DTT
APPS WEB GAMES

SINCE 2010

Ordering Words

Documentation | 18-05-2022

Table of Contents

1. Get started quickly 3

2. Introduction 4

3. Set-Up 5

4. API 8

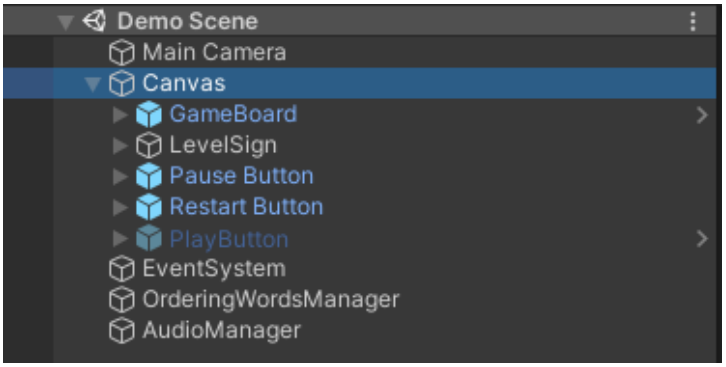
5. Known Limitations 10

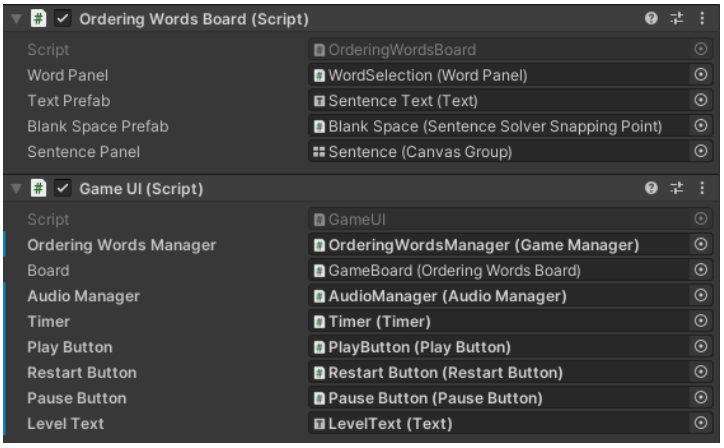
6. Support and feedback 11

1. Get started quickly

To start the game, open the demo scene found under “Demo/Scenes/Demo Scene”.

Here you can find a canvas with a board game object. This has a **GameBoard** and **GameUI** components attached to it. You can also find a game object for the level sign and timer of the game. Finally, there are also three UI buttons inside the canvas.



