Tower Defense 2D

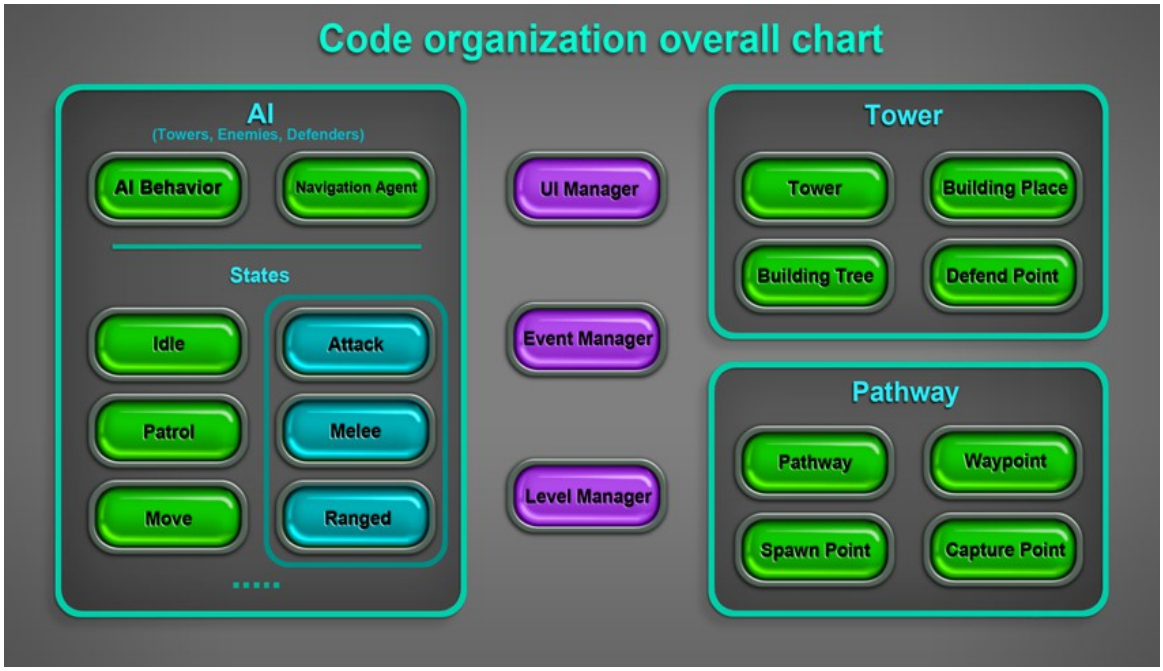
Demo

The following scenes are available:

- MainMenu - initial UI scene with “Start New Game” option;
- LevelChoose – sheet with allowed game levels;
- LevelUI – interface for game levels. It loads when level started;
- Level0 – template for game level;
- Level1 – scene with game level on simplest tower defense map;
- Level2 – one more game level with single pathway;
- Level3 – one more game level with two pathways.

Run “MainMenu” scene for asset's features demonstration.

Code organization overall chart



Editor windows

Editor windows is a user friendly interface that helps you to create tower defense game. Go to “Window->TD2D” to open the necessary editor.

Scripts description

TD2D Editor

TD2D_Levels - Editor script for levels editor window visualization. It operates only in editor mode.

TD2D_Units - Editor script for units and towers editor window visualization. It operates only in editor mode.

TD2D_Tiles - Editor script for map background prototyping. It operates only in editor mode.

Inspectors

Set of scripts that are interlayer between editor script and gameplay scripts. They operate only in editor mode.

EventManager

This is a little modified code from Unity3d tutorials. It uses to organize data transactions for all code parts without dependencies between each other.
https://unity3d.com/ru/learn/tutorials/topics/scripting/events-creating-simple-messaging-system
There are next events in game:

- UnitDie - triggers on game object with AiBehavior component destroy;
- UnitKilled – triggers when unit killed on damage taken;
- Captured – enemy reaches capture point;
- Defeat – calls when player lose the game level;
- Victory – calls when game level completed;
- TowerBuils – new tower is built;
- TowerSell – tower was sold;
- AllEnemiesAreDead – every spawn point will send this event when all specified enemies are dead;
- ButtonPressed – information about pressed UI button;
- UserClick – user click action occurred (excluding UI click);
- UserUiClick – user UI click action occurred;
- SceneQuit – new scene will load at this frame;
- GamePaused – game paused / resumed;
- WaveStart – global wave start event from timer:
- TimerEnd – wave timer stop counting;
- ActionStart – user action (spell) started;
- ActionCancel – user action (spell) canceled.

