BIX5

위젯 사용자 정의 메뉴 작성 가이드

정품을 구입하신 고객에게는 기술상담 및 지원을 제공합니다.

(주)빅스소프트

서울시 영등포구 영신로 220, 610 호 (영등포동 8 가, KnK 디지털타워)

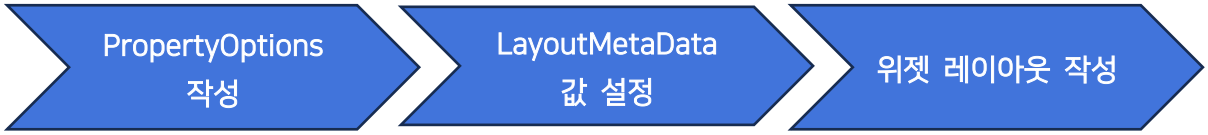
Tel. 02-2655-9784, bix5cs@bixsoft.net

BIXSOFT 빅스소프트

목차

1. 시작하며

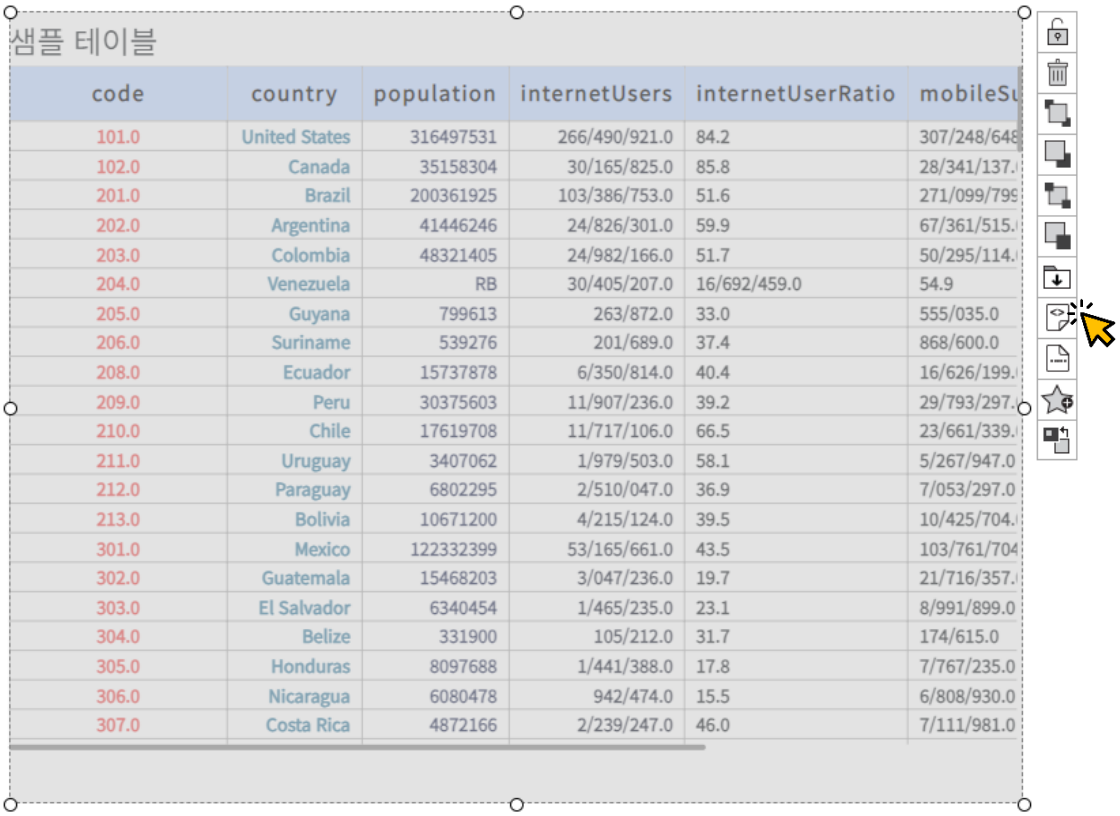
본 가이드는 대시보드 편집 화면에서 위젯 상세 메뉴의 커스터마이징을 돕기 위한 가이드입니다. 이를 통해 위젯의 상세 메뉴에 위젯을 조작할 수 있는 메뉴를 사용자가 만들 수 있습니다. 메뉴를 커스터마이징 하고 적용하는 흐름은 다음과 같습니다.



PropertyOptions 를 작성하여 메뉴에 대한 구조를 작성하고 layoutMetaData 의 값을 설정하고 위젯 레이아웃에 handlebars 문법을 사용하여 작성하게 됩니다. 제품 내부에는 handlebars 라는 자바스크립트 템플릿 엔진을 내장하고 있습니다. 이를 이용하여 위젯 레이아웃과 layoutMetaData 를 결합하여 html 을 동적으로 생성하게 됩니다.

1.1. 준비

위의 절차를 실행하려면 위젯의 코드 에디터로 접근할 권한이 있어야 합니다. 대시보드 편집 화면에서 임의의 위젯을 클릭하시어 코드 버튼이 있는지 확인하십시오. 버튼이 보이지 않으면 편집 권한이 없는 것이니 편집을 하려면 관리자에게 문의하십시오.



