PageNow使用手册V1.1

作者：黄健；最后更新日期：2021年01月04日

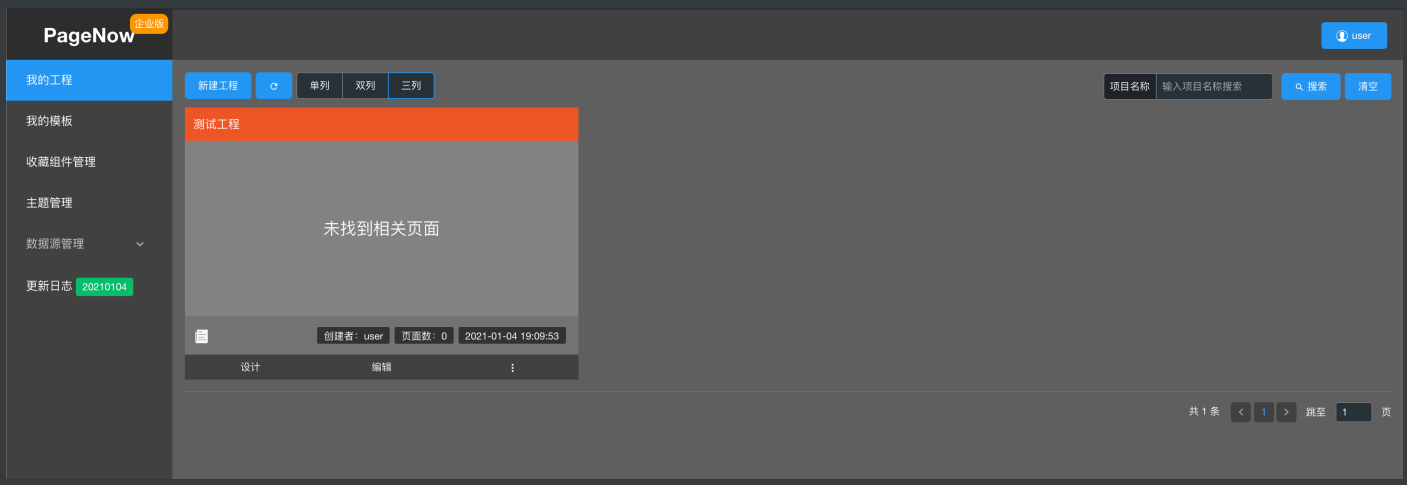
前言

在官网http://www.pagenow.cn/中，有系统使用的一些视频教程，如果看本文档还有不明白的地方，可以前往官网中进行查看相关视频教程来帮助您更快的上手PageNow的使用，也可以在官方QQ交流群内向我们提问。

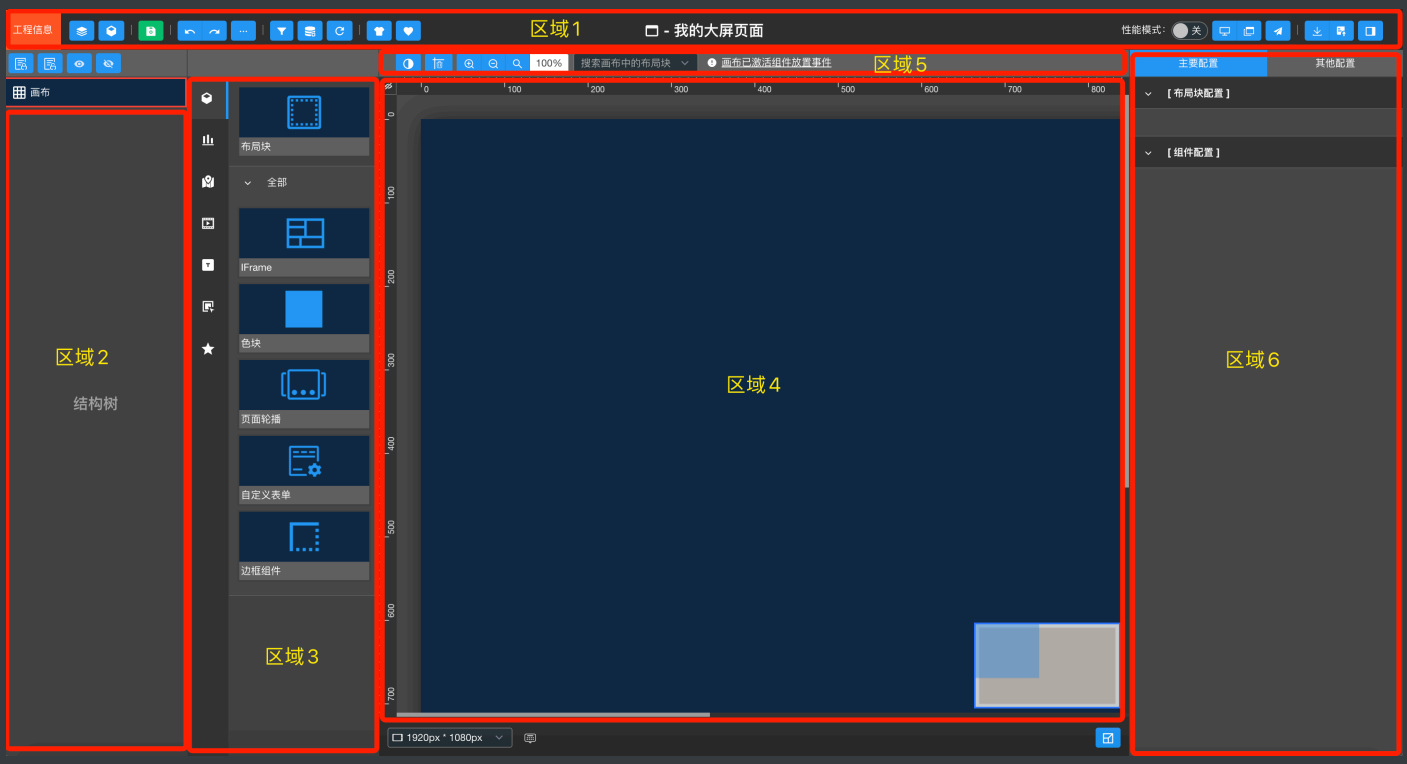
QQ官方交流群：1072768942

软件界面介绍

系统管理控制台



设计器



区域1：顶部工具栏

区域2：页面层级结构树展示

区域3：组件库

区域4：画布主要设计区域

区域5：画布工具栏

区域6：配置区域

基本概念

布局块

PageNow中，所有可配置的组件，外层都会有一层布局块，组件依托于布局块显示，当然布局块可以解绑组件单独展示

动态数据源

PageNow中，大部分组件都需要绑定数据源来实现不同的展示效果，组件默认绑定的是静态JSON数据，可以通过组件相关数据源配置，将静态数据源切换为动态数据源

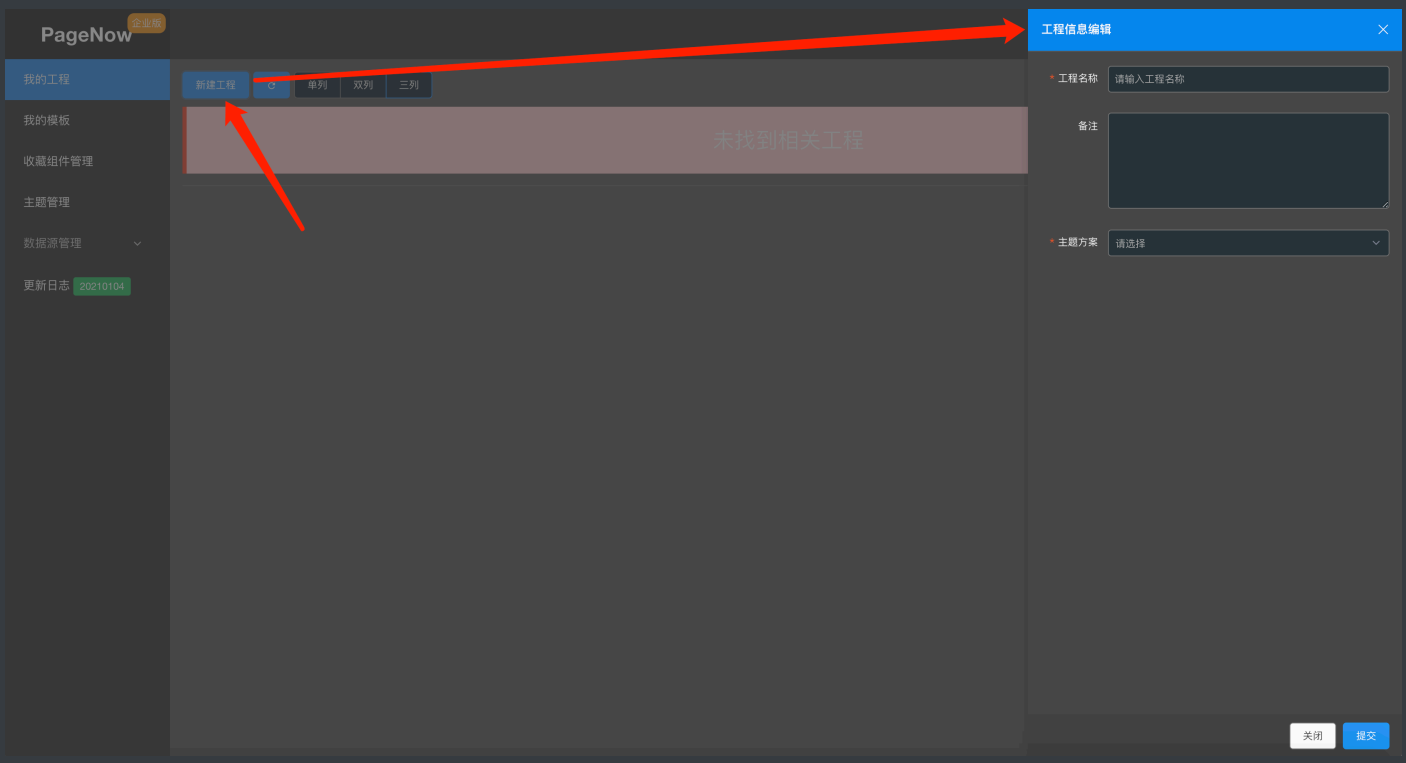
静态数据源：静态JSON数据、CSV文件

动态数据源：API接口，数据库直连（Oracle、MySQL、MsSQL、PostgreSQL）

系统管理控制台

创建工程

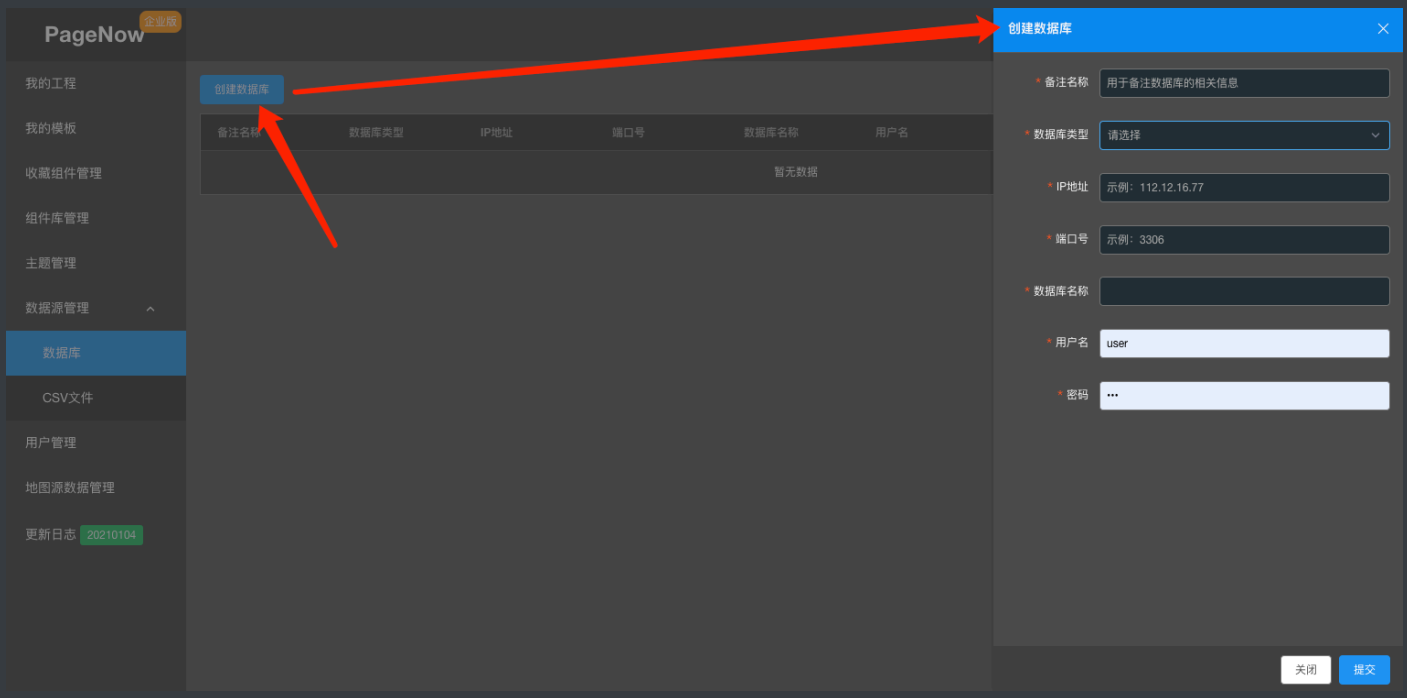
PageNow中数据大屏页面以工程为单位进行归类，工程下可以编辑任意多个页面，因此我们需要先创建一个工程，在【我的工程】界面中点击【新建工程】打开工程信息编辑表单



