# 父组件传值给子组件

## 创建子组件后，父组件获取数据

ngOnInit(): void {

    this.getData()

}

res: any

getData() {

    this.http.get('assets/data-directive.json').subscribe((response: any) => {

      this.res = response.data

    })

 }

## 通过input传给子组件

其中父组件html这么写：

<app-child [res]="res"></app-child>

子组件ts接收：

@Input() res: any;

## 子组件通过ViewChild获取元素绘制图表

1. 引入并使用ViewChild

Ts文件使用

@ViewChild("echart") echartdom: any;

Html标记

<div #echart class="echart">

</div>

1. 子组件引入并使用Echarts

import { ECharts, init } from "echarts";

echart!: ECharts;

1. 建立option

  EChartsOption = {

    tooltip: {

      trigger: "item",

      transitionDuration: 0,

    },

    legend: {

      orient: "vertical",

      left: "left",

    },

    series: [

      {

        name: "内存情况",

        data: [

          { value: 0 },

        ],

        type: "pie",

        radius: ["60%", "70%"],

        avoidLabelOverlap: false,

        label: {

          show: false,

          position: "center",

        },

        emphasis: {

          // label: {

          //   show: true,

          //   fontSize: '40',

          //   fontWeight: 'bold'

          // }

          itemStyle: {

            shadowBlur: 10,

            shadowOffsetX: 0,

            shadowColor: "rgba(0, 0, 0, 0.5)",

          },

        },

        labelLine: {

          show: false,

        },

        itemStyle: {

          normal: {

            color: function (params) {

              var colorList = ["#F52222", "#55A722", "#1FE6D4"];

              return colorList[params.dataIndex];

            },

            label: {

              show: true,

              position: "top",

              // formatter: "{b}\n{c}",

            },

          },

        },

      },

    ],

  };

1. 在子组件初始化echarts

  ngAfterViewInit(): void {

    this.echart = init(this.echartdom.nativeElement);

  }

初始化的过程在视图即将出现的生命周期函数ngAfterViewInit中

1. 数据驱动更新

下面这是更新echarts图表的函数，在ngOnChanges生命周期函数中调用。当有数据改变时就会触发更新。

  freshEcharts() {

    let arr = [];

    arr = [

      {

        value: this.res.usedStorage,

        name: "已用内存",

      },

      {

        value: this.res.availableStorage,

        name: "可用内存",

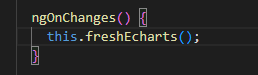
      },

    ];

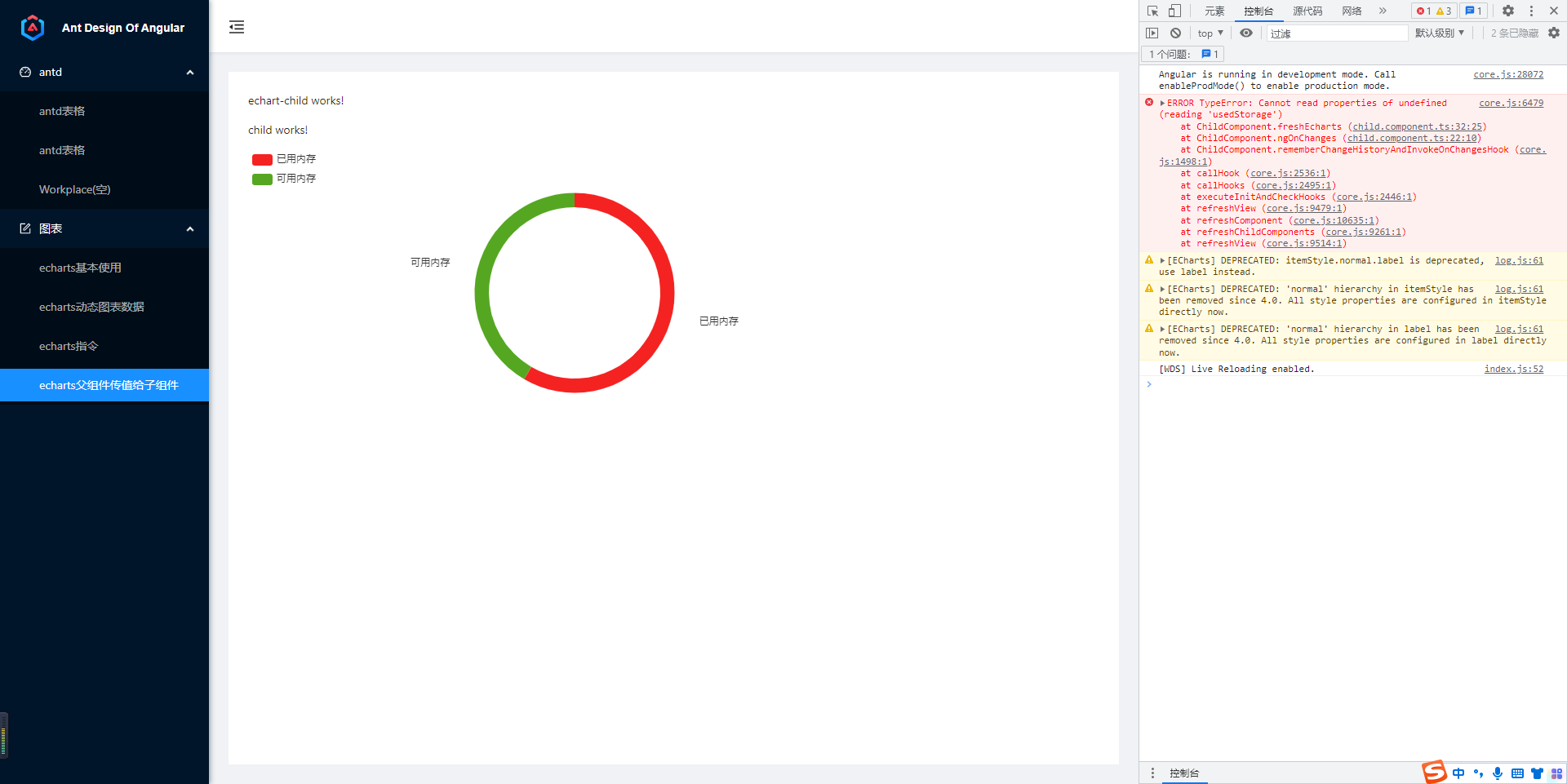
    this.EChartsOption.series[0].data = arr;