코드 에디터에서 컴포넌트 옵션 탭으로 이동합니다. 이곳에서 propertyOptions 와 layoutMetaData 를 편집하게 됩니다. 작성 방법은 2. Control 별 메뉴 작성 방법에서 설명하고 있습니다.

레이아웃 탭에서 레이아웃을 편집하게 됩니다. 작성 방법은 2. Control 별 메뉴 작성 방법에서 설명하고 있습니다.

1.2. 컨트롤 소개

메뉴는 크게 두가지로 이루어져 있습니다. 사용자로부터 입력을 받을 control 과 control 을 식별하거나 사용자에게 설명할 label 입니다. Control 의 종류는 다음과 같습니다.

Control 타입	외형	설명
TextInput	샘플 테이블	문자열을 입력 받는 컨트롤
ColorSelector		색상(hex)값을 입력받는 컨트롤
NumericStepper	1	숫자를 입력받는 컨트롤
SlideButton		Boolean(true/false)를 입력받는 컨트롤
ComboBox	중앙	하나의 옵션을 선택할 수 있는 컨트롤
ColumnComboBox	지역	데이터소스의 컬럼을 선택할 수 있는 컨트롤

2. Control 별 메뉴 작성 방법

2.1. TextInput

설명을 위해 html 위젯을 생성하였다는 가정하에 진행하겠습니다.

위젯의 코드 에디터의 컴포넌트 옵션으로 이동합니다.

레이아웃 스크립트 **컴포넌트 옵션** 사용자 데이터

```
{
  "propertyOptions": [],
  "layoutMetaData": {},
  "isMultiView": false,
  "multiViewRow": 2,
  "multiViewColumn": 2,
  "multiViewSubCategoryField": {},
  "multiViewFillType": null,
  "marginLeft": 0,
  "marginTop": 0,
  "marginRight": 0,
```

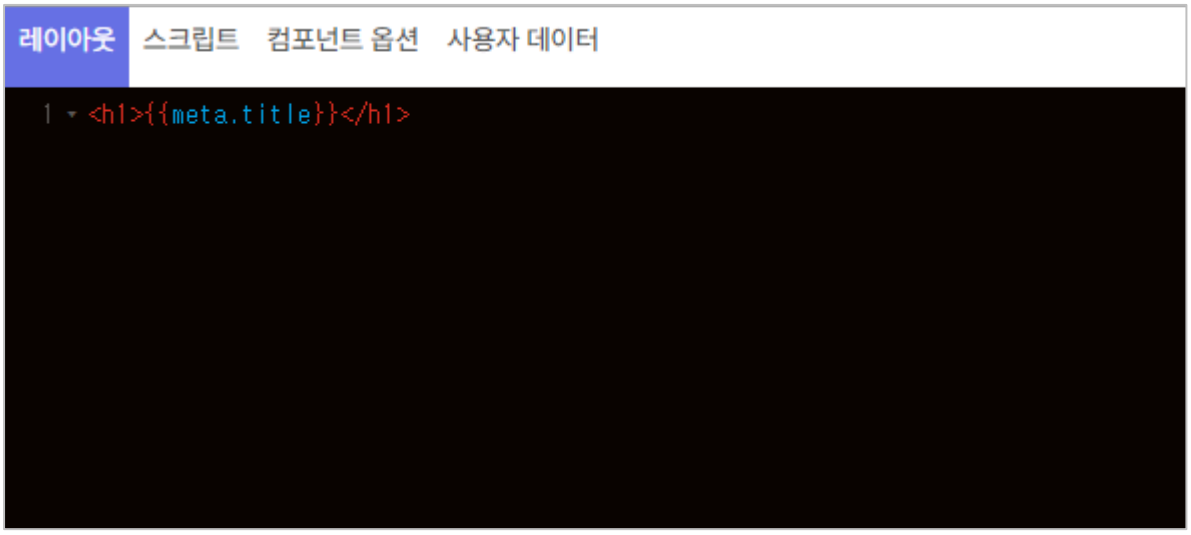
propertyOptions 의 항목을 작성합니다.

레이아웃 스크립트 **컴포넌트 옵션** 사용자 데이터

```
{
  "propertyOptions": [
    {
      "label": "제목",
      "control": "TextInput",
      "attribute": "title"
    }
  ],
  "layoutMetaData": {},
  "isMultiView": false,
  "multiViewRow": 2,
```

label	Control 을 설명하거나 식별하기 위한 라벨. 사용자가 임의로 정합니다.
control	사용자로부터 입력 받을 컨트롤. 유효한 값을 입력합니다. 유효한 값은 컨트롤 소개에서 확인하십시오.
attribute	layoutMetaData 의 key. 사용자가 임의로 정합니다.

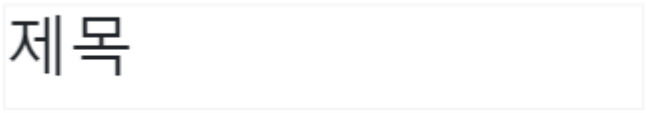
레이아웃으로 이동하여 레이아웃을 작성합니다. {{}}는 handlebars 의 문법이고 meta.title 로 layoutMetaData 의 title 의 값을 가져옵니다.



에디터의 우측 상단의 적용 버튼을 눌러 변경한 사항을 적용합니다.