主题方案：此主题方案将应用于此工程下所有页面的Echarts图表组件

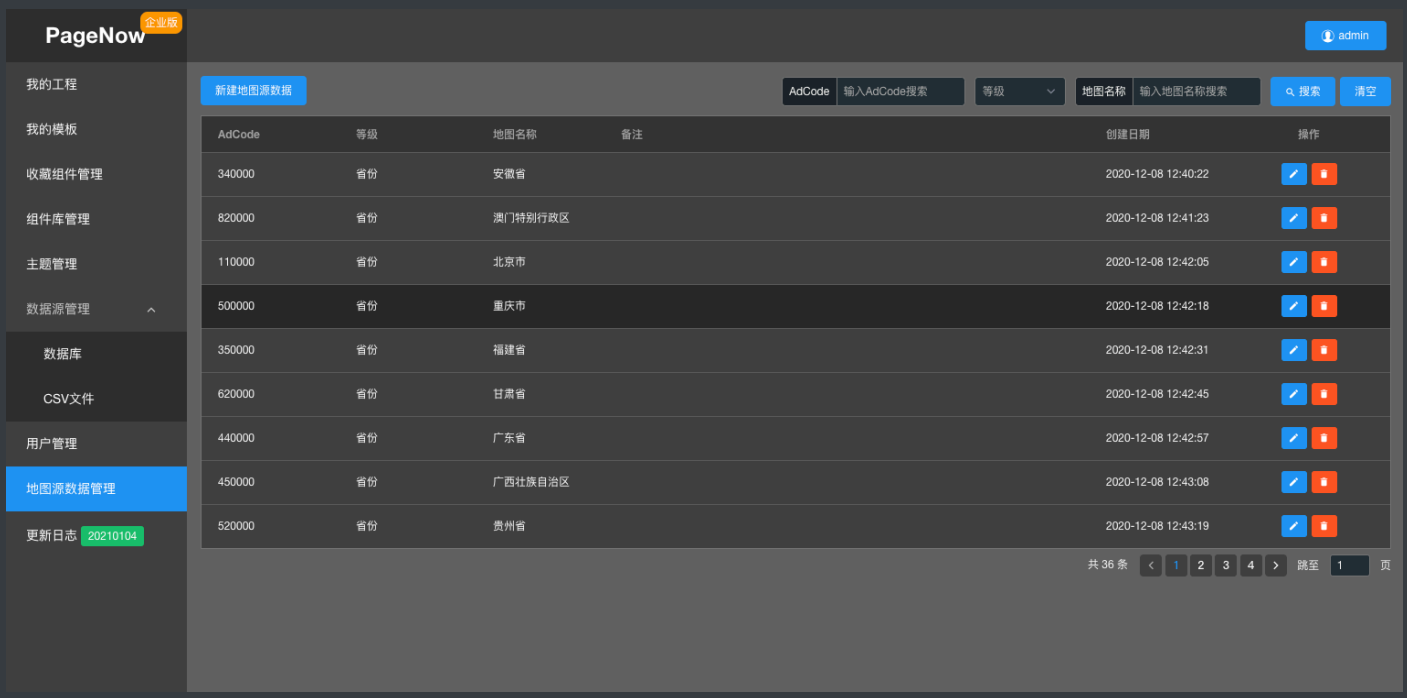
数据源管理

使用管理员账号登录系统，可以创建数据库数据源，创建的数据库，在组件需要绑定数据库作为动态数据源时可以选择使用，点击【创建数据库】，即可打开创建数据库的表单窗口



地图源数据管理

使用管理员账号登录系统，可以进行地图源数据的管理

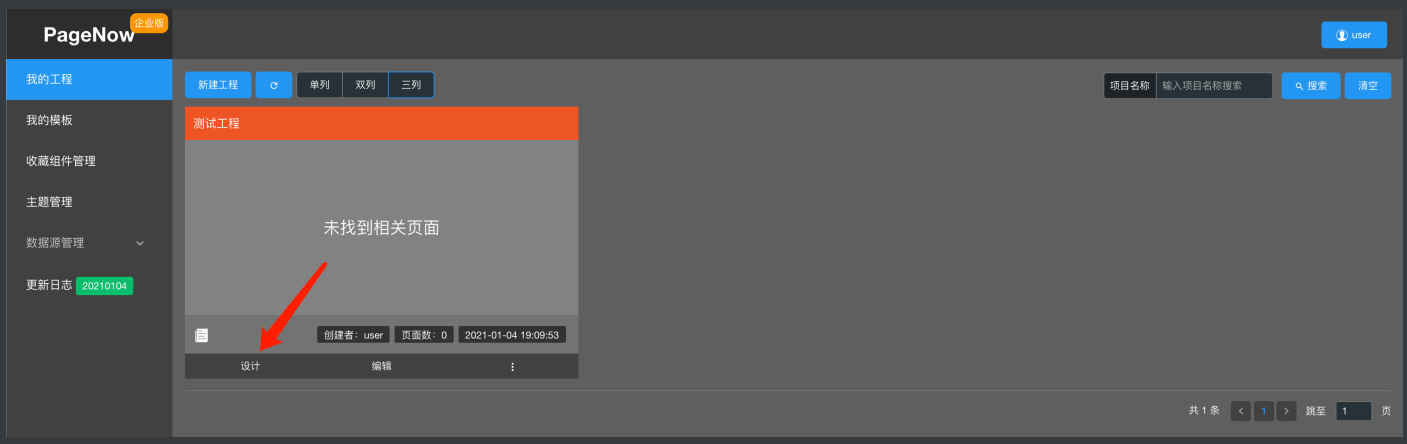


地图源数据记录中存储地区的GeoJson数据，PageNow中默认存储了世界、中国、中国各省份的GeoJson数据，这些数据在【中国省份地图】组件中会使用到。

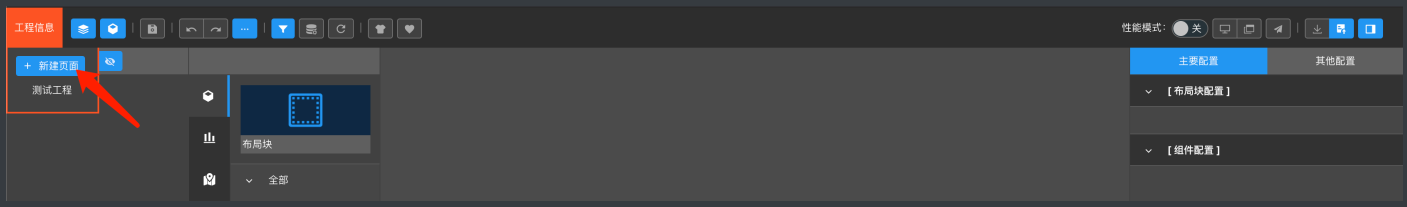
如果您需要在【中国省份地图】中展示某市区的地图数据，就需要【新建地图源数据】，我们建议通过阿里的地图选择器GeoAtlas（http://datav.aliyun.com/tools/atlas/#&lat=30.316551722910077&lng=106.75113472219310&zoom=3.5）来获取您需要的地图数据

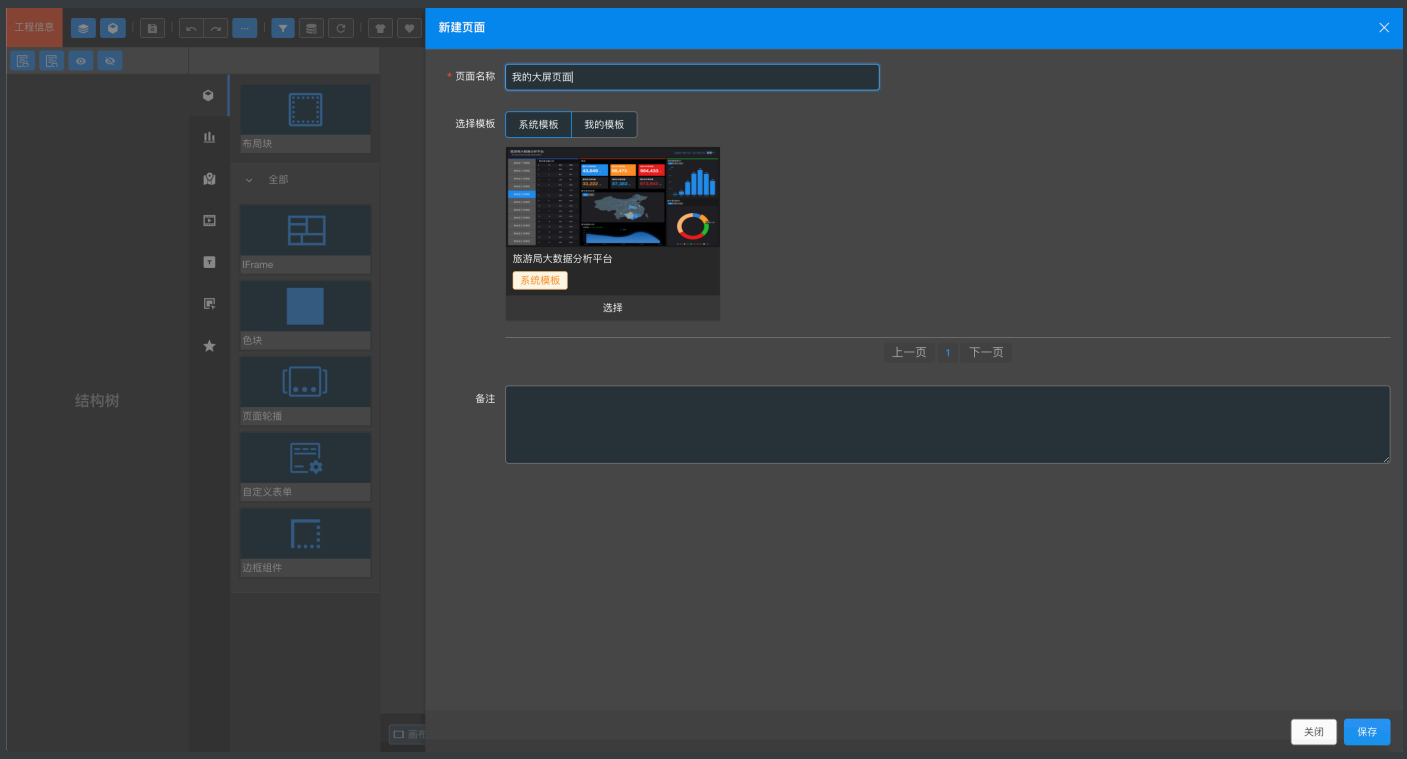
创建数据大屏页面

点击工程信息卡片的【设计】按钮，即可进入设计器界面



创建页面需要将鼠标滑入设计器界面左上角的【工程信息】处，会自动弹出工程下的页面列表窗口，点击【新建页面】即可打开创建页面的表单





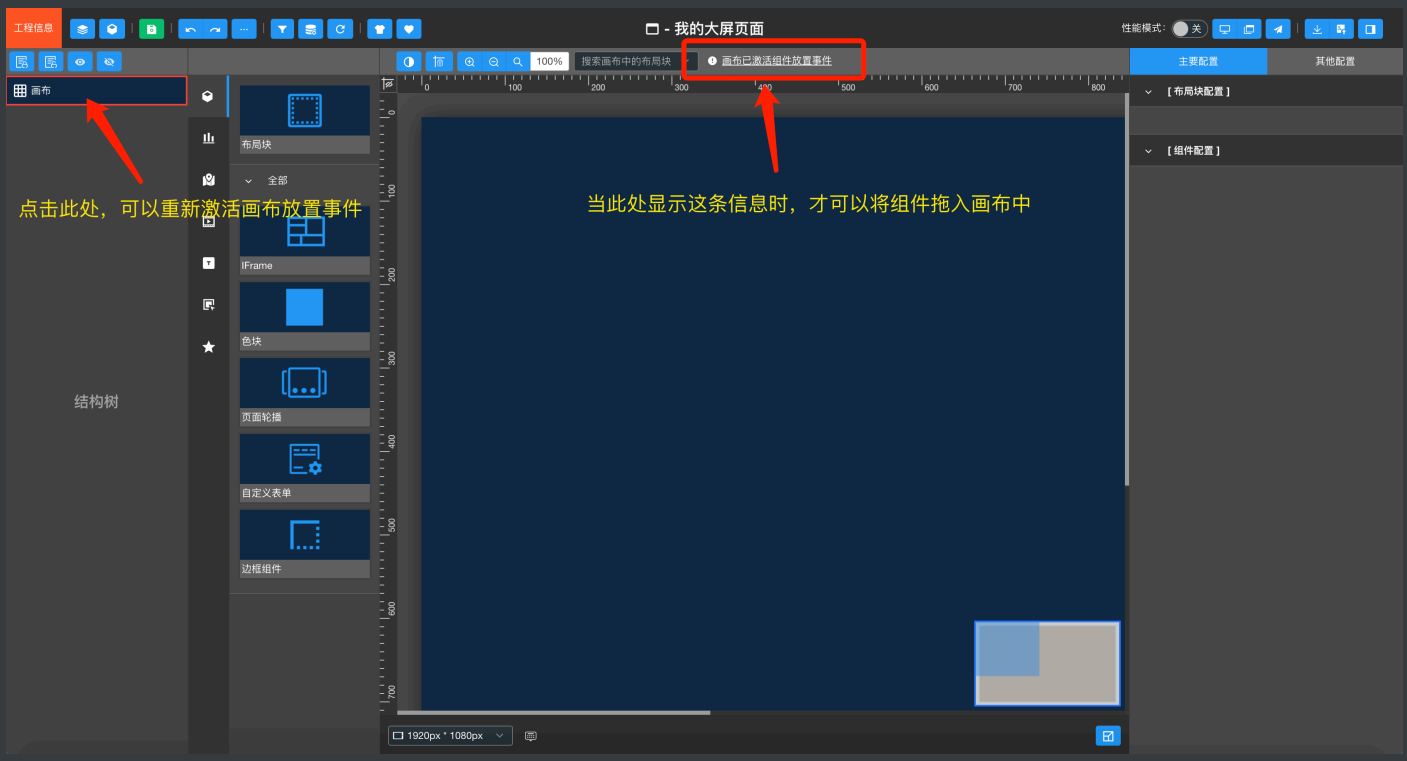
新建页面表单窗中，填入页面名称提交保存即可，当然我们也可以选择系统模板或我的模板来根据模板生成页面。

