山东大学计算机科学与技术学院

可视化技术课程实验报告

学号：202302130293	姓名：李嘉欣	班级：数据科学与大数据技术
实验题目：三、电子表格实践 I	实验日期：2025/10/17	

实验目标：
本实验旨在基于开源电子表格组件 x-data-spreadsheet 实现可视化功能的扩展。通过在原有电子表格基础上添加新的可视化（vis）函数，使表格中的数据能够被实时读取并以图的形式动态展示。

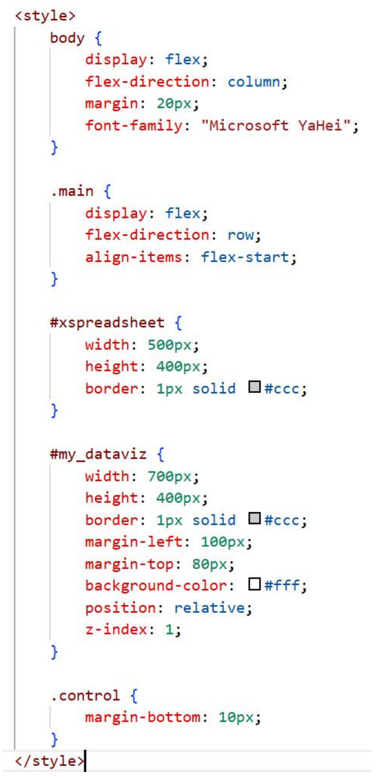
实验环境：windows 系统

实验步骤：

1.引入依赖库

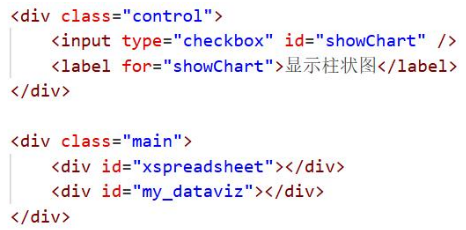
```
<link rel="stylesheet" href="https://unpkg.com/x-data-spreadsheet@1.1.5/dist/xspreadsheet.css" />
<script src="https://unpkg.com/x-data-spreadsheet@1.1.5/dist/xspreadsheet.js"></script>
<script src="https://unpkg.com/x-data-spreadsheet@1.1.9/dist/locale/zh-cn.js"></script>
<script src="https://d3js.org/d3.v6.js"></script>
```

2.布局样式



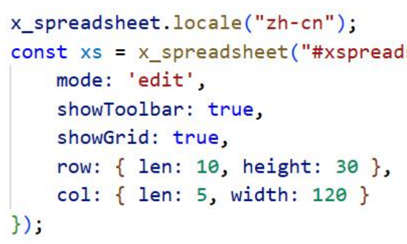
```
<style>
    body {
        display: flex;
        flex-direction: column;
        margin: 20px;
        font-family: "Microsoft YaHei";
    }

    .main {
        display: flex;
        flex-direction: row;
        align-items: flex-start;
    }

    #xspreadsheet {
        width: 500px;
        height: 400px;
        border: 1px solid #ccc;
    }

    #my_dataviz {
        width: 700px;
        height: 400px;
        border: 1px solid #ccc;
        margin-left: 100px;
        margin-top: 80px;
        background-color: #fff;
        position: relative;
        z-index: 1;
    }

    .control {
        margin-bottom: 10px;
    }
</style>
```

3.网页主体结构



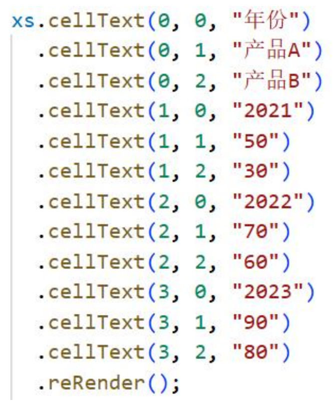
```
<div class="control">
    <input type="checkbox" id="showChart" />
    <label for="showChart">显示柱状图</label>
</div>

<div class="main">
    <div id="xspreadsheet"></div>
    <div id="my_dataviz"></div>
</div>
```

4. 初始化电子表格



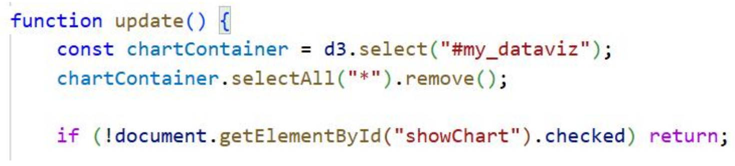
```
x_spreadsheet.locale("zh-cn");
const xs = x_spreadsheet("#xspreadsheet", {
    mode: 'edit',
    showToolbar: true,
    showGrid: true,
    row: { len: 10, height: 30 },
    col: { len: 5, width: 120 }
});
```

