

---

# DDP Element 데이터 형식

Drilldown Play Editor

---

Ver. 1.0

2021-1

(주)에스오엑스

# 문서 개요

## 1 DDP 엘리먼트 데이터 형식

: 대시보드의 엘리먼트와 Node-red 데이터 바인딩 방법 및 설정

## 2 문서 목적

- Rect 엘리먼트의 Box, Alarm 실시간 스타일 지정하는 방법 안내  
(Popup에 정보를 넣는 방법 안내 포함)
- Table, Chart 엘리먼트에 전달할 데이터 설정 및 스타일 지정 방법 안내



## DDP 관련 정보 사이트

[https://github.com/soxcorp/DDP\\_deploy](https://github.com/soxcorp/DDP_deploy)

1 DDP Editor 설치파일

2 DDP 사용법 동영상 링크

3 DDP 활용 참조 문서들 및 예제들



---

## Element 스타일 지정

---

# 1. DDP 편집 위치

1

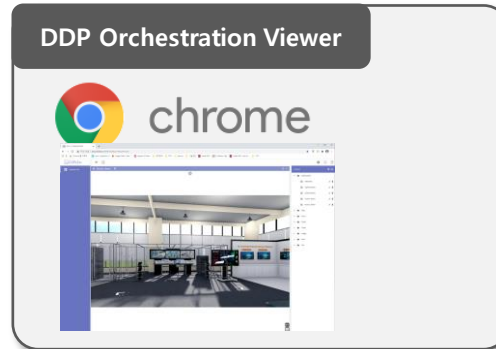
- 배경 및 엘리먼트 생성 등에 대한 그래픽 편집
- 정적인 스타일은 여기에서 설정 (테이블, 차트는 제외)



로컬 PC

2

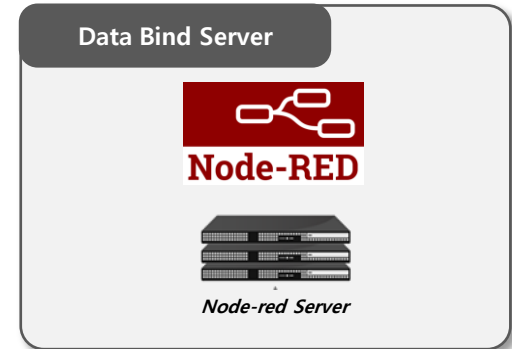
- 엘리먼트와 데이터 노드 연결 설정
- URL을 지정함



<http://naon.soxxcorp.co.kr/DrilldownPlay/#/main>

3

- 외부 데이터(데이터 소스)와 연결
- 엘리먼트에 데이터를 전송할 인터페이스 설정
- Rect, Alarm 동적 스타일 지정
- Table, Chart 스타일 지정



<http://naon.soxxcorp.co.kr:11103>

## 2. DDP 노드 편집



1

- 접속 URL을 설정함
  - 접속 주소 = **Node\_red** 접속URL + URL
- ex) [http://nano.soxcorp.co.kr:11103/factory\\_metric](http://nano.soxcorp.co.kr:11103/factory_metric)

메소드	GET
URL	/factory_health
이름	이름

2

- 엘리먼트에 전달할 데이터와 스타일 설정

```
var value = 30
msg.payload = {
  titleCaption: "" + value ,
  titleColor: '#ff0000',
  fontSize: '50px'
}
return msg;
```