设计器使用说明

本章节将对PageNow设计器的一些特别的操作进行说明

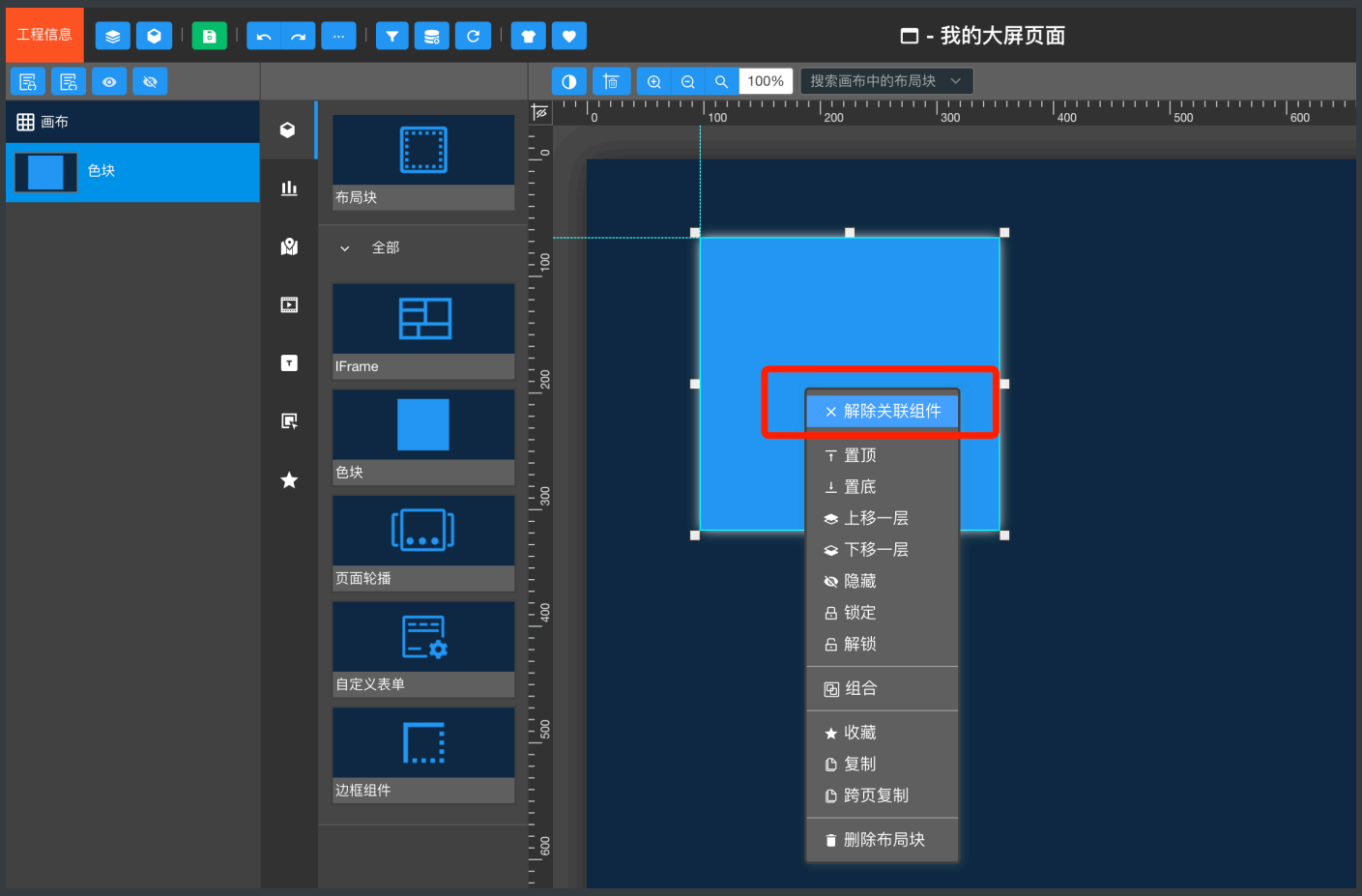
如何将组件拖入画布

PageNow设计器中，要将组件拖入画布进行排版，需要在画布工具栏中显示有【画布已激活组件放置事件】信息时才可以将组件库中的组件拖入画布中，当我们点击画布中某个具体的组件或组时，画布放置事件将会失效，此时如果我们还想继续拖入组件到画布中，需要再次点击画布空白区域或者点击层级结构树区域处的画布卡片来激活画布的放置事件



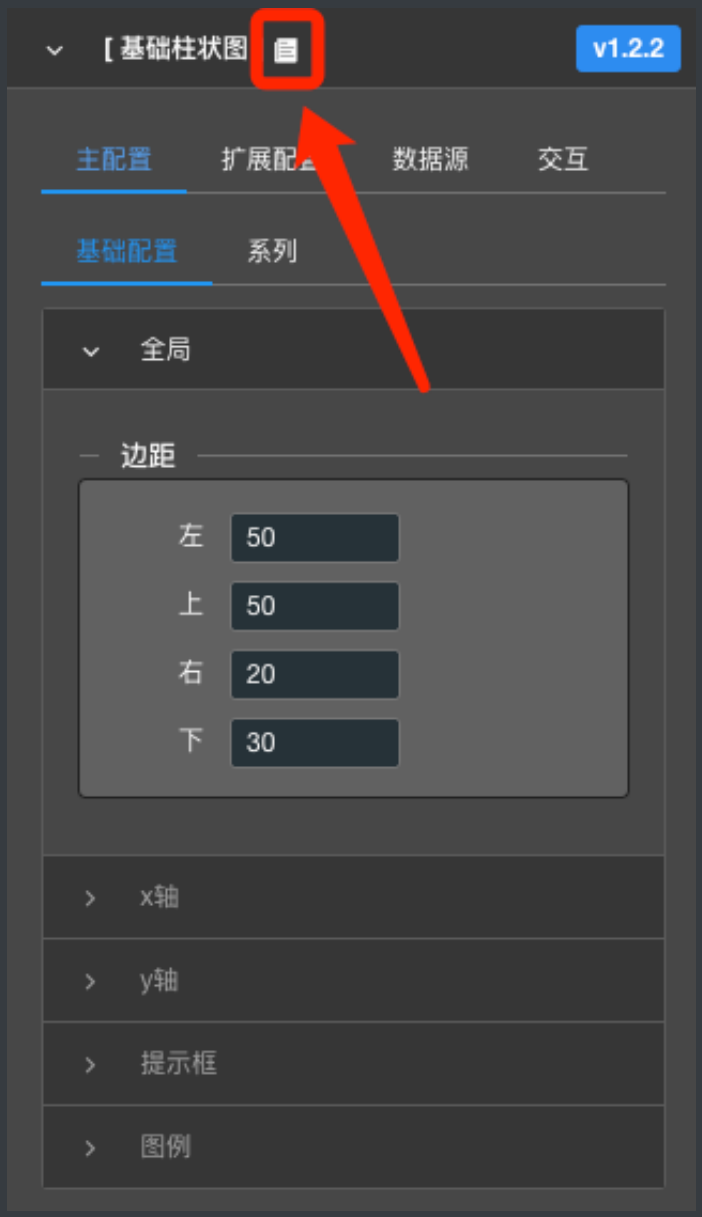
解绑关联组件

在基本概念中的布局块讲解中，说明到，PageNow中所有可拖拽至画布中的组件，外层都有一层布局块，我们可以在组件右键菜单中选择【解绑关联组件】将当前布局块绑定的组件进行解绑，然后就可以在组件库中拖拽其他的组件到此布局块中进行再绑定



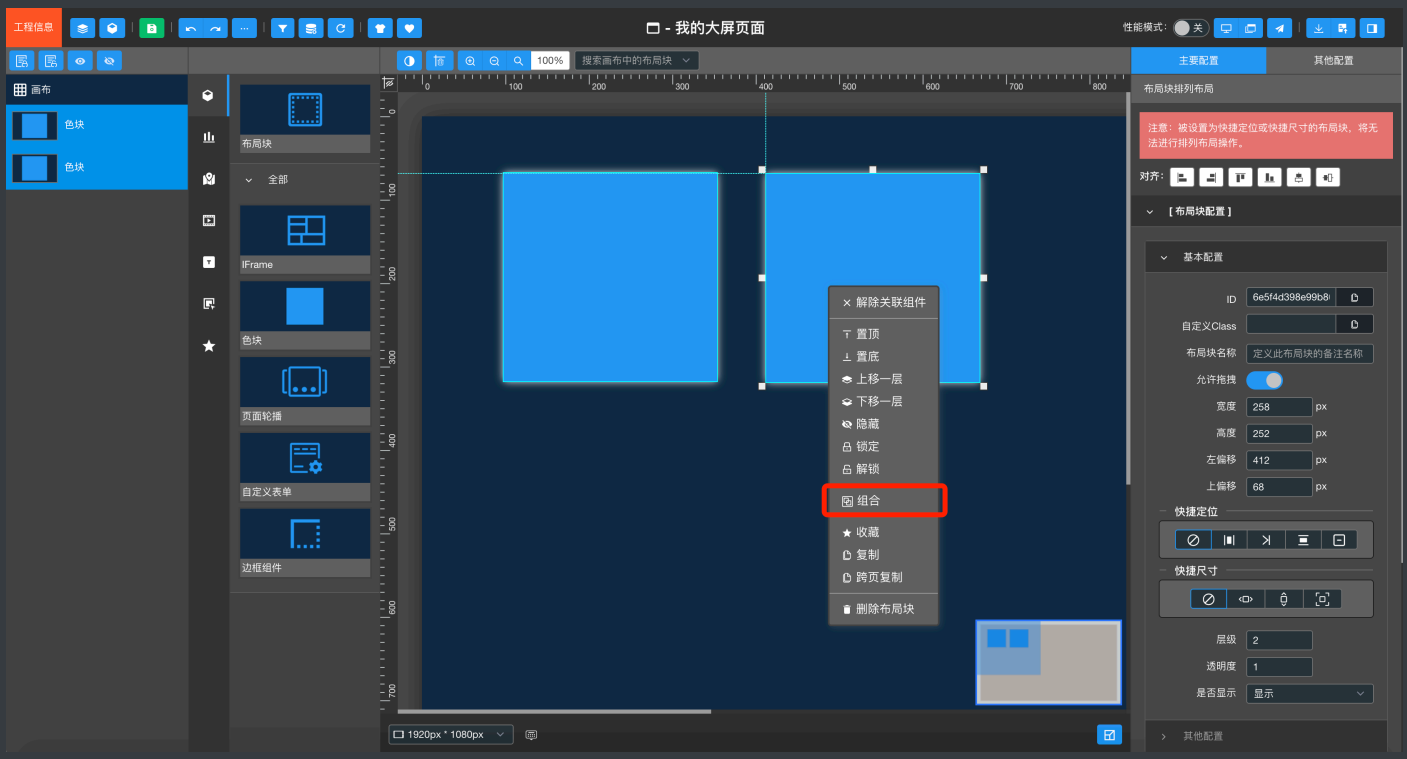
组件使用说明

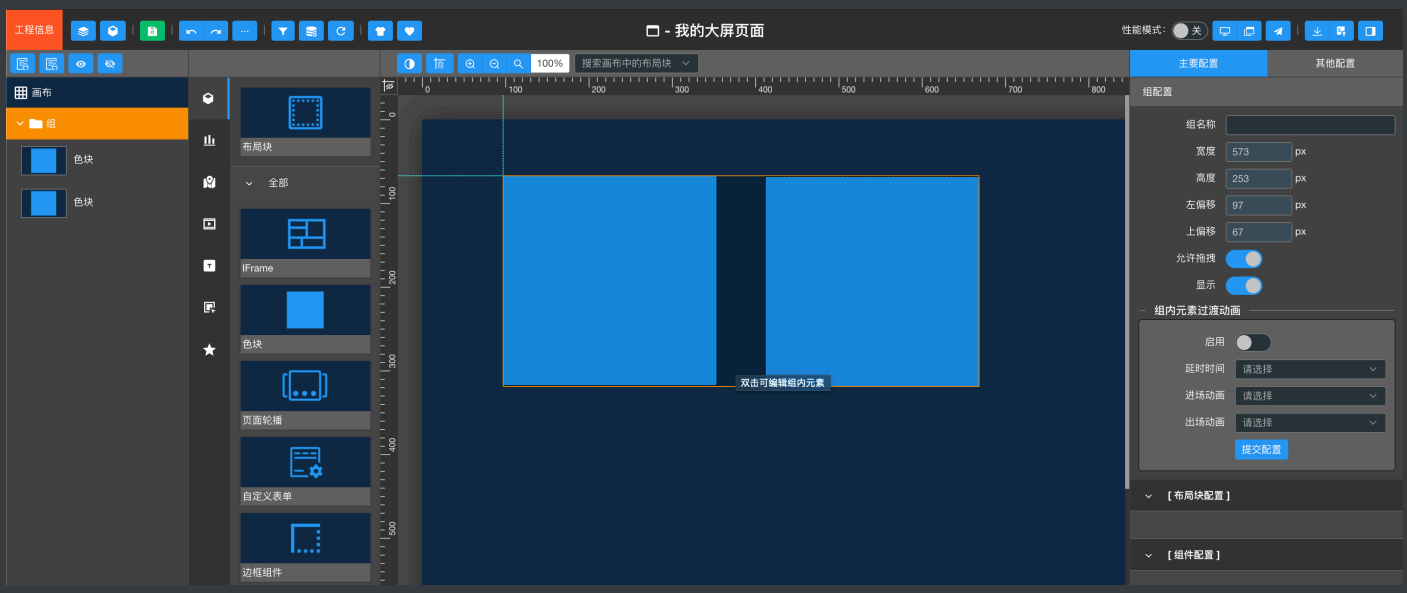
在PageNow中，大部分组件的使用通过配置都可以一目了然的了解到组件所拥有的功能，当然也有部分组件有自己特殊的使用方式，我们可以通过点击下图所示图表，查看组件的使用说明文档



