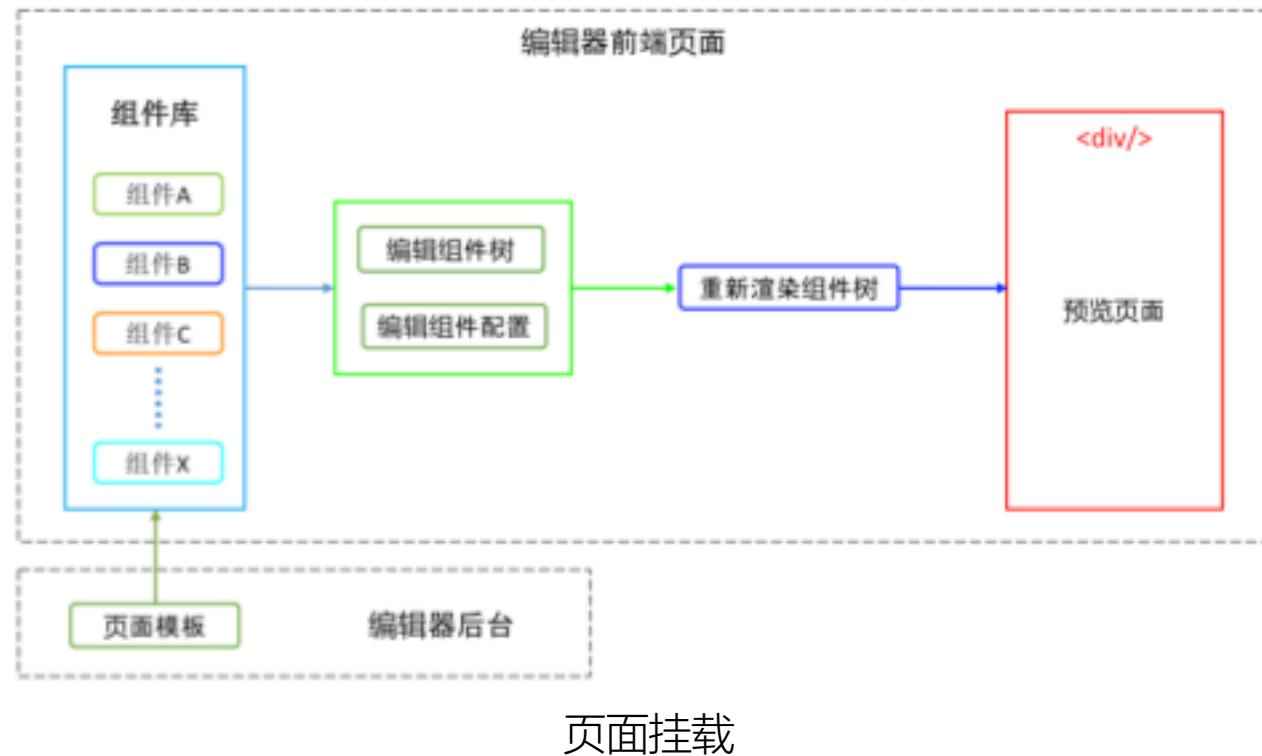
实现页面预览两种方式: **页面挂载和后台渲染**



5. 页面实时预览

页面挂载: 在编辑器前端页面的某个元素节点(div)上渲染出用户编辑结果.

后台渲染: 后台进行页面渲染和生成, 编辑器前端页面通过 iframe 展示



难点4: 如何实现后台渲染

6. 页面构建

关注页面构建效率, 一些场景需要“实时”打包

模板页面开发调试

服务端渲染

7.组件开发

需求: 根据不同业务场景进行业务组件和页面模板的自定义开发

要求:

- 页面可视化搭建工具支持业务前端框架
- 组件和模板的编写方式需遵循较简单的编写约定
- 自定义模板和组件和在开发模式下进行调试和测试

目录

1. 活动页面开发痛点(10%)
2. 可视化搭建工具前世今生(25%)
3. 可视化搭建工具技术要点(35%)
- 4. 理想的运营页面搭建工具(10%)**
5. 开源搭建框架 pipeline(20%)

理想的运营页面搭建工具

概述

运营页面搭建工具, 声明页面配置数据并提供配置表单, 通过对配置表单的数据填充, 实现基于模板的页面生成.