우측 위젯 상세 메뉴에서 값을 변경합니다.

코드 에디터의 우측 상단 '적용'버튼을 클릭하면 다음과 같은 결과를 확인할 수 있습니다.



2.2. ColorSelector

설명을 위해 html 위젯을 생성하였다는 가정하에 진행하겠습니다.

위젯의 코드 에디터의 컴포넌트 옵션으로 이동합니다.

레이아웃 스크립트 **컴포넌트 옵션** 사용자 데이터

```
{
  "propertyOptions": [],
  "layoutMetaData": {},
  "isMultiView": false,
  "multiViewRow": 2,
  "multiViewColumn": 2,
  "multiViewSubCategoryField": {},
  "multiViewFillType": null,
  "marginLeft": 0,
  "marginTop": 0,
  "marginRight": 0,
```

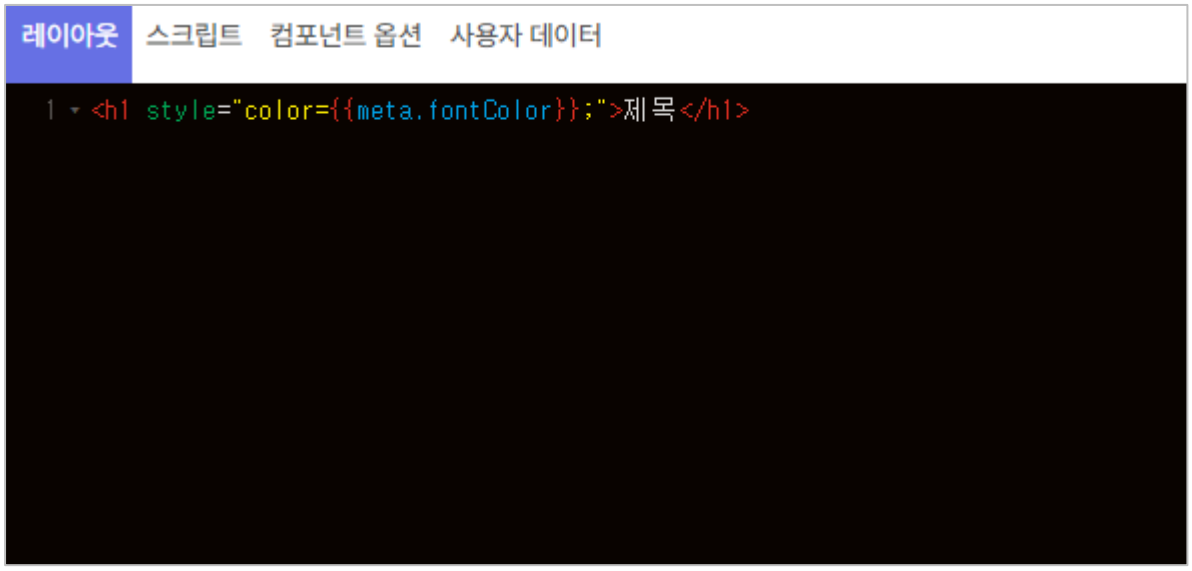
propertyOptions 의 항목을 작성합니다.

레이아웃 스크립트 **컴포넌트 옵션** 사용자 데이터

```
{
  "propertyOptions": [
    {
      "label": "글자 색상",
      "control": "ColorSelector",
      "attribute": "fontColor"
    }
  ],
  "layoutMetaData": {},
  "isMultiView": false,
  "multiViewRow": 2,
  "multiViewColumn": 2,
```

label	Control 을 설명하거나 식별하기 위한 라벨. 사용자가 임의로 정합니다.
control	사용자로부터 입력 받을 컨트롤. 유효한 값을 입력합니다. 유효한 값은 컨트롤 소개에서 확인하십시오.
attribute	layoutMetaData 의 key. 사용자가 임의로 정합니다.

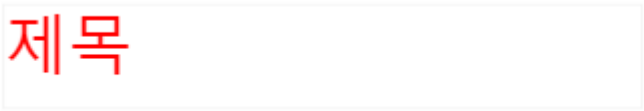
레이아웃으로 이동하여 레이아웃을 작성합니다. {{}}는 handlebars 의 문법이고 meta.fontColor 로 layoutMetaData 의 fontColor 의 값을 가져옵니다.



에디터의 우측 상단의 적용 버튼을 눌러 변경한 사항을 적용합니다.

우측 위젯 상세 메뉴에서 값을 변경합니다.

코드 에디터의 우측 상단 '적용'버튼을 클릭하면 다음과 같은 결과를 확인할 수 있습니다.



2.3. NumericStepper

설명을 위해 html 위젯을 생성하였다는 가정하에 진행하겠습니다.

위젯의 코드 에디터의 컴포넌트 옵션으로 이동합니다.

레이아웃 스크립트 **컴포넌트 옵션** 사용자 데이터

```
{
  "propertyOptions": [],
  "layoutMetaData": {},
  "isMultiView": false,
  "multiViewRow": 2,
  "multiViewColumn": 2,
  "multiViewSubCategoryField": {},
  "multiViewFillType": null,
  "marginLeft": 0,
  "marginTop": 0,
  "marginRight": 0,
```

propertyOptions 의 항목을 작성합니다.

