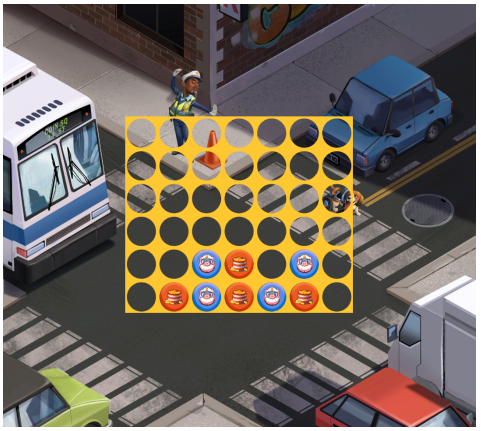
Unity Test Assignment

In this assignment you will implement a Connect 4 game. This is a game for two players - red and blue. The players are taking turns dropping colored tokens into a seven-column, six-row vertically suspended grid. The pieces fall straight down, occupying the lowest available space within the column. The objective of the game is to be the first to form a horizontal, vertical, or diagonal line of four of one's own tokens.



Part 1 - Connect 4

In this part you implement a basic “Connect 4” game.
Here are the basic game requirements:

1. 7 x 6 grid size.
2. The game has 2 players, each will be assigned with red or blue.
3. The players will take turns by choosing a column with an empty slot - blue goes first.
4. The game will end when one of the players will create a winning formation or the board is full (Draw).
5. At the end of the game, a message will appear (“Player 1 Wins!”, “Player 2 Wins!” or “It’s a Draw”).
6. Restart Game Button - restarts the game.

The game should be implemented to support the following modes:

- Player 1 vs Player 2 (local multiplayer) - 2 players playing on the same device.
- Player vs Computer (Random valid move is enough).

The software design should be open to extensions, and allow easy addition of new options (for example: play against an online player) without making modifications to the system.
Please pay attention to the performance characteristics of your solution.

Part 2 - Unit Tests

Write unit tests that will verify the game's functionality.
For example - the tests should verify the end game logic: win, lose, draw.

Part 3 - Main Menu

The game should have a "Main Menu" with the following options:

1. Choose play mode (player1 vs player 2 / player vs computer).
2. "Start" button will start the game in the selected mode.

Possible Bonus Points

- Animations (victory, time, etc.).
- Sounds (from any free sounds website).
- Settings screen that will control the game functionality.
- Support computer vs. computer play mode.

What do you get from us?

We supply you with a unity package containing the game's board scene and some basic components & functionality for you to use. Using the provided assets and code base is mandatory.

Assets

1. Connect4_Menu.unity scene - This scene contains a functional game board with all of its components and related assets.
2. Disk_A.prefab, DiskB.prefab - You will use those prefabs in order to create disks in the game.

Code

A DLL file with the following components (see also the interfaces next page):

1. Disk (mono behavior, IDisk) - Game disk component.
2. ConnectGameGrid (mono behavior, IGrid) - Game grid component.

This is the API the components above implement:

```
<summary>
Should be attached to the disk prefab.
Invokes an action on collision.
</summary>
public interface IDisk
{
    <summary>
    Invoked when the disk stops falling (hits the collider).
    </summary>
    event Action StoppedFalling;
}

<summary>
Triggers the disk Instantiation and puts it where it should be (using
colliders)
</summary>
public interface IGrid
{
    <summary>
    This event will be triggered when the user clicked one of
    the spawn buttons (clicked a column) with the respective column num.
    </summary>
    event Action<int> ColumnClicked;

    <summary>
    Instantiate a diskPrefab
    </summary>
    <param name="diskPrefab">The diskPrefab prefab we should
    instantiate</param>
    <param name="column">Which column was tapped - zero based</param>
    <param name="row">On which row should the diskPrefab be? - zero
    based</param>
    <returns>The instantiated diskPrefab game object</returns>
    IDisk Spawn(Disk diskPrefab, int column, int row);
}
```

Important things to note:

1. The solution is expected to be designed using proper software engineering principles, and should be properly documented when needed (so we know what decisions were made, and why).

2. The solution should be implemented by using C# and **Unity version 2021.3 LTS**

3. **Please don't delete any given prefabs/components - they were created to help you :)**

Submission instructions

Export a **Unity package** containing all the assets (scripts, assets, game scene) that are required to run your exercise.

Submit to:

Israel
client-home-exercise-TLV@moonactive.com

Ukraine
client-home-exercise-Kyiv@moonactive.com

Romania
client-home-exercise-Romania@moonactive.com

Questions?

Feel free to contact us with any questions or clarifications to your matching site email above.

Legal Disclaimer

This Moon Active Unity Developer test includes without limitation any all attachments and/or linked assets and by extension includes any written or oral response or presentation provided by a candidate to such test (“Test”). The Test and all other materials, communications, information and documentations, in any form, disclosed by Moon Active Ltd. (“Company ”) and/or anyone on its behalf to you, or that is otherwise learned by you in the course of conducting the Test (including, without limitation, information regarding the Company, and/o any instructions provided by Company and/or other data relating to the Company’s mobile games) (“Confidential Information”) shall be deemed confidential and you shall not disclose or use it for any other purpose other than to conduct the Test, and shall take reasonable measures to prevent disclosure or use of such Confidential Information, and is intended exclusively for you. You hereby represent and warrant that you shall not use Test in any publicity or promotional or marketing publication or personal portfolio. Company is and shall at all times remain the sole and exclusive owner of the Test, and any part thereof (including, without limitation, any content, creatives and all intellectual property rights in the foregoing and in any enhancements, modifications, updates or derivatives thereof and related know-how and any and all logos, trade names, trademarks and service marks, whether or not registered). All rights which are not expressly granted herein are reserved by Company.