TurnBasedToolKit

Documentation & API Reference for Unity3D

Version: 1.0.7
Author: K.Song tan
LastUpdated: 25th March 2014
Forum: http://goo.gl/MhnkCO
WebSite: http://song-gamedev.blogspot.co.uk/

Thanks for using TBTK. This toolkit is a collection of coding framework in C# for Unity3D. This toolkit is designed to cover most of the common turn based tactic game mechanics. Bear in mind TBTK is not a framework to create a complete game by itself, but rather, the framework tries to cover most (if not all) of the common turn based tactic gameplay mechanics. The framework does not cover other game aspects such as menu scenes, options, etc.

The toolkit is designed with the integration of custom assets in mind. Although there are some minimal amount of exemplary assets included with the toolkit for demonstration, users are expected to integrate their own audio and art assets.

TBTK should be compatible and performance friendly with all of the platforms supported by Unity3D. Please note that it's not tested vigorously on mobile platform so the performance is not guaranteed.

This framework is relatively new and actively being worked on. So there are still some rough edges and probably undetected bugs. I apologize in advance if any of this become a hinder. There will be updates and improvement in the near future.

OVERVIEW

Once the TBTK is imported into the project. A new tab 'Tools/SongGameDev/TBTK' will appear on the top panel which provide a number of useful and vital item. The first being create a new scene. Then there are a series of editor-windows which used to configured all the project wide stuff like unit, abilities and etc. Finally there's a link to the the my development blog as well as unity support forum.

The most convenient work-flow to create a new level would be to use the shortcut key to create a new scene. That would setup all the component required for the setup. From there on it's simply a matter of tinkering with each components as well as edit the grid itself to get the right setting.

There are 3 vital components in each scene, GameControl, GridManager and UITB. These 3 components will take care of other dependencies. Basically the GameControl govern the basic game rules and logic, GridManager will help to generate the grid, UITB is the UI component. Each scene must contain one and only one of this component in order to function properly. Please note the UI is based on NGUI.

***Important: Do not manually alter any files placed in TBTK/Resource folder unless you know the framework very well. Most of them are important dependencies of the framework.**

Import/Update Note:

If you are updating the TBTK and don't want the new version to overwrite your current unit, unit-ability, collectible, obstacle, damage-table. Jut uncheck the following files in "TBTK/Resources/PrefabList" folder from the import window.

UnitListPrefab – for unit
UnitAbilityListPrefab – for unit-abilty
PerkListPrefab – for in game perk
CollectibleListPrefab – for collectible
ObstacleListPrefab – for obstacle
DamageArmorList – for damage-table

Please note that since v1.0.2 there has been some name change to these files so if you are updating a project from v1.0.1 or before. You will need to manually change the name of the old files to match the new one. The change are subtle so you will know what to change.

For update from v1.0.2 to v1.0.3 and up
There has been a few name change for the script. In order to gt everything working properly, you will need to do some change on the name manually. Following are the script that need to be changed:

HexTile.cs → Tile.cs
HexGridManager.cs → GridManager.cs
HexGridManagerEditor.cs → HexGridManagerEditor.cs

NGUI Support
This package support both NGUI free and NGUI full. NGUI free are included in the package.

- To use NGUI free, unpack the package named "TDTK_NGUI_Free". Make sure there's no other existing NGUI in the project
- To use NGUI full, unpack the package named "TDTK_NGUI_Full". You need to have your own copy of NGUI version3 or later.

Without doing either of those two. You wont be able to utilise the scene as all the UI are built based on NGUI. Note that they are mutually exclusive, you cant use both at the same time.

Misc
For some reason there's a disparity for the particle system rotation across Unity4 and Unity3. Since all the particle-system based indicators in this project is set to show up properly in Unity4, it will have some rotation offset when used as default in Unity3. There particle system in question are all saved as prefabs and load in runtime. You can adjust the rotation for these particle system should you wish.

