

TRƯỜNG ĐẠI HỌC MỞ THÀNH PHỐ HỒ CHÍ MINH
KHOA CÔNG NGHỆ THÔNG TIN



BÁO CÁO BÀI TẬP LỚN
ỨNG DỤNG WEB (THIẾT KẾ WEB)

ĐỀ TÀI

Thiết kế trò chơi dò mìn – Game MineSweeper

Giảng viên bộ môn: Nguyễn Thị Mai Trang

Sinh viên thực hiện: Hoàng Phi Hùng

Lớp: DH23CS01

Mã số sinh viên: 2351010082

Thứ tư, ngày 14 tháng 8 năm 2024.

Mục lục

Lời nói đầu.....	3
I. PHẦN THÔNG TIN.....	4
II. PHẦN GIAO DIỆN	5
1. Giao diện trang web và các phần tử trong <body>	5
2. Các phần tử trong game	5
III. PHẦN MÃ NGUỒN.....	8
1. HTML và CSS để xử lý giao diện phần tử game.....	8
2. JavaScript xử lý game và chức năng trên game.....	12
a. Các biến cần có.....	12
b. Chức năng 1: Chọn cấp độ chơi, set size bảng và số lượng bom	13
c. Chức năng 2: Khởi tạo bảng trò chơi.....	14
d. Chức năng 3: Mở ô khi ô được click chuột trái	17
e. Chức năng 4: Gắn cờ và kiểm tra xem người chơi có thắng hay không	19
f. Chức năng 5: Đồng hồ tính giờ.....	20
III. KIỂM THỦ VÀ KẾT LUẬN	22
Tài liệu tham khảo	23

Lời nói đầu

Thế giới chúng ta đang bước vào thời kỳ Cách mạng 4.0, thời đại của khoa học kỹ thuật gắn liền với Internet. Internet đã thay đổi thói quen của chúng ta rất nhiều thứ về mọi mặt trong cuộc sống như đọc tin tức, giải trí, học tập và vô số khía cạnh khác. Nhắc đến Internet ta không thể không nhắc đến các website và các ngôn ngữ liên quan đến việc thiết kế website như HTML, CSS hay JavaScript. Thiết kế một website không chỉ đơn giản về mặt hình thức, trình bày các nội dung đến với tất cả mọi người trên thế giới mà còn có thể lập trình các hành động, sự kiện hoặc thậm chí là viết ra một ứng dụng chạy trên website. Bằng việc ứng dụng ngôn ngữ lập trình JavaScript cùng với ngôn ngữ đánh dấu siêu văn bản HTML và ngôn ngữ thiết kế CSS, chúng ta có thể viết ra mọi ứng dụng mong muốn chạy trên nền tảng web.

Nhắc đến ứng dụng thì có rất nhiều các loại ứng dụng khác nhau, như ứng dụng công cụ, ứng dụng hệ thống, ứng dụng trình duyệt,... Trong đó có game, là một ứng dụng giải trí thông qua các trò chơi, đem lại sự thư giãn và góp phần giải tỏa căng thẳng cho người chơi. Game có nhiều thể loại như game giải đố, dạng bảng, game AAA, game MOBA,... cùng vô số hình thức khác nhưng sức hút của các game cổ điển dường như vẫn không thay đổi. MineSweeper là một trong những game cổ điển đó, game “đòn đầu” trên máy tính Windows. Với luật chơi đơn giản nhưng cũng có phần thử thách bộ não của người chơi, MineSweeper đã thu hút một lượng lớn người hâm mộ suốt khoảng thời gian phát triển của mình.

Sau khi đã nắm được các kiến thức cơ bản về HTML, CSS và JavaScript dưới sự hướng dẫn và giảng dạy tận tình của cô Nguyễn Thị Mai Trang, với mong muốn được trau dồi kiến thức và mở rộng vốn hiểu biết của mình về mảng thiết kế web cũng như ứng dụng web, em đã thực hiện bài tập lớn này. Em chọn đề tài là “**Thiết kế trò chơi dò mìn – Game MineSweeper**” và hoàn thành bài tập theo kế hoạch đã đề ra. Đây cũng là một thử thách cho bản thân em, góp phần thực hiện ước mơ trở thành lập trình viên game trong tương lai.

Em xin gửi lời cảm ơn sâu sắc đến cô **Nguyễn Thị Mai Trang** – Giảng viên giảng dạy môn **Ứng dụng Web** của em, nhờ các bài giảng và kiến thức cô truyền đạt mà em mới có thể thực hiện bài tập lớn này.

Xin chân thành cảm ơn cô!

I. PHẦN THÔNG TIN

Liên kết truy cập trang web:

Liên kết chứa mã nguồn trang web:

Ngôn ngữ sử dụng: HTML, CSS, JavaScript.

