

图形界面计算器

Table of Contents

- [图形界面计算器](#)
 - [功能介绍](#)
 - [开发环境](#)
 - [实现效果](#)
 - [图形化界面](#)
 - [计算演示](#)
 - [核心实现](#)
 - [界面](#)
 - [语法糖](#)
 - [转换为后缀表达式](#)
 - [计算后缀表达式](#)
 - [关于作者](#)
 - [项目结构](#)
-

功能介绍

- ☒ 点击按钮输入表达式
- ☒ 运算符包括 (+、-、*、/、(、))
- ☒ 通过后缀表达式求解表达式的值
- ☒ 显示计算结果
- ☒ 输入表达式过长时动态切换字体大小
- ☒ 连续计算
- ☐

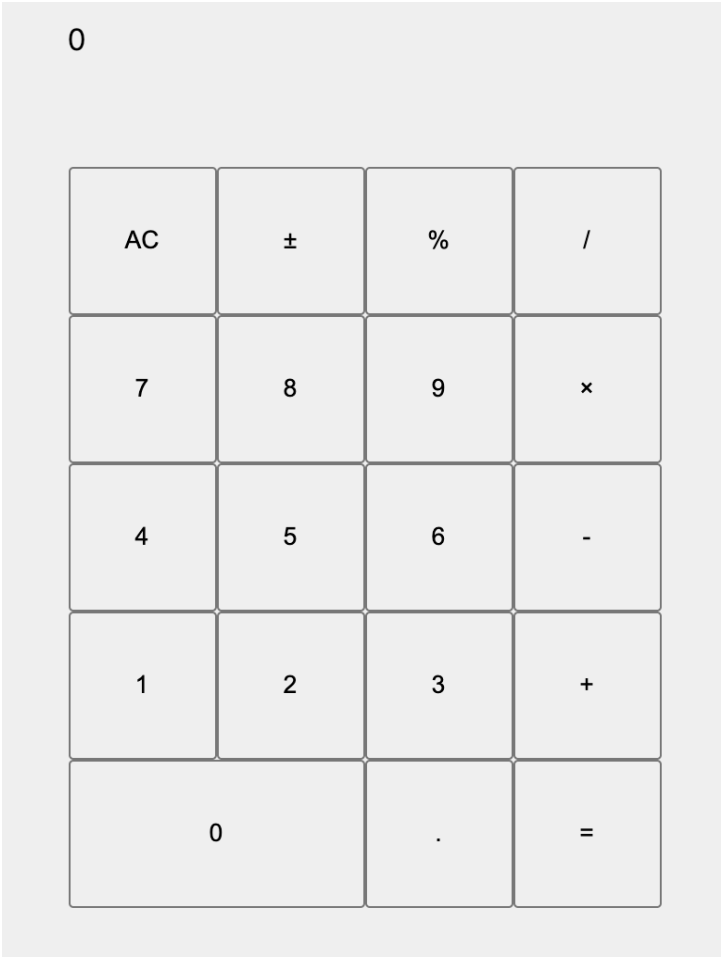
开发环境

- 操作系统
 - 开发环境: macOS Catalina 10.15.4
- IDE: Visual Studio Code 1.45.1
- 开发语言:
 - HTML5
 - CSS3
 - JavaScript

