实验 04：页面交互

一、实验目的

1、理解 JavaScript 操作页面的原理；
2、掌握 JavaScript 选择器；
3、掌握 JavaScript 事件；
4、掌握 JavaScript DOM 操作。

二、实验学时

2 学时

三、实验类型

验证性

四、实验需求

1、硬件

每人配备计算机 1 台，建议优先使用个人计算机开展实验。

2、软件

安装 Visual Studio Code，以及 Edge 浏览器。

3、网络

本地主机能够访问互联网和实验中心网络。

4、工具

无。

五、实验任务

1、完成弹出框示例。
2、完成表单验证示例；
3、完成轮播图示例。
4、完成倒计时实战。
5、完成九九乘法表。
6、完成菜单展开与收缩实战。

六、实验内容及步骤

1、弹出框示例

```
<!DOCTYPE html>
<html lang="en">
```

```
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>自定义弹出框</title>
    <style>
        /* 弹出框背景样式 */
        .custom-popup-overlay {
            display: none;
            position: fixed;
            top: 0;
            left: 0;
            width: 100%;
            height: 100%;
            background-color: rgba(0, 0, 0, 0.5);
            justify-content: center;
            align-items: center;
            animation: fadeIn 0.3s ease;
        }

        /* 弹出框内容样式 */
        .custom-popup {
            background-color: white;
            padding: 20px;
            border-radius: 8px;
            box-shadow: 0 4px 8px rgba(0, 0, 0, 0.2);
            max-width: 400px;
            animation: slideUp 0.3s ease;
        }

        /* 弹出框标题样式 */
        .custom-popup-title {
            font-size: 20px;
            font-weight: bold;
            margin-bottom: 10px;
        }

        /* 弹出框消息样式 */
        .custom-popup-message {
            margin-bottom: 20px;
        }

        /* 弹出框按钮容器样式 */
        .custom-popup-buttons {
            display: flex;
            justify-content: flex-end;
        }
```

```
        /* 弹出框按钮样式 */
        .custom-popup-button {
            padding: 8px 16px;
            margin-left: 10px;
            border: none;
            border-radius: 4px;
            cursor: pointer;
        }

        /* 确认按钮样式 */
        .custom-popup-button.confirm {
            background-color: #007BFF;
            color: white;
        }

        /* 取消按钮样式 */
        .custom-popup-button.cancel {
            background-color: #ccc;
        }

        /* 淡入动画 */
        @keyframes fadeIn {
            from {
                opacity: 0;
            }
            to {
                opacity: 1;
            }
        }

        /* 上滑动画 */
        @keyframes slideUp {
            from {
                transform: translateY(20px);
                opacity: 0;
            }
            to {
                transform: translateY(0);
                opacity: 1;
            }
        }
    </style>
</head>

<body>
    <button id="open-popup">打开弹出框</button>

    <!-- 自定义弹出框 -->
    <div class="custom-popup-overlay" id="custom-popup">
```

```
        <div class="custom-popup">
            <div class="custom-popup-title">提示</div>
            <div class="custom-popup-message">这是一个自定义弹出
框的示例。</div>
            <div class="custom-popup-buttons">
                <button class="custom-popup-button cancel" id="ca
ncel-button">取消</button>
                <button class="custom-popup-button confirm" id="c
onfirm-button">确认</button>
            </div>
        </div>
    </div>

    <script>
        // 获取打开弹出框的按钮
        const openPopupButton = document.getElementById('open-p
opup');
        // 获取弹出框元素
        const popup = document.getElementById('custom-popup');
        // 获取取消按钮
        const cancelButton = document.getElementById('cancel-butto
n');
        // 获取确认按钮
        const confirmButton = document.getElementById('confirm-bu
tton');

        // 打开弹出框函数
        function openPopup() {
            popup.style.display = 'flex';
        }

        // 关闭弹出框函数
        function closePopup() {
            popup.style.display = 'none';
        }

        // 为打开按钮添加点击事件监听器
        openPopupButton.addEventListener('click', openPopup);
        // 为取消按钮添加点击事件监听器
        cancelButton.addEventListener('click', closePopup);
        // 为确认按钮添加点击事件监听器
        confirmButton.addEventListener('click', function () {
            // 这里可以添加确认按钮点击后的具体逻辑
            alert('你点击了确认');
            closePopup();
        });

        // 点击弹出框背景关闭弹出框
        window.addEventListener('click', function (event) {
```

```
            if (event.target === popup) {
                closePopup();
            }
        });
    </script>
</body>

</html>
```

