山东大学计算机科学与技术学院

大数据分析实践课程实验报告

学号：202300300198	姓名：朱宣玮	班级：数据 23
实验题目：电子表格实践 I		
实验学时：2	实验日期：2025.10.10	

实验目标：

Add a new vis function based on the open source spreadsheet codes:https://github.com/myliang/x-spreadsheet

实验环境：

Window，vscode，利用 x-spreadsheet 进行表格操作，利用 d3 进行可视化

实验步骤与结果：

实验步骤

1. 导入需要的官方库

导入 x-spreadsheet 电子表格库和 D3.js 可视化库，分别对应表格和可视化模块

```
<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. 加入复选框选择可视化类型

添加两个复选框，允许用户选择柱状图或折线图进行数据可视化

```
<div class="checkbox-group">

    <label><input type="checkbox" class="checkbox" value="barchart">柱状图</label>

    <label><input type="checkbox" class="checkbox" value="linechart">折线图</label>

</div>
```

3.初始化电子表格

使用 x-spreadsheet 库创建电子表格，设置编辑模式、工具栏、网格显示等参数

```
// 初始化电子表格
x_spreadsheet.locale("zh-cn");
const xs = x_spreadsheet("#xspreadsheet", {
    mode: 'edit',
    showToolbar: true,
    showGrid: true,
    showContextmenu: true,
    view: {
        height: () => 500,
        width: () => document.getElementById('xspreadsheet').clientWidth,
    },
    row: {
        len: 15,
        height: 25,
    },
    col: {
        len: 8,
        width: 100,
```

填充初始示例数据

```
// 设置初始数据
xs.cellText(0, 1, "计算机")
  .cellText(0, 2, "法学")
  .cellText(0, 3, "经济学")
  .reRender();

xs.cellText(1, 0, "2017")
  .cellText(1, 1, "23")
  .cellText(1, 2, "15")
  .cellText(1, 3, "32")
  .reRender();
```

4. 实现数据获取函数

创建 getSpreadsheetData()函数，从电子表格中提取行列标题和数值数据，验证数据并格式化。

```
function getSpreadsheetData() {
    const data = [];
    const yTitle = [];
    const xTitle = [];

    // 获取列标题（第一行）
    for (let j = 1; j < 20; j++) {
        const cell = xs.cell(0, j);
        if (!cell || !cell.text || cell.text.trim() === "") {
            break;
        }
        xTitle.push(cell.text);
    }

    // 获取行标题（第一列）和数据
    for (let i = 1; i < 20; i++) {
        const rowTitleCell = xs.cell(i, 0);
        if (!rowTitleCell || !rowTitleCell.text || rowTitleCell.text.trim() === "") {
            break;
        }
        yTitle.push(rowTitleCell.text);

        const rowData = [];
        for (let j = 1; j <= xTitle.length; j++) {
            const cell = xs.cell(i, j);
            if (!cell || !cell.text || isNaN(+cell.text)) {
                rowData.push(0); // 将无效数据设为0
            } else {
                rowData.push(+cell.text);
            }
        }
        data.push(rowData);
    }

    return { data, xTitle, yTitle };
}
```

5. 实现可视化绘制函数

分别实现柱状图和折线图的绘制函数，使用 D3.js 创建交互式可视化图表，设置坐标轴、图例、数据标签等元素。

```
// 绘制柱状图
function drawBarChart(data, xTitle, yTitle) {
    d3.select("#my_dataviz").html("");

    const margin = { top: 40, right: 150, bottom: 60, left: 50 };
    const width = 600 - margin.left - margin.right;
    const height = 400 - margin.top - margin.bottom;

    const svg = d3.select("#my_dataviz")
        .append("svg")
        .attr("width", width + margin.left + margin.right)
        .attr("height", height + margin.top + margin.bottom)
        .append("g")
        .attr("transform", `translate(${margin.left},${margin.top})`);

    // 计算最大值
    let maxValue = 0;
    data.forEach(row => {
        row.forEach(value => {
            if (value > maxValue) maxValue = value;
        });
    });
```