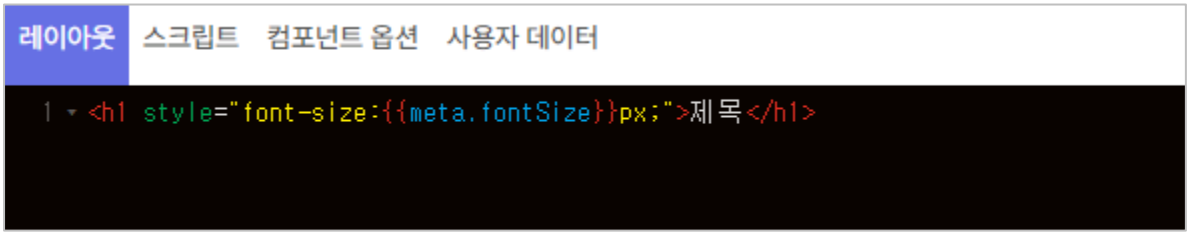
레이아웃 스크립트 **컴포넌트 옵션** 사용자 데이터

```
{
  "propertyOptions": [
    {
      "label": "글자 크기",
      "control": "NumericStepper",
      "minimum": 0,
      "maximum": 1,
      "stepSize": 0.1,
      "attribute": "fontSize"
    },
  ],
  "layoutMetaData": {},
  "isMultiView": false,
  "multiViewRow": 2,
```

label	Control 을 설명하거나 식별하기 위한 라벨. 사용자가 임의로 정합니다.
control	사용자로부터 입력 받을 컨트롤. 유효한 값을 입력합니다. 유효한 값은 컨트롤 소개에서 확인하십시오.
attribute	layoutMetaData 의 key. 사용자가 임의로 정합니다.

minimum	NumericStepper 의 minimum. 0 이면 0 미만으로 설정할 수 없습니다. 설정하지 않은 경우 기본값은 0 입니다.
maximum	NumericStepper 의 maximum. 100 이면 100 초과로 설정할 수 없습니다. 설정하지 않은 경우 기본값은 10 입니다.
stepSize	값 변경 시 변경되는 크기. 1 이면 1,2,3,4… 0.1 이면 0.1, 0.2, 0.3 으로 증가/감소 합니다. 설정하지 않은 경우 기본값은 1 입니다.

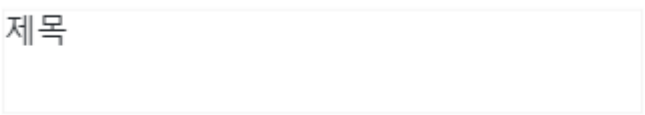
레이아웃으로 이동하여 레이아웃을 작성합니다. {{}}는 handlebars 의 문법이고 meta.fontSize 로 layoutMetaData 의 fontSize 의 값을 가져옵니다.



에디터의 우측 상단의 적용 버튼을 눌러 변경한 사항을 적용합니다.

우측 위젯 상세 메뉴에서 값을 변경합니다.

코드 에디터의 우측 상단 '적용'버튼을 클릭하면 다음과 같은 결과를 확인할 수 있습니다.



2.4. SlideButton

설명을 위해 html 위젯을 생성하였다는 가정하에 진행하겠습니다.

위젯의 코드 에디터의 컴포넌트 옵션으로 이동합니다.

레이아웃 스크립트 **컴포넌트 옵션** 사용자 데이터

```
{
  "propertyOptions": [],
  "layoutMetaData": {},
  "isMultiView": false,
  "multiViewRow": 2,
  "multiViewColumn": 2,
  "multiViewSubCategoryField": {},
  "multiViewFillType": null,
  "marginLeft": 0,
  "marginTop": 0,
  "marginRight": 0,
```

propertyOptions 의 항목을 작성합니다.

레이아웃 스크립트 **컴포넌트 옵션** 사용자 데이터

```
{
  "propertyOptions": [
    {
      "label": "보이기/숨기기",
      "control": "SlideButton",
      "attribute": "isVisible"
    }
  ],
  "layoutMetaData": {},
  "isMultiView": false,
  "multiViewRow": 2,
```

label	Control 을 설명하거나 식별하기 위한 라벨. 사용자가 임의로 정합니다.
control	사용자로부터 입력 받을 컨트롤. 유효한 값을 입력합니다. 유효한 값은 컨트롤 소개에서 확인하십시오.
attribute	layoutMetaData 의 key. 사용자가 임의로 정합니다.

레이아웃으로 이동하여 레이아웃을 작성합니다. {{}}는 handlebars 의 문법이고 meta.isVisible 로 layoutMetaData 의 isVisible 의 값을 가져옵니다.

에디터의 우측 상단의 

버튼을 눌러 변경한 사항을 적용합니다.

코드 에디터의 우측 상단 '적용'버튼을 클릭하면 다음과 같은 결과를 확인할 수 있습니다.

2.5. ComboBox

설명을 위해 html 위젯을 생성하였다는 가정하에 진행하겠습니다.

위젯의 코드 에디터의 컴포넌트 옵션으로 이동합니다.