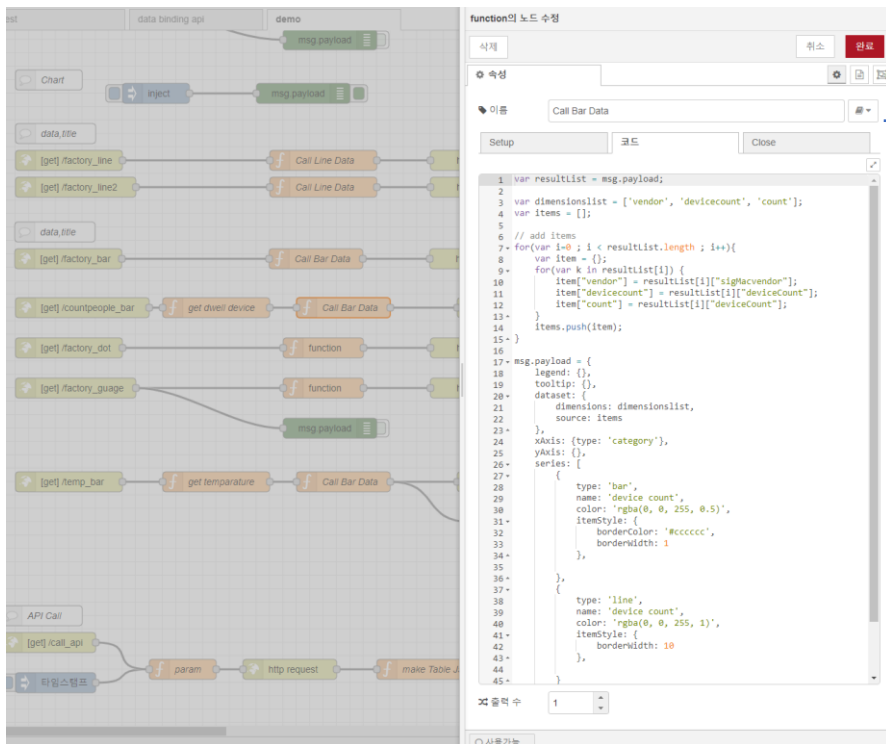
\* 전달할 데이터는 msg.payload에 담는다.

## 2. DDP 노드 편집 방법

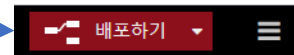
1 함수선택 (더블클릭)



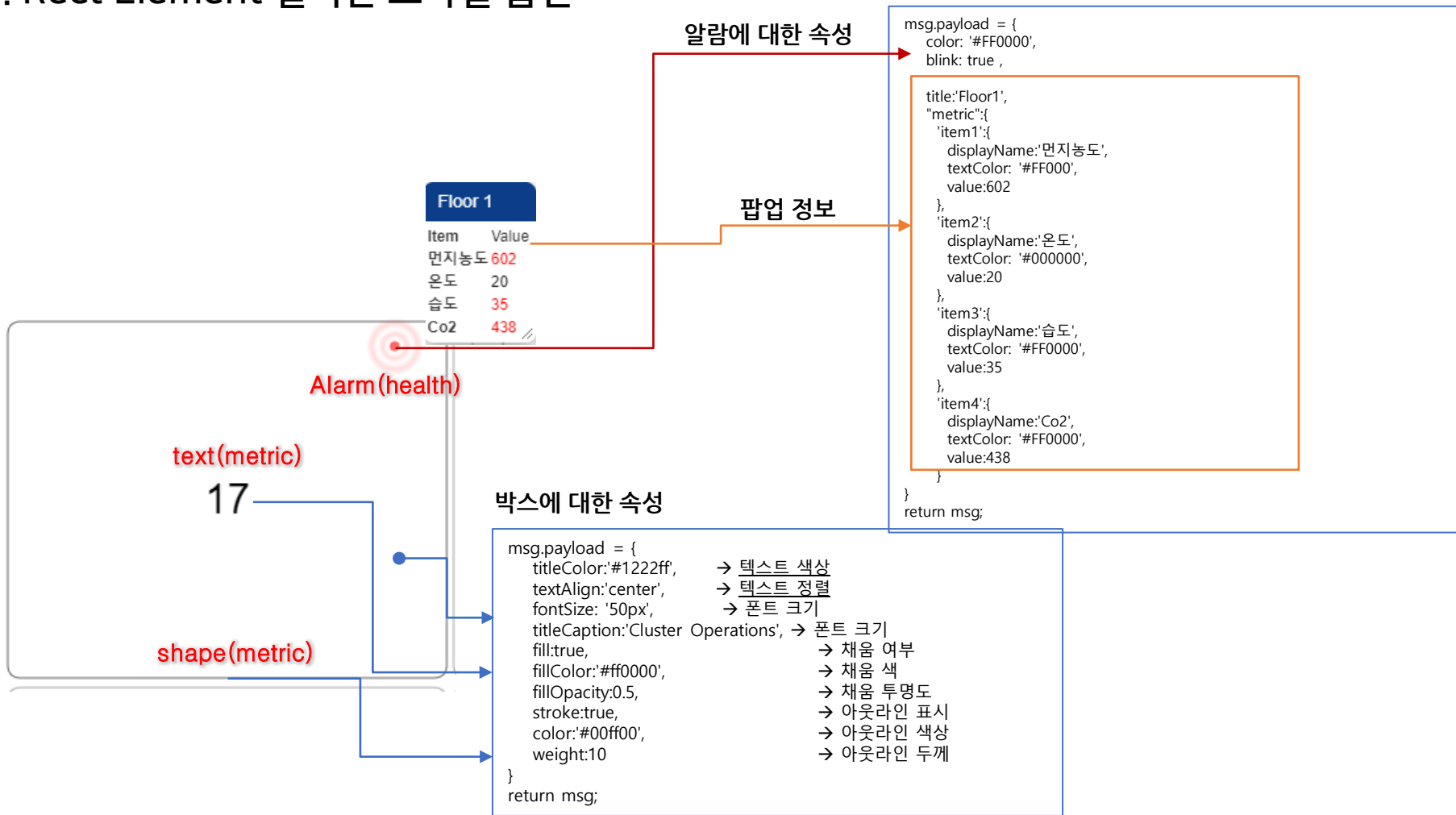
2 내용 편집 (완료로 저장)



