Space Invaders

We will have 4 delivery points:

- *Prototype*: Monday 21:30
- *Score*: Thursday 21:30
  The sprint base-score will be determined by this delivery – please do your best to have a working game
- *Bonus(optional)*: Saturday 21:30
- *Presentation:* Monday 17:30
  We will go through all projects, review the feature, and get some chirring up from everyone

Preface

Your challenge is to create the game Space-Invaders.
Play a version of the game for a little bit, get to know how it functions.

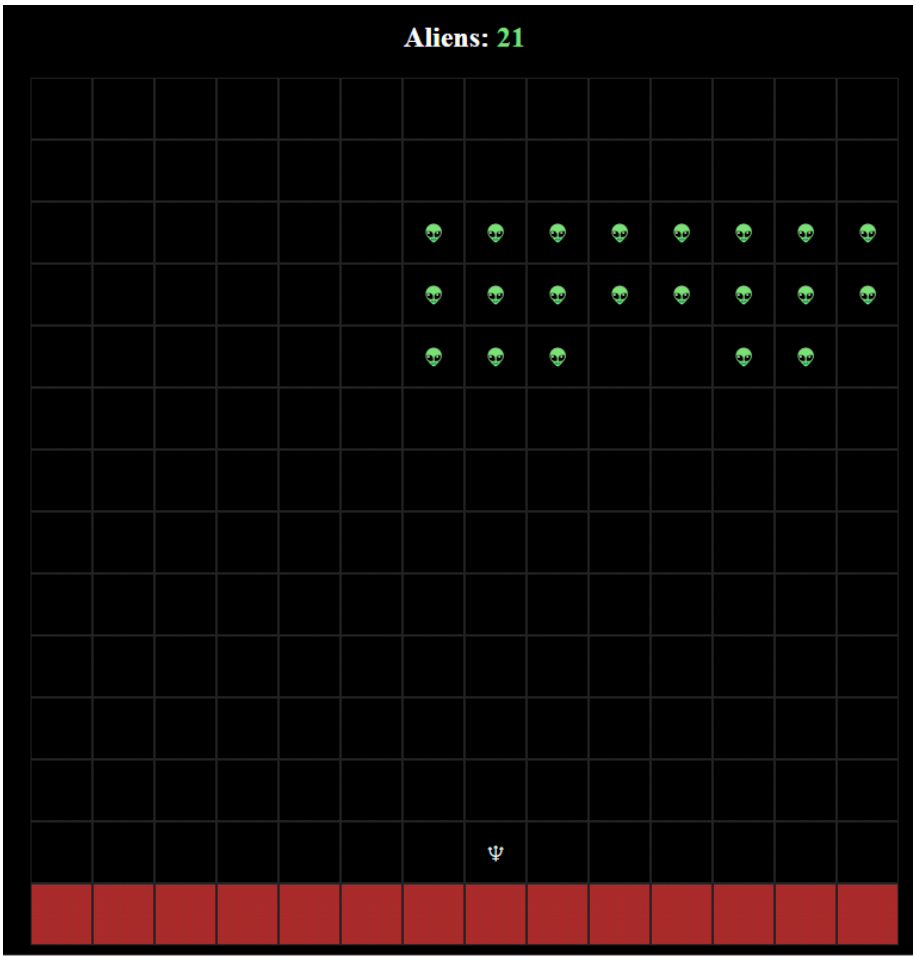
Notes for our version of the game

- We call the invaders – aliens
- All alien rows move on the same direction

Space Invaders – Basic intro

The goal of the game is to clear all the aliens before they get to the earth.
You can move the hero left and right and shoot the aliens, The aliens move together from side to side and then down

Space Invaders – Basic user interface



Functionality and Features

- Left and right arrows move the hero on the board.
- Spacebar shoots a laser towards the aliens,
    - hero can shoot only 1 laser at a time
    - make sure to clear the interval when hitting an alien or reaching the top
    - hero get points when clearing aliens
- Player wins if all aliens were cleared
- Aliens move from right to left and again from left to right once they hit the edge, every time they hit the edge they also go down by one row.
- Player loose when an alien reached the hero row (landed on earth)

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).

game.js

```
const BOARD_SIZE = 14
const ALIEN_ROW_LENGTH = 8
const ALIEN_ROW_COUNT = 3

const HERO = '♆'
const ALIEN = '👽'
const LASER = '⟰'

// Matrix of cell objects. e.g.: {type: SKY, gameObject: ALIEN}
var gBoard
var gGame = {
    isOn: false,
    alienCount: 0
}
```

```
// Called when game loads
function init() {}
// Create and returns the board with aliens on top, ground at bottom
// use the functions: createCell, createHero, createAliens
function createBoard() {}

// Render the board as a <table> to the page
function renderBoard(board) {}

// Returns a new cell object. e.g.: {type: SKY, gameObject: ALIEN}
function createCell(gameObject = null) {
    return {
        type: SKY,
        gameObject: gameObject
    }
}
// position such as: {i: 2, j: 7}
function updateCell(pos, gameObject = null) {
    gBoard[pos.i][pos.j].gameObject = gameObject
    var elCell = getElCell(pos)
    elCell.innerHTML = gameObject || ''
}
```