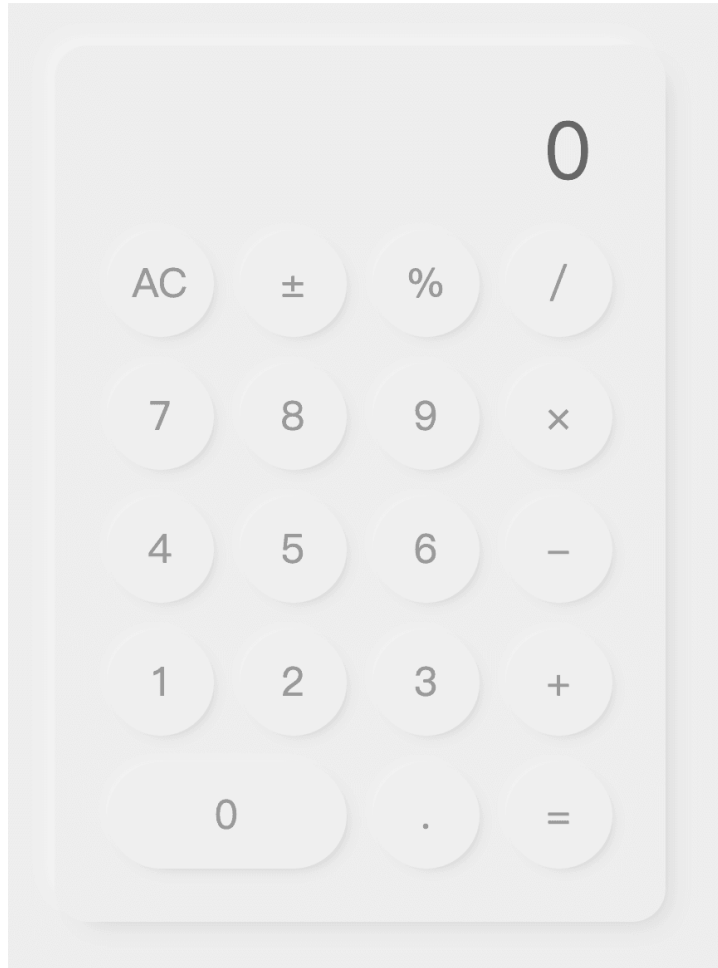
实现效果

图形化界面

- 基础grid布局

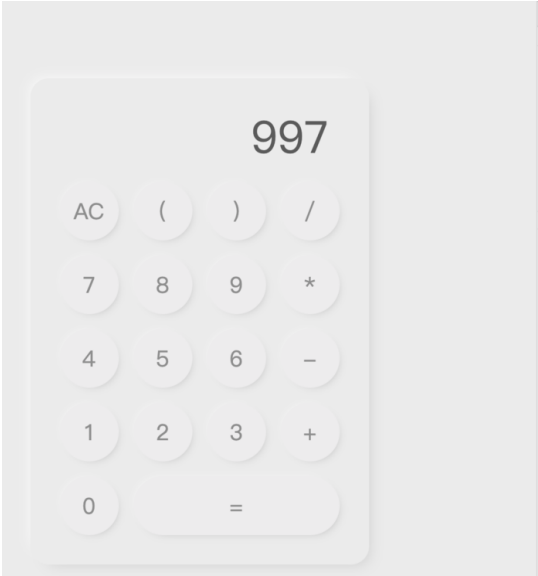


- 拟态风格计算器



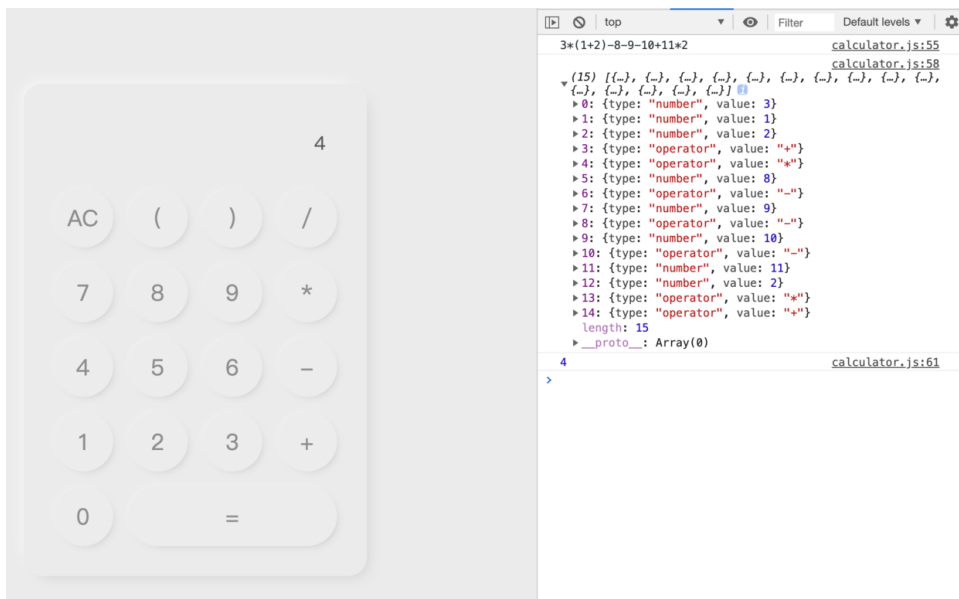
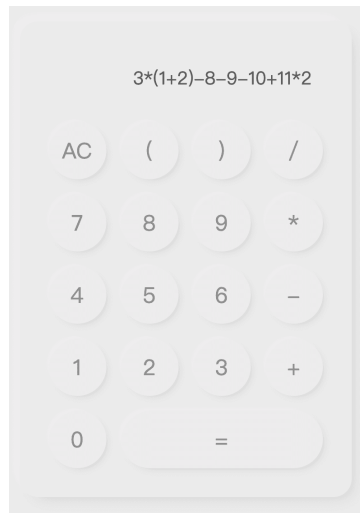
计算演示

- 普通表达式

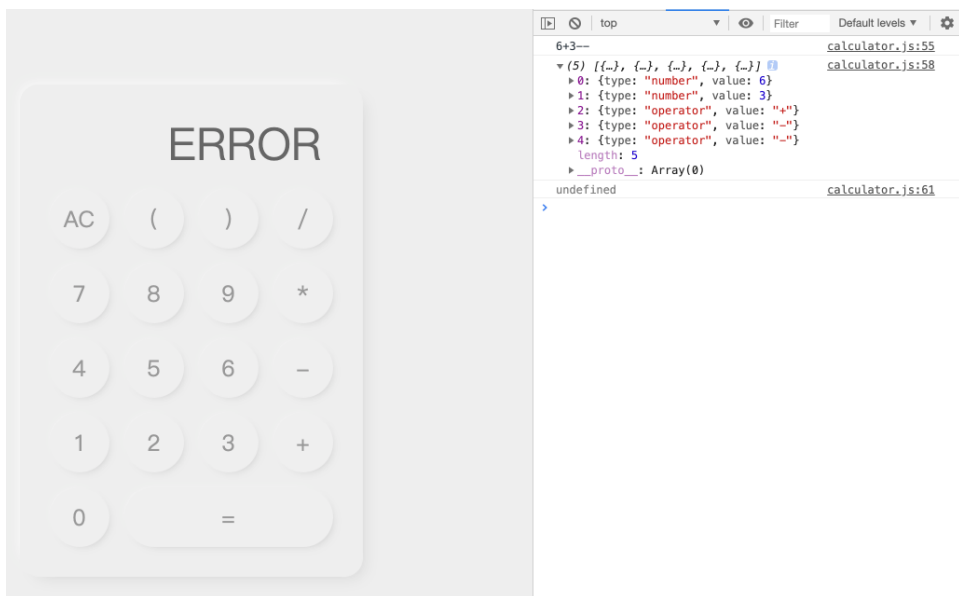


```
996+3*2-5
calculator.js:55
calculator.js:58
▼ (7) [(-), (-), (-), (-), (-), (-), (-)]
  ▶ 0: {type: "number", value: 996}
  ▶ 1: {type: "number", value: 3}
  ▶ 2: {type: "number", value: 2}
  ▶ 3: {type: "operator", value: "+"}
  ▶ 4: {type: "operator", value: "*"}
  ▶ 5: {type: "number", value: 5}
  ▶ 6: {type: "operator", value: "-"}
  length: 7
  ▶ __proto__: Array(0)
997
calculator.js:61
>
```