3 '배포하기'로 동작 활성화

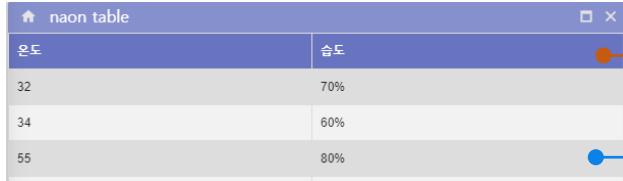


### 3. Rect Element 실시간 스타일 옵션





## 4. Table Element 데이터 형식



온도	습도
32	70%
34	60%
55	80%

```
var headerlist = [  
  {text:'온도', value:'temp'},  
  {text:'습도', value:'humid'}  
];
```

헤더형식

```
var items = [  
  {temp:32, humid:'70%'},  
  {temp:34, humid:'60%'},  
  {temp:55, humid:'80%'},  
];
```

항목데이터 형식

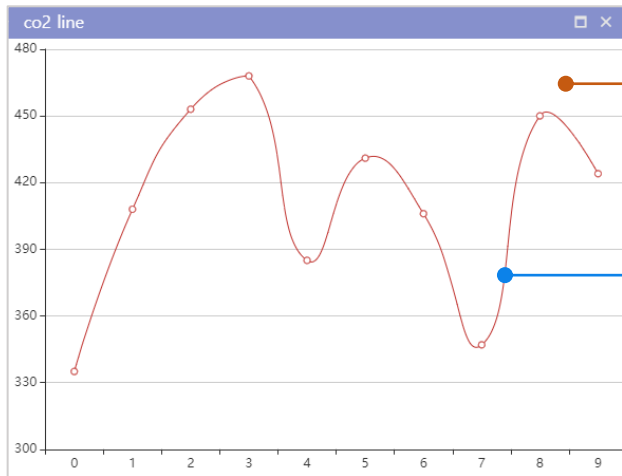
```
msg.payload = {  
  headers: headerlist,  
  items:items  
};
```

전송형식

```
return msg;
```

\* 테이블은 아직 스타일 변경을 지원하지 않음

## 5. Chart Element 데이터 형식 (Line Chart)



\* dataList에 표현할 데이터를 1차원 배열로 담아 주면 됨

```
var dataList = [26,30,21,15,14,19,25,30,45,55];  
var title = '타이틀';
```

데이터 형식

```
msg.payload = {  
  grid: {  
    top:10,  
    left:'10%',  
    right:'0%',  
    bottom:'10%'  
  },  
  xAxis:{  
    type: "category",  
    data:[]  
  },  
  yAxis:{  
    type:'value',  
    scale: true,  
  },  
}
```

차트 옵션 : 건드리지 않음

```
series: [{  
  name:title,  
  animation: false,  
  data: dataList,  
  type: 'line',  
  smooth: true,
```