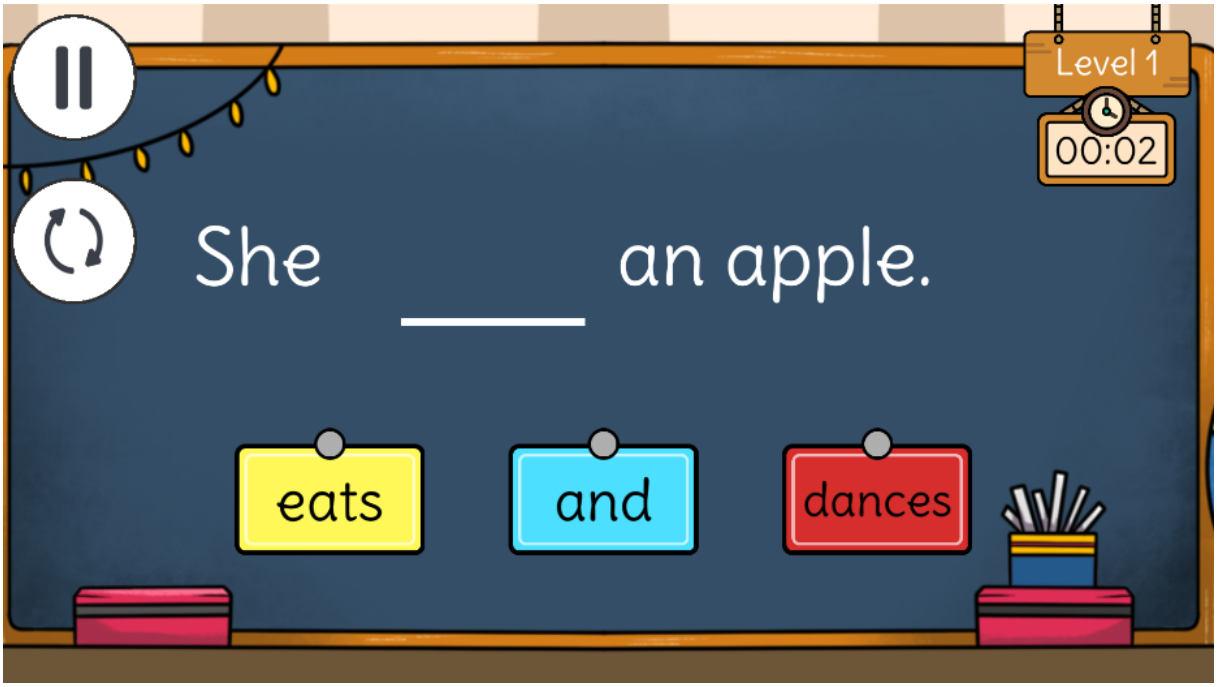


In the scene, the ordering words manager game object holds the **GameManager** component that is used to handle the Playing/Restarting/Pausing of the game. It holds a **GameStarter** with a reference to the GameSettings scriptable object. This component is used to start the game automatically.

The **AudioManager** game object and component is used for playing sound effects and makes use of the **AudioAsset** scriptable objects to play the audio clips.

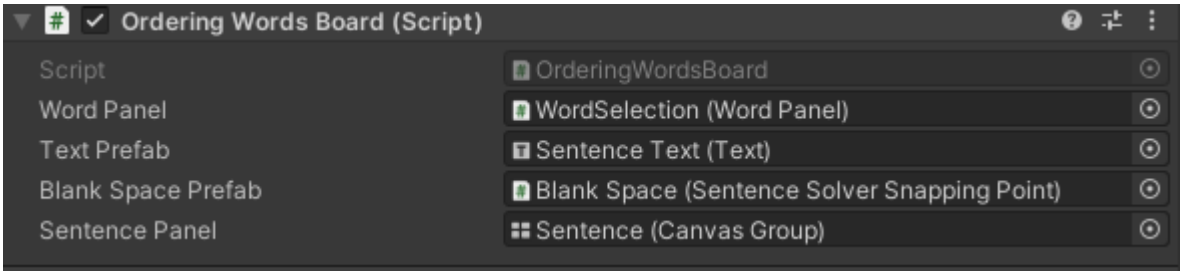
2. Introduction

DTT **Ordering Words** is a Unity asset that allows you to easily implement a game to set the missing words in the sentence. Using scriptable objects you can edit some features of the game which allows every level to have a different difficulty. You can customize the word options and the amount shown to the player, the sentence, and the amount of blank space. Additionally, you can choose the possible colors for the word options. All of this allows for an easy way to implement a word ordering game into your project.