组合

可以多选多个组件之后，然后右键菜单中选择成组，将多个组件进行成组，方便对多个组件进行整体的尺寸、位置等信息的编辑

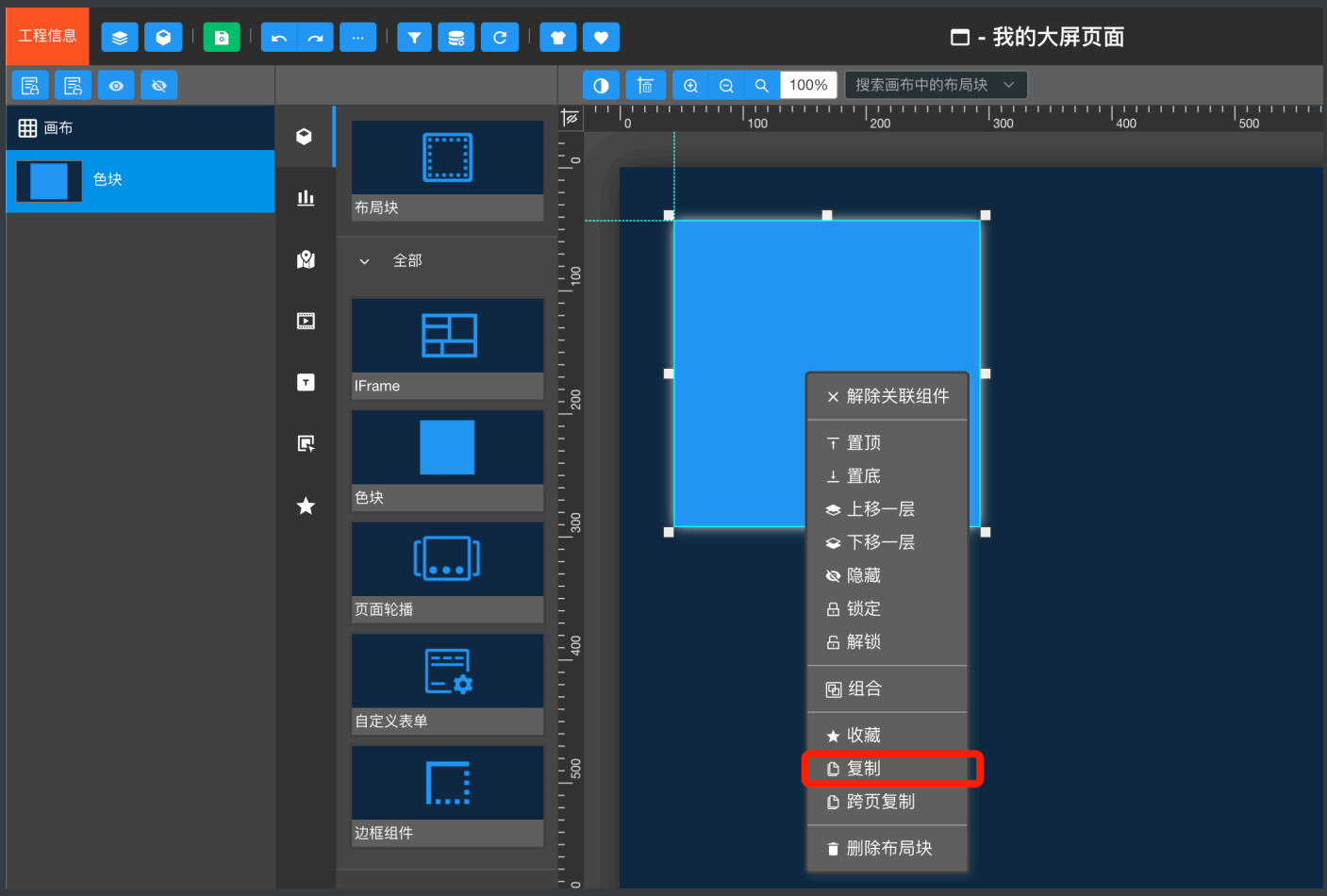




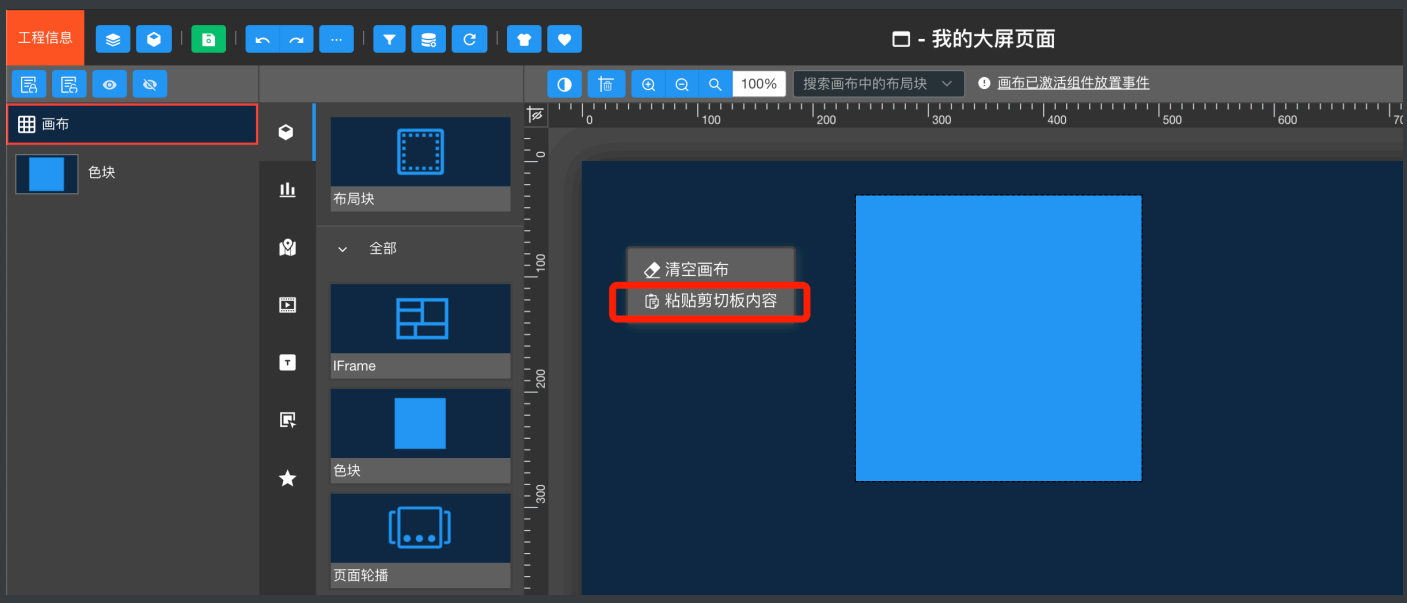
当多个组件被成组之后，我们点击组时，右侧的【主要配置】区域会自动显示组的相关配置，这里我们可以给组进行名称自定义，以及设置组的【允许拖拽】、【显示】等配置，这些配置将会级联影响组下所有的组件，如需编辑组内的组件，我们在组上双击，即可打开组内元素的编辑限制

组件的复制与跨页复制

选中一个或多个组件，在右键菜单中点击【复制】，即可在当前页面复制生成选中的组件



点击【跨页复制】，即可将多个组件复制至剪切板，然后我们可以在其他任意页面中，在画布空白区域右键，选择【粘贴剪切板内容】将复制的组件粘贴至当前页面中



保存页面

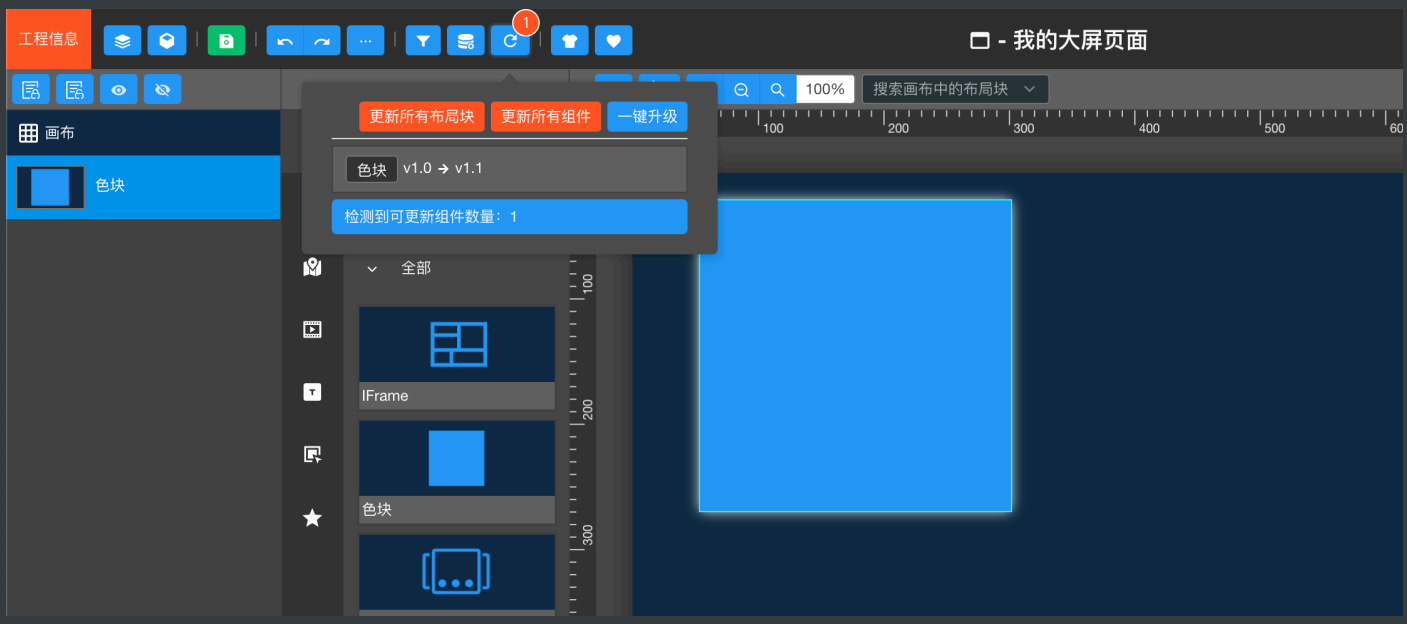
在设计器中编辑数据大屏页面时，系统是不会自动对编辑的操作进行保存的，我们需要手动在顶部工具栏中点击【保存】按钮对编辑的状态进行保存，也可以使用快捷键Ctrl/Cmd+S进行保存

撤销、单步恢复

设计器中可以撤销相关的操作，目前PageNow支持多步撤销，但恢复只支持单步恢复，也可以使用快捷键Ctrl/Cmd+Z或Ctrl/Cmd+Y进行撤销与恢复操作

升级组件

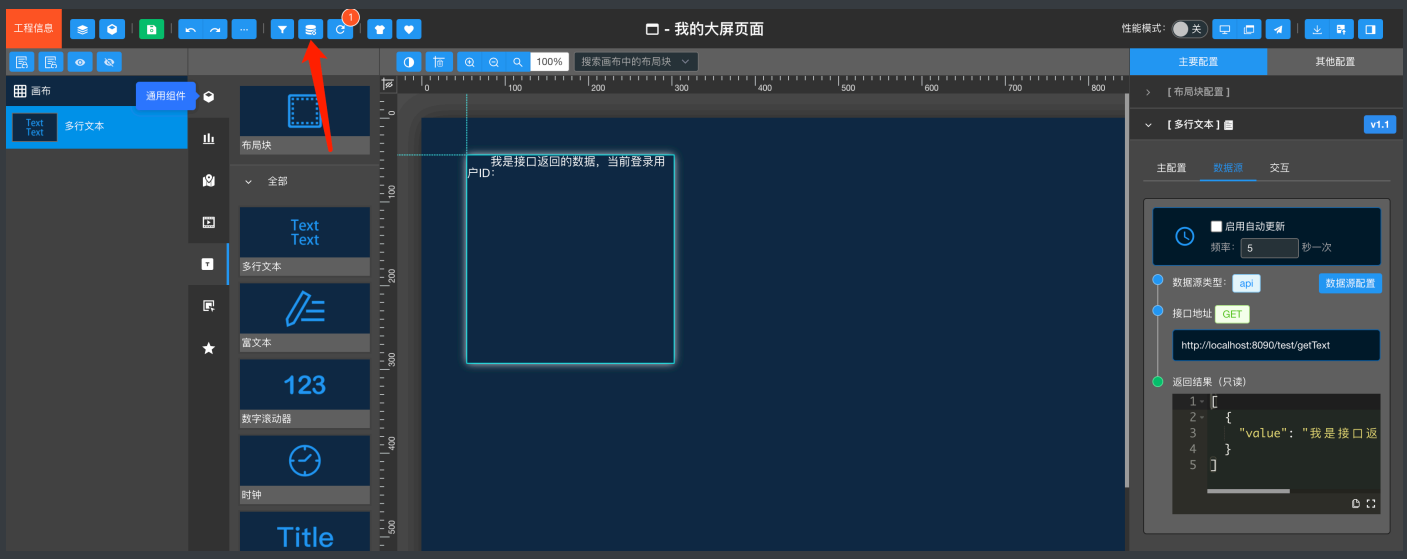
当前编辑页面如果存在可升级的组件时，会在顶部工具栏的【升级组件】按钮上显示可升级的组件数量

