DTT
APPS WEB GAMES

SINCE 2010

Bubble Shooter

Documentation | 18-07-2022

Table of Contents

1. Get started quickly

To get the minigame running quickly, follow the steps provided below.

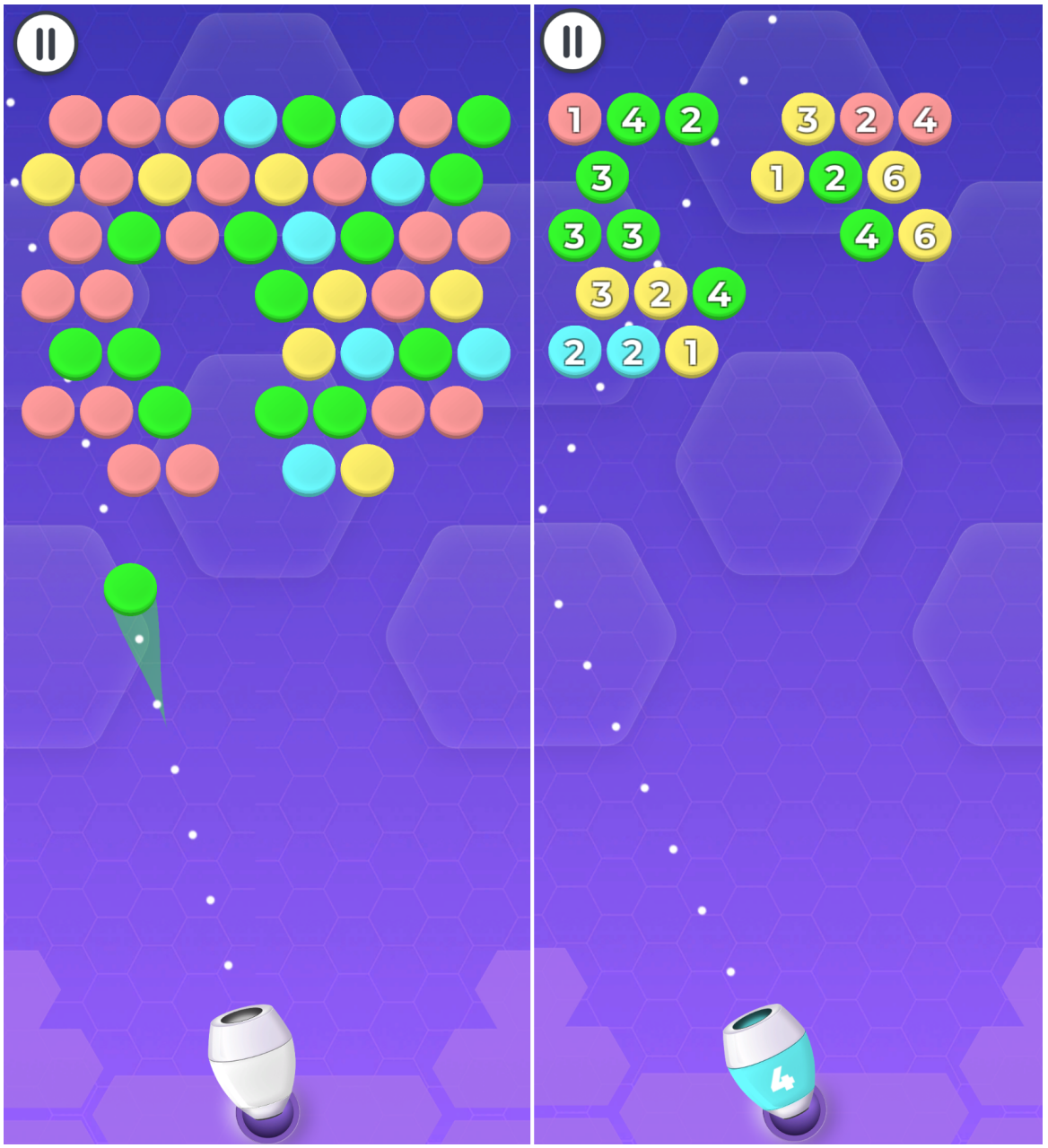
1. Open the demo scene that was provided with the package. You can run the game from here and see the asset in action.
2. Select the *Managers* object from the hierarchy.
3. Select the reference to the config in the **BubbleShooterManager** component.
4. The config used for this demo will now be highlighted in your projects tab. From here, select it and modify the values of the configuration in the inspector.

