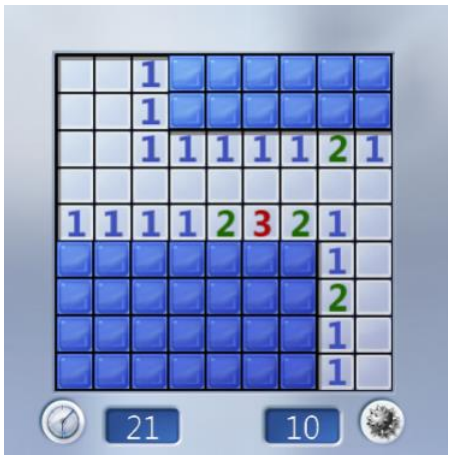
Sprint 1 Challenge

Mine Sweeper

Preface

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

First, play [the game] a little bit, get to know it and relax



Software Delivery Phases - Instructions

In this sprint **we work alone**, we code our own solution and bring our own knowledge and tools - please respect the rules.

Delivery is done through github

We will have 4 delivery points:

1. *Prototype*: Wednesday 21:00
2. *Score*: Thursday 23:00
   The sprint base-score will be determined by this delivery – please do your best to have a working game
3. *Bonus*: Saturday 21:00
4. *Presentation:* Sunday 8:30
   We will go through all projects, review the feature and get some cheering up from everyone

Minesweeper – Basic intro

The goal of the game is to uncover all the squares that do not contain mines without being "blown up" by clicking on a square with a mine underneath.
Our Minesweeper basic functionality is based on the reference game

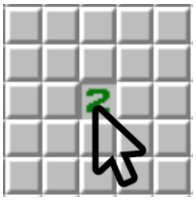
Functionality and Features

- Show the board
- Left click **reveals** the cell's content
- Right click flags/unflags a suspected cell (cannot **reveal** a flagged cell)

- Game ends when:
  - LOSE: when clicking a mine, all the mines are revealed

- WIN: all the mines are flagged, and all the other cells are shown

- Support 3 levels of the game
  - Beginner (4 * 4 with 2 MINES)
  - Medium (8 * 8 with 14 MINES)
  - Expert (12 * 12 with 32 MINES)
- Expanding: When **left clicking** on cells there are 3 possible cases we want to address:
  - MINE – reveal the mine clicked
  - Cell with neighbors – reveal the cell alone
  - Cell without neighbors – expand it and its 1st degree neighbors


Development - Tips and Guidelines

As you know, there is usually more than one way to approach a challenge.

But as a guideline, we suggest having the following functions (it is ok to have more functions as needed).

`onInit()`	This is called when page loads
`buildBoard()`	Builds the board Set the 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)`	Called when a cell is right-clicked See how you can hide the context menu on right click
`checkGameOver()`	Game ends when all mines are marked, and all the other cells are shown
`expandShown``(board, elCell, i, j)`	When user clicks a cell with no mines around, we need to open not only that cell, but also its neighbors. NOTE: start with a basic implementation that only opens the non-mine 1st degree neighbors BONUS: if you have the time later, try to work more like the real algorithm (see description at the Bonuses section below)

Here are the global variables you might be using:

gBoard - A Matrix containing cell objects:
Each cell: {
minesAroundCount: 4,
isShown: false,
isMine: false,
isMarked: true
 	
}
	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,
shownCount: 0,
markedCount: 0,
secsPassed: 0
}
	This is an object in which you can keep and update the current game state:
	isOn: Boolean, when true we let the user play
	shownCount: How many cells are shown
	markedCount: How many cells are marked (with a flag)
	secsPassed: How many seconds passed

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 mines using `renderBoard()` function.