UiManager

This script is responsible for GUI displaying and for all player interactions, such as pointer click and camera drag. It coordinates transitions to:

- Pause menu
- Victory menu
- Defeat menu
- Quit to main menu

Also UiManager monitors gold amount for towers construction.

LevelManager

Controls level progress and conditions, such as:

- Defeat conditions and victory conditions;
- Starting gold amount for level;
- Allowed enemies types for this level (they will be randomly generated by spawn points)

On level start LevelUI also adds into scene. Every level also need WavesInfo to specify timeouts between enemy waves.

AudioManager

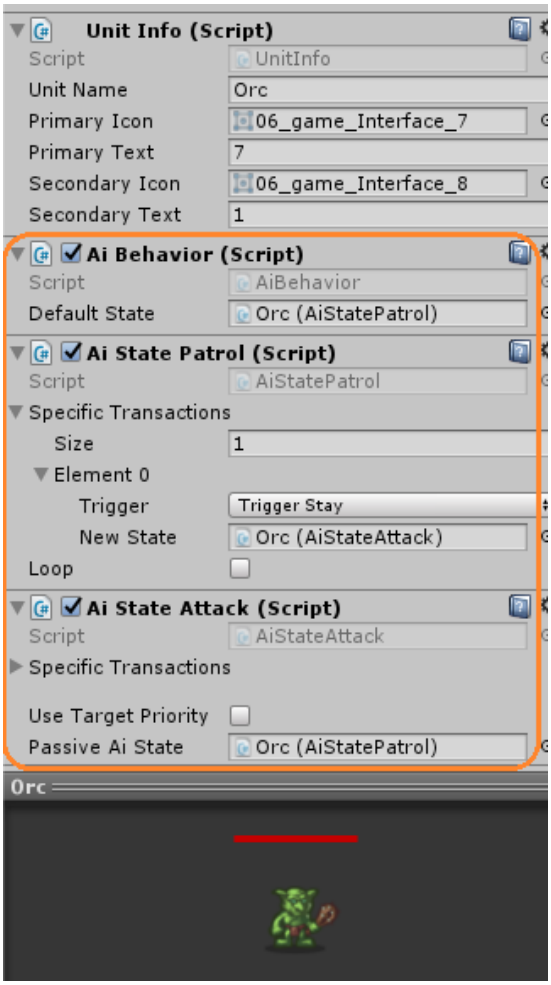
This script used for all sound effects playing. It has two audio source: one for soundtrack and one for all sound effects. It plays and stop soundtrack on game pause. Also it limits the number of simultaneous sound effects (such as attack and die). The volume settings are available for music and for sounds.

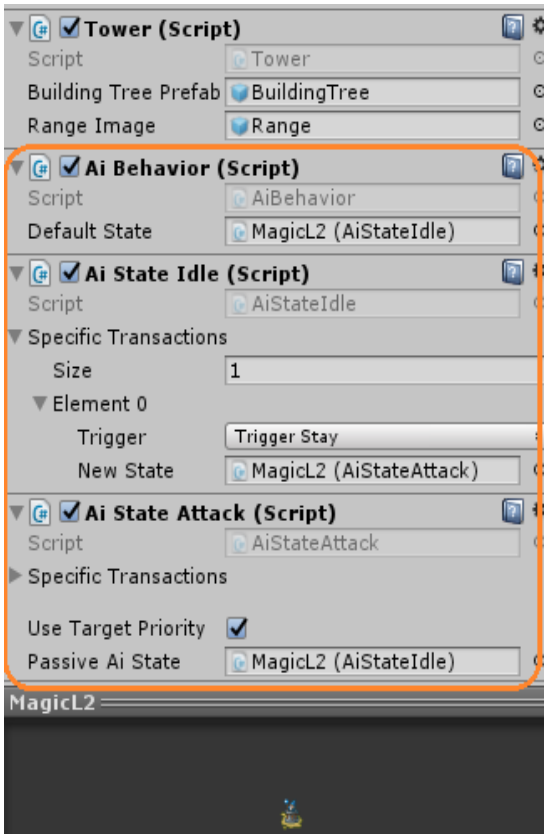
DataManager

Modified script from Unity3d tutorials. Allows to save and load game progress.
https://unity3d.com/ru/learn/tutorials/topics/scripting/persistence-saving-and-loading-data

AI

There is combination of scripts that helps to organize Artificial Intelligence for all units in game: towers, enemies and defenders.





AiBehavior

This is mandatory script for AI operation. It controls AI states switching and provides one active state at the moment. AI Behavior must have default AI State that is activated on game start.

NavAgent

This script operates movement and turning in 2D space, based on destination controlled by other scripts.