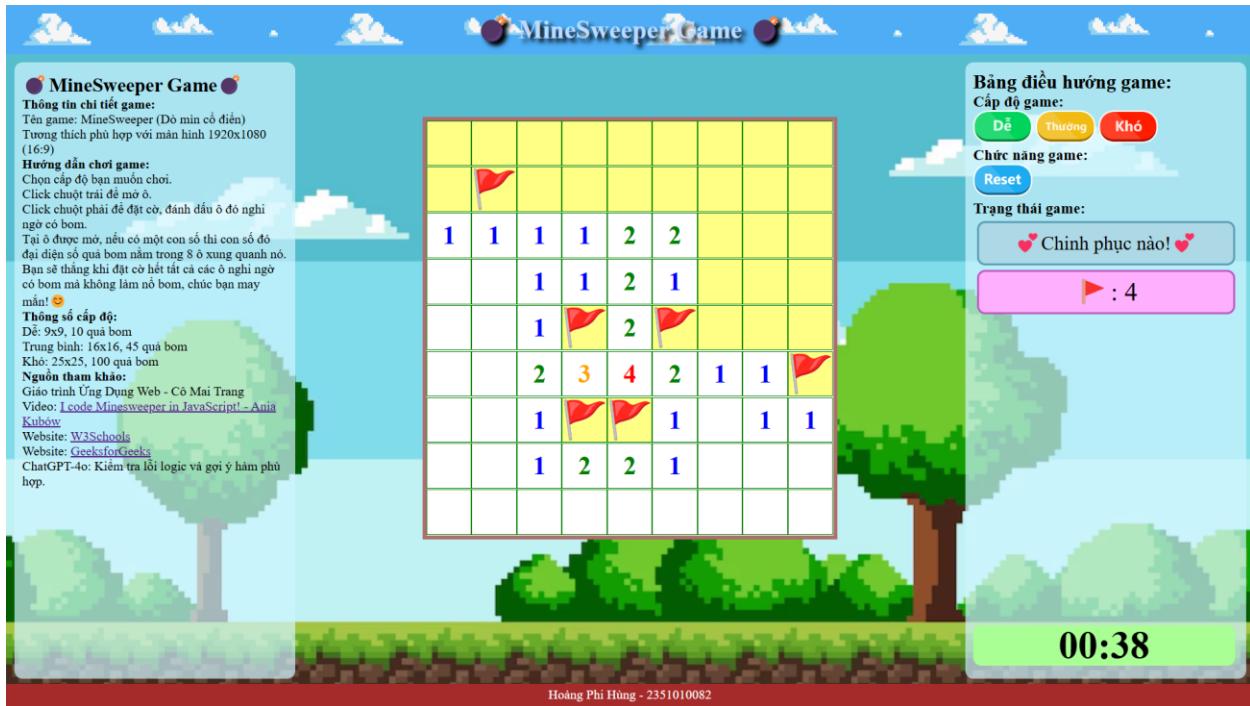
Ý tưởng trò chơi: Game dò mìn là game dạng bảng, người chơi thực hiện các thao tác trên các ô vuông chứa trong bảng bao gồm mở ô, gắn cờ. Game có các chức năng như chọn cấp độ chơi, tải lại trò chơi, đếm thời gian, thông báo cho người chơi biết trạng thái hiện tại của game.

Luật chơi:

- Người chơi sẽ click vào các ô vuông trong ma trận ô vuông game để mở ô.
- Trong ma trận này sẽ có ngẫu nhiên bom nằm rải rác khắp ma trận, nhiệm vụ của người chơi là sẽ gắn cờ vào các ô vuông chứa bom này.
- Người chơi có thể tìm ra ô chứa bom bằng cách mở các ô khác.
- Khi một ô được mở, nếu ô đó gần các quả bom thì sẽ có một con số đại diện cho số quả bom có trong 8 ô xung quanh nó, nếu không có quả bom nào thì sẽ tự động mở các ô liền kề không chứa bom hoặc chứa số.
- Người chơi sẽ chiến thắng khi đặt đủ số cờ vào các ô chứa bom.
- Người chơi sẽ thua khi click phải ô chứa bom hoặc thời gian vượt quá 60 phút.
- Tại mỗi một màn chơi đã chọn, người chơi không thể chọn lại màn chơi, trừ khi tải lại trang để reset trò chơi mới.
- Click chuột trái để mở ô.
- Click chuột phải để gắn cờ. Trong trường hợp ô đã được gắn cờ, khi click chuột phải thì cờ sẽ được thu hồi.

II. PHẦN GIAO DIỆN

1. Giao diện trang web và các phần tử trong <body>

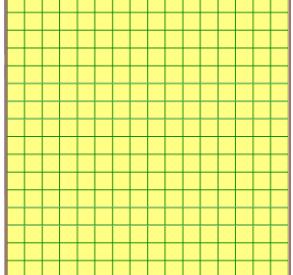


Hình 2.1.1 – Giao diện chính của website

- Header: Chứa tiêu đề của toàn bộ trang web là tên game MineSweeper
- Body: Bố cục gồm phần, mỗi phần được tổ chức theo một cột, em sử dụng flex để tổ chức phần này.
 - Phải: Chứa thông tin về game, cách chơi, thông số các cấp độ, nguồn tham khảo để xây dựng nên trò chơi.
 - Giữa: Chứa bảng trò chơi, đây là phần chính của toàn bộ trang web.
 - Trái: Chứa các nút lệnh chọn cấp độ, reset lại trò chơi và các trạng thái của game, bao gồm thông báo, số lá cờ còn lại và đồng hồ hiển thị thời gian đang chơi.
- Footer: Chứa tên và mã số sinh viên của em.