Step2 – counting neighbors:

1. Create `setMinesNegsCount()` and store the numbers
2. Update the `renderBoard()` function to also display the neighbor count and the mines
3. Add a `console.log` – to help you with debugging

Step3 – click to reveal:

1. When clicking a cell, call the `onCellClicked()` function.
2. Clicking a safe cell reveals the `minesAroundCount` of this cell

Step4 – randomize mines' location:

1. Add some randomicity for mines location
2. After you have this functionality working– its best to comment the code and switch back to static location to help you focus during the development phase

Step5 –

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

Continue to **Functionality and Features**, then to **Further Tasks**, and if you went that far, do go ahead and check the **Bonus Tasks**.

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 matrix (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, there is an indication to the user that he clicked a mine. The LIVES counter decreases. The user can continue playing.

The Smiley button

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

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

- Sunglasses – WIN 😎

Bonus Tasks – if time permits

Add support for HINTS

The user has 3 hints

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

Now, when a cell (unrevealed) is clicked, the cell and its neighbors are revealed **for a second**, and the clicked hint disappears.

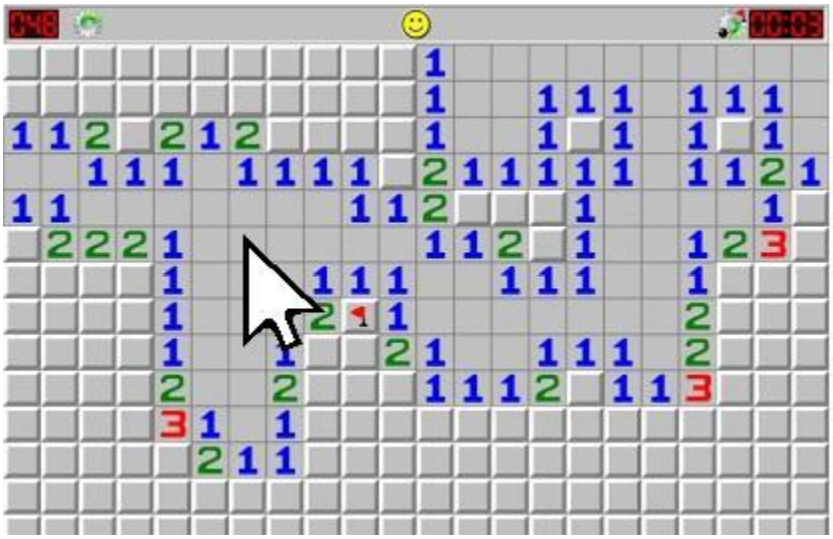
Best Score

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

Full Expand

When an empty cell is clicked, open all empty cells that are connected and their numbered neighbors

Expand like in the real game ("Full expand"):



Think about a recursion.

Safe click

Add a **Safe-Click** Button:

The user has 3 **Safe-Clicks**

Clicking the **Safe-Click** button will mark a random covered cell (for a few seconds) that is safe to click

Present the remaining **Safe-Clicks** count

Manually positioned mines

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

Undo

Add an “UNDO” button, each click on that button takes the game back by one step (can go all the way back to game start).

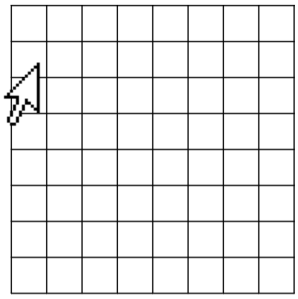
DARK MODE

Implement Dark-Mode for the game

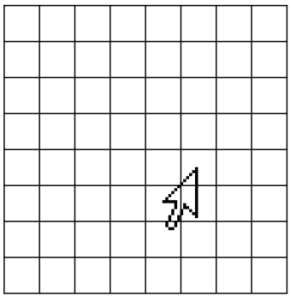
MEGA HINT

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.

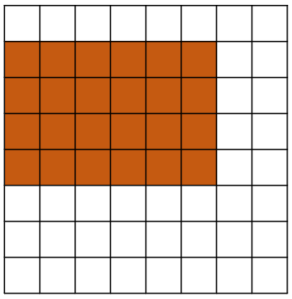
Step1: Click the Mega-Hint button



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.

We will need re-calculation of neighbors-count

אזהרה – הספרינט הזה ייקח אותך רחוק משחשבת