##### 实现两个节点

###### 1.1、html部分

<template>

  <div class="wrap">

    <div id="container"></div>

  </div>

</template>

###### 1.2、js部分（放在mouted方法内）

 const data = {  // 数据

      nodes: [ //节点数据

        {

          id: "node1",

          label: "node1",

        },

        {

          id: "node2",

          label: "node2",

        },

      ],

      edges: [ // 连线数据

        {

          source: "node1", // 连线的起始位置

          target: "node2", // 连线的终点位置

        },

      ],

    };

    const container = document.getElementById("container"); // DAG图容器

    const width = container.scrollWidth || 400;

    const height = container.scrollHeight || 500;

    const graph = new G6.Graph({  // graph 实例化

      container: "container",

      width: width,

      height: height,

      modes: { // 模式 如下有（画布拖拽、节点拖拽）

        default: ["drag-canvas", "drag-node"],

      },

      defaultNode: { // 节点默认样式

        type: "rect",

      },

      defaultEdge: { // 连线默认样式

        type: "cubic",

        style: {

          stroke: "#F6BD16",

          lineWidth: 2,

          endArrow: { // 箭头默认样式

            path: G6.Arrow.vee(),

            fill: "#F6BD16",

          },

        },

      },

    });

    graph.data(data); // 填充数据

    graph.render();  // 渲染

    if (typeof window !== "undefined")

      window.onresize = () => {

        if (!graph || graph.get("destroyed")) return;

        if (!container || !container.scrollWidth || !container.scrollHeight)

          return;

        graph.changeSize(container.scrollWidth, container.scrollHeight);

      };

1. 自定义节点

G6.registerNode(

typeName: string, // 该新节点类型名称

nodeDefinition: object, // 该新节点类型的定义

extendedTypeName: string // 被继承的节点类型，可以是内置节点类型名

)

3、e.propagationStopped = true, 阻止事件冒泡