Em định dạng các thẻ <div> cùng với id bằng CSS để tổ chức giao diện như trên.

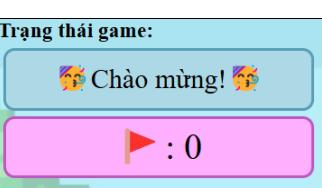
2. Các phần tử trong game

	<p>Bảng trống: Đây là bảng trống khi chưa được load các ô vuông.</p>
	<p>Bảng đã chứa ô: Đây là bảng chứa các ô sau khi đã được load, game đã bắt đầu và người chơi có thể thao tác. Các ô chưa được chọn em định dạng cho nó có nền vàng.</p>
	<p>Các ô chứa số: Mỗi một số đại diện cho số quả bom có xung quanh nó. Với mỗi số em cho màu chữ khác nhau trên nền trắng để dễ phân biệt. Riêng số 0 em cho không hiện số lên.</p>
	<p>Ô chứa bom: Ô chứa bom em định dạng cho ảnh nền của nó là một quả bom trên nền trắng.</p>
	<p>Ô được gắn cờ: Em định dạng cho ô này background hình lá cờ trên nền vàng.</p>



Hình 2.2.6: Bảng điều hướng

Bảng điều hướng game: Mỗi một nút em định dạng nó là một ảnh, khi nhấn lên thì sẽ thực thi các hàm JavaScript.



Hình 2.2.7: Bảng trạng thái

Bảng trạng thái: Mỗi một ô là một thẻ <div> với id, em dùng các sự kiện trong JavaScript để thay đổi nội dung trong 2 ô này.



Hình 2.2.8: Đồng hồ tính giờ

Đồng hồ tính giờ: Là một thẻ <div> với id, em dùng để hiển thị thời gian mà người chơi chơi được trong bao lâu, em dùng hàm JavaScript để thay đổi thời gian hiển thị mỗi giây.

II. PHẦN MÃ NGUỒN

1. HTML và CSS để xử lý giao diện phần tử game

```
<!DOCTYPE html>
<html lang="vi">
  <head>
    <meta charset="UTF-8" />
    <meta name="viewport" content="width=device-width, initial-scale=1.0" />
    <link rel="icon" type="image/x-icon" href="./GameAssets/bomb.png">
    <title>Minesweeper Game - Marcoh05P</title>
    <link rel="stylesheet" href="./style.css" />
    <script src="./script.js"></script>
  </head>
  <body>
    <header>MineSweeper Game </header>
    <div class="body">
      <div class="left">
        <h2>MineSweeper Game</h2>
        <b>Thông tin chi tiết game:</b>
        <br>Tên game: MineSweeper (Dò mìn cổ điển) <br />Tương thích phù hợp với màn hình 1920x1080 (16:9)
        <br><b>Hướng dẫn chơi game:</b>
        <br>Chọn cấp độ bạn muốn chơi.
        <br>Click chuột trái để mở ô.
        <br>Click chuột phải để đặt cờ, đánh dấu ô đó nghi ngờ có bom.
        <br>Tại ô được mở, nếu có một con số thì con số đó đại diện số quả bom nằm trong 8 ô xung quanh nó. <br>Bạn sẽ thắng khi đặt cờ hết tất cả các ô nghi ngờ có bom mà không làm nổ bom, chúc bạn may mắn! 😊
        <br><b>Thông số cấp độ:</b>
        <br>Để: 9x9, 10 quả bom
        <br>Trung bình: 16x16, 45 quả bom
        <br>Khó: 25x25, 100 quả bom
        <br><b>Nguồn tham khảo:</b>
        <br>Giáo trình Ứng Dụng Web - Cô Mai Trang
        <br>Video: <a href="https://youtu.be/jS7iB9mRvcc?feature=shared" target="_blank">I code Minesweeper in JavaScript! - Ania Kubów</a>
        <br>Website: <a href="https://www.w3schools.com/" target="_blank">W3Schools</a>
        <br>Website: <a href="https://www.geeksforgeeks.org/" target="_blank">GeeksforGeeks</a>
        <br>ChatGPT-4o: Kiểm tra lỗi logic và gợi ý hàm phù hợp.
      </div>
      <div class="game">
        <div id="board"></div>
      </div>
      <div class="right">
        <h2>Bảng điều hướng game:</h2>
        <h3>Cấp độ game:</h3>
        
        
        
        <br />
        <h3>Chức năng game:</h3>
      </div>
    </div>
  </body>
</html>
```

```


<h3>Trạng thái game:</h3>
<div id="result">👋 Chào mừng! 👋</div>
<div id="flag">▶ : <span id="flag_left">0</span></div>

    <div id ="timer"><span id = "clock">00:00</span></div>
</div>
</div>
<footer>Hoàng Phi Hùng - 2351010082</footer>
</body>
</html>