    this.echart.setOption(this.EChartsOption);

  }



1. 此时页面显示出图表，但是页面有报错



## 解决报错

（1）属性不存在

core.js:6479 ERROR TypeError: Cannot read properties of undefined (reading 'usedStorage')

我们前面是把整个返回的数据传给子组件的，这里说读取不到这个属性，那么我们可以在父组件进行定义一个空值。代码如下：

  res: any ={

    usedStorage:0,

  }

（2）无法设置图表配置项setOption

core.js:6479 ERROR TypeError: Cannot read properties of undefined (reading 'setOption')

这里就是子组件ngOnChanges使用问题了。我们打印看看这里使用的具体信息。

  ngOnChanges(changes: SimpleChanges) {

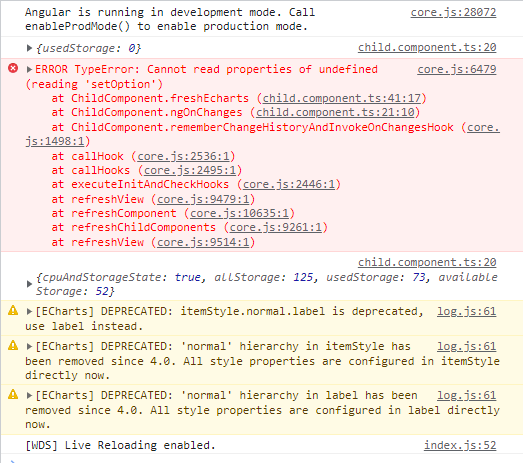
    const current = changes.res && changes.res.currentValue

    console.log(current);

    this.freshEcharts();

  }

这里打印changes表现不是特别直观，我们就打印current的值。



由打印结果可以得知，第一次打印输出为未get获取到数据的父组件初始值，第二次打印输出为正常获取到的数据，但是在第一次未获取正常数据之前，已经进行了一次更新图表，这就是报错的原因。解决方案很简单，让更新图表的函数选择性执行就行。编码思路：