EDITORS-IN-INSPECTOR

GameControl (GameControlTB.cs)

GameControl is the controlling component of the logic and state of the game. It initiates the game, determines the turn logic and when the game is won or lost, etc. A scene must always contain just one GameControl. GameControl by default will require these script component: UnitControl. AbilityManager, DamageTable, AIManager.

CONFIGURABLE

Property	Description
Turn Mode	Turn mode to be used in this scene. Can be either from following: **FactionAllUnitPerTurn** - Each faction take turn to move all its unit in each round **FactionSingleUnitPerTurnSingle** - Each faction take turn to move a single unit in each round **FactionSingleUnitPerTurnAll** - Each faction take turn to move a single unit in each turn. The round is completed when all unit has moved **SingleUnitPerTurn** - All units (regardless of faction) take turn to move according to the stats (TurnPriority). The round is completed when all unit has moved
Move Order	Unit move order to be used in this scene. Not applicable when TurnMode is set to SingleUnitPerTurn Can be either from following: **Free** - Randomise move order for the unit Unit switching is enabled **FixedRandom** - Randomise move order for the unit. Unit switching is disabled **FixedStatsBased** - Arrange the move order based on unit's stats (TurnPriority). Unit switching is not allowed
Data Load Mode	The data loading mode used for this level. Can be either from following: **UsePersistantData** – Use data carried from previous progress. Upon complete this level, the data will be saved and carry to subsequent scene. **UseTemporaryData** – Use temporary data set in any previous scene, if there's any. This data will not be carried forward. **UseCurrentData** – Use data set in UnitControl. This data will not be carried forward. Data is referring to unit composition as well as player's point(currency).

*** HotSeat Mode & Player Faction ID**	Unique numerical ID of the faction belongs to player. If none of the faction in the game has the ID specified. The entire game will be run by AI. When HotSeatMode is enabled, multiple PlayerFactionID can be specified, each respresent a different playerable faction in the scene. This subsequently allow multiple startingUnit line up in UnitControl that corresponded to each playerFactionID specified. These playerFactionIDs will also corresponded to units pre-placed in the scene or the unit generated procedurally so long as they have a matching factionID. Basically, units group with factionID that matches any of the playerFactionID specified will be regarded as a player controlled faction. Units group with factionID with no match with any of the playerFactionID will be regarded as AI faction. Please note that
Movement AP Rule	The AP cost of unit's movement. Can be either from following: **none** – Unit's movement doesn't cost any AP. **PerMove** – Unit's movement cost just one AP regardless of how far the unit move. **PerTile** – Each movement across a tile cost the unit one AP.
Attack AP Rule	The AP cost of unit's attack. Can be either from following: **none** – Unit's attack doesn't cost any AP. **PerMove** – Each attack cost the unit one AP.
Win Point Reward	point reward for winning this battle
Next Level	The name of scene to be load upon completing this level
Main Menu	The name of scene to be load as Main Menu
Enable Unit Placement	Check to enable player to hand place the starting unit before the battle start
Enable Counter-Attack	Check to enable unit to perform counter attack. Only available when attacker is within attack range. Each unit has it's own limited number of counter-attack in each round.
Restore AP at New Round	Check to restore unit ap to full at every new round. Otherwise each unit will restore AP at it's own rate, as specified by UnitEditor
Enable Vertical Component *** see note at bottom of table**	Check to use enable vertical axis in the game. Height difference of tiles will be taken into consideration. Unit on high ground will gain hit bonus, traverse through tile with different height will cost extra movement, etc.