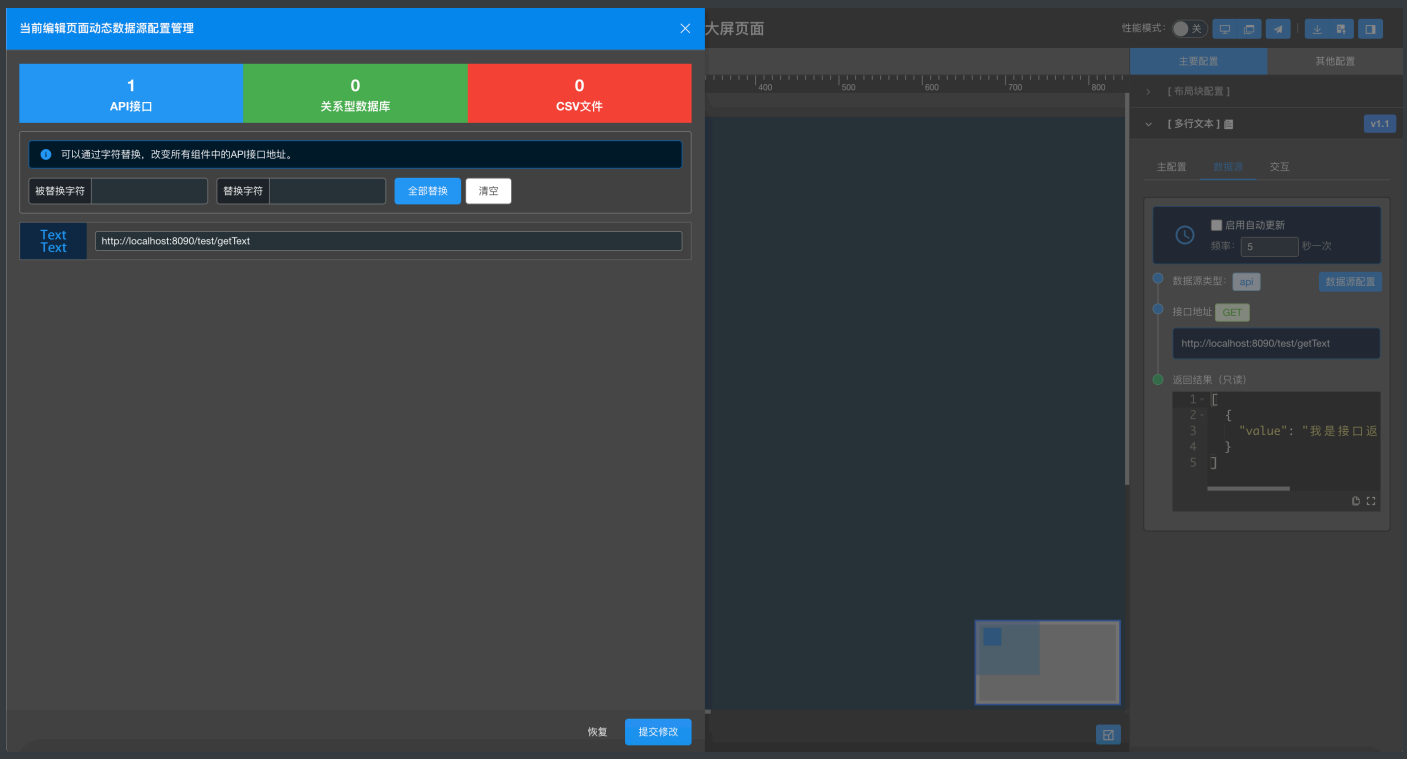


点击【一键升级】可以将可更新的组件升级至对应版本，点击【更新所有组件】可以强制更新当前页面内所有组件的版本，点击【更新所有布局块】可以强制更新当前页面内所有的布局块的版本

温馨提示：如果当前编辑页面存在可更新的组件时，只管升级即可，因为部分组件如果不升级至最新版，可能会影响组件的正常功能

当前编辑页面动态数据源配置管理





此功能可以统计当前编辑页面内所有组件绑定的数据源的数量，当页面中，存在有组件使用API接口数据源时，会罗列出这些组件内绑定的接口地址，可以通过【被替换字符】与【替换字符】将这些组件绑定的API接口地址进行全局的字符替换。例如：将所有组件绑定的API接口地址从测试环境地址切换为正式环境地址这样的场景下，就会用到此项功能

（后续还会对此功能进行扩展，实现SQL脚本的统一管理等）

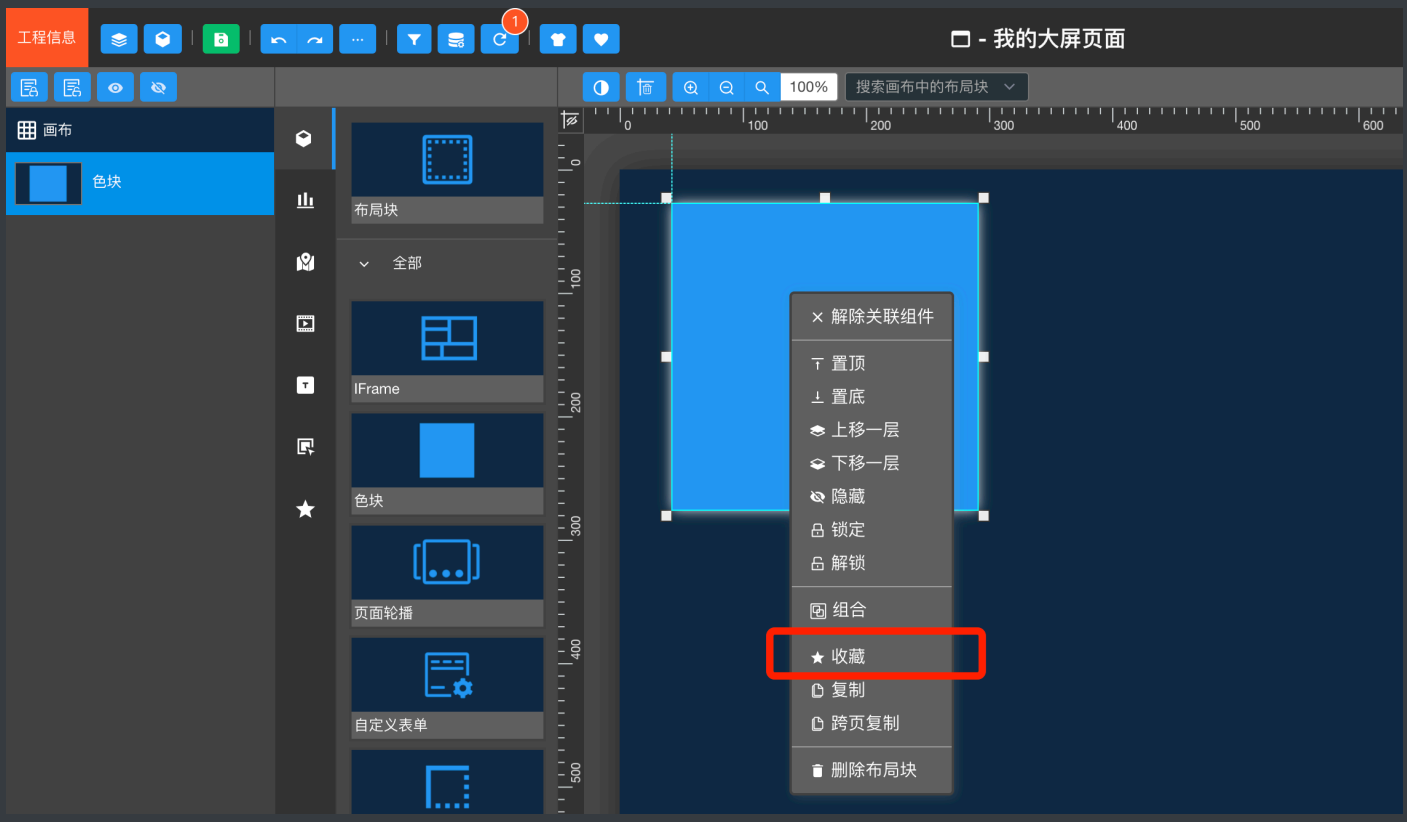
页面收藏为模板

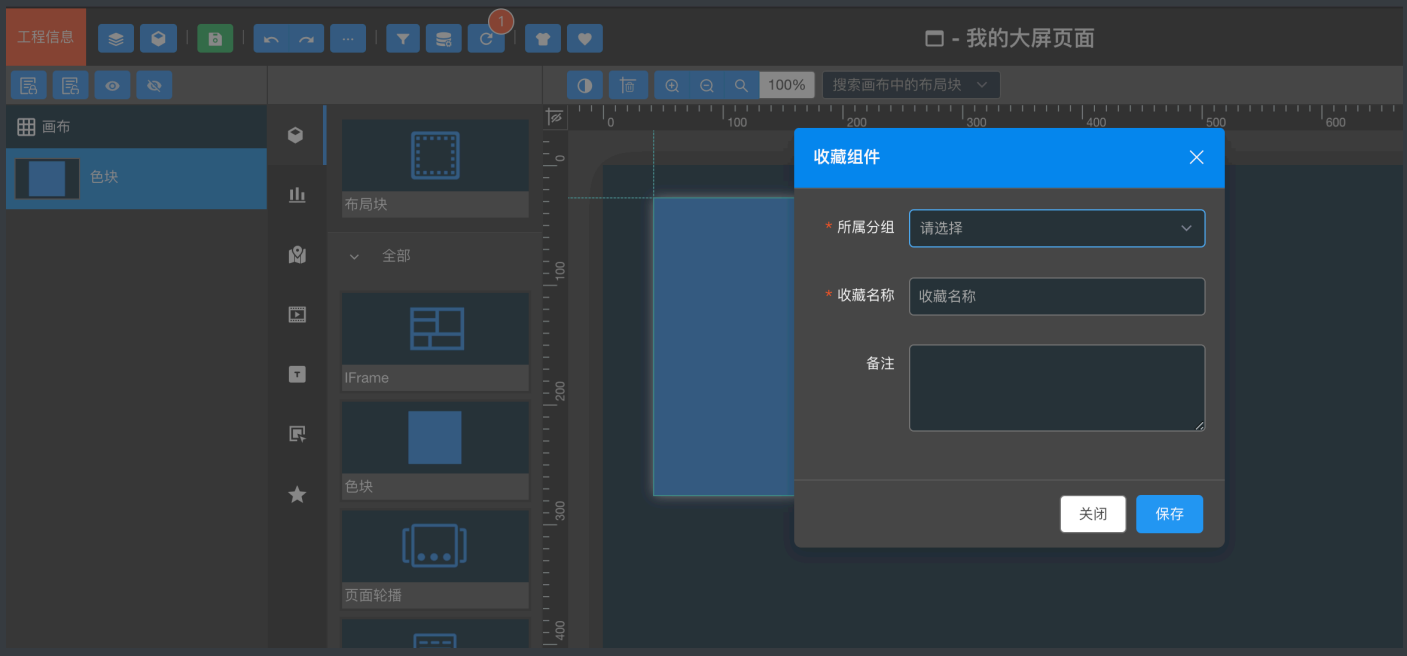
可以将编辑的页面保存为模板，当我们在创建页面时，可以选择自己保存的模板或系统模板来生成页面，用户保存的模板，在系统管理控制台【我的模板】中可以进行查看管理

组件收藏

可以将配置好的组件或组进行收藏，此后即可以在任意页面编辑中，从组件库【收藏组件】中拖入收藏的组件，在此之前，我们需要在系统管理控制台【收藏组件管理】中新建一个分组