```

/*Định dạng style cho các thẻ gốc của html*/

```

* {
    margin: 0px;
    padding: 0px;
}

img {
    height: 40px;
}

img:hover {
    cursor: pointer;
}

header {
    color: rgb(191, 218, 255);
    text-shadow: 5px 5px 5px black;
    width: 100%;
    font-size: 2em;
    text-align: center;
    font-weight: bold;
    line-height: 7vh;
    height: 7vh;
    background-image: url("./GameAssets/headerbg.png");
    background-size: 25%;
}

footer {
    width: 100%;
    height: 3vh;
    line-height: 3vh;
    color: white;
    text-align: center;
    background-color: brown;
    position: fixed;
    bottom: 0;
    padding-bottom: 4px;
}

/*Định dạng style cho phần thân của web, bao gồm trái, phải và phần chứa game*/

```

```
.body {
    display: flex;
    flex-direction: row;
    background-image: url("./GameAssets/bodybg.png");
    background-repeat: repeat-x;
    background-size: cover;
    height: 90vh;
}

.left {
    width: 25%;
    border: 1px solid none;
    margin: 10px;
    padding: 10px;
    background-color: rgba(207, 245, 255, 0.7);
    border-radius: 10px;
}

.game {
    width: 50%;
    margin: 5%;
    display: flex;
    justify-content: center;
    flex-wrap: wrap;
}

.right {
    width: 25%;
    border: 1px solid none;
    margin: 10px;
    padding: 10px;
    background-color: rgba(207, 245, 255, 0.7);
    border-radius: 10px;
}

/*Định dạng style cho các phần tử của game*/
#board /*Khung chứa các ô*/
{
    width: 510px;
    height: 510px;
    background-color: white;
    border: 5px solid rgba(146, 60, 60, 0.74);

    display: flex;
    flex-wrap: wrap;
}

#board div /*Các ô vuông*/
{
    width: 100%;
    height: 100%;
    border: 1px solid green;
    font-size: 2em;
    font-weight: bold;
}
```

```

    text-align: center;
    line-height: 100%;
    margin: 0px;
}

#flag, #result{
    margin: 5px;;
}

#flag /*Khung chứa số lượng cờ hiện có*/
{
    background-color: rgb(254, 176, 255);
    height: 50px;
    width: 96%;
    font-size: 2em;
    line-height: 50px;
    text-align: center;
    border-radius: 10px;
    border: 3px solid rgb(189, 89, 190);
}

#result /*Khung trạng thái thông báo*/
{
    background-color: lightblue;
    height: 50px;
    width: 96%;
    font-size: 1.5em;
    line-height: 50px;
    text-align: center;
    border-radius: 10px;
    border: 3px solid rgb(89, 158, 180);
}

#timer /*Khung thời gian hiện hành*/
{
    background-color: rgb(170, 255, 148);
    height: 50px;
    width: 21%;
    font-size: 3em;
    font-weight: bold;
    line-height: 50px;
    text-align: center;
    border-radius: 10px;
    position: absolute;
    bottom: 6%;
}

.valid,
.bomb /*Ô trống và ô chứa bom*/
{
    background-color: rgb(255, 255, 133);
}

.boom /*Ô chứa bom đã được mở*/
{

```

```

background-image: url("./GameAssets/bomb.png");
background-size: 100%;
}

.checked /*Ô đã được mở*/
{
background-color: white;
}

.flaged /*Ô đã được gắn cờ*/
{
background-image: url("./GameAssets/flag.png");
background-size: 100%;
}

/*Định dạng màu sắc cho các chữ số gần bom*/
.one {
color: blue;
}

.two {
color: green;
}

.three {
color: orange;
}

.four {
color: red;
}

.five {
color: purple;
}

.six {
color: aqua;
}

.seven {
color: chocolate;
}

.eight {
color: darkred;
}

```

2. JavaScript xử lý game và chức năng trên game

a. Các biến cần có

```

let size = 0; //Biến chứa độ lớn của ô game (level)
let bomb = 0; //Biến chứa số lượng bomb
let flags = 0; //Biến chứa số lượng cờ

```

```

let trangthai = 0; //Biến chứa trạng thái level đã chọn hay chưa (Ngăn chặn
người chơi chọn level khác)
let squares = []; //Mảng chứa tất cả các ô, dùng để thao tác
var gameOver = false; //Biến trạng thái game kết thúc
var intervalId; //Đối tượng trạng thái interval
var time; //Biến thời gian chạy

```

b. Chức năng 1: Chọn cấp độ chơi, set size bảng và số lượng bom

Với chức năng này, em tạo ra 2 nhóm hàm chính là set cấp độ và chỉnh sửa style

- Set cấp độ: Bao gồm cập nhật thông tin về size bảng, số bom, gọi hàm chỉnh sửa style và hàm tạo bảng. Trong đó em cập nhật lại biến “trangthai” để tránh xung đột khi người chơi đang chơi lại click vào nút cấp độ khác.
- Chỉnh sửa style: Chỉnh sửa lại các thông số về ô vuông cho đẹp và khớp với giao diện.