我们刚刚打印得知，前后相差几个变量，那么我们判断这相差的变量是否存在即可。

代码如下：

  ngOnChanges(changes: SimpleChanges) {

    const current = changes.res && changes.res.currentValue

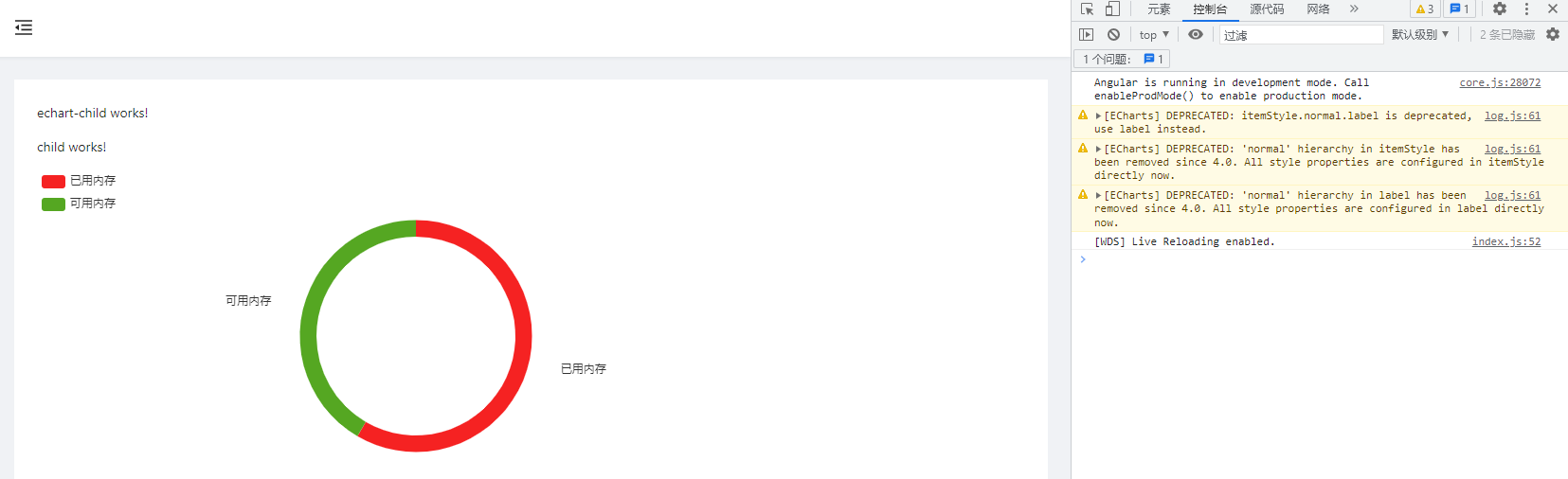
    if ('availableStorage' in current) {

      this.freshEcharts();

    }

  }

这样就可以正常显示而且没有报错了。



# Echarts使用水球图

## 下载插件

yarn add echarts-liquidfill

## 在main.ts中注册

import 'echarts-liquidfill';

## 进行使用

使用方式和echarts没有太大区别，主要是在option中修改参数。

Series数组再加入一个对象：

{

        name: "内存占用",

        outline: {

          show: false,

        },

        radius: "60%",

        type: "liquidFill",

        data: [

          {

              value: 0.6,

              direction: 'left', //波浪方向

          },

        ],

        amplitude: '15 %', //波浪的振幅

        color: ["#dbc7b5"],

        label: {

          formatter: function (param) {

            return `${param.seriesName}: ${param.value \* 100}%`;

          },

          fontSize: 14,

        },

        tooltip: {

          formatter(param) {

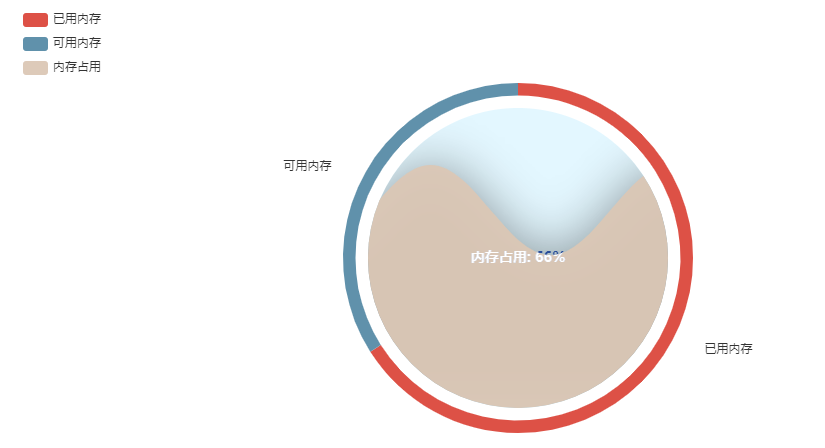
            return `${param.seriesName}: ${param.value \* 100}%`;

          },

        },

},

## 效果图如下



## 参数讲解

详细参数可以阅读github讲解<https://github.com/ecomfe/echarts-liquidfill>

### type: 'liquidFill',

这是水球图实现的关键，type声明了这个的图的属性。

官方讲解中这里的type有这些属性：

line（折线图）、bar（柱状图）、pie（饼图）、scatter（散点图）、effectScatter（气泡图）、radar（雷达图）、tree（树图）、treemap（树状数据图）、sunburst（旭日图）、boxplot（箱形图）、candlestick（K线图）、heatmap（热力图）、map（地图）、parallel（平行坐标系）、lines（路径图）、graph（关系图）、sankey（桑基图）、funnel（漏斗图）、gauge（仪表盘）、pictorialBar（象形柱图）、themeRiver（主题河流）、custom（自定义系列）。

### 多重波浪

var option = {

series: [{

type: 'liquidFill',

data: [0.6, 0.5, 0.4, 0.3]

}]};