不嵌套的组件

组件铺满页面宽度，在页面高度方向顺序排列



配置表单生成

按照 JSON Schema 对 JSON 数据的描述, 动态渲染出配置表单
对编辑后的数据做格式校验, 避免编辑错误



理想工具

采用组件化和页面模板实现页面生成效率的提升

采用不嵌套的组件层级简化数据流和样式布局

采用 JSON Schema 声明配置数据, 自动生成配置表单

采用后台渲染, 使编辑系统与组件前端框架解耦

在遵循编辑系统约定下, 组件可以自由拓展, 前端框架可以自由选择

目录

1. 活动页面开发痛点(10%)
2. 可视化搭建工具前世今生(25%)
3. 可视化搭建工具技术要点(35%)
4. 理想的运营页面搭建工具(10%)
5. **开源搭建框架 pipeline**(20%)

pipeline



项目地址: <https://github.com/page-pipeline>

体验地址: <https://page-pipeline.github.io/pipeline-editor/dist/#/>

本地部署: [pipeline-document](#)

[pipeline-editor](#)

页面可视化搭建框架的web编辑器 -- <https://page-pipeline.github.io/pipeline-editor/dist/#/>

JavaScript ★ 130 ⚡ 16 Updated 2 days ago

[pipeline-document](#)

页面可视化搭建框架的文档

Updated 10 days ago

[pipeline-mongo](#)

页面可视化搭建框架的 mongo 数据库

JavaScript ⚡ 1 Updated on Nov 16, 2018

[pipeline-node-server](#)

页面可视化搭建框架的后台服务

JavaScript ★ 4 ⚡ 3 Updated on Nov 16, 2018

[pipeline-template-react](#)

页面可视化搭建框架的页面模板 - 基于 React

JavaScript ⚡ 1 Updated on Jul 30, 2018

[pipeline-template](#)

页面可视化搭建框架的页面模板 - 基于 Vue

JavaScript ★ 6 ⚡ 3 Updated on Jul 29, 2018

pipeline

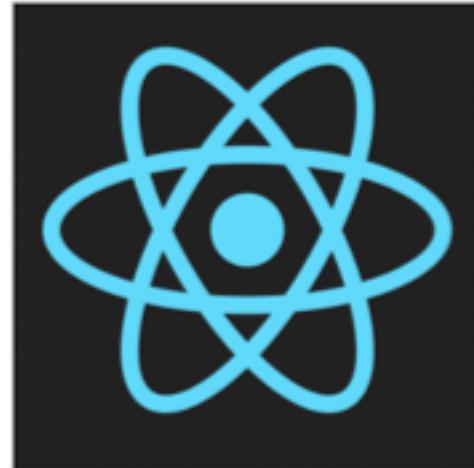
支持的前端框架模板列表

pipeline 支持不同前端框架开发的页面模板, 无缝对接业务现有前端技术栈.



Vue框架模板

使用



React框架模板

使用

开源页面可视化搭建框架

pipeline

组件库

- ★ 间隔区示例
- ★ 头部区示例
- ★ 信息区示例
- ★ 查询天气示例

(拖拽或双击)