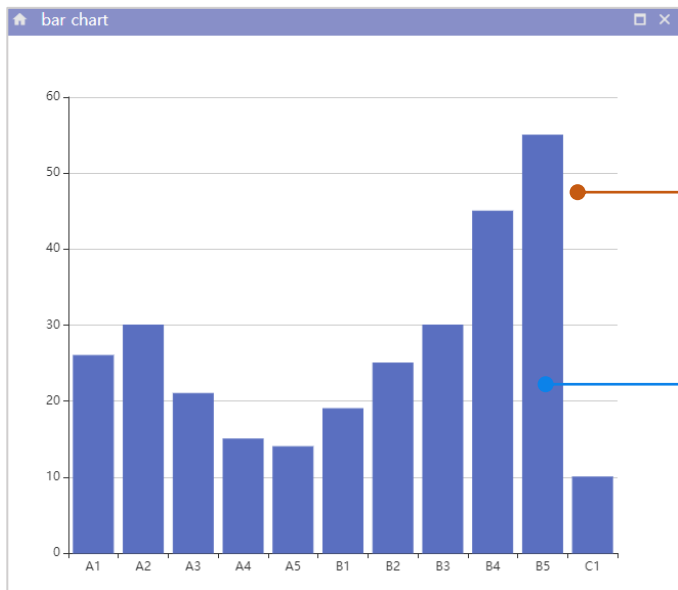
true : 곡선 / false : 직선

```
  lineStyle: {  
    color: '#5470C6',  
    width: 2,  
  },  
  symbol: 'circle',  
  symbolSize: 5,  
  itemStyle: {  
    borderWidth: 1,  
    borderColor: '#5470C6',  
    color: '#5470C6'  
  }  
}
```

라인스타일 옵션

```
  }  
}  
  
return msg;
```

## 5. Chart Element 데이터 형식 (Bar Chart)



\* dataList에 표현할 데이터를 1차원 배열로 담고  
X축에 표시할 내용을 xRowList에 담음

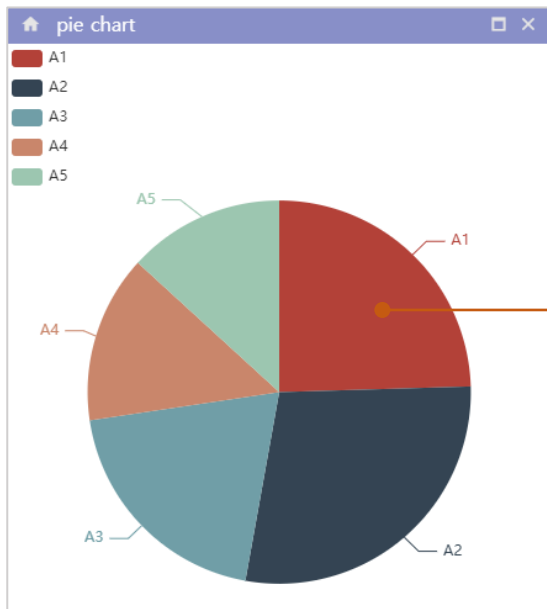
```
var xRowList = ["A1","A2","A3","A4","A5","B1","B2","B3","B4","B5","C1"]; X축 표시값  
var dataList = [26,30,21,15,14,19,25,30,45,55,10]; 데이터
```

```
msg.payload = {  
  tooltip: {},  
  xAxis: {  
    type: 'category',  
    data: xRowList  
  },  
  yAxis: {  
    type: 'value'  
  },  
  series: [  
    {  
      name: 'data1',  
      data: dataList,  
      type: 'bar',  
      barWidth: '80%',  
      barGap: 0,  
      itemStyle: {  
        borderWidth: 1,  
        borderColor: '#5470C6',  
        color: '#5470C6'  
      }  
    }  
  ]  
};
```

바 스타일 옵션

```
return msg;
```

## 5. Chart Element 데이터 형식 (Pie Chart)



\* dataList에 표현할 데이터를 1차원 배열로 담고  
라벨에 표시할 내용을 RowList에 담음

```
var rowList = ["A1","A2","A3","A4","A5"];  
var dataList = [26,30,21,15,14];  
var items = [];
```

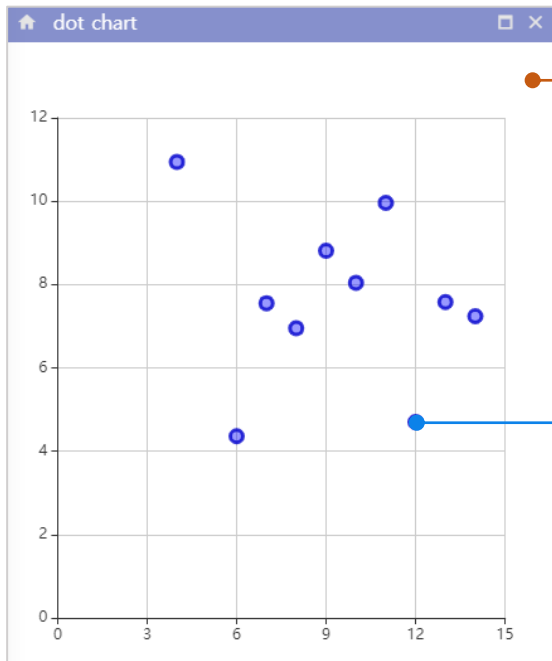
데이터

```
// add items  
for(var i=0 ; i < dataList.length ; i++){  
    var item = {};  
    item["value"] = dataList[i];  
    item["name"] = rowList[i];  
    items.push(item);  
}
```

