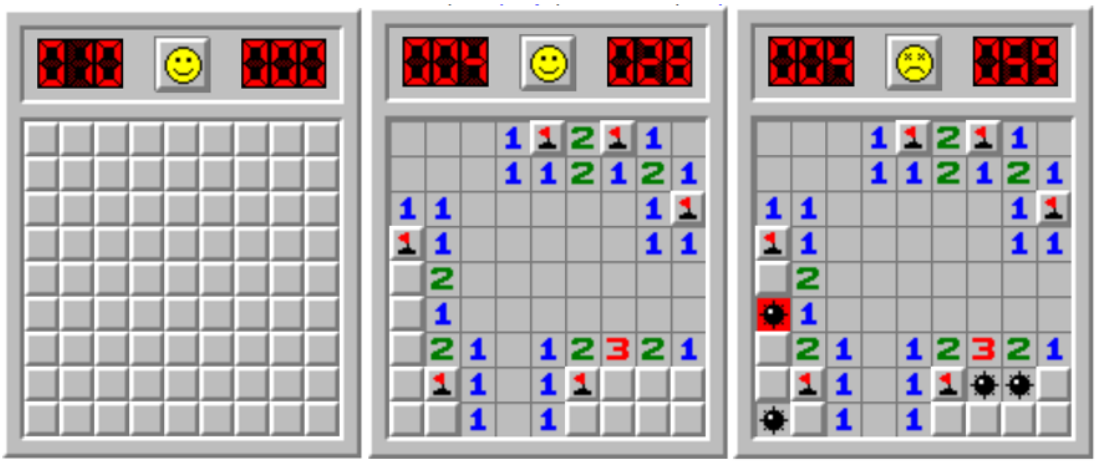
Sprint 1 Challenge

Mine Sweeper

Preface

Let's create a super version of the **Minesweeper** game.

First, play the game for a little while, get to know it, and relax.



Recommendations Before Diving In

Building projects is the best way to learn how to code. It's OK to start even if you don't understand everything yet, the material will become clearer as you practice and implement it.

Play the game until you feel comfortable with how it works. Remember: playing a game is not the same as programming a game.

Read this PDF from the beginning several times. Make sure you know exactly what you're trying to build before you start. Break the project into small tasks, and then break those into even smaller sub-tasks.

Reuse code you've already written. Work with open references such as previous exercises, solutions, in-class examples and slides. Make sure your code follows our conventions. There's a difference between code that works and good-quality code.

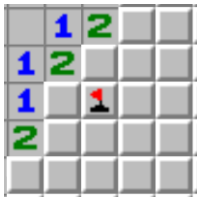
Minesweeper – Basic intro

The goal of the game is to reveal all the cells that do not contain mines without being "blown up" by clicking on a cell with a mine underneath.

Our Minesweeper basic functionality is based on the reference game

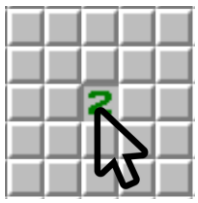
Functionality and Features

- Show the board
- Left-click reveals the cell's content
    - The cell may contain a mine (bomb)
    - Or it can be a safe cell
- Right-click flags/unflags a (suspected) cell



- Game ends when:
    - Clicking a mine - all the mines are revealed (User lost)
    - All the mines are flagged, and all the other cells are revealed (User's victory)

- When left-clicking on a cell, reveal it and then:
  - Cell with a mine – reveal all the mines and end the game
  - Cell that has some neighbors that are mines – show the number of mines
  - Cell that has no mines in his neighbors – also expand to reveal it's 1st degree neighbors


- Support 3 levels of the game
  - Beginner (4 * 4 with 2 MINES)
  - Medium (8 * 8 with 14 MINES)
  - Expert (12 * 12 with 32 MINES)

Development - Tips and Guidelines

Here are some initial global variables:

gBoard – A Matrix containing cell objects:
{
minesAroundCount: 4,
isRevealed: false,
isMine: false,
isMarked: false
}
	The model

gLevel = {
SIZE: 4,
MINES: 2
}
	This is an object by which the board size is set (in this case: 4x4 board and how many mines to place)

gGame = {
isOn: false,
revealedCount: 0,
markedCount: 0,
secsPassed: 0
}
	Holds the current game state:
	isOn: true when game is on
	revealedCount: How many cells are revealed
	markedCount: How many cells are marked (with a flag)
	secsPassed: How many seconds passed

As a guideline, here are some initial functions that we suggest:

`onInit()`	Called when page loads
`buildBoard()`	Builds the board Set some mines Call `setMinesNegsCount()` Return the created board
`setMinesNegsCount(board)`	Count mines around each cell and set the cell's **minesAroundCount**.
renderBoard(board)	Render the board as a <table> to the page
`onCellClicked(elCell, i, j)`	Called when a cell is clicked
onCellMarked(elCell, i, j)	Called when a cell is right-clicked See how you can hide the context menu on right click

`checkGameOver()`	The game ends when all mines are marked, and all the other cells are revealed
`expandReveal``(board, elCell, i, j)`	When the user clicks a cell with no mines around, reveal not only that cell, but also its neighbors. NOTE: start with a basic implementation that only reveals the non-mine 1st degree neighbors BONUS: Do it like the real algorithm (see description at the Bonuses section below)

Development – How to start?

Breaking-down the task to small tasks is a key success factor.
In our case – we recommend starting from the following steps:

Step1 – the seed app:

1. Create a 4x4 gBoard Matrix containing Objects.
2. Set 2 of them to be mines.
3. Present the board using **`renderBoard()`** function.