It is advised to make a copy of the scene and adjust objects from there.

2. Introduction

DTT Bubble Shooter is a minigame template that generates a grid populated with randomly generated bubbles. The player gets the ability to shoot bubbles from a turret to match connected bubbles and make them pop as long as they match.

By using a Scriptable Object, a game can be set up very dynamically to adjust the needs of each game. By adjusting the size of the grid and how far the grid can reach until it is game over can all be setup using the configuration. This will then define the difficulty of a game.



3. Set-Up

Down below is a basic procedure described on how to implement the asset to a working example.

Required steps

The steps provided below will make the game run in-code. After the required steps have been completed, the optional steps can be followed to get instructions on how to visualize the game.

1. Create a **Bubble Shooter Config** configuration file to adjust the settings of a game to your liking through **DTT/Mini Game/Bubble Shooter/Config**.
2. After creating the config file, create an empty **Game Object** in the hierarchy and attach the **BubbleShooterManager** component to the object.
3. Drag the earlier created configuration file to the **BubbleShooterManager** component to set the configuration variable.

Optional steps

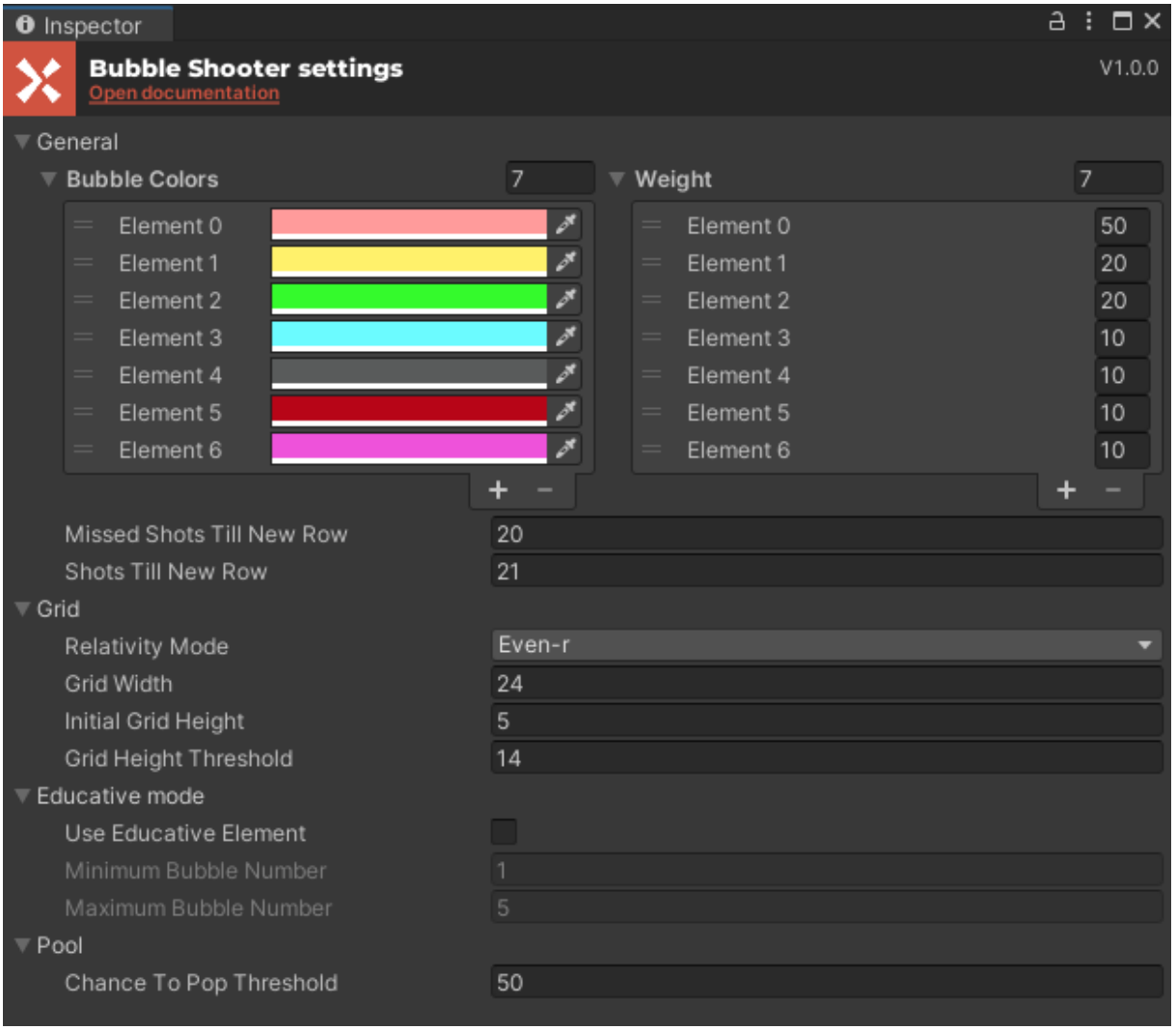
The steps provided below will go through the process of an example of how to implement the visualization of the game.