Mã nguồn:

```

function createEasyLevel() //Khởi tạo bảng chế độ dễ
{
    if (trangthai == 0) {
        size = 9;
        bomb = 10;
        createBoard();
        changeEasyLevelStyle();
        trangthai = 1;
    }
}

function changeEasyLevelStyle() //Thay đổi style của ô board cho đẹp, có tham
khảo từ ChatGPT
{
    document.getElementById("board").style.width = "501px";
    var divs = document.querySelectorAll("#board div");
    divs.forEach(function (div)
    {
        div.style.width = "54px";
        div.style.height = "54px";
        div.style.fontSize = "2em";
        div.style.lineHeight = "54px";
    });
}

function createMediumLevel() //Khởi tạo bảng ở chế độ thường
{
    if (trangthai == 0) {
        size = 16;
        bomb = 45;
        createBoard();
        changeMediumLevelStyle();
        trangthai = 1;
    }
}

```

```

function changeMediumLevelStyle()
{
    document.getElementById("board").style.width = "507px";
    var divs = document.querySelectorAll("#board div");
    divs.forEach(function (div)
    {
        div.style.width = "30px";
        div.style.height = "30px";
        div.style.fontSize = "1.5em";
        div.style.lineHeight = "30px";
    });
}

function createHardLevel() //Khởi tạo bảng ở chế độ khó
{
    if (trangthai == 0) {
        size = 25;
        bomb = 100;
        createBoard();
        changeHardLevelStyle();
        trangthai = 1;
    }
}

function changeHardLevelStyle()
{
    document.getElementById("board").style.width = "490px";
    var divs = document.querySelectorAll("#board div");
    divs.forEach(function (div)
    {
        div.style.width = "18px";
        div.style.height = "18px";
        div.style.fontSize = "1em";
        div.style.lineHeight = "18px";
    });
}

```

c. Chức năng 2: Khởi tạo bảng trò chơi

Chức năng 2.1: Tạo ma trận bom cho bảng

Ở chức năng này, em sử dụng mảng 1 chiều để tạo ra mảng tham chiếu vị trí bom cho bảng. Tạo ra 2 mảng và gộp 2 mảng lại ngẫu nhiên để tạo thành mảng này.

Mã nguồn:

```
//Tạo mảng 1 chiều lưu giá trị để tham chiếu vị trí đặt bom vào
const bombArray = Array(bomb).fill("bomb");
const emptyArray = Array(size * size - bomb).fill("valid");
const gameArray = bombArray.concat(emptyArray).sort(() => Math.random() -
0.5);
```

Sau khi đã có mảng tham chiếu vị trí bom, trong vòng lặp để xử lý trên từng ô em thêm các giá trị tương ứng các phần tử của mảng này để ghi vào class của 1 đối tượng square, sau đó em thôi đối tượng này vào mảng đối tượng squares dùng để lưu ma trận bom.

Mã nguồn:

```
//Tạo mảng 1 chiều chứa ma trận bom
const square = document.createElement("div");
square.id = i;
square.classList.add(gameArray[i]);
document.getElementById("board").appendChild(square);
squares.push(square);
```

Chức năng 2.2: Xử lý sự kiện khi click chuột trái hoặc chuột phải vào ô

Em gọi các hàm để xử lý khi click chuột trái hoặc chuột phải. Click chuột trái gọi hàm click() để mở ô và addFlag() để gắn cờ.

Mã nguồn:

```
//Xử lý sự kiện khi click chuột trái
square.addEventListener("click", function () {
    click(square);
});

//Xử lý sự kiện khi click chuột phải
square.addEventListener("contextmenu", function (event) {
    event.preventDefault();
    addFlag(square);
});
```

Chức năng 2.3: Đếm số bom xung quanh và ghi số bom đã đếm vào tất cả các ô trong bảng

Ý tưởng thực hiện: Tại một ô không chứa bom, ta có thể đếm các ô xung quanh nó như sau cùng với các ràng buộc

- Tạo một biến đếm để đếm bom.
- Tại một ô sẽ có nhiều nhất 8 ô xung quanh nó nếu id của nó lớn hơn size và bé hơn size * (size - 1) và sẽ ít hơn nếu nó nằm ở các cạnh của bảng.
- Để ngăn chặn việc đếm các ô xung quanh không hợp lệ, em chia 8 ô này thành 8 trường hợp khác nhau để xử lý, nếu thỏa điều kiện thì tăng biến đếm bom lên 1 đơn vị, sau khi kiểm tra các ô thì ghi số bom vào data. Gọi i là id của ô hiện tại đang kiểm tra, size là chiều dài cạnh bảng.