2、表单验证示例

```
<!DOCTYPE html>
<html lang="en">

<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>用户添加表单校验</title>
    <style>
        body {
            font-family: Arial, sans-serif;
        }

        .error {
            color: red;
            font-size: 0.9em;
        }
    </style>
</head>

<body>
    <form id="userForm">
        <label for="username">用户名:</label>
        <input type="text" id="username" required>
        <div class="error" id="usernameError"></div>
        <br>
        <label for="email">邮箱:</label>
        <input type="email" id="email" required>
        <div class="error" id="emailError"></div>
        <br>
        <label for="phone">手机号:</label>
        <input type="text" id="phone" required>
        <div class="error" id="phoneError"></div>
        <br>
        <label for="password">密码:</label>
        <input type="password" id="password" required>
        <div class="error" id="passwordError"></div>
        <br>
        <label for="repeatPassword">重复密码:</label>
```

```
        <input type="password" id="repeatPassword" required>
        <div class="error" id="repeatPasswordError"></div>
        <br>
        <label for="idCard">身份证号:</label>
        <input type="text" id="idCard" required>
        <div class="error" id="idCardError"></div>
        <br>
        <input type="submit" value="提交">
    </form>

    <script>
        const form = document.getElementById('userForm');
        const username = document.getElementById('username');
        const email = document.getElementById('email');
        const phone = document.getElementById('phone');
        const password = document.getElementById('password');
        const repeatPassword = document.getElementById('repeatPas
sword');
        const idCard = document.getElementById('idCard');

        const usernameError = document.getElementById('usernameE
rror');
        const emailError = document.getElementById('emailError');
        const phoneError = document.getElementById('phoneError');
        const passwordError = document.getElementById('passwordE
rror');
        const repeatPasswordError = document.getElementById('repe
atPasswordError');
        const idCardError = document.getElementById('idCardError');

        // 实时验证函数
        function validateInput(input, errorElement, regex, message) {
            const value = input.value;
            if (!regex.test(value)) {
                errorElement.textContent = message;
            } else {
                errorElement.textContent = '';
            }
        }

        // 用户名验证
        username.addEventListener('input', function () {
            const usernameRegex = /^[a-zA-Z0-9]{3,20}$/;
            const message = '用户名必须由 3 - 20 位字母或数字组成';
            validateInput(username, usernameError, usernameRegex,
message);
        });

        // 邮箱验证
```

```
        email.addEventListener('input', function () {
            const emailRegex = /^[^\s@]+@[^\s@]+\.[^\s@]+$/;
            const message = '请输入有效的邮箱地址';
            validateInput(email, emailError, emailRegex, message);
        });

        // 手机号验证
        phone.addEventListener('input', function () {
            const phoneRegex = /^1[3-9]\d{9}$/;
            const message = '请输入有效的 11 位手机号';
            validateInput(phone, phoneError, phoneRegex, message);
        });

        // 密码验证
        password.addEventListener('input', function () {
            const passwordRegex = /^(?=.*[a-z])(?=.*[A-Z])(?=.*\d)[a-
zA-Z\d]{8,}$/;
            const message = '密码至少 8 位，包含至少一个大写字母、一
个小写字母和一个数字';
            validateInput(password, passwordError, passwordRegex,
message);
            // 当密码输入框内容改变时，同时检查重复密码
            if (repeatPassword.value) {
                validateRepeatPassword();
            }
        });

        // 重复密码验证
        repeatPassword.addEventListener('input', validateRepeatPassw
ord);

        function validateRepeatPassword() {
            if (password.value!== repeatPassword.value) {
                repeatPasswordError.textContent = '两次输入的密码不
一致';
            } else {
                repeatPasswordError.textContent = '';
            }
        }

        // 身份证号验证
        idCard.addEventListener('input', function () {
            const idCardRegex = /(^\d{15}$)|(^\d{18}$)|(^\d{17}(\d|X|
x)$)/;
            const message = '请输入有效的身份证号';
            validateInput(idCard, idCardError, idCardRegex, message);
        });

        // 表单提交验证
```

```
        form.addEventListener('submit', function (event) {
            const validations = [
                { input: username, regex: /^[a-zA-Z0-9]{3,20}$/, mess
age: '用户名必须由 3 - 20 位字母或数字组成', errorElement: usernameE
rror },
                { input: email, regex: /^[^\s@]+@[^\s@]+\.[^\s@]+
$/, message: '请输入有效的邮箱地址', errorElement: emailError },
                { input: phone, regex: /^1[3-9]\d{9}$/, message: '请
输入有效的 11 位手机号', errorElement: phoneError },
                { input: password, regex: /^(?=.*[a-z])(?=.*[A-Z])(?=.*\
d)[a-zA-Z\d]{8,}$/, message: '密码至少 8 位，包含至少一个大写字母、一个
小写字母和一个数字', errorElement: passwordError },
                { input: idCard, regex: /(^\d{15}$)|(^\d{18}$)|(^\d{17}
(\d|X|x)$)/, message: '请输入有效的身份证号', errorElement: idCardError
 }
            ];

            let hasError = false;
            validations.forEach(({ input, regex, message, errorElement
 }) => {
                if (!regex.test(input.value)) {
                    errorElement.textContent = message;
                    hasError = true;
                } else {
                    errorElement.textContent = '';
                }
            });

            // 检查重复密码
            if (password.value!== repeatPassword.value) {
                repeatPasswordError.textContent = '两次输入的密码不
一致';
                hasError = true;
            }

            if (hasError) {
                event.preventDefault();
            }
        });
    </script>
</body>

</html>
```