5. 填充初始数据



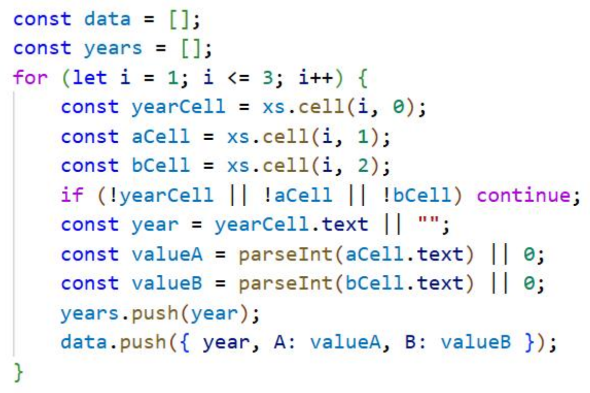
```
xs.cellText(0, 0, "年份")
  .cellText(0, 1, "产品A")
  .cellText(0, 2, "产品B")
  .cellText(1, 0, "2021")
  .cellText(1, 1, "50")
  .cellText(1, 2, "30")
  .cellText(2, 0, "2022")
  .cellText(2, 1, "70")
  .cellText(2, 2, "60")
  .cellText(3, 0, "2023")
  .cellText(3, 1, "90")
  .cellText(3, 2, "80")
  .reRender();
```

6. 定义更新函数



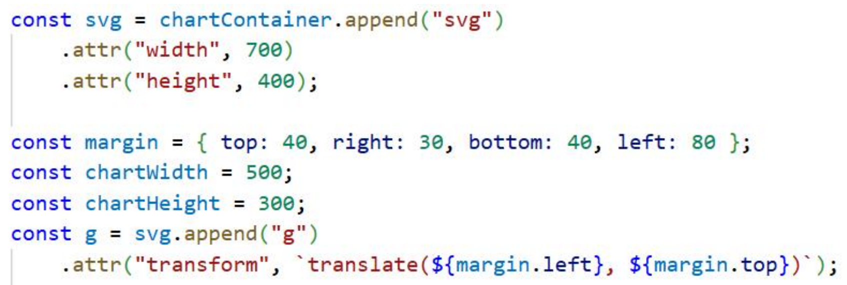
```
function update() {
    const chartContainer = d3.select("#my_dataviz");
    chartContainer.selectAll("*").remove();

    if (!document.getElementById("showChart").checked) return;
```

7. 从表格读取数据



```
const data = [];
const years = [];
for (let i = 1; i <= 3; i++) {
    const yearCell = xs.cell(i, 0);
    const aCell = xs.cell(i, 1);
    const bCell = xs.cell(i, 2);
    if (!yearCell || !aCell || !bCell) continue;
    const year = yearCell.text || "";
    const valueA = parseInt(aCell.text) || 0;
    const valueB = parseInt(bCell.text) || 0;
    years.push(year);
    data.push({ year, A: valueA, B: valueB });
}
```

8. 创建 SVG 绘图区域



```
const svg = chartContainer.append("svg")
    .attr("width", 700)
    .attr("height", 400);

const margin = { top: 40, right: 30, bottom: 40, left: 80 };
const chartWidth = 500;
const chartHeight = 300;
const g = svg.append("g")
    .attr("transform", `translate(${margin.left}, ${margin.top})`);
```

9. 设置分层结构

```
const bgLayer = g.append("g").attr("class", "background-layer");
const axisLayer = g.append("g").attr("class", "axis-layer");
const barLayer = g.append("g").attr("class", "bar-layer");
const labelLayer = g.append("g").attr("class", "label-layer");
```

10. 绘制白色背景

```
bgLayer.append("rect")
    .attr("x", -60)
    .attr("y", -30)
    .attr("width", chartWidth + 120)
    .attr("height", chartHeight + 80)
    .attr("fill", "white")
    .attr("stroke", "#ddd");
```

11. 定义比例尺与坐标轴

```
const x = d3.scaleBand()
    .domain(years)
    .range([0, chartWidth])
    .padding(0.2);
const maxValue = d3.max(data, d => Math.max(d.A, d.B)) * 1.1;
const y = d3.scaleLinear()
    .domain([0, maxValue])
    .range([chartHeight, 0]);
axisLayer.append("g")
    .attr("transform", `translate(0, ${chartHeight})`)
    .call(d3.axisBottom(x));
axisLayer.append("g").call(d3.axisLeft(y));
```