- Minimum Height	Minimum height difference (in height level, as determined based on heightUnit) required before the hit bonus/penalty is applied. See Height Unit below for more info.
- High2Low Bonus	Hit bonus gained when a unit is attacking from high ground\ntakes value from 0-1, 1 being 100%
- Low2High Penalty	Hit penalty gained when a unit is attacking from low ground\ntakes value from 0-1, 1 being 100%
- Height Unit	Base height difference unit used to determine height level for each tile. Example: when set to 1, any tile with y-axis pos between 0-1 is considered at level-0, any tile with y-axis pos between 1-2 is considered at level-1 and so on
- Move Cost	Movement range required to traverse each height level in an adjacent tile ie. if a neighbouring tile is 2 level higher/lower, then it will cost a unit 2 movement range to move to that tile.
Enable Cover System	Check to enable cover system. When enable, unit face a hit penalty when attacking target behind a cover, and a critical chance bonus when attacking target not in cover.
- Half Cover Bonus	hit penalty when attacking a target behind a half cover
- Full Cover Bonus	hit penalty when attacking a target behind a Full cover
- Flanking Crit Bonus	Critical bonus gain when attacking an unit not in cover
Enable Fog-of-war	Check to enable fog of war system in the scene. Unit will only be able to see hostile unit within range and line-of-sight
Allow Movement After Attack	Check to allow unit to move after attack
Allow Ability After Attack	Check to allow unit to use ability after attack
Action Cam Frequency	The rate at which action-cam will be used. value from 0-1. 0 being no action-cam, 1 being always action-cam

* **Note:**
HotSeat-Mode and Fog-of-war is mutually exclusive.
Unit selection before gameplaye scene is not supported by HotSeatMode, yet.

* **Note on Height Component**
To create a scene with height component, the only requirement is collider (be it terrain or any object) with terrain layer (10) assigned. GridManager will detect the collider in the scene and adjust the grid accordingly. Collider that act as terrain can be used as cover, simply attach script Obstacle.cs to it. Note you can select the cover type (full/half) using the script.

Please see example scene “ExHeightComponent” for reference

GridManager (GridManager.cs)

GridManager is the component that govern and control the grid. On the editor end, it's uses to procedurally generate the grid in any scene. In runtime, it's responsible for all the logic concerning the grid and tile.

CONFIGURABLE

Property	Description
Randomize Grid Upon Start	Check to regenerate the grid every time the scene is loaded/played. This will override the existing grid layout and regenerate all the units on the grid as well
Randomize Unit Upon Start	Check to regenerate the units on the grid every time the scene is loaded/played. This will replace the existing unit on the grid.
Show Gizmo	Check to show the relevant gizmo on SceneView
Grid Type	The Type of grid to be generated. Can be either of the following: **Hex** – The tile is a hexagon intended for hex-grid **Square** – The tile is a square intended for square-grid
Grid Size	The size of the tile.
Num of Column	The number of column of tiles in the grid (x-axis)
Num of Row	The number of row of tiles in the grid (y-axis)
Area Height	The position of the grid on y-axis (vertical-axis)
Grid To Tile Size Ratio	The ratio of the grid layout to the tile size. Adjust this to have the tile spaced apart or overlapped. Recommended range with default tile prefab are 0.9 to 1.1
Unwalkable Tile Rate	The percentage of unwalkable tile on the grid
Tile Prefab	The tile prefab used to formed the grid. If left em
Player Placement Area	A Rect class defining the area covering the tile which player can place the starting unit, if there's any and manual unit placement is enabled. Shown by gizmo in SceneView if ShowGizmo is enabled.
Indicator	The indicator used for various purpose. This can be left blank and default indicator will be used.
Faction	The configuration for generated faction in the scene. Each faction must have a unique ID with 0 being the player's faction. Faction with same ID will be treated as same faction. The color is only for gizmo purpose and have to bearing in the game. Each faction can have multple SpawnArea. Each area can have its own designated spawn unit type and spawn count. The spawn area are shown on the SceneView with the faction's color if ShowGizmo flag is enabled.

GridPainter (HexGridManager.cs)

GridPainter is a helper script to edit the grid manually. GridPainter is by default required by HexGridManager and is dependent on HexGridManager. A GridPainter component wouldnt work when attached on a gameObject without HexGridManager component.

When a HexGridManager object with GridPainter is selected, user can directly modify the grid via SceneView with simple mouse clicking a la paint style. The extend of modifycation included state of individual tile, visibility/border of the grid, units and collectible on the grid.

*note, when in unit and collectible mode, right click on any tile containing any unit/collectible will have the unit/collectible removed.
*when in tile mode, for all feature editing except tiile state, right-click on any tile set the selected feature value back to default value (zero).