1. Create a new **MonoBehaviour** script and add it to a **GameObject**. This script will handle incoming updates from the grid once changes on it occur.
2. Make a reference from the newly created script to the **BubbleShooterManager** component to access the grid instance.
3. In this script, attach a delegate to the *GameInitialized* event in the **BubbleShooterManager** component.
4. In this delegate, attach another delegate to the *Updated* event to get notified when a cell in the grid gets updated. By using these events, you can visualize the game by keeping track of updates that happen in code.

Note: To save yourself some time, it is recommended to use the demo scene as a starting point and expand further from there.

4. Editor

DTT Bubble Shooter allows you to set up a simple minigame where a grid will be generated using the data from a Scriptable Object. A custom editor has been created in order to make the experience of setting up a game easier for the developer.



If you would like to dive a little deeper on the meaning of all values in the configuration, see the table below.

Option	Description
Bubble Colors	The colors applied to bubbles in the grid.
Weight	The weight of each color for each bubble generation. If all is set to 0, color generation is fully random.
Missed Shots Till New Row	The amount of missed shots the turret has to shoot until a new row appears.
Shots Till New Row	The amount of shots the turret had to shoot until a new row appears.
Relativity Mode	The mode used to properly get adjacent cells in a specific offset hexagon grid.
Grid Width	Represents the amount of bubbles each row will contain.
Initial Grid Height	The height of rows the grid of bubbles will be instantiated with once the game starts.
Grid Height Threshold	The height of the grid that has to be reached for the game to be considered game over and lost.
Use Educative Element	Indicates whether to use numbers on bubbles for an educative way to play the game.
Minimum Bubble Number	The minimum value a numbered bubble can have upon initialization of the game.
Maximum Bubble Number	The maximum value a numbered bubble can have upon initialization of the game.

Chance To Pop Threshold	The minimum chance a set of bubbles has to have from the bubble pool to be generated in the turret. The higher this value, the easier the shots will be.

5. API

The classes described below show relevant information on how they relate to one another. These can be used to adjust the flow of the minigame to your liking.

BubbleShooterManager

Property Name	Type	Description
Config	BubbleShooterConfig	A reference to the configuration passed through the inspector.
Grid	HexagonGrid	the HexagonGrid instance that bubbles are placed on.
Pool	BubblePool	Reference to an instance of BubblePool that holds possible answers for popping bubbles.
Turret	Turret	The Turret instance that is responsible for shooting bubbles.
TimeElapsed	float	The time the bubble shooter game has been running for in seconds.
IsPaused	bool	Whether the game is paused or not.
IsGameActive	bool	Whether the game is active or not. The game is also active if the game is paused.
Finish	Action<BubbleShooterResult>	The event fired when the game is finished. Additionally, it gives results of the game.
Initialized	Action	The event fired when all game references have been set up, but the game has not started yet.

		Called before the Started event.
Started	Action	The event fired when the game is done initializing and started. Called after the Initialized event.
Paused	Action	The event fired when the game is being paused.
Continued	Action	The event fired when the game is being unpaused.