AI States

The combination of AI States determines behavior of unit. Each state is individual component added to GameObject. Switching between states is determined by states themselves and specified triggers.

AiStateIdle

The simplest AI State. Unit does not perform any actions, waiting for some events.

AiStateMove

This state is used for moving defenders after spawning. Unit moves to destination (defend point) and then go to passiveState.

AiStatePatrol

This state requires specified Pathway on scene. Unit moves along pathway from one Waypoint to another.

AiStateAttack

This state allows unit to attack other GameObjects. Attack state operates two attack types: melee attack and ranged attack. GameObject must have at least one attack type (ore both types). Attack types are attached components with IAttack compatibility (AttackMelee, AttackRanged).
Targets that can be attacked (collide) are specified by one more script AiColliderTrigger. This component has a list of variables that determines allowed targets tags (for example: there can be different targets specified for melee attack and ranged attack).

Towers

Towers are static units that are interactive for player. They can be built and upgraded.

BuildingPlace

Empty script is used to organize parent GameObject for tower building place.

DefendPoint

Every building place must have defend point. It is used for Barracks when defenders are spawned. Every defend point may have positions for several defenders.

DefenderSpawner

It used by Barracks Tower for defenders spawning. It requires defender prefab, specified spawning cooldown and defenders maximum number (at one moment on scene).

Tower Actions

Set of scripts for tower actions tree (upgrade, sell and active skills).

Pathway

This is a set of GameObjects which organize enemies waves.

Pathway

The parent GameObject, includes spawn point and way points. It is possible to have several pathways on scene.

SpawnPoint

This is an enemies waves generator for current pathway. Enemy waves have settings for waves number, personal delay before every wave, enemies counter and enemies prefabs. If enemy prefab not set the random enemy will be randomly generated from list, specified in LevelManager. The global timeouts between waves is controlled by WavesTimer script.

Waypoint

Local part of pathway. The first child Waypoint is the start of pathway.

CapturePoint

This is defeat condition. If enemy reached this point – game will end. It is possible to set up the number of capture triggers before defeat (in LevelManager).

Bullets

All ranged attacks have bullets prefabs. IBullet components control bullet fly.

BulletArrow

Allows to move bullet along ballistic trajectory;

BulletBold

It is identical BulletArrow, but bold can fly through targets.

AoeDamage

Area of Effect damage. May be attached to any bullet. On destroy bullet will cause damage to nearest targets.

AoeEffect

Area of Effect. May be attached to any bullet. On destroy bullet will apply effect to nearest targets.

Enemies

This scripts operates independently from AI states and help to organize additional functionality for enemies.

EffectControl

This script must be used for applying effects (such as slowdown, stun, immobilize and so on). It controls duration of all effects and has necessary methods for effects handling. The effects can be two types: boolean (stun and so on) and float (slowdown and so on).

AloneSpeedUp

If there is no other units around the unit speed will be increased.

AoeHealer

Heals all nearest units on cooldown.

Healer

Heals one random unit on cooldown.

Clouded and CloudOnDamage

CloudOnDamage script make clouds that cover all the nearest units.

User actions (Spells)

Spells help user to defeat the enemies.

UserActionIcon

Operates spell icon interactive and controls cooldown.

AirStrike

The type of spell suitable for meteorite, starburst and so on spells.

Other scripts

DamageTaker

This component's owner can receive damage and die. This script also controls the health bar.

Price

This simple script with one variable “price”. It is used for towers building and also for gold increment on unit die.

SpriteSorting

This script controls unit position on screen and sets sprite sorting order depending on Y position. Units with larger Y position will be overlapped by other units.

CameraControl

This script resizes camera to fit the game map and operates camera moving.

MainMenu

It is used in Main Menu scene to control new game start and application exit.

LevelChoose

It operates visual organization of game level choosing. The predefined list of levels visualization is needed. The name of level prefab must be equal the level's scene name.

UnitInfo

This component allows to set game description for object. This info will be displayed by ShowInfo in level UI when object is clicked.

ShowInfo

Displays unit information in special UI sheet.