3. Set-Up

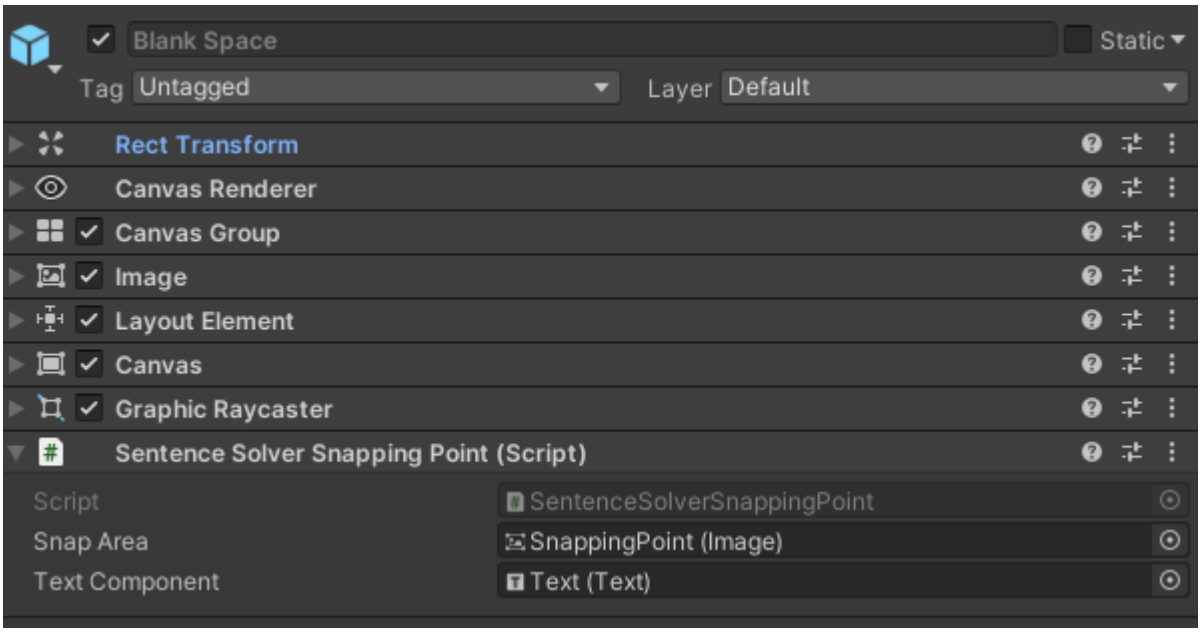
Create a game object and add the GameManager component to it. This requires a reference to a GameBoard. So start with the board, create a canvas and then create a game object under it. Add the GameBoard component to it. This holds a reference to several components.

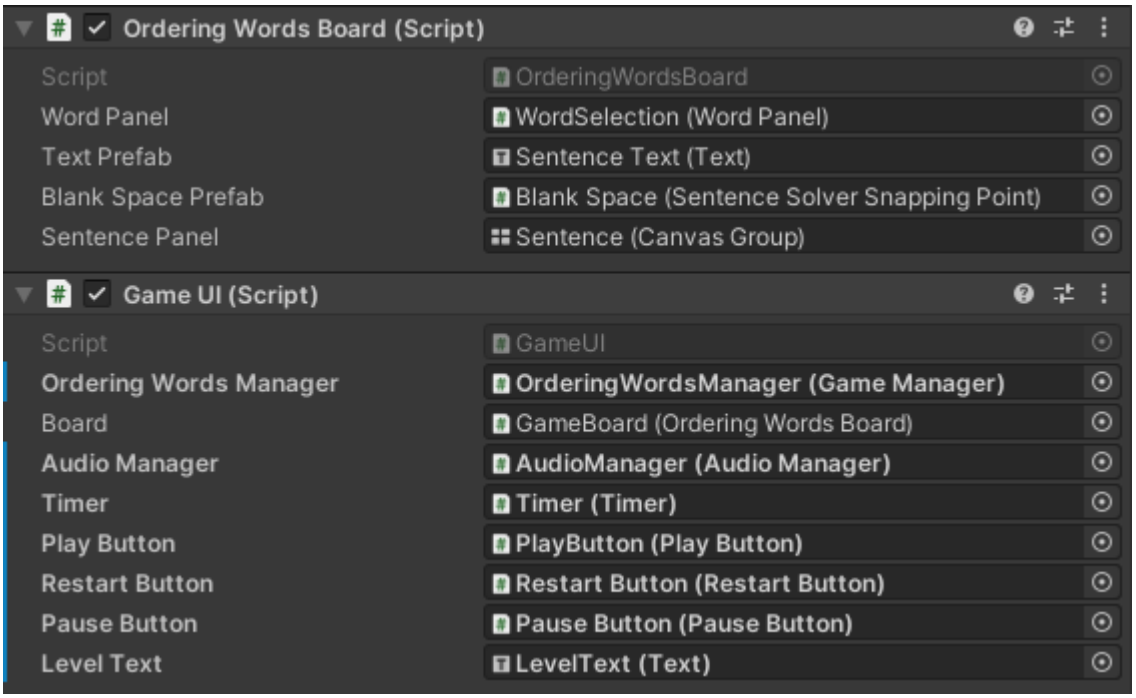


Create a game object as a child of the board and add the WordPanel component to it. This holds a reference to a Word prefab and a draggable panel. For the draggable panel, create a game object and place it where the words should be spawned in the canvas.