3、轮播图示例

```
<!DOCTYPE html>
<html lang="en">

<head>
```

```
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>多功能轮播图</title>
    <style>
        /* 轮播图容器样式 */
        .slider {
            width: 500px;
            height: 300px;
            margin: 0 auto;
            position: relative;
            overflow: hidden;
        }

        /* 图片样式 */
        .slider img {
            width: 100%;
            height: 100%;
            position: absolute;
            opacity: 0;
            transition: opacity 1s;
        }

        /* 激活图片样式 */
        .slider img.active {
            opacity: 1;
        }

        /* 切换按钮样式 */
        .slider button {
            position: absolute;
            top: 50%;
            transform: translateY(-50%);
            background-color: rgba(0, 0, 0, 0.5);
            color: white;
            border: none;
            padding: 10px;
            cursor: pointer;
        }

        /* 上一页按钮位置 */
        .slider button.prev {
            left: 10px;
        }

        /* 下一页按钮位置 */
        .slider button.next {
            right: 10px;
        }
```

```
        /* 指示器容器样式 */
        .indicators {
            position: absolute;
            bottom: 10px;
            left: 50%;
            transform: translateX(-50%);
            display: flex;
        }

        /* 指示器样式 */
        .indicator {
            width: 10px;
            height: 10px;
            background-color: rgba(255, 255, 255, 0.5);
            border-radius: 50%;
            margin: 0 5px;
            cursor: pointer;
        }

        /* 激活指示器样式 */
        .indicator.active {
            background-color: white;
        }
    </style>
</head>

<body>
    <!-- 轮播图容器 -->
    <div class="slider">
        <!-- 轮播图片 -->
        <img src="https://via.placeholder.com/500x300?text=Slide+1"
 alt="Slide 1">
        <img src="https://via.placeholder.com/500x300?text=Slide+2"
 alt="Slide 2">
        <img src="https://via.placeholder.com/500x300?text=Slide+3"
 alt="Slide 3">
        <!-- 上一页按钮 -->
        <button class="prev">&lt;</button>
        <!-- 下一页按钮 -->
        <button class="next">&gt;</button>
        <!-- 指示器容器 -->
        <div class="indicators"></div>
    </div>

    <script>
        // 获取所有轮播图片
        const slides = document.querySelectorAll('.slider img');
        // 获取上一页按钮
```

```
        const prevButton = document.querySelector('.slider button.prev');
        // 获取下一页按钮
        const nextButton = document.querySelector('.slider button.next');
        // 获取指示器容器
        const indicatorsContainer = document.querySelector('.indicators');
        // 当前显示的幻灯片索引
        let currentSlide = 0;

        // 创建指示器
        slides.forEach((_, index) => {
            const indicator = document.createElement('div');
            indicator.classList.add('indicator');
            if (index === 0) {
                indicator.classList.add('active');
            }
            indicator.addEventListener('click', () => {
                showSlide(index);
            });
            indicatorsContainer.appendChild(indicator);
        });

        // 显示指定索引的幻灯片
        function showSlide(index) {
            slides.forEach((slide) => {
                slide.classList.remove('active');
            });
            slides[index].classList.add('active');

            const indicators = document.querySelectorAll('.indicator');
            indicators.forEach((indicator) => {
                indicator.classList.remove('active');
            });
            indicators[index].classList.add('active');

            currentSlide = index;
        }

        // 切换到上一张幻灯片
        function prevSlide() {
            currentSlide = (currentSlide - 1 + slides.length) % slides.length;
            showSlide(currentSlide);
        }

        // 切换到下一张幻灯片
        function nextSlide() {
```

```
            currentSlide = (currentSlide + 1) % slides.length;
            showSlide(currentSlide);
        }

        // 为上一页按钮添加点击事件监听器
        prevButton.addEventListener('click', prevSlide);
        // 为下一页按钮添加点击事件监听器
        nextButton.addEventListener('click', nextSlide);

        // 自动轮播
        setInterval(nextSlide, 3000);

        // 初始显示第一张幻灯片
        showSlide(currentSlide);
    </script>
</body>

</html>
```