i-size-1	i-size	i-size+1
i-1	i	i+1
i+size-1	i+size	i+size+1

- Kiểm tra ô bên trái có id là $[i - 1]$: $i > 1$, không được là cạnh trái và tại ô này chưa bom.
- Kiểm tra ô bên phải có id là $[i + 1]$: $i < (size * size) - 1$, không được là cạnh phải và ô này chưa bom.
- Kiểm tra ô bên trên có id là $[i - size]$: $i \geq size$ và ô này chưa bom
- Kiểm tra ô bên dưới có id là $[i + size]$: $i < (size * size) - 1$ và ô này chưa bom
- Kiểm tra ô bên trên phía bên trái có id là $[i - size - 1]$: $i \geq size$, không được là cạnh trái và ô này chưa bom.
- Kiểm tra ô bên trên phía bên phải có id là $[i - size + 1]$: $i \geq size$, không được là cạnh phải và ô này chưa bom
- Kiểm tra ô bên dưới phía bên trái có id là $[i + size - 1]$: $i < size * (size - 1)$, không được là cạnh trái và ô này chưa bom.
- Kiểm tra ô bên dưới phía bên phải có id là $[i + size + 1]$: $i < size * (size - 1)$, không được là cạnh phải và ô này chưa bom

Mã nguồn:

```
//Vòng lặp đếm số bom xung quanh và ghi số bom đã đếm vào tất cả các ô trong bảng
for (let i = 0; i < squares.length; i++) {
  let dembom = 0;
  const canhTrai = (i % size === 0); //Đánh dấu ô là ô nằm ở cạnh trái
  const canhPhai = (i % size === size - 1); //Đánh dấu ô là ô nằm ở cạnh phải

  if (squares[i].classList.contains("valid")) {
    if (i > 0 && !canhTrai && squares[i - 1].classList.contains("bomb"))
      dembom++; //Kiểm tra ô bên trái
    if (i < (size * size) - 1 && !canhPhai && squares[i + 1].classList.contains("bomb"))
      dembom++; //Kiểm tra ô bên phải
    if (i >= size && squares[i - size].classList.contains("bomb"))
      dembom++; //Kiểm tra ô bên trên
    if (i < size * (size - 1) && squares[i + size].classList.contains("bomb"))
      dembom++; //Kiểm tra ô bên dưới
    if (i >= size && !canhTrai && squares[i - 1 - size].classList.contains("bomb"))
      dembom++; //Kiểm tra ô trên trái
    if (i >= size && !canhPhai && squares[i + 1 - size].classList.contains("bomb"))
      dembom++; //Kiểm tra ô trên phải
    if (i < size * (size - 1) && !canhTrai && squares[i - 1 + size].classList.contains("bomb"))
      dembom++; //Kiểm tra ô dưới trái
    if (i < size * (size - 1) && !canhPhai && squares[i + 1 + size].classList.contains("bomb"))
      dembom++; //Kiểm tra ô dưới phải
    squares[i].setAttribute("data", dembom); //Ghi số bom vào ô
  }
}
```

d. Chức năng 3: Mở ô khi ô được click chuột trái

Ý tưởng thực hiện: Khi một ô click chuột trái vào sẽ có các sự kiện xảy ra và thực hiện nó, các trường hợp xảy ra và ràng buộc như sau.

- Không làm gì cả nếu như game đã kết thúc, ô đã được mở hoặc ô đã được gán cờ.
- Nếu click phải bom thì dừng game, thông báo cho người dùng, và mở các ô chưa bom trong bảng (thay đổi style ô hiện tại thông qua việc thay đổi class).
- Nếu click phải ô chưa số thì đánh dấu ô đã được mở và mở ô này (thay đổi style ô hiện tại thông qua việc thay đổi class).
- Nếu click phải ô trống thì gọi hàm moOTrong() để mở các ô trống và ô số theo một vùng.
 - Mở 8 ô xung quanh ô hiện tại, ý tưởng thực hiện như việc đếm bom nhưng điều kiện có khác biệt là sẽ không mở các ô chưa bom.
 - Gọi đệ quy lại hàm click() để lan truyền theo một vùng.