```
// 绘制折线图
function drawLineChart(data, xTitle, yTitle) {
    d3.select("#my_dataviz").html("");

    const margin = { top: 40, right: 150, bottom: 60, left: 50 };
    const width = 600 - margin.left - margin.right;
    const height = 400 - margin.top - margin.bottom;

    const svg = d3.select("#my_dataviz")
        .append("svg")
        .attr("width", width + margin.left + margin.right)
        .attr("height", height + margin.top + margin.bottom)
        .append("g")
        .attr("transform", `translate(${margin.left},${margin.top})`);

    // 计算最大值
    let maxValue = 0;
    data.forEach(row => {
        row.forEach(value => {
            if (value > maxValue) maxValue = value;
        });
    });
```

6. 实现更新机制

创建 update()函数作为核心更新机制，监听电子表格内容变化和复选框状态变化，实时更新可视化显示。

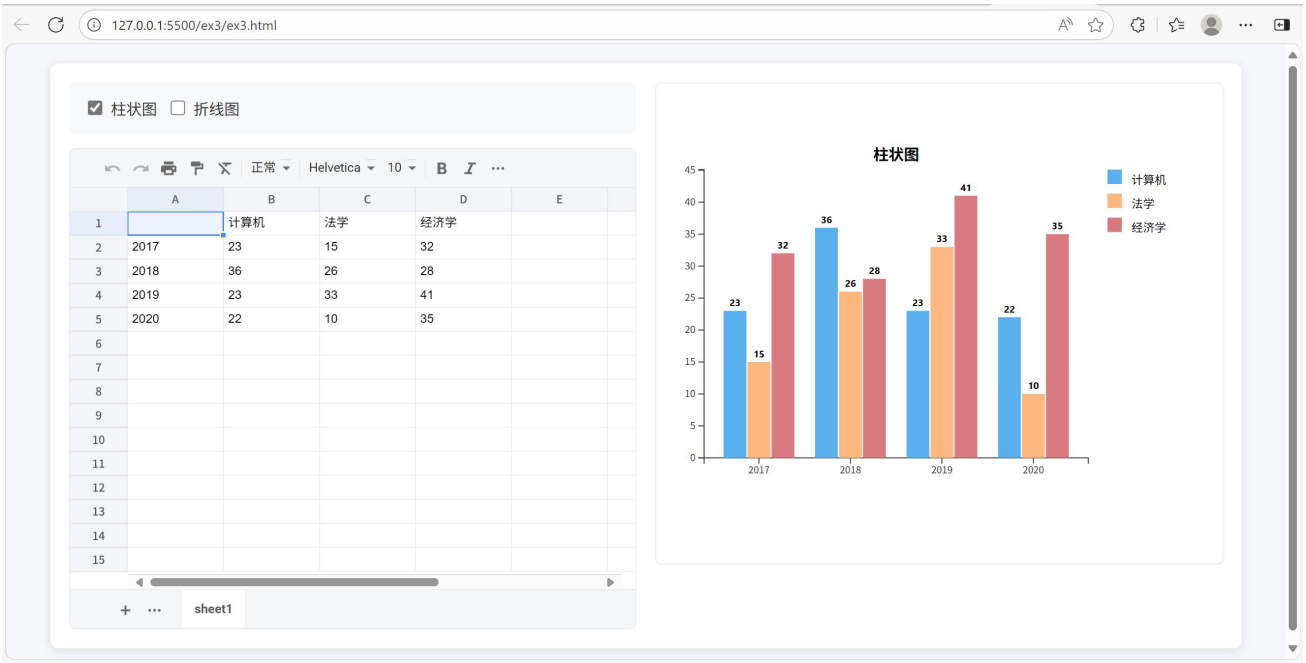
```
// 更新可视化
function update() {
    const barChartChecked = d3.select('input[value="barchart"]').property("checked");
    const lineChartChecked = d3.select('input[value="linechart"]').property("checked");

    if (!barChartChecked && !lineChartChecked) {
        d3.select("#my_dataviz").html("<div style='text-align: center; color: #7f8c8d;'><p>请选择可视化类型并输入数据</p></div>");
        return;
    }

    const { data, xTitle, yTitle } = getSpreadsheetData();

    // 确保数据有效
    if (data.length === 0 || xTitle.length === 0 || yTitle.length === 0) {
        d3.select("#my_dataviz").html("<div style='text-align: center; color: #e74c3c;'><p>数据格式不正确，请检查电子表格</p></div>");
        return;
    }

    if (barChartChecked) {
        drawBarChart(data, xTitle, yTitle);
    } else if (lineChartChecked) {
        drawLineChart(data, xTitle, yTitle);
    }
}
```