4、倒计时实战

实现一个功能完备的倒计时器，用户能够根据自身需求设定倒计时的时长，在倒计时过程中，页面会实时更新显示剩余时间。当倒计时结束时，系统会通过特定的提示方式告知用户。

- 设计一个简洁美观的页面布局，包含一个输入框，用于用户输入倒计时的时长（单位：秒）。
- 提供一个 “开始” 按钮，点击该按钮后开始倒计时。
- 有一个专门的显示区域，用于实时展示倒计时的剩余时间，格式为 “XX:XX”（分：秒）。
- 当倒计时结束时，显示区域可变为特定颜色（如红色），以突出显示倒计时结束状态。

5、九九乘法表实战

在网页上以美观、规范的格式展示九九乘法表。乘法表应按照传统的数学格式排列，每个乘法算式的结果正确显示，并且可以通过一定的样式设置增强视觉效果。

- 设计一个整齐的表格来展示九九乘法表，表格的每一行和每一列对应乘法表中的一个算式。
- 为表格添加合适的边框和背景颜色，使乘法表更加清晰易读。
- 可以对表格的表头和不同行、列的样式进行区分，如表头加粗、交替行背景颜色不同等。

6、菜单展开与收缩实战

创建一个多级菜单系统，菜单具有展开和收缩的功能。用户可以通过点击菜单的父级项来展开或收缩其子菜单，方便浏览和操作。

- 设计一个美观的菜单界面，菜单采用树形结构展示，父级菜单项后面有一个箭头图标，用于指示菜单的展开和收缩状态。
- 菜单的样式要与整体页面风格相协调，可使用 CSS 进行样式设置，如菜单的背景颜色、文字颜色、边框等。

- 当鼠标悬停在菜单项上时，菜单项的样式可以发生变化（如背景颜色变深），以增强交互性。

七、实验考核

本实验考核采用【实验随堂查】方式开展。

每个实验完成后，在实验课上通过现场演示的方式向实验指导教师进行汇报，并完成现场问答交流。

每个实验考核满分 100 分，其中实验成果汇报 60 分，现场提问交流 40 分。

实验考核流程：

（1）学生演示汇报实验内容的完成情况，实验指导老师现场打分。

（2）指导老师结合实验内容进行提问，每位学生提问 2-3 个问题，根据回答的情况现场打分。

（3）实验考核结束后，进行公布成绩。

八、创作说明

本实验指导书由河南中医药大学信息技术学院（智能医疗行业学院）智能医疗教研室与郑州泰来信息科技有限公司联合创作。

作者：冯顺磊（郑州泰来信息科技有限公司）

审核：王昂（河南中医药大学信息技术学院）