레이아웃 스크립트 **컴포넌트 옵션** 사용자 데이터

```
{
  "propertyOptions": [],
  "layoutMetaData": {},
  "isMultiView": false,
  "multiViewRow": 2,
  "multiViewColumn": 2,
  "multiViewSubCategoryField": {},
  "multiViewFillType": null,
  "marginLeft": 0,
  "marginTop": 0,
  "marginRight": 0,
```

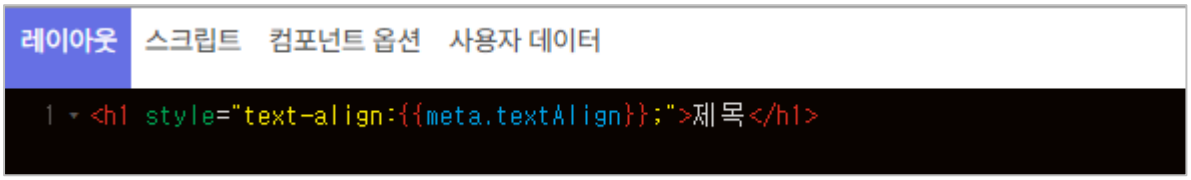
propertyOptions 의 항목을 작성합니다.

레이아웃 스크립트 **컴포넌트 옵션** 사용자 데이터

```
{
  "propertyOptions": [
    {
      "label": "텍스트 정렬",
      "control": "ComboBox",
      "attribute": "textAlign",
      "options": [
        {
          "label": "왼쪽",
          "value": "left"
        },
        {
          "label": "중앙",
          "value": "center"
        },
        {
          "label": "오른쪽",
          "value": "right"
        }
      ]
    }
  ],
```

label	Control 을 설명하거나 식별하기 위한 라벨. 사용자가 임의로 정합니다.
control	사용자로부터 입력 받을 컨트롤. 유효한 값을 입력합니다. 유효한 값은 컨트롤 소개에서 확인하십시오.
attribute	layoutMetaData 의 key. 사용자가 임의로 정합니다.
options	콤보박스의 목록 입니다. 배열 형태이며 label, value 를 작성합니다. Label 은 사용자에게 보이는 라벨이고 value 는 해당 옵션의 값입니다.

레이아웃으로 이동하여 레이아웃을 작성합니다. {{}}는 handlebars 의 문법이고 meta.textAlign 으로 layoutMetaData 의 textAlign 의 값을 가져옵니다.



에디터의 우측 상단의 적용 버튼을 눌러 변경한 사항을 적용합니다.

우측 위젯 상세 메뉴에서 값을 변경합니다.

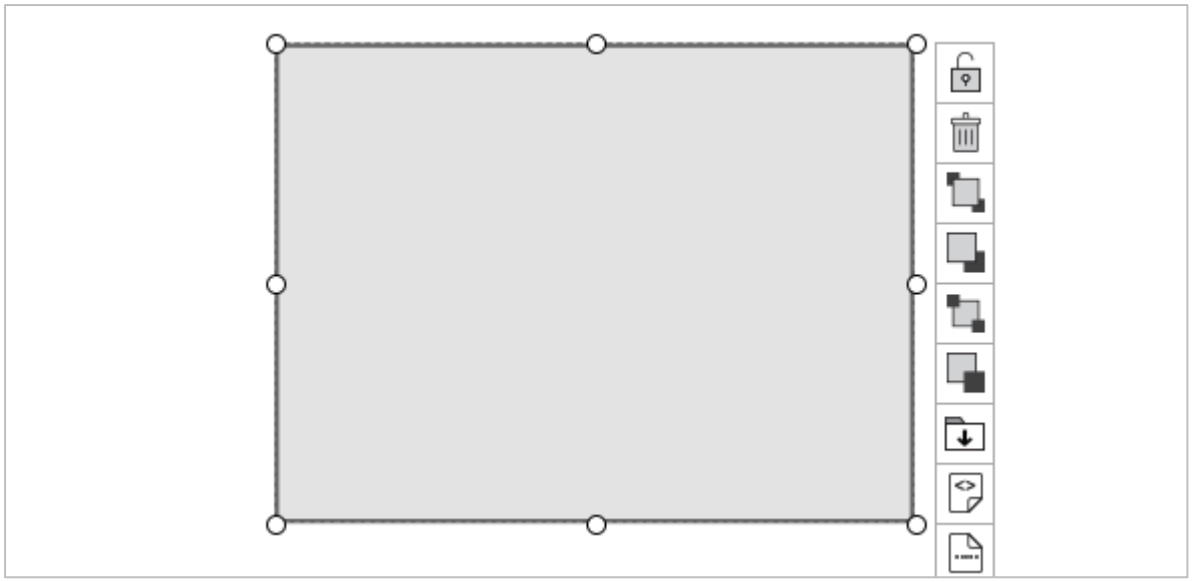
코드 에디터의 우측 상단 '적용'버튼을 클릭하면 다음과 같은 결과를 확인할 수 있습니다.

2.6. ColumnComboBox

설명을 위해 html 위젯을 생성하였다는 가정하에 진행하겠습니다.