CONFIGURABLE

Property	Description
Enabled	Check to enable editing of the script using this component.
Mode	Select a feature on the grid to edit. Can be either of following: **Tile** – Edit the state of each individual tile on the grid **Unit** – Place or remove unit on the grid **Collectible** – Place or remove collectible on the grid
Tile (Mode)	
Edit Feature	The feature of the tile to be modified. Can be either of the following: **State** – The state of the tile to be set, can be default, open for placement or unwalkable **HPGainModifier** – Set the HPGainModifier of the tiles. +ve value grant extra HP per round (as specified), -ve value do otherwise **APGainModifier** – Set the APGainModifier of the tiles\n+ve value grant extra AP per round (as specified), -ve value do otherwise **DamageModifier** – Set the damageModifier of the tiles\n+ve value grant damage bonus (as specified), -ve value do otherwise **AttRangerModifier** – Set the attRangeModifier of the tiles\n+ve value grant extra attack range bonus (for range unit only), -ve value do otherwise **AttackModifer** – Set the attackModifier of the tiles\n+ve value grant attack bonus, -ve value do otherwise **DefendMofier** – Set the defendModifier of the tiles\n+ve value grant defend bonus, -ve value do otherwise **CriticalModifer** – Set the criticalModifier of the tiles\n+ve

	value grant critical bonus, -ve value do otherwise **CriticalImmunityModifier** – Set the criticalDefModifier of the tiles\n+ve value grant critical immunity bonus, -ve value do otherwise
Tile State (for state setting only)	The state of the tile to switch to when **left click** on. Can be either of the following: **Default** – Default tile state **UnitPlacement** – Player can placed unit on the tile during unit placement phase of the game **Unwalkable** – The tile is unwalkable *for unit placement state, each tile open for placement can be assigned a factionID which corresponds to to the factionID for the player's faction setup in GameControl. Only unit with matching factionID can be placed on the tile.
Unwalkable Type **(only applicable when Tile State is set to Unwalkable)**	The type of walkable to be set to the tile. A selection of available will be shown based on obstacle added to the ObstacleManager. There are two default options: **empty(visible)** – blank unwalkable tile, visible on the grid **empty(invisible)** – invisible unwalkable tile, the tile wont be visible in play.
Value (for all features except state)	The value to be set on the tile for the feature selected.
Unit (Mode)	
Unit	The unit prefab to be placed on the grid. The list is taken from UnitManager. Only unit that has been added to UnitManager can be used.
Unit Faction ID	The faction of unit being placed. Zero being player's unit, else it would be consider a hostile unit to player.
Collectible (Mode)	
Collectible	The collectible prefab to be placed on the grid. The list is taken from CollectibleManager. Only collectible that has been added to CollectibleManager can be used.
Item/Unit Rotation	The rotation of the the unit/item to be placed on to the grid, If there's any item/unit to be placed.

Tile (Tile.cs)

Tile is the script component attached on tile prefab. There's no specific configurable on the script, except the material used to assign on the tile in various state. They are as follow:

CONFIGURABLE

Property	Description
Type	The type of tile. Can be either of the following: **Hex** – The tile is a hexagon intended for hex-grid **Square** – The tile is a square intended for square-grid
Mat Normal	Material to assign to when the tile is in default state
Mat Walkable	Material to assign to when the tile is in walkable state. (To indicate that a selected player unit can walk on it)
Mat Unwalkable	Material to assign to when the tile is in unwalkable state. (To indicate the tile is inaccessible to all unit)
Mat Hostile	Material to assign to when the tile is in Hostile state. (To indicate the unit occupied the tile can attacked by selected unit)
Mat Ability Range	Material to assign to when the tile is within range of a unit abiility

UnitControl (UnitControl.cs)

UnitCotrol is the script responsible for governing the units and faction in a level. Any player starting unit for the level, if there's any, are to be setup here.