Mã nguồn:

```
function click(square) //Hàm xử lý khi click chuột trái vào ô (Mở ô)
{
    if (gameOver || square.classList.contains("checked") || square.classList.contains("flagged"))
        return;

    if (square.classList.contains("bomb")) //Click trúng phải bom
    {
        document.getElementById("result").innerHTML = "💣 Thua rồi! 💣";
        gameOver = true; //Đừng game
        //Mở hết tất cả các ô có bom còn lại
        squares.forEach(function (square)
        {
            if (square.classList.contains("bomb"))
            {
                square.classList.remove("bomb");
                square.classList.add("boom");
            }
        });
    } else //Không phải bom và mở ô đó ra
    {
        let so = square.getAttribute("data");
        //Mở các ô trống nếu tại ô đó xung quanh có chứa bom, đổi màu sắc cho chữ số bằng cách ghi thêm class
        if (so != 0)
        {
            square.classList.add("checked");
            if (so == 1) square.classList.add("one");
            if (so == 2) square.classList.add("two");
            if (so == 3) square.classList.add("three");
            if (so == 4) square.classList.add("four");
            if (so == 5) square.classList.add("five");
            if (so == 6) square.classList.add("six");
            if (so == 7) square.classList.add("seven");
            if (so == 8) square.classList.add("eight");
            square.innerHTML = so;
        }
    }
}
```

```

        return;
    }
    //Mở các ô trống khác và lan truyền đi
    moCacOTrong(square);
}
square.classList.add("checked");
}

function moCacOTrong(square) //Hàm mở các ô trống nếu tại ô đó số bom xung
quanh là 0
{
    const IDHienTai = parseInt(square.id);
    const canhTrai = IDHienTai % size === 0;
    const canhPhai = IDHienTai % size === size - 1;

    //Dùng setTimeout để tạo hiệu ứng khai phá ô cho đẹp
    setTimeout(function (){
        //Tương tự như việc đếm bom, hàm này sẽ mở 8 ô xung quanh nó với các điều
        //kiện ràng buộc, thực hiện gọi đệ quy lại hàm clicked() để lan truyền, điều
        //kiện là ô này không chứa bom (Đám bám mở ra được các ô trống và chứa số)
        if (IDHienTai > 0 && !canhTrai) //Mở ô bên trái
        {
            const newId = IDHienTai - 1;
            const newSquare = document.getElementById(newId.toString());
            if (newSquare && !newSquare.classList.contains("bomb"))
                click(newSquare);
        }

        if (IDHienTai < size * size - 1 && !canhPhai) //Mở ô bên phải
        {
            const newId = IDHienTai + 1;
            const newSquare = document.getElementById(newId.toString());
            if (newSquare && !newSquare.classList.contains("bomb"))
                click(newSquare);
        }

        if (IDHienTai >= size) //Mở ô bên trên
        {
            const newId = IDHienTai - size;
            const newSquare = document.getElementById(newId.toString());
            if (newSquare && !newSquare.classList.contains("bomb"))
                click(newSquare);
        }

        if (IDHienTai < size * (size - 1)) //Mở ô bên dưới
        {
            const newId = IDHienTai + size;
            const newSquare = document.getElementById(newId.toString());
            if (newSquare && !newSquare.classList.contains("bomb"))
                click(newSquare);
        }

        if (IDHienTai >= size + 1 && !canhTrai) //Mở ô trên trái
        {
            const newId = IDHienTai - size - 1;
            const newSquare = document.getElementById(newId.toString());
            if (newSquare && !newSquare.classList.contains("bomb"))
                click(newSquare);
        }
    }, 100);
}

```

```

    const newSquare = document.getElementById(newId.toString());
    if (newSquare && !newSquare.classList.contains("bomb"))
        click(newSquare);
}

if (IDHienTai >= size && !canhPhai) //Mở ô trên phải
{
    const newId = IDHienTai - size + 1;
    const newSquare = document.getElementById(newId.toString());
    if (newSquare && !newSquare.classList.contains("bomb"))
        click(newSquare);
}

if (IDHienTai < size * (size - 1) && !canhTrai) //Mở ô dưới trái
{
    const newId = IDHienTai + size - 1;
    const newSquare = document.getElementById(newId.toString());
    if (newSquare && !newSquare.classList.contains("bomb"))
        click(newSquare);
}

if (IDHienTai < size * (size - 1) - 1 && !canhPhai) //Mở ô dưới phải
{
    const newId = IDHienTai + size + 1;
    const newSquare = document.getElementById(newId.toString());
    if (newSquare && !newSquare.classList.contains("bomb"))
        click(newSquare);
}
}, 10);
}

```

e. Chức năng 4: Gắn cờ và kiểm tra xem người chơi có thắng hay không

Ý tưởng thực hiện: Khi một ô click chuột trái vào sẽ có các sự kiện xảy ra và thực hiện nó, các trường hợp xảy ra và ràng buộc như sau.

- Không làm gì cả nếu game đã kết thúc hoặc ô đã được mở.
- Sử dụng biến đếm để đếm số cờ đã đặt.
- Không cho phép người dùng tiếp tục đặt cờ nếu số cờ đã đặt vượt quá số bom.
- Nếu tại ô đó chưa có cờ thì gắn cờ lên ô, tăng biến đếm cờ lên 1 và đánh dấu ô này (thay đổi style ô hiện tại thông qua việc thay đổi class).
- Nếu tại ô đó có cờ thì gỡ cờ, trả về style cũ và giảm biến đếm cờ đi 1.
- Liên tục kiểm tra điều kiện thắng qua hàm checkForWin().
 - Nếu các ô được gắn cờ trùng với các ô chứa bom thì người dùng chiến thắng, thông báo và dừng trò chơi.