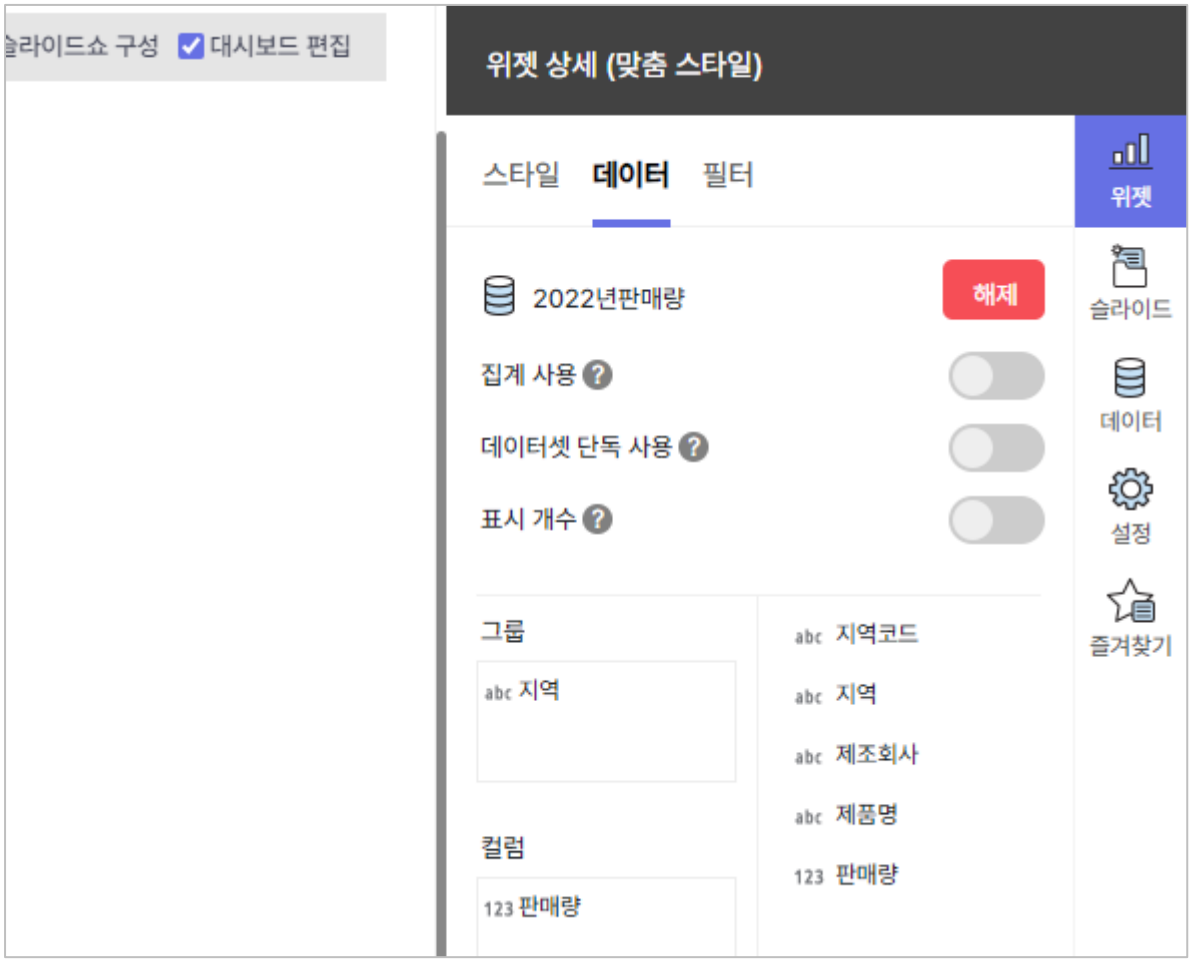
ColumnComboBox 를 사용하기 위해서 위젯에 데이터를 설정해야 합니다.
위젯을 선택한 후 우측 위젯 상세 메뉴의 '데이터'탭으로 이동합니다. '선택'버튼을 눌러 데이터소스를 선택합니다.

위젯 선택



데이터 소스 선택 전

데이터 소스 선택 후



데이터 형태

데이터 보기

날짜	지역코드	지역	제조회사	제품명	판매량	매출액
2022-01-01	4200000000	강원	삼성	갤럭시 S23	20	20000000
2022-01-01	4200000000	강원	삼성	갤럭시 Ultra	19	19000000
2022-01-01	4200000000	강원	삼성	갤럭시 Z Fold 4	36	36000000
2022-01-01	4200000000	강원	애플	iPhone 14	23	23000000
2022-01-01	4200000000	강원	애플	iPhone 14 Plus	43	43000000
2022-01-01	4200000000	강원	애플	iPhone 14 Pro	10	10000000
2022-01-01	4100000000	경기	삼성	갤럭시 S23	95	95000000
2022-01-01	4100000000	경기	삼성	갤럭시 Ultra	103	103000000
2022-01-01	4100000000	경기	삼성	갤럭시 Z Fold 4	127	127000000

위젯의 코드 에디터의 컴포넌트 옵션으로 이동합니다.