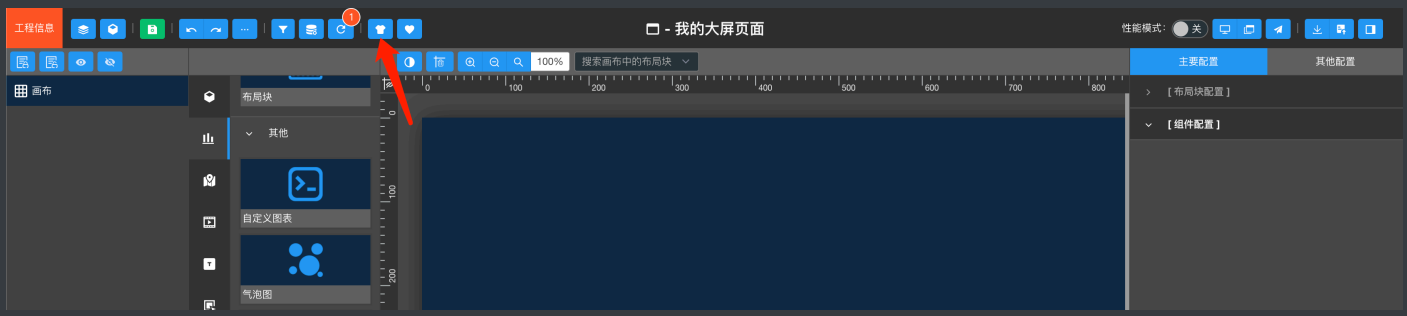
有了分组之后，在设计器中，就可以在特定组件或组上右键，对组件或组进行收藏操作

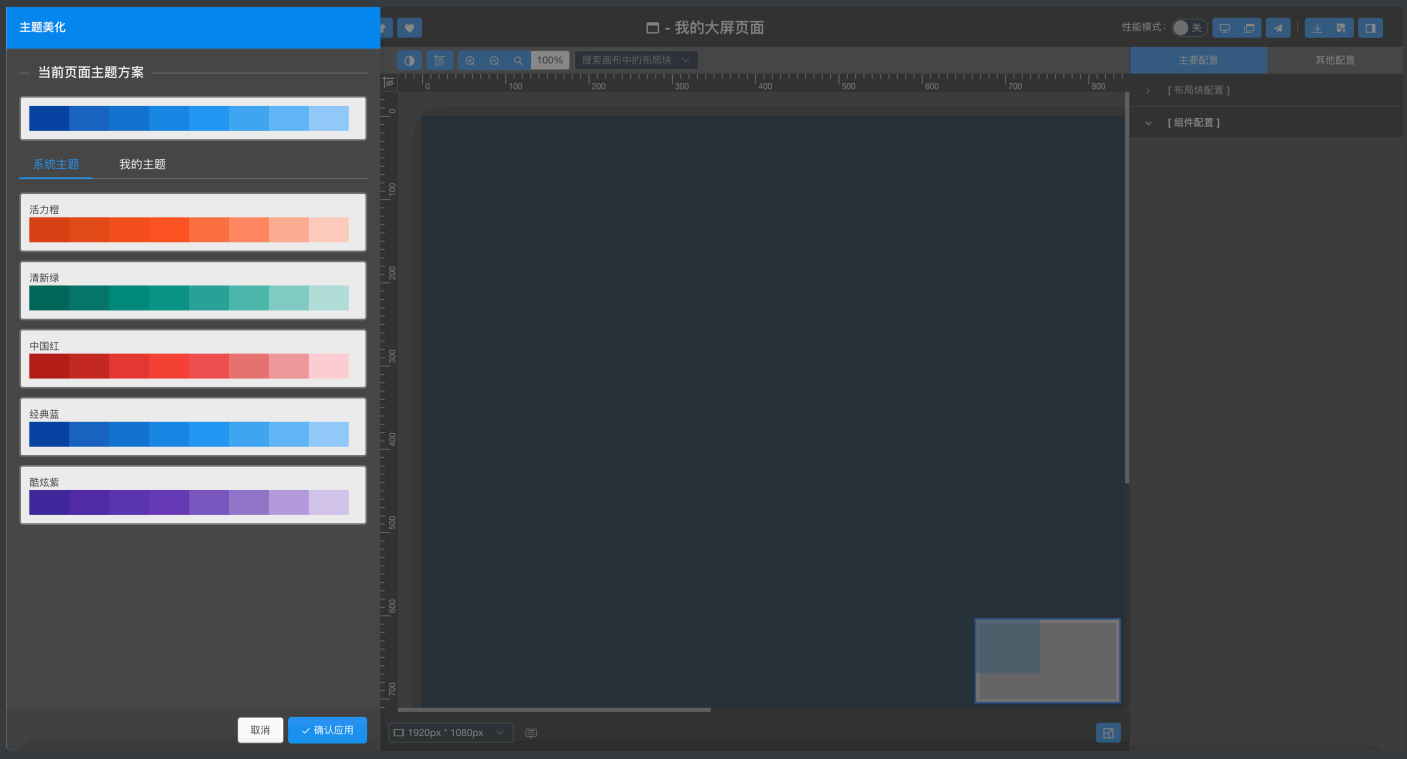




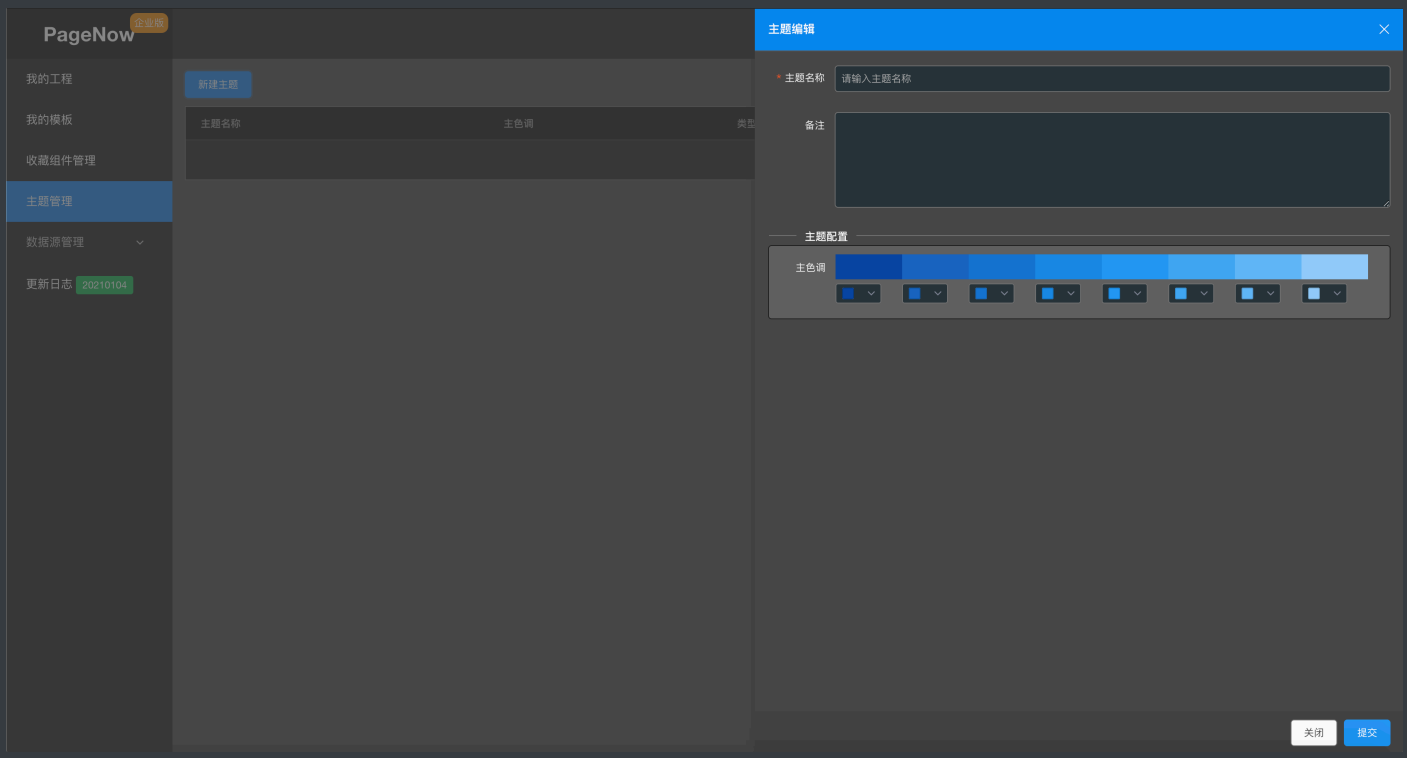
主题美化

当我们创建工程时，默认选择了一个主题方案，工程的主题方案会作为创建页面时默认使用的主题方案，我们可以在页面编辑时，切换当前编辑页面所使用的主题方案，主题方案会作用于大部分图表组件与部分功能组件



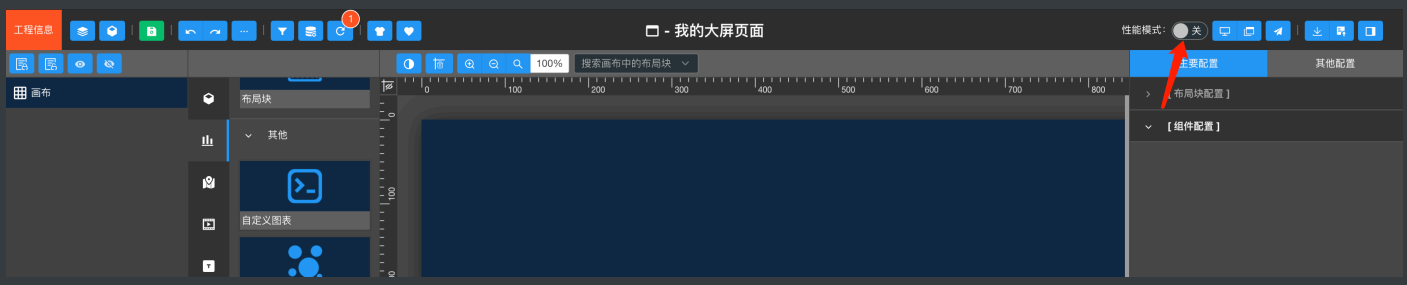


用户可以在系统管理控制台【主题管理】中，新增自定义的主题配色方案



性能模式

开启性能模式可以降低浏览器占用的内存资源，此模式下所有的代码编辑器将会切换为简约模式，简约模式下的代码编辑器将使用原始的textarea作为编辑框。

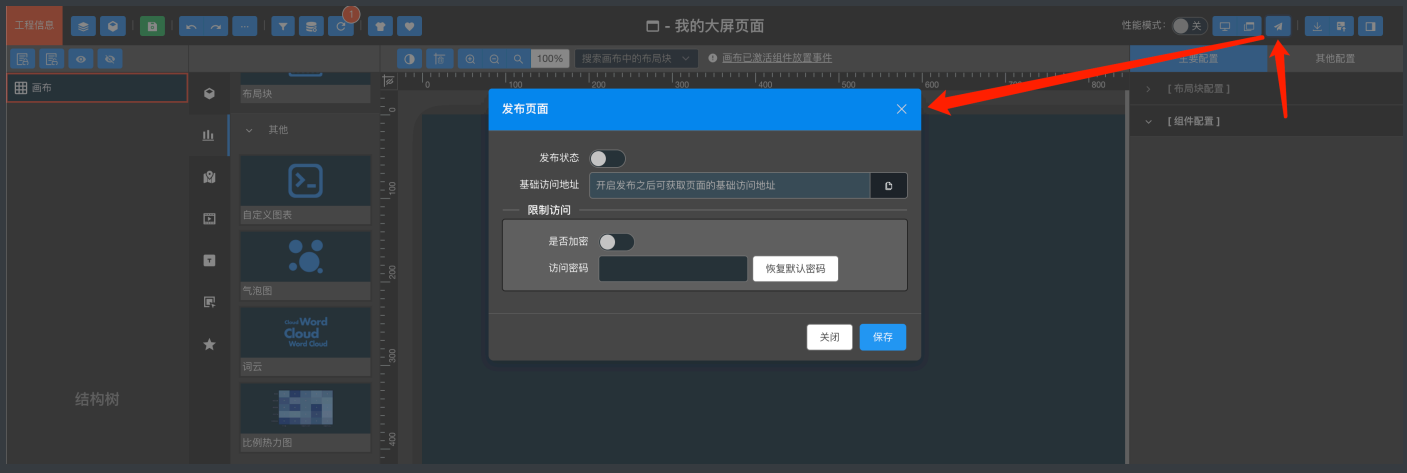


页面预览

预览分为当前窗口预览和新窗口预览，PageNow中每打开一次预览模式，只会预览打开预览窗口前的编辑状态，也就是说，如果您点击的是新窗口预览，那么当你回到设计器再次进行页面编辑时，需要把打开的预览页面关闭，重新再打开，浏览器F5刷新是无法获取到最新的编辑状态的。

页面发布

未发布的页面是无法在设计器之外访问的，要开放设计的数据大屏页面的公共访问，需要将页面进行发布，发布后的页面，系统会自动为其生成一个独立的URL地址，通过此URL即可访问设计好的数据大屏页面