hero.js

```
const LASER_SPEED = 80
var gHero = {pos: {i:12, j: 5}, isShoot: false}

// creates the hero and place it on board
function createHero(board) {}
// Handle game keys
function onKeyDown(ev) {}
// Move the hero right (1) or left (-1)
function moveHero(dir) {}
// Sets an interval for shutting (blinking) the laser up towards aliens
function shoot() {}
// renders a LASER at specific cell for short time and removes it
function blinkLaser(pos) {}
```

alien.js

```
const ALIEN_SPEED = 500
var gIntervalAliens

// The following two variables represent the part of the matrix (some r
ows)
// that we should shift (left, right, and bottom)
// We need to update those when:
// (1) shifting down and (2) last alien was cleared from row
var gAliensTopRowIdx
var gAliensBottomRowIdx

var gIsAlienFreeze = true

function createAliens(board) {}
function handleAlienHit(pos) {}
function shiftBoardRight(board, fromI, toI) {}
function shiftBoardLeft(board, fromI, toI) {}
function shiftBoardDown(board, fromI, toI) {}

// runs the interval for moving aliens side to side and down
// it re-renders the board every time
// when the aliens are reaching the hero row - interval stops
function moveAliens() {}
```

util.js

```
// Returns a new cell object. e.g.: {type: SKY, gameObject: ALIEN}
function createCell(gameObject = null) {
  return {
    type: SKY,
    gameObject: gameObject
  }
}


function getElCell(pos) {
  return document.querySelector(`[data-i='${pos.i}'][data-
j='${pos.j}']`)
}
```

Development – How to start?

Breaking-down the project to small tasks and figuring the right order is a key success factor.
We recommend starting from the following steps:

Step1 – Setup the board

- Create a 14x14 matrix containing strings (later on you can switch to a more advanced model of cells as described in code above). Place a single line of few aliens on top, no need for them to move just yet
- Code the function `createHero(board)` so that it adds the hero to the board
- Code the function `createAliens(board)` so that it adds the aliens to the board
- Present the board using `renderBoard()` function.

Step2 – Move the hero.

- Update the model and DOM using the function `updateCell`

Step3 – Shoot Aliens.

- Implement the function `blinkLaser` that uses `updateCell` twice: to display and remove the laser in a specific cell
- Implement the function `shoot`. It calls the function `blinkLaser` with an interval
    - Do not allow shooting if there is already a laser in progress
    - Clear the interval when reaching the top or hitting an alien.
- Hero gain points by clearing an alien, each alien worth 10 points. Show the score.
- If all aliens cleared show a victory message with a Restart game

Step4 – Let the aliens move

Note – when aliens are moving, debug becomes harder.

- implement the freeze feature early so you can stop them for debug (the aliens-interval can keep running but when `gIsAlienFreeze` is true, it returns immediately without performing any action)
- start with implementing the *shiftBoard* functions (3 functions) and prove they work with Unit Testing, make sure you cover the different possible states (empty, alien, laser)
- Code the function `moveAliens` that runs the interval for moving aliens side to side and down, it re-renders the board every time, when the aliens are reaching the hero row - interval stops

Make sure you covered the **Functionality and Feature** and that you have a nice-looking game.

Good idea to save a backup before moving to further tasks.

Further Tasks

Note – It will be easier to develop, test and debug some of these features when aliens are frozen

- **Start / Restart Buttons** - The game will start after pressing on the start button and should allow restart when done
- **Blow up neighbors** - when player press **'n'** – if the laser has neighbors that are aliens – they are all cleared
- **Super mode** – When player press **'x'** his laser symbol become **'^'** and it flies faster. You have 3 Super Attacks.

Bonus Tasks – If time permits

Space candies

Appear for 5 seconds, every 10 seconds on the first row
If hit, player gains an extra 50 points

Aliens Freeze

When space candy is hit, aliens freeze for 5 seconds

Go Extreme

Aliens Variety
Make the aliens rows look different from each other

Levels
Support 3 levels (Easy, Normal, Hard) those will affect aliens count, and speeds

Super mode
When player press **'x'** he can shoot multiple lasers for 3 seconds. Player has 3 Super Attacks. TIP: you will need to keep intervals in an array

Aliens Shoot
Aliens can randomly throw rocks down

Lives
Add support for “LIVES” - The user has 3 LIVES:
When a laser hits your spaceship, there is an indication to the user that he got hit. The LIVES counter decrease. The user can continue playing

Shields
When player press **'z'** his symbol change and he is safe for 5 seconds. Player has 3 Shields.

Bunkers
Add bunkers to your board, when aliens, rocks or lasers engage with it, they destroy it.

Customize
Allow the player to customize the game: Theme, Alien images, Bunkers