Mã nguồn:

```

function addFlag(square) //Hàm xử lý khi click chuột phải vào ô (Hàm gắn cờ
đánh dấu ô nghi ngờ chứa bom)
{
    if (gameOver || square.classList.contains("checked")) return;

```

```

if (flags < bomb) //Ngăn chặn việc đặt cờ nhiều hơn số bom
{
    if (!square.classList.contains("flaged")) //Gắn cờ nếu chưa đánh dấu
    {
        square.classList.add("flaged");
        flags++;
        document.getElementById("flag_left").innerHTML = bomb - flags;
        checkForWin(); //Kiểm tra điều kiện thắng
    }
    else //Thu hồi lại cờ đã đánh dấu
    {
        square.classList.remove("flaged");
        flags--;
        document.getElementById("flag_left").innerHTML = bomb - flags;
    }
}
}

function checkForWin() //Hàm kiểm tra điều kiện thắng
{
    let matches = 0; //Đếm số cờ đã đánh dấu đúng vị trí của bom

    for (let i = 0; i < squares.length; i++)
    {
        if (squares[i].classList.contains("flaged") &&
squares[i].classList.contains("bomb"))
            matches++;
    }

    if (matches == bomb) //Chiến thắng khi đặt hết cờ đúng vị trí
    {
        document.getElementById("result").innerHTML = "🎉Bạn đã thắng! 🎉";
        gameOver = true;
    }
}

```

f. Chức năng 5: Đòng hồ tính giờ

Ý tưởng thực hiện: Đồng hồ sẽ chạy khi game bắt đầu (người chơi chọn cấp độ và bảng được tải) và dừng lại khi kết thúc game.

- Em sử dụng một biến đếm thời gian, biến này sẽ được cập nhật mỗi giây thông qua việc sử dụng hàm setInterval().

Mã nguồn:

```

function clock() {
    var x;
    time = 0;
    clearInterval(intervalId);
    intervalId = setInterval(function () //Tăng lên 1 mỗi giây khi game chưa
thắng hoặc chưa reset
    {
        if (time == 3600)

```

```
    gameOver = true;
else
{
    time++;
    x = ( Math.floor(time/60) < 10? "0" + Math.floor(time/60):
Math.floor(time/60) ) + ":" + (time%60 < 10? "0"+time%60:time%60)
    document.getElementById("clock").innerHTML = x;
}
if (gameOver) {
    document.getElementById("result").innerHTML = "Thua rồi!";
    clearInterval(intervalId);
}
}, 1000);
}
```

III. KIỂM THỬ VÀ KẾT LUẬN

Giao diện: Giao diện đồ họa tương đối dễ nhìn và trực quan. Do sử dụng các thông số cố định nên khi có sự thay đổi về độ lớn của cửa sổ thì sẽ xảy ra lỗi, cho nên trang web này đẹp nhất với màn hình 1920x1080, tỉ lệ màn hình 16:9 và tỉ lệ thu phóng 100%.

Chọn cấp độ chơi và reset: Người chơi chọn cấp độ khi trang web được tải. Reset sẽ tải lại trang. Khi người dùng đã chọn cấp độ thì không thể chọn lại cấp độ, trừ khi tải lại trang. Em sẽ phát triển thêm phần này, không cần tải lại trang mà vẫn reset được cấp độ hoặc chọn cấp độ mới.

Khởi tạo bảng trò chơi: Các ô được khởi tạo lấp đầy bảng hiện hành, tương đối đẹp. Tuy nhiên do em sử dụng cố định kích thước theo pixel nên khi có sự thay đổi về kích thước cửa sổ thì các ô vuông sẽ bị tràn.

Các chức năng thao tác:

- **Mở ô khi click chuột trái và lan truyền:** Chương trình thực thi tương đối tốt, lan truyền hợp lý, hiển thị ổn.
- **Gắn cờ khi click chuột phải:** Chức năng diễn ra bình thường, chưa phát hiện lỗi.
- **Thắng và thua game:** Chương trình thực thi đúng với yêu cầu game.
- **Đồng hồ tính giờ:** Hoạt động ổn và không xảy ra lỗi.

Tài liệu tham khảo

- Giáo trình môn Ứng dụng Web – Cô Nguyễn Thị Mai Trang.
- Video: [I code Minesweeper in JavaScript! - Ania Kubów](#) (Tham khảo ý tưởng game)
- Website: <https://www.w3schools.com/> (Tra cứu các hàm để tạo các chức năng)
- Website: <https://www.geeksforgeeks.org/>
- Công cụ hỗ trợ: ChatGPT-4o, em dùng để kiểm tra các lỗi logic và tìm kiếm các hàm và thông số phù hợp để ứng dụng cho game (Tỉ lệ sử dụng khoảng 10% cho toàn bộ bài tập lớn).

Thứ tư, ngày 14 tháng 8 năm 2024



Hoàng Phi Hùng