CONFIGURABLE

Property	Description
Hide Unit When Killed	When checked, the unit will appear off screen when it's killed. Otherwise, the unit will stay in the tile where they are killed.
Starting Unit	The list contains the starting unit for player. These unit will need to be placed onto the grid upon the level started. Only valid if data load mode in GameControl is set to **UseCurrentData**. If left empty, player will only have unit that are already placed in the scene. When hotseat mode is enabled, multiple list will appear according to how many player has been specified. The factionID indicate which faction the unit in the list will belongs to.

Faction Colors	The colors assigned to each faction in runtime. This is optional. Random color will be used if no color has been specified.

ShootObject (ShootObjectTB.cs)

ShootObject is the component required by any range attack unit for the shootObject.

CONFIGURABLE

Property	Description
Type	The type of the shootObject, affect the behaviour of the shootObject in runtime. Can be either of following: **Projectile** – ShootObject is an object which travel to shootPoint to targetPoint using a fixed trajectory with elevation **Missile** – ShootObject travel from shootPoint to targetPoint with varied trajectory (with addition rotation in y-axis and varied speed over the trajectory) **Beam** – ShootObject takes the form of a beam from shootPoint to targetPoint. A LineRenderer component on the shootObject is required
Shoot Effect	The gameObject intend as visual effect to be spawned at the shootObject's position when the shootObject is fired.
Hit Effect	The gameObject intend as visual effect to be spawned at the shootObject's position when the shootObject hit it's target.

Projectile (Type)

Speed	The travel speed for the projectile
Max Shoot Angle	The maximum elevation angle for the shootObject's trajectory in x-axis. Actual value is based on the range of the target
Max Shoot Range	The maximum range intended for this shootObject. Has no actual effect on game mechanic, it's just used to factor the elevation of the trajectory based on the MaxShootAngle. If the target range is beyond this range, maximum angle will be used. Otherwise the elevation angle will be adjusted porportionally

Missile (Type)	
Speed	The travel speed for the missile
Shoot Angle X	The maximum elevation angle for the shootObject's trajectory in x-axis. Actual value is randomized from zero to this value.
Shoot Angle Y	The maximum deviation angle in Y-axis of the trajectory. Actual value is randomized from -ve to +ve of this value.
Initial Speed Boost	Check to give the missile a speed boost during the initial phase of the trajectory
Beam (Type)	
Beam Duration	The active duration of the beam
Beam Length	The length of the beam. This will never exist the actual range from the attacking unit to the target. Set it to infinity and the beam will always ends at target position
LineRenderer	The LineRenderer component for this beam shootObject. Must be within the hierarchy of this prefab

** The projectile shoot elevation angle are dependent on the target's distance from the shootPoint. At MaxRange, the projectile will then be shoot at the MaxAngle.*

AIManager (AIManager.cs)

AIManager is the componet that runs the AI logic and control the AI behaviour.

CONFIGURABLE	
Property	**Description**
AI Stance	The behaviour of AI. Can be either from following: **Passive** - AI faction/unit will not do anything until hostile unit has come into range **Active** - AI faction/unit will actively move around seeking out target

PerkManager (PerkManagerTB.cs)

PerkManager is an optional script component contains the function related to Perks configured in PerkEditor. The configurable parameters in PerkManager is pretty self-explained.

There's two way in which PerkManager can be used. First is from a level to level basis where every level require a PerkManager. It can be also used for a progressive gameplay where the progress are carried forward to next level. To do this, place a PerkManager at the first scene of the game and check DontDestroyOnLoad flag on PerkManager. When this is used, subsequent PerkManager component in any level loaded will be override.

Each individual perk can be set to available/unavailable in the level (or in the game if persistent setting is used). The perks can also be set to unlocked/locked from start.

EDITORS-WINDOW

DamageTableEditor

Damage Table is an editor window used to configure the damage and armor type use in the entire project. It support multiple damage and armor type and allows adding and removing of new damage and armor type. The effectiveness of each damage type against armor type can also be configured via a set of numerical modifier. You don't need to do any setup in a scene to get damage table to work with TDTK. The functionality has already been built into the framework. You just need to configure the stats using the DamageAmorTableEditor and assign each tower/creep prefab with the specific armor/damage type ID.