模板组件

- 头部区示例
- 信息区示例
- 间隔区示例
- 查询天气示例
- 间隔区示例
- 头部区示例

刷新 新页面预览

Hello Pipeline

页面可视化搭建框架



框架特性

- 可视化页面搭建框架
- 支持自由拓展页面组件
- 自定义页面可配置字段

模板工程/编辑器/后台服务解偶

头部区 折叠

头部标题

页面可视化搭建框架

头部图片

<https://avatars3.githubusercontent.com/u/3>

试试输入新的图片URL:
<https://vuejs.org/images/logo.png>

头部图片点击跳转

<https://github.com/page-pipeline>

提交配置数据

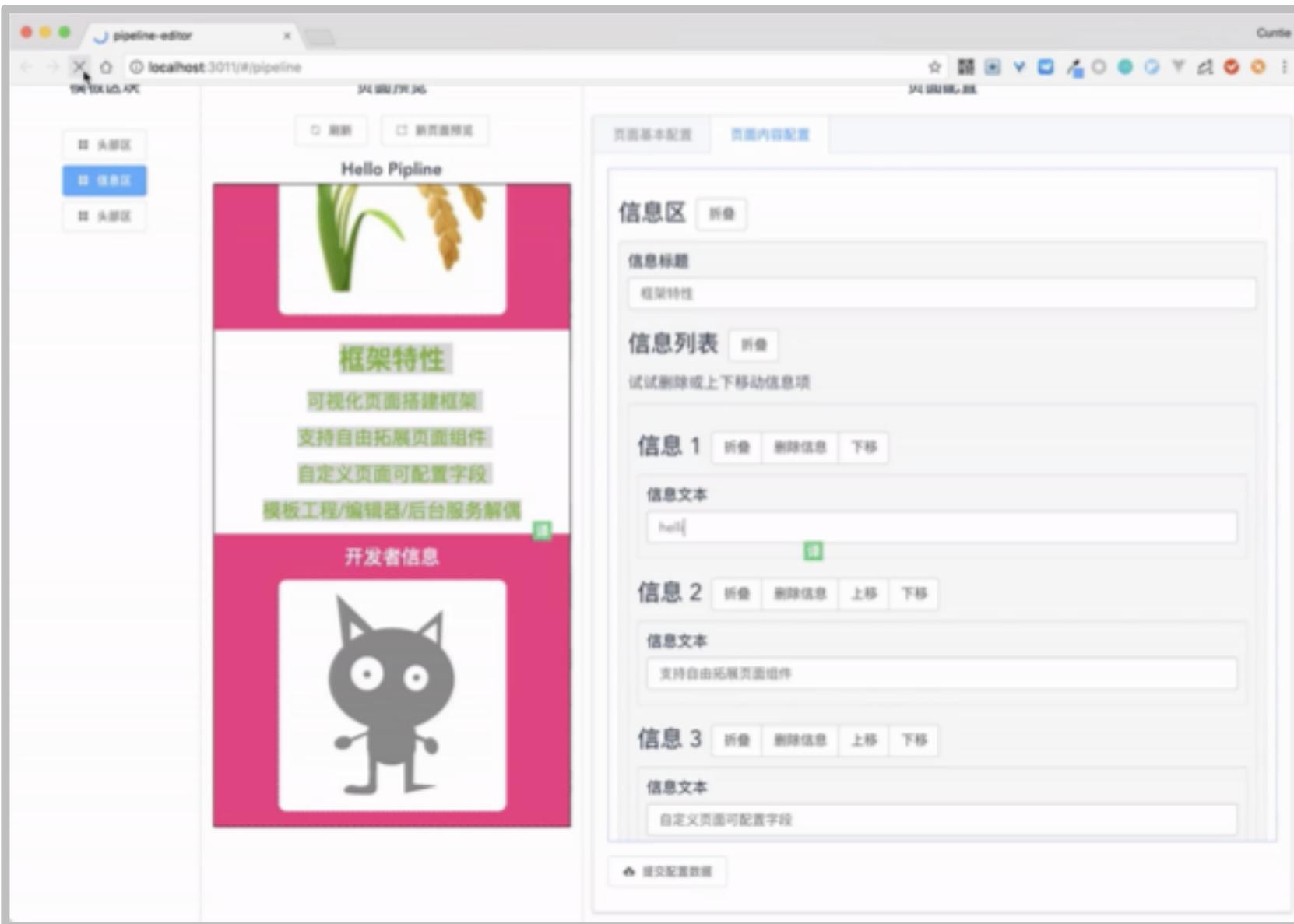
开源页面可视化搭建框架

pipeline

该图展示了Pipeline开源页面可视化搭建框架的界面，分为三个主要部分：

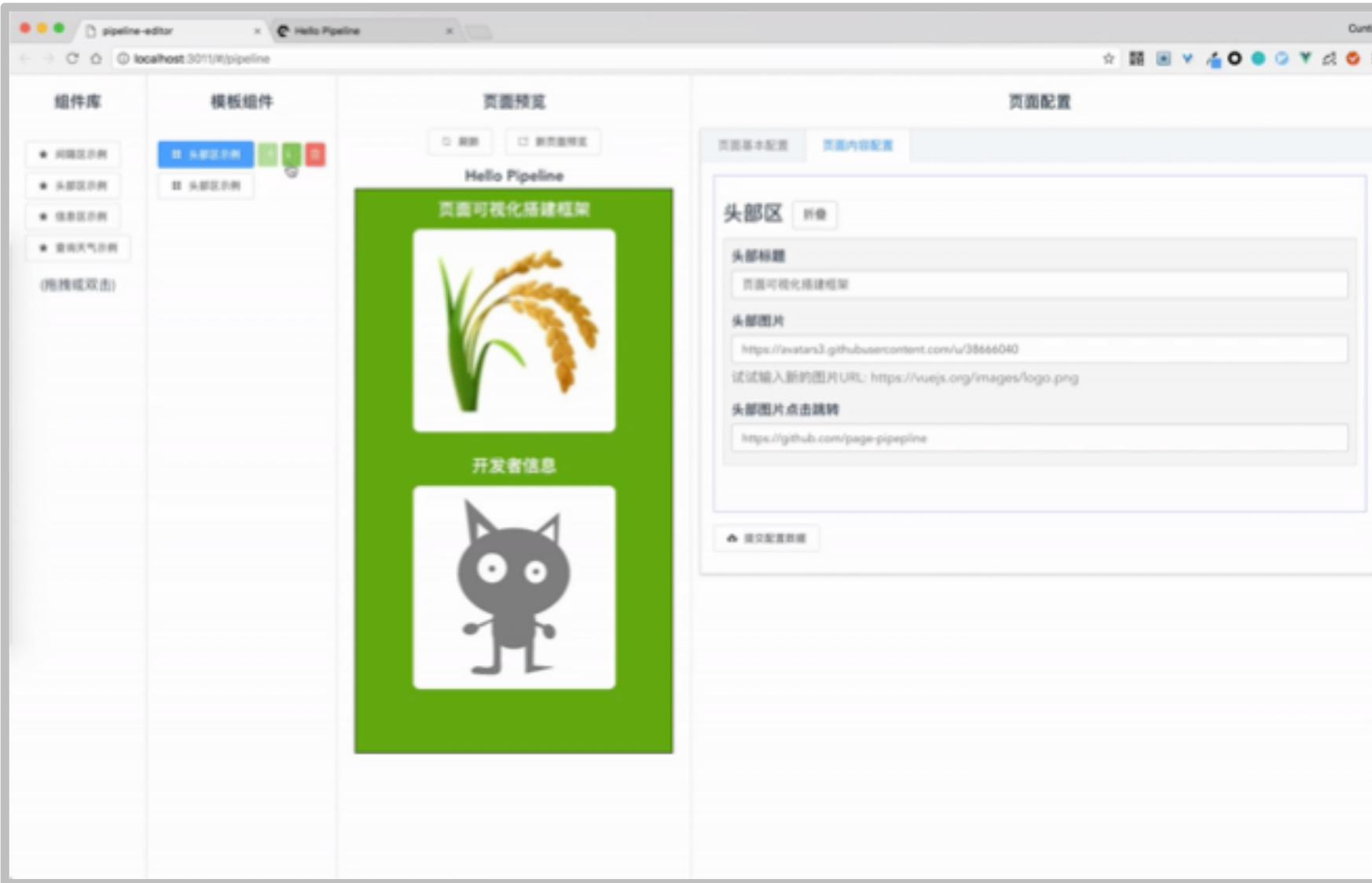
- 左侧工具栏：**包含“组件库”和“模板组件”两个模块。组件库中列出了“间隔区示例”、“头部区示例”、“信息区示例”、“查询天气示例”以及“(拖拽或双击)”。模板组件中列出了“头部区示例”、“信息区示例”、“间隔区示例”、“查询天气示例”、“间隔区示例”和“头部区示例”。右侧有两块蓝色标签：“页面预览”和“页面配置表单”，分别指向中间和右侧的子界面。
- 中间区域（由“页面预览”标签指向）：**显示了一个预览窗口，标题为“Hello Pipeline”，下方有“页面可视化搭建框架”的说明文字和一幅稻穗插画。下方文字包括：“框架特性”、“可视化页面搭建框架”、“支持自由拓展页面组件”、“自定义页面可配置字段”和“模板工程/编辑器/后台服务解耦”。