데이터를 2차원으로 구성

```
msg.payload = {  
    legend: {  
        orient: 'vertical',  
        left: 'left',  
    },  
    tooltip: {  
        trigger: 'item'  
    },  
    series: [  
        {  
            name: '점유율',  
            type: 'pie',  
            radius: '70%',  
            data: items,  
            emphasis: {  
                itemStyle: {  
                    shadowBlur: 10,  
                    shadowOffsetX: 0,  
                    shadowColor: 'rgba(0, 0, 0, 0.5)'  
                }  
            }  
        }  
    ]  
};  
return msg;
```

## 5. Chart Element 데이터 형식 (Dot Chart)



```
var rowList = ["A1","A2","A3","A4","A5","B1","B2","B3","B4","B5"];
var xList = [10, 8, 13, 9, 11, 14, 6, 4, 12, 7];
var yList = [8.04, 6.95, 7.58, 8.81, 9.96, 7.24, 4.36, 10.94, 4.7, 7.55];
var dataList = [200, 100, 30, 40, 120, 70, 80, 90, 150, 60];
```

데이터

```
var items = [];
for(var i=0 ; i < dataList.length ; i++){
    var item = [];
    item.push(xList[i]);
    item.push(yList[i]);
    item.push(dataList[i]);
    item.push(rowList[i]);
    items.push(item);
}
```

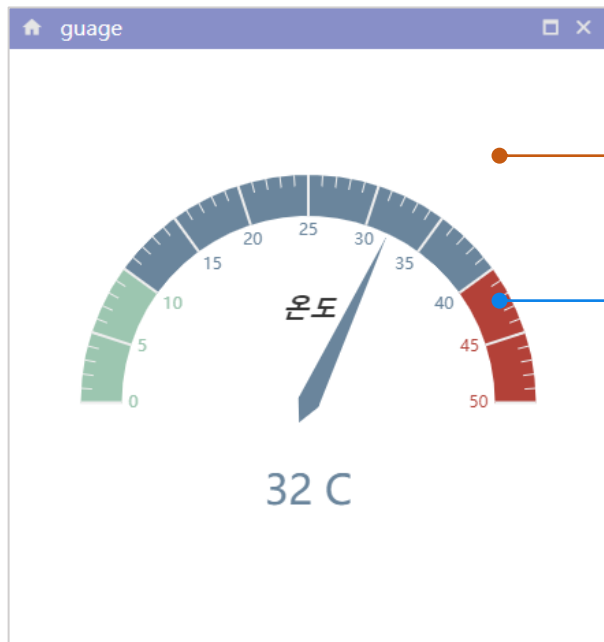
데이터를 2차원으로 구성

```
msg.payload = {
  xAxis: {},
  yAxis: {},
  series: [{
    color: 'rgba(0, 0, 255, 0.5)',
    itemStyle: {
      borderColor: '#0000cc',
      borderWidth: 3
    },
    symbolSize: function (data) {
      return data[2];
    },
    emphasis: {
      label: {
        show: true,
        formatter: function (param) {
          return param.data[3] + " : " + param.data[2];
        },
        position: 'top'
      }
    },
    data: items,
    type: 'scatter'
  }]
};
return msg;
```

도트 스타일 옵션

- \* 데이터 항목 형식은 [x좌표, y좌표, 값, id]
- x좌표 List와 y좌표 List는 필수적임

## 5. Chart Element 데이터 형식 (Gauge Chart)



var value = 10; **데이터**

```
msg.payload = {  
  tooltip: {  
    formatter: '{a} <br/>{b} : {c}%'  
  },  
  series: [  
    {  
      name: 'Progress',  
      type: 'gauge',  
      min: 0,  
      max: 50,  
      startAngle: 180,  
      endAngle: 0,  
      title: {  
        fontWeight: 'bolder',  
        fontSize: 20,  
        fontStyle: 'italic'  
      },  
      detail: {formatter: '{value} C'},  
      data: [{value: value, name: '온도'}]  
    }  
  ]  
};  
return msg;
```

게이지 범위 및 각도 조정

데이터 표시 형식