```
// 添加事件监听器
xs.on('cell-edited', update);
d3.selectAll(".checkbox").on("change", function() {
    // 确保只有一个复选框被选中
    if (this.value === "barchart" && this.checked) {
        d3.select('input[value="linechart"]').property("checked", false);
    } else if (this.value === "linechart" && this.checked) {
        d3.select('input[value="barchart"]').property("checked", false);
    }
    update();
});
```

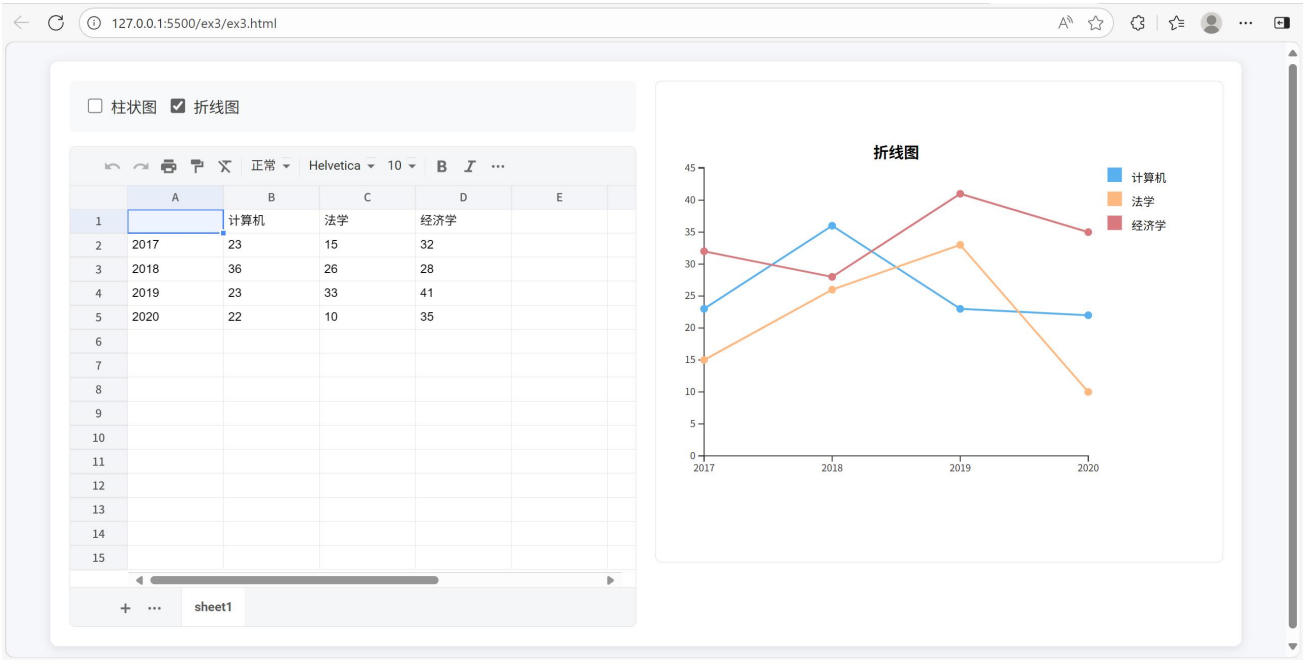
实验结果

用户可编辑表格中的数据并调整样式，选择柱状图或折线图进行可视化，交互式实时更新。

选择柱状图可视化



选择折线图可视化



结论分析与体会：

结论分析

本次实验成功基于 x-spreadsheet 和 D3.js 实现了电子表格与可视化图表的集成系统。

系统支持实时数据编辑和动态可视化更新，可自动处理数据验证和错误提示，能够根据用户选择生成柱状图或折线图，页面美观，交互体验良好。

体会

掌握了事件驱动机制在数据可视化中的应用，通过监听表格编辑和复选框变化实现实时更新。

认识到数据验证的重要性，必须对电子表格输入进行严格的类型检查和异常处理。

多图表类型选择功能需要合理的页面布局和交互设计，避免视觉混乱