- 带括号的表达式



- 错误表达式



核心实现

界面

```
<div class="calculator">
  <div id="result" class="result" style="grid-area: result;">
    0
  </div>
  <button id="ac" style="grid-area: ac;">AC</button>
  <button style="grid-area: left-bracket;">(</button>
  <button style="grid-area: right-bracket;">)</button>
  <button style="grid-area: add;">+</button>
  <button style="grid-area: subtract;">-</button>
  <button style="grid-area: multiply;">*</button>
  <button style="grid-area: divide;">/</button>
  <button id="equal" style="grid-area: equal;">=</button>

  <button style="grid-area: number-1;">1</button>
  <button style="grid-area: number-2;">2</button>
  <button style="grid-area: number-3;">3</button>
  <button style="grid-area: number-4;">4</button>
  <button style="grid-area: number-5;">5</button>
  <button style="grid-area: number-6;">6</button>
  <button style="grid-area: number-7;">7</button>
  <button style="grid-area: number-8;">8</button>
  <button style="grid-area: number-9;">9</button>
  <button style="grid-area: number-0;">0</button>
</div>
```

```
.calculator {
  --button-width: 80px;
  --button-height: 80px;
  display: grid;
  grid-template-areas:
    "result result result result"
    "ac left-bracket right-bracket divide"
    "number-7 number-8 number-9 multiply"
    "number-4 number-5 number-6 subtract"
    "number-1 number-2 number-3 add"
    "number-0 equal equal equal";
  grid-template-columns: repeat(4, var(--button-width));
  grid-template-rows: repeat(6, var(--button-height));

  box-shadow: -8px -8px 16px -10px rgba(255, 255, 255, 1), 8px 8px 16px
-10px rgba(0, 0, 0, .15);
  padding: 24px;
  border-radius: 20px;
}
```

语法糖

```
function $(id){
    return document.getElementById(id);
}
```

```
document.querySelectorAll("button:not(#ac):not(#equal)").forEach(function(elem
, index){
    elem.onclick = function(){
        appendChar(elem.innerHTML);
    };
});
```

转换为后缀表达式

```
function toPostfix(exp){
    let op_stack = [];
    _current = 0;

    exp += "#";
    op_stack.push("#");

    let postfix_exp = [];

    let elem = getNextContent(exp);
    while(op_stack.length !== 0) {
        let ch = elem.value
        if(elem.type === "number") {
            postfix_exp.push(elem);
            elem = getNextContent(exp);
        } else if(elem.type === "operator"){
            let topch = op_stack[op_stack.length-1];
            if(isp(topch) < icp(ch)) {
                op_stack.push(ch);
                elem = getNextContent(exp);
            } else if (isp(topch) > icp(ch)) {
                postfix_exp.push({
                    type: "operator",
                    value: op_stack.pop()
                });
            } else {
                if(op_stack[op_stack.length-1] === '(') {
                    elem = getNextContent(exp);
                }
                op_stack.pop();
            }
        }
    }
}
```

```
}

return postfix_exp;
}
```

计算后缀表达式

```
function calculatePostfixExpression(postfix_exp){
    let num_stack = [];
    postfix_exp.forEach(function(elem){
        if(elem.type === "number"){
            num_stack.push(elem.value);
        } else {
            let right = num_stack.pop();
            let left = num_stack.pop();
            if(isNaN(parseInt(right)) || isNaN(parseInt(left))){
                return "ERROR";
            }

            let buf = null;
            try {
                buf = parseInt(eval(left + elem.value + right));
            } catch (e) {
                return "ERROR";
            } finally {
                num_stack.push(buf);
            }
        }
    });
    return num_stack.pop();
}
```

关于作者

Item	Info
Name	张喆
ID	1754060
Academy	软件学院
Course Name	2020人工智能微专业 - 数据结构
Email	dbzdbz@tongji.edu.cn

项目结构

```
.
├── README.md
├── calculator.html
├── doc
│   └── 图形界面计算器说明文档.pdf
└── static
    ├── css
    │   └── calculator.css
    └── js
        ├── calculator.js
        ├── stack.js
        └── util.js
```