Step2 – counting neighbors:

1. Create **`setMinesNegsCount()`** and store the **`minesAroundCount`** property in each cell
2. Update the **`renderBoard()`** function to also display the neighbors count and the mines

Step3 – click to reveal:

1. When clicking a cell, call the **`onCellClicked()`** function.
2. Reveal the cell.

Step4 – randomize mines' location:

1. Add some randomicity for mines locations.

2. After you have this functionality working– it's best to comment the code and switch back to static location to help you focus during the development phase

Step5 – Upload initial game

1. Add a footer with your name
2. Upload to git

UI Guidelines

This sprint is not a UI-centered project, however, do your best to make it look nice

Further Tasks

First click is never a Mine

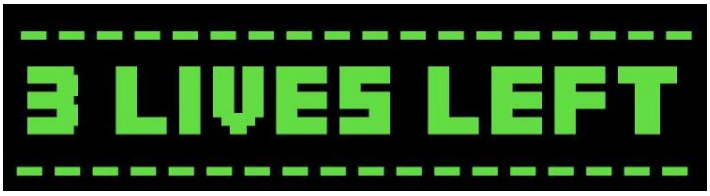
The first clicked cell is never a mine

HINT: We need to start with an empty board (no mines) and then place the mines and count the neighbors only on first click.

Lives

Add support for "LIVES" -

The user has 3 LIVES:



When a MINE is clicked:

- Show an indication to the user that he clicked a mine
- The LIVES counter decreases
- The cell is being unrevealed
- The user can mark it and continue playing

The Smiley button

Add the smiley button - clicking the smiley resets the game
here are some smiley states:

- Normal 😀
- Sad & Dead – LOSE 😵 (stepped on a mine and have no life left)
- Sunglasses – WIN 😎

Bonus Tasks

Add support for HINTS

The user has 3 hints

When a hint is clicked, it changes its look, example: 

Now, when an unrevealed cell is clicked, the cell and its neighbors are revealed for 1.5 seconds, and the clicked hint disappears.

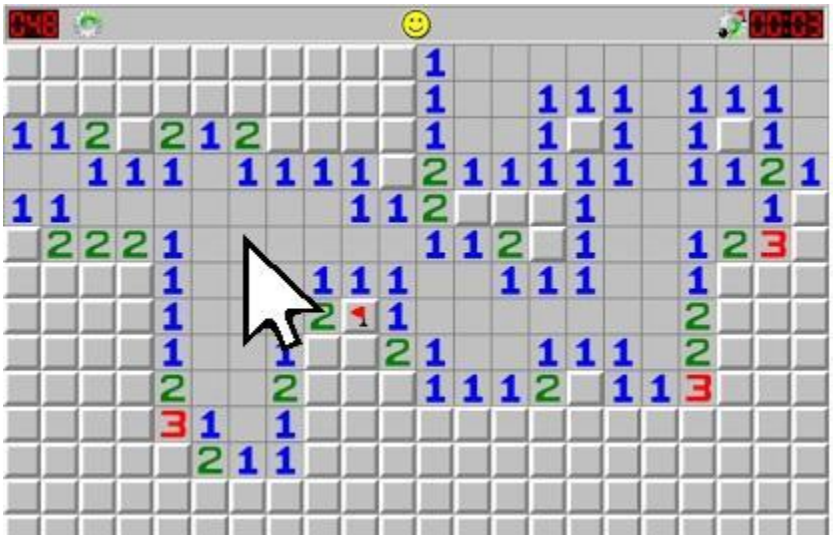
Best Score

Keep the best score in local storage (per level) and show it on the page

Full Expand

When a cell that has no mines in his neighbors is clicked – reveal the cell and also expand to all it's neighbors in a recursive way.

Here is an example:



Hint: Think about a recursion.

Safe click

The user has 3 **Safe-Clicks:**

Clicking the **Safe-Click** button will mark a random (unrevealed) safe cell (for 1.5 seconds) so the user knows that he can safely click that cell.

Dark Mode

The user should be able to toggle between Dark-Mode and Light-Mode

Undo

Add an "UNDO" button, so the user can undo (some of) his moves

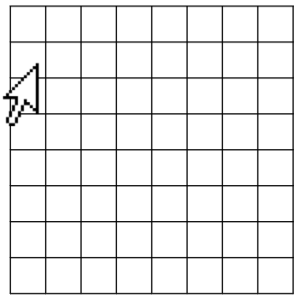
Manually positioned mines

Create a "manually create" mode in which the user first positions the mines (by clicking cells) and then plays.

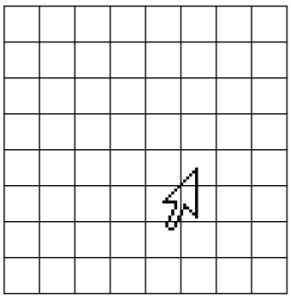
MEGA HINT

Mega-Hint works only once every game. It is used to reveal an area of the board for 2 seconds. Functionality description: (1) Click the "Mega Hint" button (2) then click the area's top-left cell (3) then click bottom-right cell. The whole area will be revealed for 2 seconds.

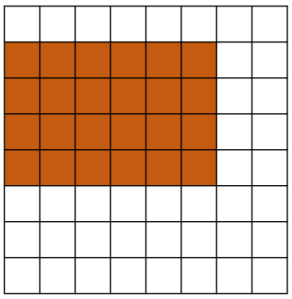
Step1: Click the Mega-Hint button



Step2: Click the area's top-left corner



Step3: Click the area's bottom-right corner



Result: the selected area's content will be revealed for 2 seconds

MINE EXTERMINATOR

Clicking the "Exterminator" button, eliminate 3 of the existing mines, randomly. These mines will disappear from the board. Re-calculation of neighbors-count is needed

אזהרה – הספרינט הזה ייקח אותך רחוק משחשבת