레이아웃 스크립트 **컴포넌트 옵션** 사용자 데이터

```
{
  "propertyOptions": [],
  "layoutMetaData": {},
  "isMultiView": false,
  "multiViewRow": 2,
  "multiViewColumn": 2,
  "multiViewSubCategoryField": {},
  "multiViewFillType": null,
  "marginLeft": 0,
  "marginTop": 0,
  "marginRight": 0,
```

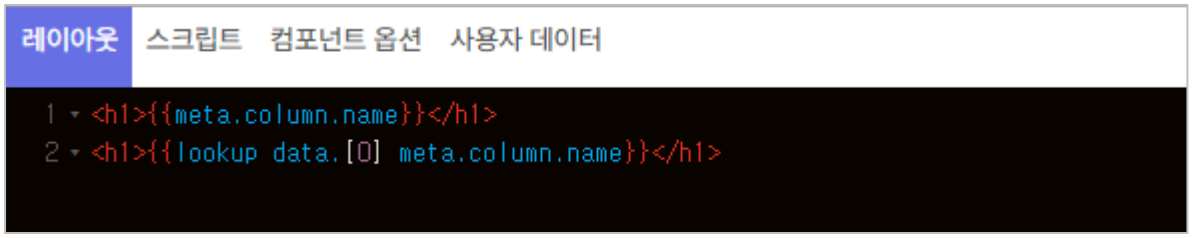
propertyOptions 의 항목을 작성합니다.

레이아웃 스크립트 **컴포넌트 옵션** 사용자 데이터

```
{
  "propertyOptions": [
    {
      "label": "컬럼",
      "control": "ColumnComboBox",
      "attribute": "column"
    }
  ],
  "layoutMetaData": {
    "columns": [
      {
        "id": 603,
        "name": "지역",
        "label": "지역",
```

label	Control 을 설명하거나 식별하기 위한 라벨. 사용자가 임의로 정합니다.
control	사용자로부터 입력 받을 컨트롤. 유효한 값을 입력합니다. 유효한 값은 컨트롤 소개에서 확인하십시오.
attribute	layoutMetaData 의 key. 사용자가 임의로 정합니다.

레이아웃으로 이동하여 레이아웃을 작성합니다.



lookup 은 특정 객체 내에서 속성(키)을 찾아 그 값을 반환하는 handlebars 의 내장 함수입니다.
{{lookup 객체 '키'}}로 사용합니다. 예시에서 사용하는 data.[0]은 위젯에 바인딩 된 데이터 소스의 0 번째 데이터를 가리킵니다. meta.column.name 은 사용자가 선택한 컬럼의 필드명입니다.

에디터의 우측 상단의 

버튼을 눌러 변경한 사항을 적용합니다.
적용 후 위젯 메뉴에서 컬럼을 설정합니다.

컬럼을 설정하게 되면 layoutMetaData 에 다음과 같이 설정됩니다.

※ 해당 값을 임의로 변경하지 말아주십시오

```
"layoutMetaData": {
  "column": {
    "id": 606,
    "name": "판매량",
    "label": "판매량",
    "dataType": "numeric",
    "memo": "판매량",
    "aggregate": "",
    "alias": "판매량",
    "category": "measure"
  },
```

설정 후 다음과 같은 결과를 확인할 수 있습니다.

2.7. ResourceSelector

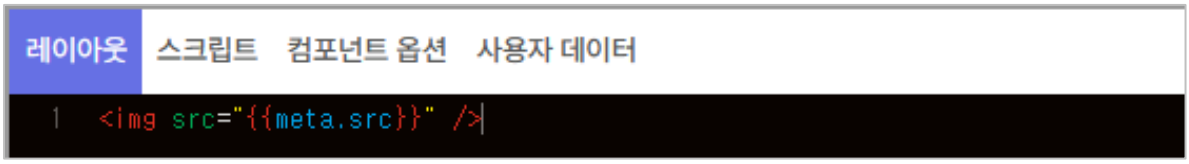
설명을 위해 html 위젯을 생성하였다는 가정하에 진행하겠습니다.

위젯의 코드 에디터의 컴포넌트 옵션으로 이동합니다.

propertyOptions 의 항목을 작성합니다.