ButtonHandler

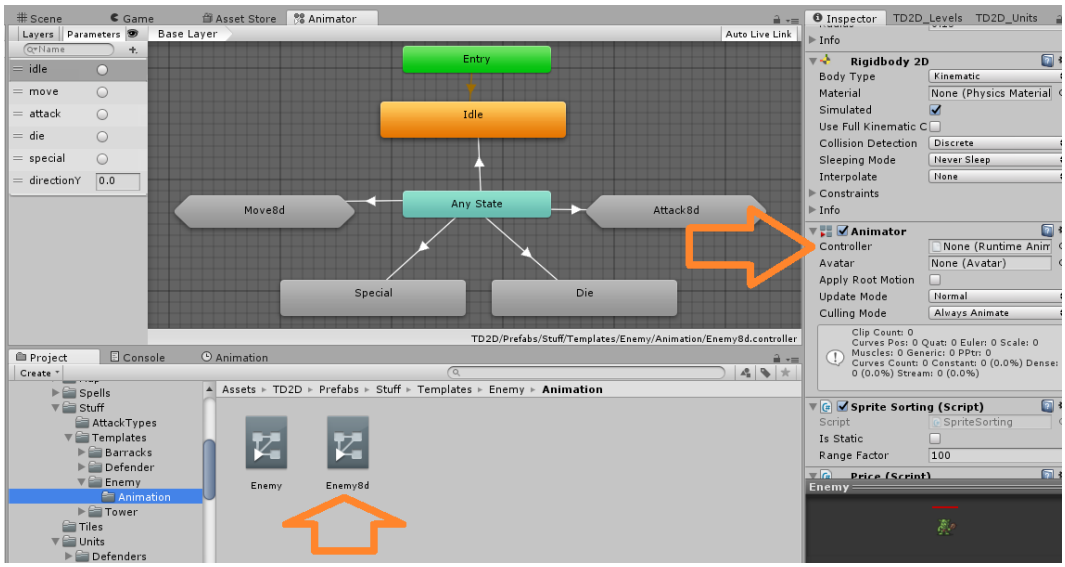
Just sends global event with information about button pressing.

WavesTimer

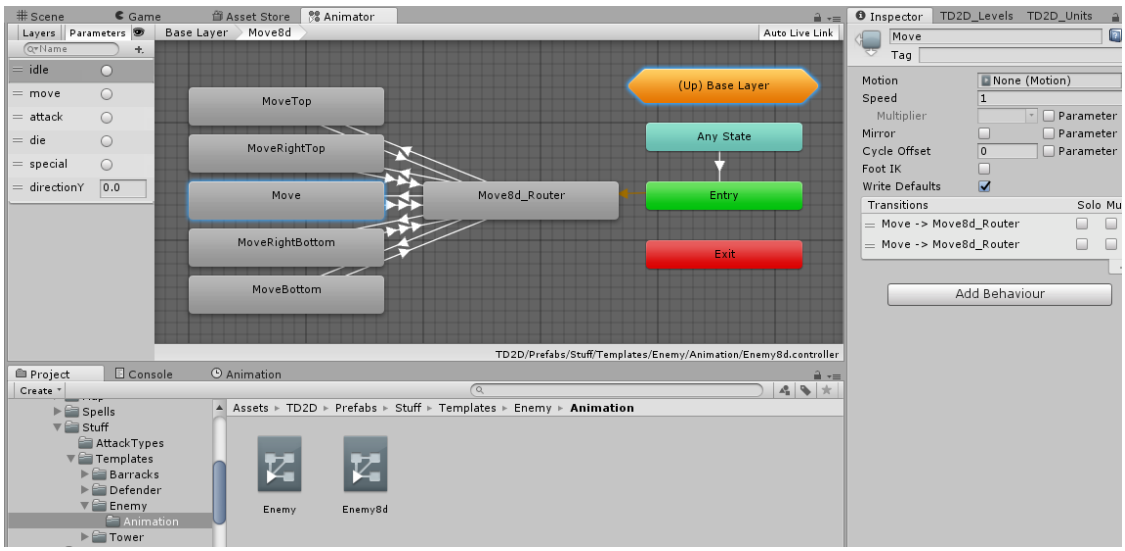
Displays information about current wave and maximum number of waves. The number of waves is equal to maximum number given in SpawnPoints. Timer uses WavesInfo to display waves information for current level.

8D animations

Asset supports 8 directions animation for units and towers. It is actual for unit's Move and Attack animation and for ranged tower's Attack animation.
When you created new unit using TD2D_Units editor window, it has two animator controllers in corresponding project folder. To implement 8D animation just set 8D controller in inspector for created unit.



8D controller has two subcontrollers Move8d and Attack8d. Double click on it to look at states.



There are 5 states for movement directions: top, right-top, right (Move), right-bottom, bottom.
Other 3 states (left-top, left, left-bottom) will be automaticaly made via scripts by flipping unit on X-axis.

The same as in tutorial you need to create animations and set it to corresponding states. Please be sure that at least one clip has name “Move” (and “Attack” for attack animations).

Fill all move animation states with corresponding clips.
Router state must have no clip (None).

You may fill several states with same animations to decrease number of needed animation directions.

The same rules for attack animations.