For the sentence panel, create a game object as a child of the board and add the Horizontal Layout Group and Canvas Group components.

Finally, add the references for the Text prefab and Blank Space prefabs. For the Text prefab, create a game object with the Text component. To create a Blank Space add the components shown on the image below.





In the Demo, a GameUI component can be also found on the Game Board prefab. This has a reference to the **GameManager**, the **GameBoard**, a **Timer**, an **AudioManager,** and other button components to handle the Playing/Restarting/Pausing of the game.

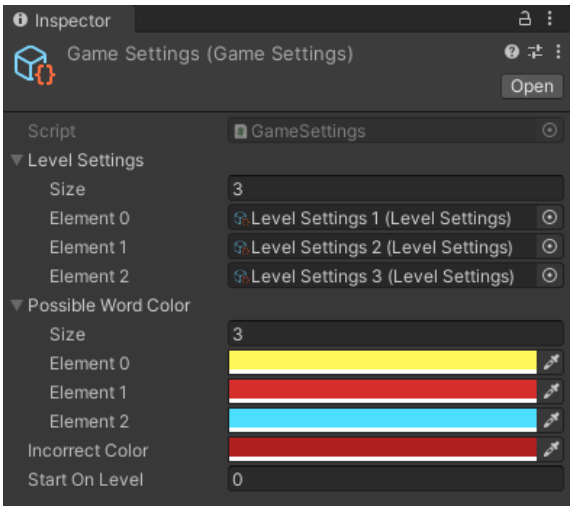
For the buttons, you can create a new game object as a child of the canvas and use the components found under Demo/Scripts/GameUI.

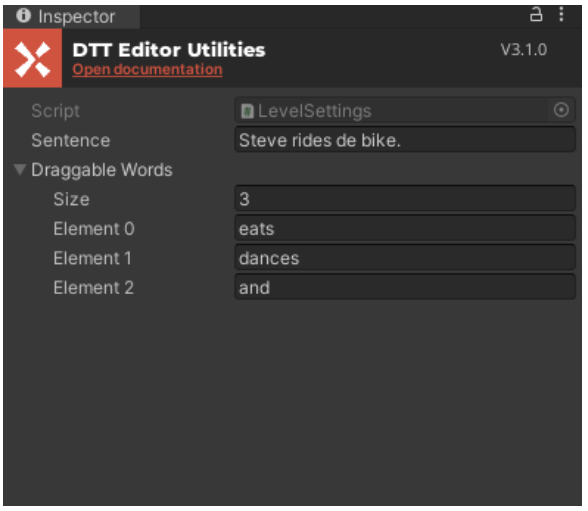
To start the game, you can add the **GameStarter** component found under Demo/Scripts. This requires a reference to **GameManager** and a **GameSettings** scriptable object.

The **GameSettings** holds an array of **LevelSettings**, one for each level of the game. To create these, right-click somewhere on the project folder and follow the options shown in the image below.

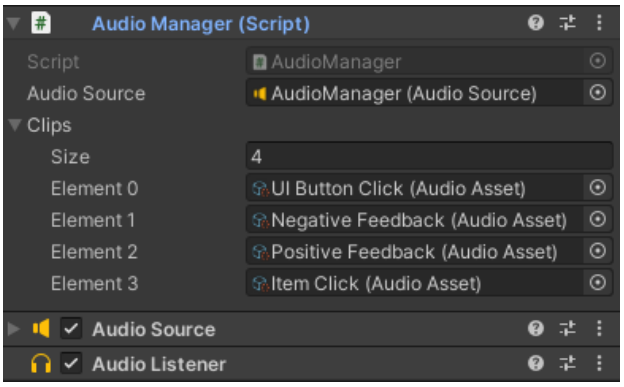
You can add colors for the word options on the **GameSettings** and choose what level to start the game on. On **LevelSettings,** you can create a sentence and add the word