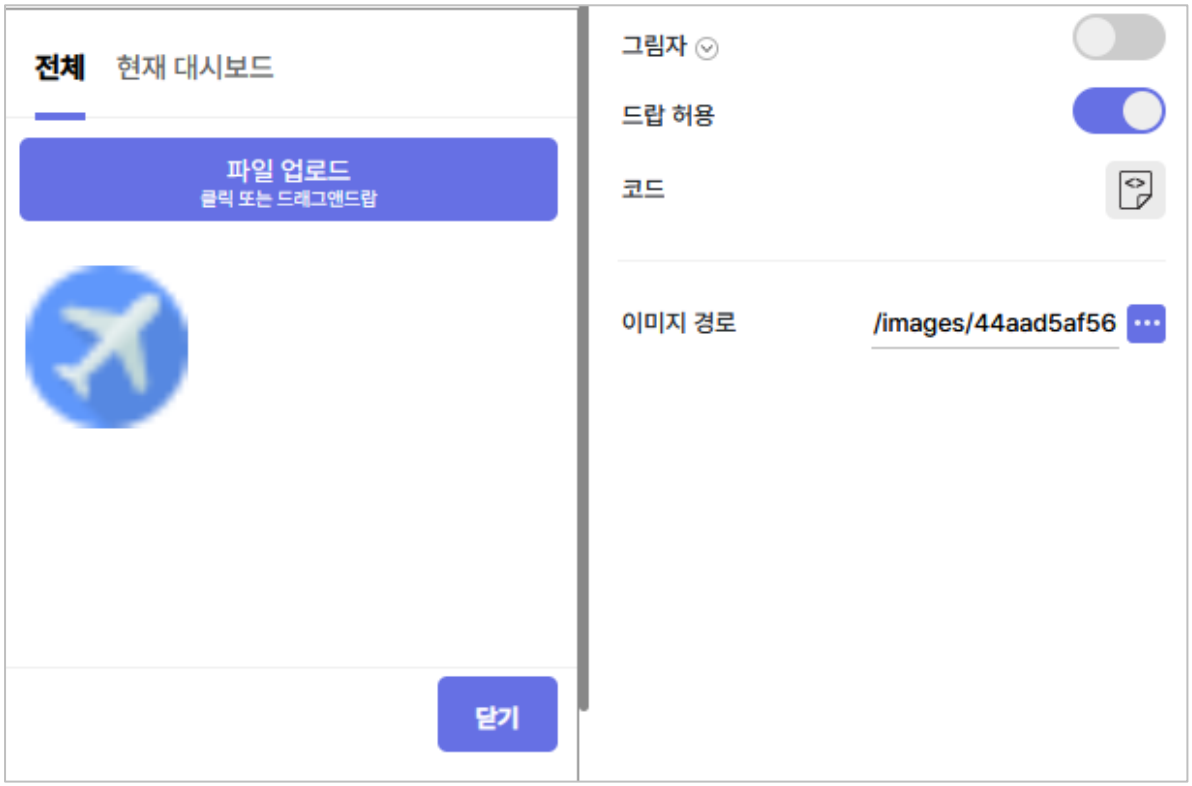
label	Control 을 설명하거나 식별하기 위한 라벨. 사용자가 임의로 정합니다.
control	사용자로부터 입력 받을 컨트롤. 유효한 값을 입력합니다. 유효한 값은 컨트롤 소개에서 확인하십시오.
attribute	layoutMetaData 의 key. 사용자가 임의로 정합니다.
type	ResourceSelector 의 타입입니다. 기본값:image, 유효값:image,sound,video

레이아웃으로 이동하여 레이아웃을 작성합니다. {{}}는 handlebars 의 문법이고 meta.imgSrc 로 layoutMetaData 의 imgSrc 의 값을 가져옵니다.

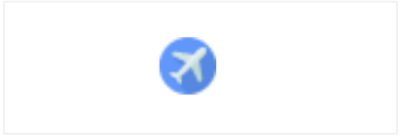


에디터의 우측 상단의 적용 버튼을 눌러 변경한 사항을 적용합니다.

우측 위젯 상세 메뉴에서 자원 경로를 입력할 수 있고 ••• 버튼을 눌러 업로드한 자원을 선택할 수 있습니다.



설정을 완료하면 다음과 같은 결과를 확인할 수 있습니다.



3. 그룹 메뉴 작성 방법

3.1. repeatTarget

데이터가 바인딩 된 상태에서 propertyOption 에 repeatTarget 을 column 으로 설정하면 그룹 메뉴를 작성할 수 있습니다.

기존 property 작성에 "repeatTarget":"column"을 추가합니다.
layoutMetaData 에 attribute 값을 key 값으로 추가 후 값이 여러 개이기 때문에 배열 [] 을 설정합니다.
ColumnComboBox 는 repeatTarget 을 지원하지 않습니다.

레이아웃 스크립트 **컴포넌트 옵션** 사용자 데이터

```
{
  "propertyOptions": [
    {
      "label": "글자 색상",
      "control": "ColorSelector",
      "attribute": "fontColors",
      "repeatTarget": "column"
    }
  ],
  "layoutMetaData": {
    "fontColors": [],
    "columns": [
      {
```

레이아웃 샘플입니다.

레이아웃 스크립트 컴포넌트 옵션 사용자 데이터

```
{{#if meta.columns.length}}
{{#each meta.columns as |column|}}
<h1 style="color:{{lookup @root.meta.fontColors @index}};">{{column.label}}</h
{{/each}}
{{/if}}
```

meta.columns 의 값이 있는지 체크 후 있으면 내장함수인 each 함수로 meta.columns 의 length 만큼 반복하여 실행합니다. 반복문 내부에서 column 이라는 이름으로 반복 시 현재 요소로 사용하게 됩니다.

<h1>태그 안에 {{column.label}}은 column 객체의 label 속성을 출력합니다. 그리고 style 에 color 를 @root.meta.fontColors 의 값을 반복적으로 꺼내어 설정합니다.

내장함수인 lookup 함수는 특정 객체 내에서 속성(키)을 찾아 그 값을 반환하는 함수입니다. 사용법은 {{lookup 객체 '키'}}입니다.

meta 앞에 @root 를 사용하는 이유는 반복문 내부에서 context 가 달라져서 meta 데이터에 접근하기 위함 입니다.

@index 인데 반복문 진입 후 현재 index 를 반환합니다.

결과는 다음과 같습니다.

4. 유의사항

- 컴포넌트 옵션 작성시 JSON 형식에 맞춰 작성해야 합니다. 따옴표를 사용할 때는 큰따옴표(")를 사용하여야 오류가 나지 않습니다.
- JSON 에서 key 작성시에는 꼭 큰따옴표로 감싸 작성하여 주십시오.

5. 참고 사이트

Handlebars 사이트 https://handlebarsjs.com/

감사합니다