PageNow中提供了页面的加密发布，设置了加密的页面，只有录入对应的加密密码，才可以正常对页面进行访问

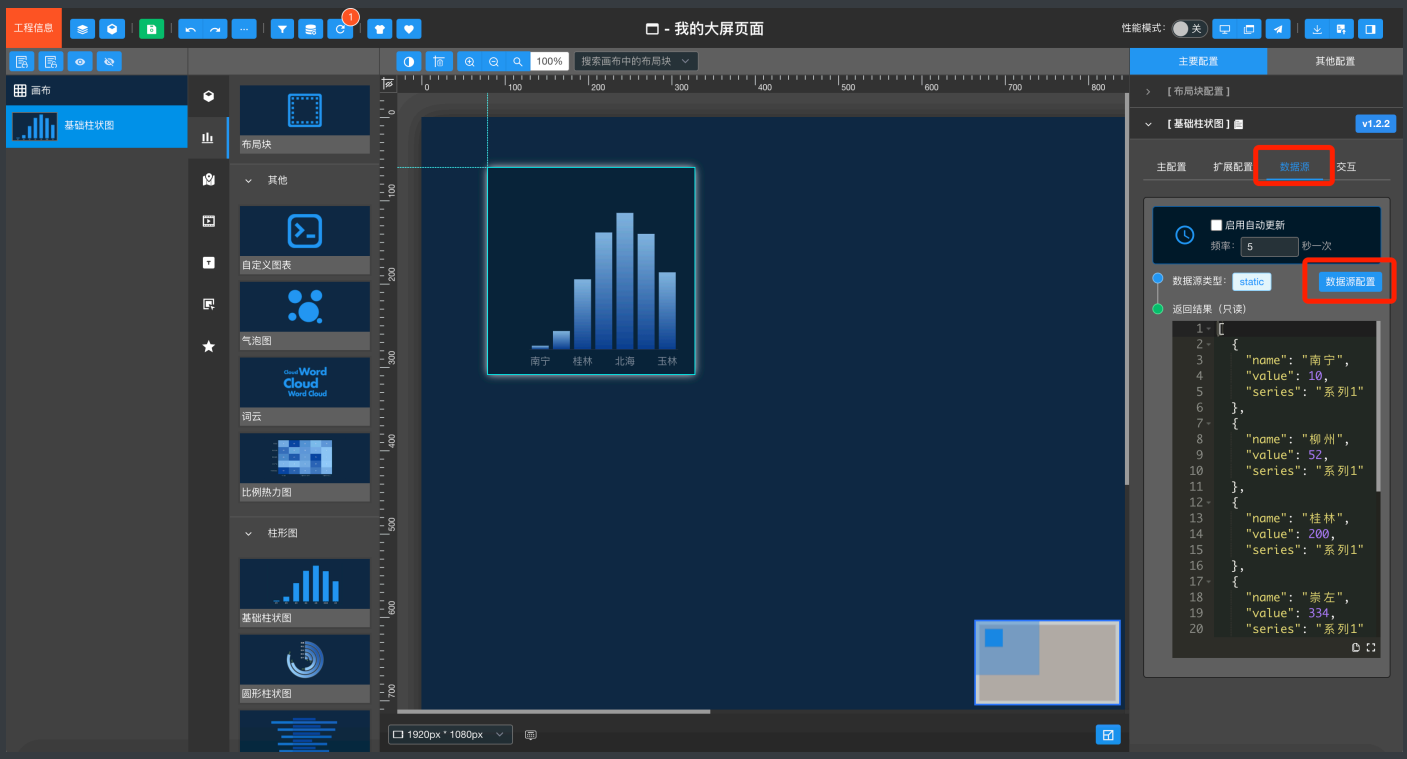
页面的导入与导出

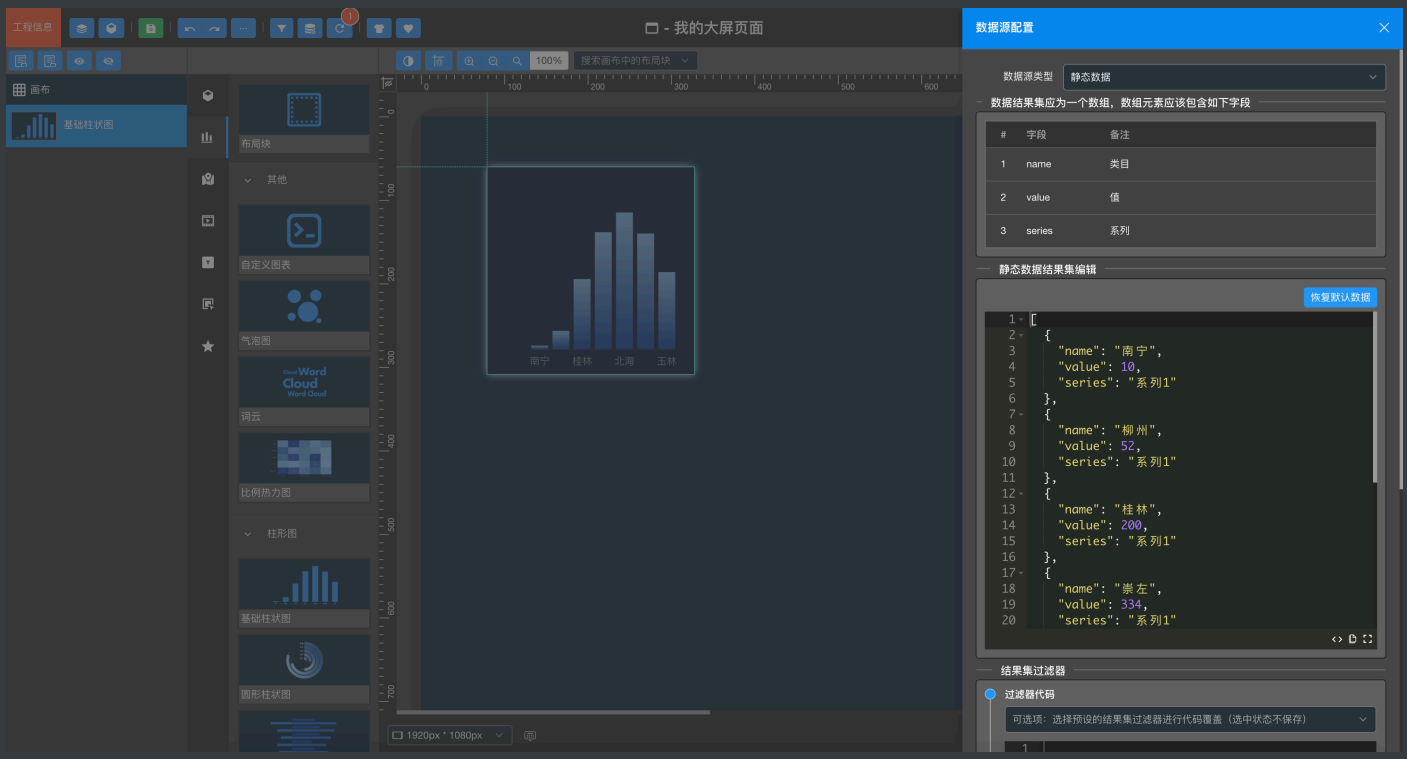
可以将编辑好的页面进行导出，导出的页面压缩包，可以在PageNow中进行导入

温馨提示：被导出的页面中，如果涉及使用了数据库数据源的组件，那么可能会在导入到其他地址下的PageNow中，造成组件数据源无法正常调用的情况，使用API接口的不受影响

数据源绑定

PageNow的组件库中，大部分组件都支持数据源的配置，在组件的配置区域【数据源】页签中可以进行数据源的配置





可以进行数据源配置的组件，默认均使用的是静态数据作为初始化数据源

结果集格式

PageNow中每种组件要求的数据源返回的数据都有其各自的格式，我们称其为结果集格式，不同组件的结果集格式各不相同，需要注意的是，每种组件的结果集格式是固定的，不论采用何种数据源类型，返回的结果集必须满足组件要求的结果集格式，才能被组件正常解析展示

温馨提示：如何判断组件要求的结果集格式是怎么样的？最简单的办法就是，看组件默认绑定的静态数据，按照静态数据的结果集格式即可

值得注意的是，所有的组件返回的结果集对象都必须为一个**数组**，数组中包含一个或多个对象存储相应的信息。

例如单行文本组件的结果集要求格式为：

```
[
  {
    "value": "我是动态数据源返回的数据"
  }
]
```

基本柱状图的结果集要求格式为：

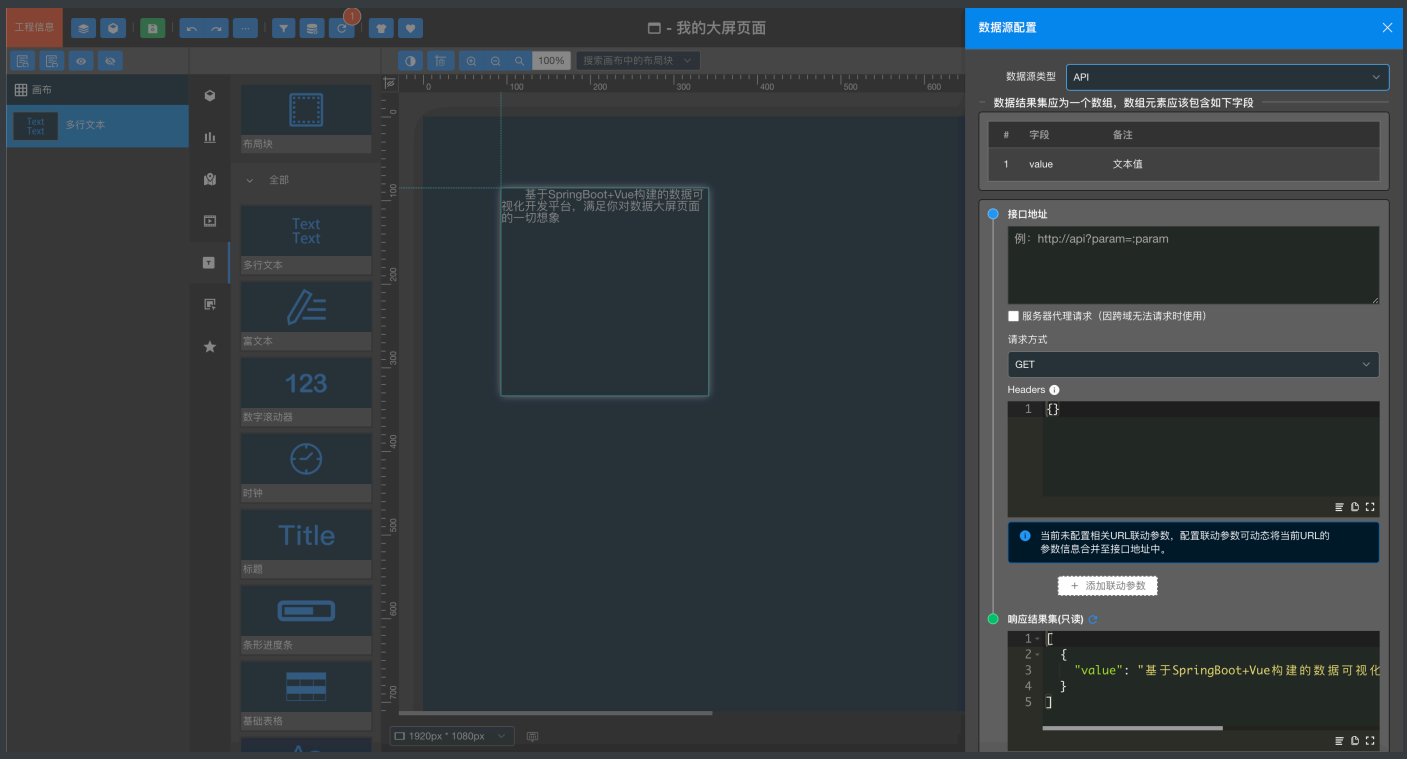
```
[
  {
    "name":"南宁",
    "value":10,
    "series":"系列1"
  },
  {
    "name":"柳州",
    "value":20,
    "series":"系列1"
  },
  ...
]
```

数据源类型

目前PageNow中支持四种数据源类型：静态数据、API接口、数据库、CSV文件，默认拖入画布的组件使用的都是静态数据

API接口

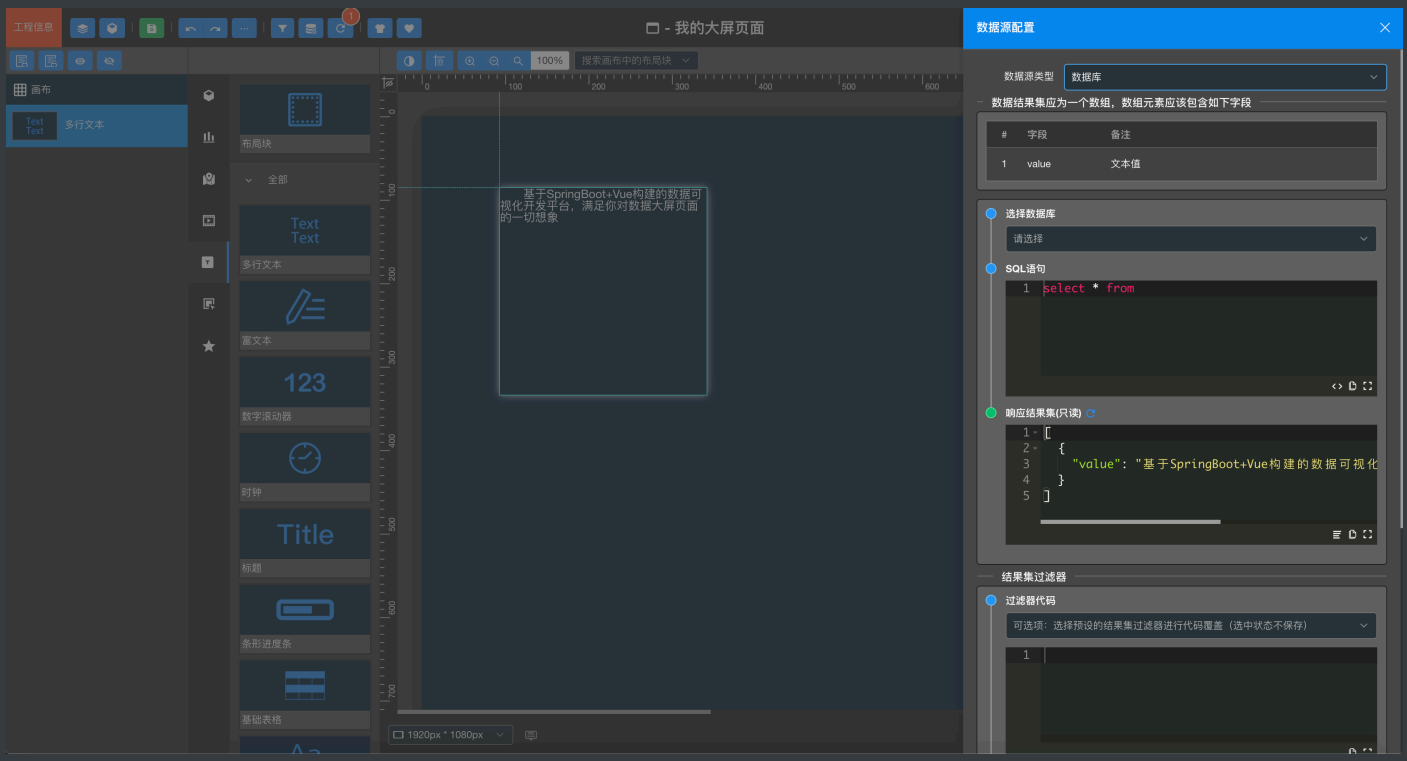
在数据源类型下拉框中选择【API】，即可切换为API接口数据源



- 接口地址：API的接口地址，注意必须带http://
- 服务器代理请求：使用服务器代理对接口进行数据请求，接口因跨域无法请求数据时可以勾选此选项
- 请求方式：可以切换接口的请求方式，支持GET和POST请求
- Headers：将会以请求的头信息一同提交
- 添加联动参数：使用此功能，我们可以获取当前页面的浏览器URL地址栏中的参数合并至接口地址中，例如：当我们打开某个数据大屏页面时的URL地址为：http://xxxxx/pagenow/57490378404?msg=hello，那么如果此页面下有组件使用API接口数据源并且添加了联动参数msg，那么这个组件在接口调用时，会自动添加一个URL参数msg=hello