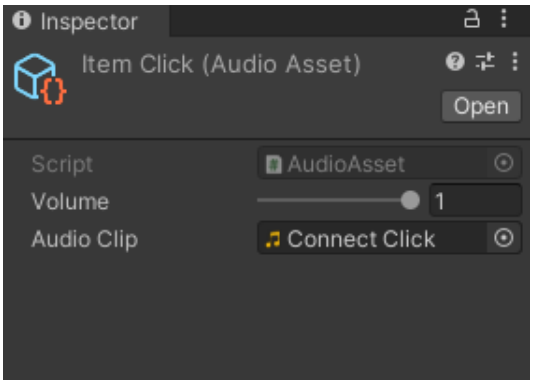
options. Adding a word that is also found in the sentence will create a blank space that needs to be filled to complete an exercise.





For audio purposes, you can use the AudioManager component found under Demo/Scripts/Audio. This holds a reference to an AudioSource and multiple AudioAsset prefabs that hold the audio clips.





4. API

The GameManager component handles the state of the game and allows you to start, pause, and finish the game.

StartGame	Method that allows you to start the game and will invoke the Started event.
Start level	Method that allows you to start at a specific level.
NextLevel	Method that allows you to proceed to the following level.
Restart	Method that allows you to restart the game.
Pause	Method that allows you to pause the game.
Continue	Method that allows you to continue the game if paused.
FinishLevel	Method that allows you to finish the current level.
ForceFinish	method allows you to finish the game and Finish will be invoked.

You can listen to the following events:

```
/// <summary>
/// Manager class for the ordering words minigame, handles starting, finishing, pausing and resuming the game.
/// </summary>
public class GameManager : MonoBehaviour, IMinigame<GameSettings, OrderingWordsResult>
{
    /// <summary>
    /// Reference to the ordering words controller.
    /// </summary>
    [SerializeField]
    [Tooltip("Reference to the ordering words controller")]
    private OrderingWordsBoard _board;

    /// <summary>
    /// Is called when the game has started.
    /// </summary>
    public event Action Started;

    /// <summary>
    /// Is called when the game has been Paused;
    /// </summary>
    public event Action Paused;

    /// <summary>
    /// Is called when the game has finished.
    /// </summary>
    public event Action<OrderingWordsResult> Finish;

    /// <summary>
    /// Is called when the level has been finished;
    /// </summary>
    public event Action LevelEnded;
```

The **GameBoard** component handles the board of the game. This has some events that can be listened to:

```
/// <summary>
/// Event for when the correct word answers are given.
/// </summary>
public event Action CorrectAnswerGiven;

/// <summary>
/// Event for when an incorrect word answer is given.
/// </summary>
public event Action IncorrectAnswerGiven;
```

5. Known Limitations

- Sentences created in the **LevelSettings** can be up to 100 words.

6. Support and feedback

If you have any questions regarding the use of this asset, we are happy to help you out. Always feel free to contact us at:

unity-support@d-tt.nl

(We typically respond within 1-2 business days)

We are actively developing this asset, with many future updates and extensions already planned. We are eager to include feedback from our users in future updates, be they 'quality of life improvements, new features, bug fixes, or anything else that can help you improve your experience with this asset. You can reach us at the email above.

Reviews and ratings are very much appreciated as they help us raise awareness and to improve our assets.

DTT stands for Doing Things Together

DTT is an app, web, and game development agency based in the center of Amsterdam. Established in 2010, DTT has over a decade of experience in mobile, game, and web-based technology.

Our game department primarily works in Unity where we put significant emphasis on the development of internal packages, allowing us to efficiently reuse code between projects. To support the Unity community, we are publishing a selection of our internal packages on the Asset Store, including this one.

More information about DTT (including our clients, projects, and vacancies) can be found here:

https://www.d-tt.nl/en/