12. 绘制柱状图

```
barLayer.selectAll(".barA")
    .data(data)
    .enter()
    .append("rect")
    .attr("class", "barA")
    .attr("x", d => x(d.year))
    .attr("y", d => y(d.A))
    .attr("width", x.bandwidth() / 2 - 5)
    .attr("height", d => chartHeight - y(d.A))
    .attr("fill", "#3366cc");

barLayer.selectAll(".barB")
    .data(data)
    .enter()
    .append("rect")
    .attr("class", "barB")
    .attr("x", d => x(d.year) + x.bandwidth() / 2 + 5)
    .attr("y", d => y(d.B))
    .attr("width", x.bandwidth() / 2 - 5)
    .attr("height", d => chartHeight - y(d.B))
    .attr("fill", "#dc3912");
```

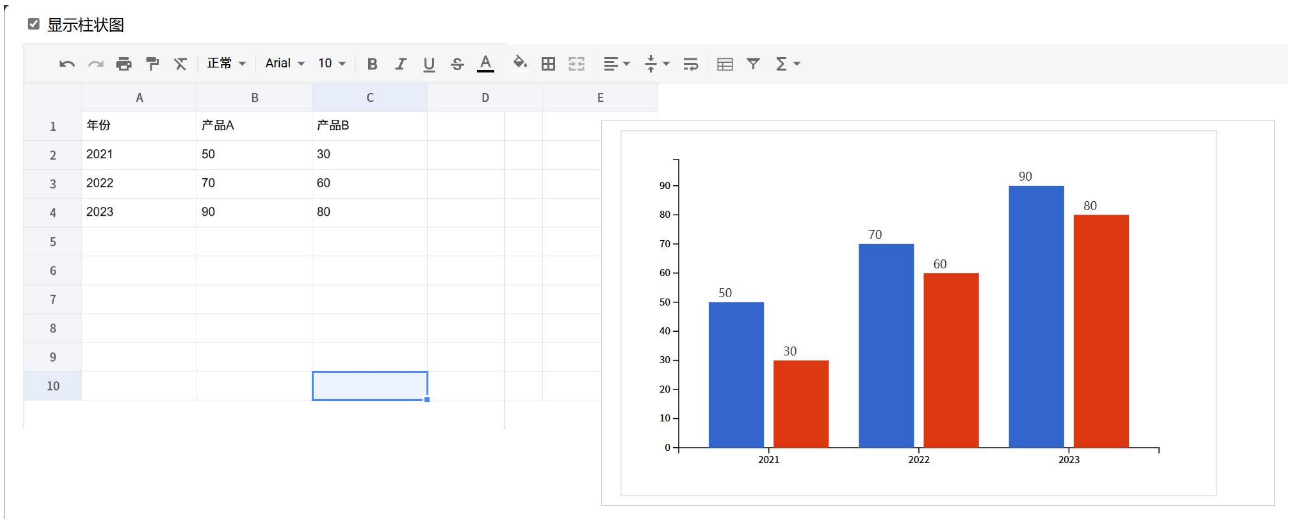
13.添加数据标签

```
labelLayer.selectAll(".labelA")
    .data(data)
    .enter()
    .append("text")
    .attr("x", d => x(d.year) + 10)
    .attr("y", d => y(d.A) - 5)
    .attr("fill", "#333")
    .attr("font-size", "12px")
    .text(d => d.A);

labelLayer.selectAll(".labelB")
    .data(data)
    .enter()
    .append("text")
    .attr("x", d => x(d.year) + x.bandwidth() / 2 + 15)
    .attr("y", d => y(d.B) - 5)
    .attr("fill", "#333")
    .attr("font-size", "12px")
    .text(d => d.B);
```

14.绑定事件更新

```
xs.on('cell-edited', update);
document.getElementById("showChart").addEventListener("change", update);
```

输出：



☑ 显示柱状图

	A	B	C	D	E
1	年份	产品A	产品B		
2	2021	50	30		
3	2022	70	60		
4	2023	90	80		
5					
6					
7					
8					
9					
10					

实验分析与体会：

本实验通过在开源电子表格组件 x-data-spreadsheet 的基础上添加柱状图可视化功能，深入理解了数据交互与动态可视化的实现过程。实验中利用 D3.js 构建了数据驱动的绘图函数，实现了表格数据与柱状图的实时联动更新；同时通过分层绘制方式解决了图形遮挡与层级问题，使得背景、坐标轴、柱状图和标签的显示更加清晰。实验过程中遇到的主要困难在于比例尺设置与布局协调，通过调整坐标范围和布局参数得以解决。通过本次实践，我掌握了开源组件的二次开发方法，对前端数据可视化的原理与交互机制有了更直观的理解，也提升了综合运用 HTML、JavaScript 与 D3.js 的能力。