Method name	Return Type	Parameters	Description
Continue	void	config : BubbleShooterConfig	Handle everything necessary for starting a new game.
ForceFinish	void	-	Pauses the game.
Stop	void	-	Continue/resumes the game.
Pause	void	-	Restarts the game.
StartGame	void	-	Finish the game and invoke the finish Action with the results.
Restart	void	-	Reinitializes the game and starts a new instance.
InitializeGame	void	-	Sets up all necessary related references as preparation to start a game.
ShootTurret	void	direction : Vector2	Shoots the turret into a given direction.
GetColor	Color	-	Return a random color, taking weight into consideration if set up.

HexagonGrid

Property Name	Type	Description
Cells	HexagonCell[,]	A collection of cells that is held by the grid.
Width	int	Indicates the width of the grid.
Height	int	Indicates the height of the grid, excluding empty rows.
RealHeight	int	Indicates the height of the grid, including empty rows.
Updated	Action<IEnumerable<HexagonCell>, HexagonRelativityMode, bool>	The event fired when cell changes occur in the grid.
Attached	Action<Bubble, Vector2Int, bool, List<HexagonCell>>	The event fired when a new Bubble instance attaches itself to the grid.

Method name	Return Type	Parameters	Description
Clear	void	-	Clears all cells in the grid.
GetAdjacentCells	IEnumerable<HexagonCell>	position : Vector2Int	Retrieves all adjacent cells of a given position in the grid.
GetAdjecantCellsRecursively	IEnumerable<HexagonCell>	position : Vector2Int predicate : Func<HexagonCell, bool> includeSelf : bool excludedCells : List<HexagonCell>	Recursively looks for adjacent cells as long as the adjacent cell meets the predicate function.
NotifyUpdate	void	instant : bool	Notifies listeners for the Updated event that the grid has been updated.

ForceNotifyUpdate	void	-	Notifies listeners for the Updated event that the entire grid has been updated.
Touch	bool	position : Vector2Int	Tests if the cell on the given position has any adjacent cells with bubbles.
Attach	void	bubble : Bubble position : Vector2Int	Attaches a bubble to a given HexagonCell in the grid and potentially initiates a pop.
AddRow	void	-	Adds a new row of randomly generated bubbles to the top of the grid.

HexagonCell

Property Name	Type	Description
Context	HexagonGrid	The HexagonGrid context of this cell.
Position	Vector2Int	Defines where the cell is placed in the grid.
Node	Bubble	The node that is held by the cell.

Bubble

Property Name	Type	Description
Popped	Action	The event fired when the bubble pops in the grid.

Method name	Return Type	Parameters	Description
AddMatches	void	predicate : Func<T, bool> connectPredicate : Func<IEnumerable<T>, bool>	Adds a predicate function as a match for a type of bubble to match with this bubble and a predicate function that tests if the bubble should be part of a group and pop.
InvokeToPop	void	-	Notifies the listening grid that the bubble needs to be popped from the grid.
Pop	void	-	Notifies listeners for the Popped event that the bubble has popped.
IsMatch	bool	other : Bubble	Invokes matching predicate functions to test whether bubbles match one another.
IsConnectionMatch	bool	otherGroup : IEnumerable<Bubble>	Tests if the bubble is a successful poppable bubble in combination with another attached group.
Clone	Bubble	-	Copies the properties of the instance to a new Bubble instance.

Turret

Property Name	Type	Description
Bubble	Bubble	Holds the currently loaded Bubble instance that acts as ammunition.
Shot	Action<Bubble, Vector2>	The event fired when the turret shoots a bubble into a given direction.
Reloaded	Action<Bubble>	The event fired when a new bubble ammunition is put in the turret.

Method name	Return Type	Parameters	Description
Reload	void	bubble : Bubble	Grabs a new Bubble instance and sets it as ammunition for the turret.

BubblePool

Method name	Return Type	Parameters	Description
Recompute	void	-	Clears the current pool and repopulates the pool with new entries of BubblePoolSet instances.
PickBubble	Bubble	-	Picks a generated bubble from a BubblePoolSet as long as it reaches the given threshold. If no sets can be found that reach the threshold, a random set will be picked.

6. Known Limitations

- With the current implementation, the turret allows a two-dimensional direction, meaning that the turret can not shoot in the z-axis in world space.
- Due to heavy operations of the grid, increasing the size of it will exponentially reduce performance.

7. 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 centre 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/