数据库

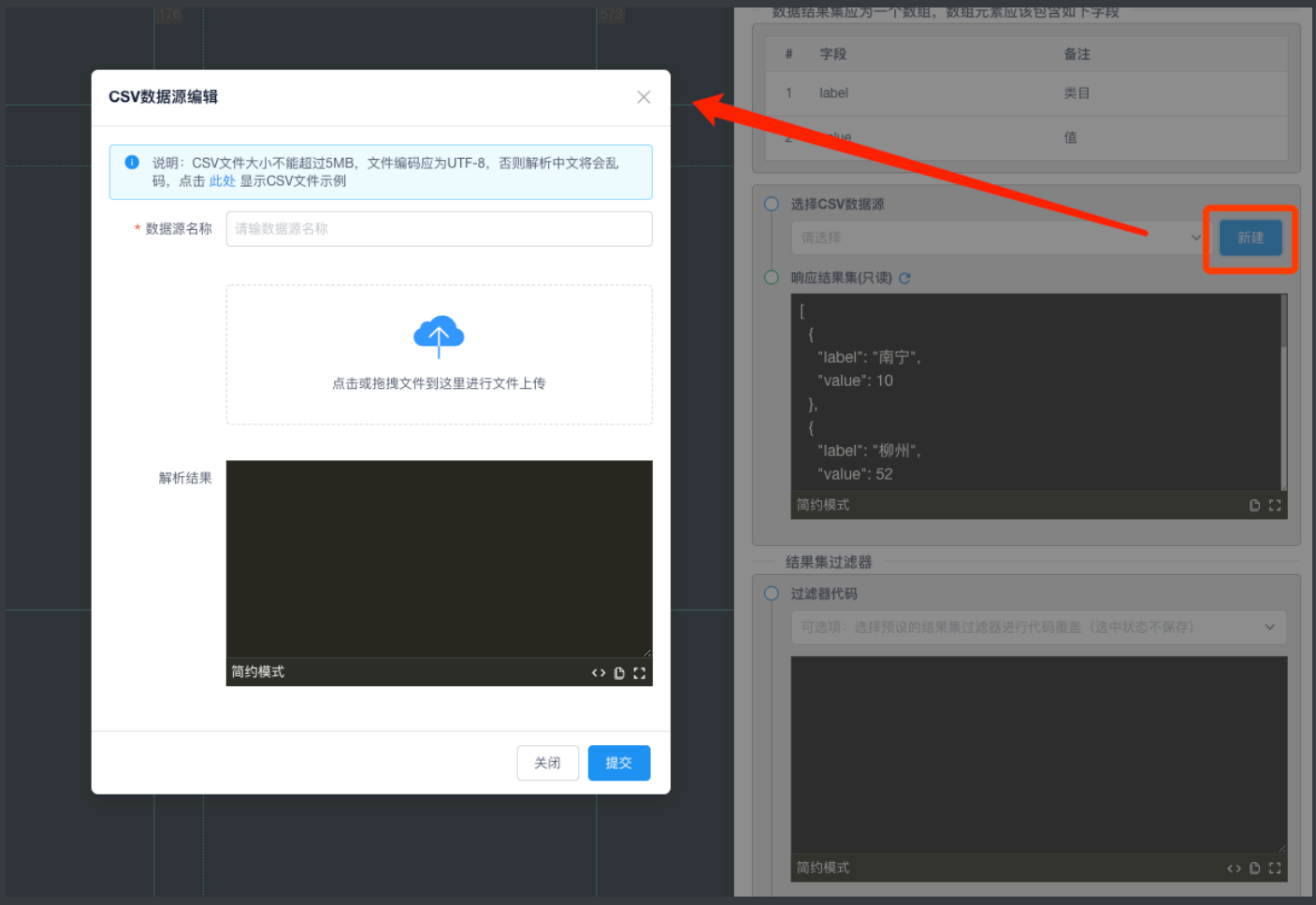
在数据源类型下拉框中选择数据库，即可切换为数据库数据源，此时我们可以通过选择相应的数据库并且编写SQL语句来获取结果集数据：



数据库数据源需要使用管理员账号在管理界面的数据源管理中进行添加才可在组件中进行选择，选择对应的数据库之后，编写相应的SQL语句即可。

CSV文件

在数据源类型下拉框中选择CSV文件，即可切换为CSV文件数据源，首先我们需要点击【新建】按钮上传CSV数据文件：



以基本柱状图为例，基本柱状图要求的结果集格式为：

```
[
  {
    "label":"南宁",
    "value":10
  },
  {
    "label":"柳州",
    "value":20
  }
  ...
]
```

那么我们的CSV文件中应该以如下图所示的格式进行编辑：

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q
1	label	value															
2	南宁	10															
3	柳州	20															
4																	
5																	
6																	

结果集过滤器

一般我们在使用动态数据源时，数据源返回的数据格式不一定都能与组件对应要求的结果集格式保持一致，这时候我们就可以通过结果集过滤器，实现结果集数据结构的转换、筛选和一些简单的计算，实现与组件要求的结果集格式的适配

在结果集过滤器代码编辑框中，可以编写我们的结果集过滤器代码，结果集过滤器代码的语法规则如下：

过滤器中通过【ds_resultObj】变量来存储数据源返回的默认结果集对象，我们只需要对ds_resultObj进行重构，然后重新赋值即可，下面举例来进行讲解：

单行文本组件：单行文本组件要求的结果集格式为

```
[
  {
    "value":"文本"
  }
]
```

如果我们数据源返回的结果集格式为

```
[
  {
    "text":"文本"
  }
]
```

显然是不符合组件要求的结果集格式的，那么我们的结果集过滤器中则可以如下编辑进行过滤：

```
let newData = [
  {
    value: ds_resultObj[0].text
  }
]
ds_resultObj = newData
```

此过滤器中，我们new了一个新的对象newData，然后在此对象中构建与单行文本组件要求的结果集格式一致的数据，此次我们通过ds_resultObj获取数据源返回的数据text赋值给value，然后将newData重新赋值给ds_resultObj即可。

交互

在PageNow中，组件之间的交互行为只能通过JS脚本编辑实现，这在一定程度上提高了编程人员编辑数据大屏的可扩展性，但也降低了非编程人员的使用成本

常用实例与变量

组件实例

每一个组件在编辑交互的JS脚本时，均可以通过this获取当前组件的Vue实例对象，在部分脚本编辑中，有可能会使用其他变量名来指向this，例如图表的【数据项点击时运行脚本】中，一般如果在系统中没有特殊标注，都是用的this来指向当前组件的Vue实例

this.chart：目前PageNow中，几乎所有的图表组件均使用的是Echarts组件实现，this.chart指向了当前图表组件的图表容器实例，通过此实例，我们就可以根据Echarts相关API对图表进行一些扩展操作，例如注册Tooltip轮播事件等等

全局对象

在PageNow中，我们提供了一些全局对象，帮助您在编写JS脚本实现交互效果时提供便利的函数调用，这些全局变量有：$、EventBus、PnUtil等，一般比较常用的是EventBus，这是实现组件数据交互的核心对象

初始化运行脚本

PageNow中，组件的生命周期与Vue的生命周期是保持一致的，组件的【初始化运行脚本】即是在组件的mounted生命周期钩子函数内执行的脚本，在此脚本内，我们可以对组件执行全局事件总线监听、获取Echarts图表容器实例并注册Tooltip轮播事件等操作

组件的配置属性

PageNow中，组件内通过compConfigData存储组件的配置属性，我们在编辑交互脚本时，经常会用到如下语法

```
this.component.compConfigData.ds_apiPath = ""
```

PageNow中如需要查看组件有那些配置信息，我们可以在组件的【初始化运行脚本】中输入如下代码：

```
console.log(this.component.compConfigData)
```

然后我们预览当前编辑页面，在浏览器控制台中就能够看到当前组件内的配置属性有哪些。

配置属性的作用：组件的配置区域中，配置的就是组件compConfigData中存储的数据，PageNow中，通过修改组件的compConfigData配置属性，来实现组件的个性化配置，因此，我们也可以在编辑交互脚本时，通过手动修改配置属性，然后通过组件重绘函数来实现自定义的组件配置修改，例如：组件API接口地址的修改

常用组件配置属性：

- ds_type：组件所使用的数据源类型
- ds_apiPath：组件API接口地址
- ds_apiMethod：API接口提交方式
- ds_sql：SQL查询语句

...

布局块的配置属性

与组件的配置属性一样，组件外层布局块也有自己的配置属性，通过layoutItemConfigData存储布局块的配置属性，通过以下语法，我们可以获取当前组件布局块的配置属性信息

```
this.layoutItem.layoutItemConfigData
```

常用布局块配置属性：

- className：自定义样式类名
- draggableEnabled：是否可拖拽
- resizableEnabled：是否可调整尺寸
- width：宽度
- height：高度
- left：左偏移
- top：右偏移
- backgroundColor：背景色
- padding：内边距
- borderWidth：边框宽度
- borderColor：边框颜色

...

与组件配置属性一样，我们也可以在组件的【初始化运行脚本】中，编写控制台打印语句来查看组件布局块有哪些可配置的属性

组件之间如何进行数据通信

在PageNow中，通过事件总线来实现组件之间的数据通信，一个通信过程包括发送事件总线事件和接收事件总线事件。

下面的代码为发送事件总线事件代码：

```
EventBus.send('', '')
```

send函数有两个参数：

- 第一个参数为事件名称：这个名称可以以英文形式任意命名，一般我们建议以发送的数据的作用来命名，例如我们发送的这个事件需要携带一个”hello，你好“的文本信息，那么我们这个事件的名称可以命名为helloMsg，当然怎么命名由你来决定，但是需要注意的是，同一个页面下，不同组件之间，尽量不要出现发送同名事件的情况。
- 第二个参数为事件的携带信息（或者叫载荷信息）：这个参数可以是任意形式的JS对象，可以是一个字符串，或者一个数组或者一个JSON对象都可

下面的代码为接收事件总线事件代码：

```
EventBus.receive([''], (params) => {

})
```

receive函数的参数为一个字符串数组，用于填入需要监听的事件名称，可以监听多个事件总线事件，也就是说，除了可以监听上面发送的helloMsg事件之外，如果我们在其他组件也发送了其他的事件，例如helloMsg2，那么我们只需要在数组中再新增一个字符填入helloMsg2即可。

在回调函数 (params) => {} 中，可以通过params来获取接收到的所携带过来的信息（载荷信息）。

注意：在PageNow中，一个事件发送与接收的过程是点对点触发的，也就是说，发送事件总线事件与接收事件总线事件，是一个主动触发的过程，只有通过某种动作触发发送事件，才可以被接收总线事件的回调中进行处理，举个简单的例子说明：

当我们页面中有两个按钮组件按钮1和按钮2，这两个按钮组件分别发送自己的事件helloMsg和helloMsg2，这时候我们在一个多行文本组件中监听这两个事件，当我们点击按钮1时，多行文本组件中的receive函数的回调函数中，params只能获取到helloMsg发送出来的信息，不能获取到helloMsg2发送出来的信息。

初始化运行脚本中特有变量的使用

在PageNow中，接收事件总线事件一般都是在组件的初始化运行脚本中去编写的，在组件的初始化运行脚本中有三个特有的变量：

- original_ds_apiPath：当组件使用的是API数据源时，此变量用于存储原始的API接口地址
- original_ds_sql：当组件使用的是数据库数据源时，此变量用于存储原始的SQL语句
- original_ds_resublObj：此变量用于存储原始的结果集数据

findCompVmById函数的使用

在PageNow中，除了普通功能组件和Echarts图表组件之外，在普通功能组件下还有一种特别的组件，叫交互组件，例如按钮、下拉选择器、输入框组件、Tab列表、日期选择等，这些组件中，除了按钮之外，都可以理解为是一种表单组件，这种交互组件或叫表单组件，是实现多条件数据源查询的重要方式。

例如这么一种场景，页面有一个日期选择器和一个输入框以及一个基本柱状图组件，柱状图通过API接口来获取数据，接口地址需要同时获取一个日期和一个文本来作为参数。

首先我们需要在日期选择器组件中的【选中值变化时触发】脚本中发送一个事件总线事件：

```
EventBus.send('dateValue', value)
```

同理，输入框组件的【按下回车键时触发】脚本中也发送一个事件总线事件：

```
EventBus.send('textValue', value)
```

然后我们需要在基本柱状图中的初始化运行脚本中去监听这两个事件并且使用【findCompVmById】函数去获取日期选择器组件和输入框组件的数据值，通过这两个数据值，来重构柱状图组件的API接口地址并重绘图表，下面是示例代码：

```
EventBus.receive(['dateValue', 'textValue'], (params) => {
  let date =
this.findCompVmById('f0187e892dd864ea93d2ba21556e61bf').value;
  let text =
this.findCompVmById('7d03f973a9e575e2d574eb1f368eb386').value;
  this.component.compConfigData.ds_apiPath = original_ds_apiPath + '?
date=' + date + '&text' + text;
  this.redrawComp()
})
```

以上这段代码中有几个注意点：

- 在receive的回调函数中，可以发现，我们并没有使用params来获取接收到的事件信息，因为在组件之间如何进行数据通信章节中我们已经提到过，事件的发送和接收是点对点触发的，如果我们通过params来获取事件参数，那么会存在只能获取到一个表单组件的数据值的情况，所以就需要使用findCompVmById函数来获取日期组件和输入框组件的Vue实例，然后通过.value来获取其数据值。
- findCompVmById函数需要传入一个id参数，这个id参数通过复制对应的组件ID粘贴进来，例如下面截图中，分别为日期选择组件的ID和输入框组件的ID

- 柱状图组件的数据源类型必须选择为API接口，original_ds_apiPath变量中存储的就是原始填入的API接口地址，例如我们柱状图默认调用的接口是http://localhost:8090/getBarData，那么original_ds_apiPath存储的就是这个地址，然后我们通过JS语法给这个地址添加了两个URI参数date和text。
- 重设柱状图组件的API接口地址时候，我们还需要调用【this.redrawComp()】来对图表进行重绘。