- 右侧区域（由“页面配置表单”标签指向）：**显示了一个配置表单，包含“页面基本配置”和“页面内容配置”两个子项。在“头部区”配置项下，有“头部标题”输入框（值为“页面可视化搭建框架”）、“头部图片”输入框（值为“<https://avatars3.githubusercontent.com/u/3>”）和“头部图片点击跳转”输入框（值为“<https://github.com/page-pipeline>”）。底部有一个“提交配置数据”按钮。

基本操作 demo



修改页面全局配置
可视化修改组件内容
页面实时预览
即刻获取结果页面
页面支持业务逻辑

组件拖拽 demo



动态增删页面组件
可视化的组件拖拽
页面支持业务逻辑

框架特点

- 模板工程/编辑器/后台服务解偶
- 支持自由拓展页面组件
- 自定义页面可配置字段
- 组件动态增减, 组件拖拽
- 模板工程前端框架无关: 支持 vue 和 react

下一步工作

- 撰写项目使用文档
- 提供更多前端框架模板
- 提供更多的可视化交互

Q&A

陈韩杰 [cntchen](https://www.cntchen.com)
cntchen@gmail.com



References

项目地址: <https://github.com/page-pipeline>

体验地址: <https://page-pipeline.github.io/pipeline-editor/dist/#/>

本地部署: [pipeline-document](#)