The editor can be open via the Top Panel by selecting Tools/SongGameDev/TBTK->DamageArmorTable. The editor window is pretty much self-explained. Each button/entry-field has a dedicated tooltip

Each unit can now be assigned with a damage and armor (this is done in UnitEditor) type setup in this editor.

UnitManager

UnitManager is an editor window used to manage the unit to be used in the game. All unit prefab must have UnitTB.cs attached to it before it can be added to UnitManager. All unit prefab must be added to UnitManager before they can be used in any other editor (to be edit in UnitEditor, add to grid in GridManager or GridPainter, etc.)

The editor can be open via the Top Panel by selecting Tools/SongGameDev/TBTK->UnitManager.

UnitEditor

UnitEditor is an editor window used to edit various attribute on a unit prefab. Any unit prefab must be added to UnitManager before they can be edited in UnitEditor. All the editable field on the editor are pretty much self-explained and comes with tooltip

The editor can be open via the Top Panel by selecting Tools/SongGameDev/TBTK->UnitEditor.

UnitAbilityEditor

UnitAbilityEditor is an editor window used to add/edit ability that can be used by the units. Abilities created in this edtior can then be assigned to each unit.

The editor can be open via the Top Panel by selecting Tools/SongGameDev/TBTK->UnitAbilityEditor.

PerkEditor

PerkEditor is an editor window used to add/edit perk that can be unlocked by player in game. Each individual perk can then be enable/disabled in the PerkManager in each scene/level.

The editor can be open via the Top Panel by selecting Tools/SongGameDev/TBTK->UnitAbilityEditor.

CollectibleManager

CollectibleManager is an editor window used to manage the collectible prefab to be used in the game. All collectible prefab must have CollectibleTB.cs attached to it before it can be added to CollectibleManager. All collectible prefab must be added to CollectibleManager before they can be used in any other editor (to be edit in CollectibleEditor, add to grid in GridManager or GridPainter, etc.)

The editor can be open via the Top Panel by selecting Tools/SongGameDev/TBTK->CollectibleManager.

CollectibleEditor

CollectibleEditor is an editor window used to edit various attribute on a collectible prefab. Any collectible prefab must be added to CollectibleManager before they can be edited in CollectibleEditor. All the editable field on the editor are pretty much self-explained and comes with tooltip

The editor can be open via the Top Panel by selecting Tools/SongGameDev/TBTK->CollectibleEditor.

ObstacleManager

ObstacleManager is an editor window used to manage the obstacle prefab to be used on the grid. All obstacle prefab must have Obstacle.cs attached to it before it can be added to ObstacleManager. All unit prefab must be added to ObstacleManager before they can be used in GridPainter.

These obstacle is very much tied to the game mechanic. Each obstalce will occupied one tile. It will be used to determine unit ling-of-sight if fog-of-war is enabled and cover bonus if cover system is enabled. The obstacle can only be add to the grid via GridPainter.

The editor can be open via the Top Panel by selecting Tools/SongGameDev/TBTK->ObstacleManager.

OPTIONAL COMPONENTS

These are the optional components of the TD gameplay.

CameraControl (CameraControl.cs)

Script component that allows manipulation of the camera. The camera control in this script works pretty much like a 3rd person character. it pan's around the horizontal plane, zoom in/out along the view direction, and rotate around a centre anchor point. The component supports iOS as well for moving and zooming.

The component is not designed to be used on the camera component itself. Rather, the camera component should be a child transform of the gameObject with the component attached. The parent transform will act as the actor point where the rotation and zooming will be centered around on. When moving, the parent transform is moved instead of the camera transform.

CONFIGURABLE

Property	Description
Pan Speed	The speed of the camera when moving in horizontal plane.
Zoom Speed	The speed of the camera when moving zooming in/out.
Rotate Speed	Rotate speed of the camera when turning left and right
Enable Key Rotate	Enable rotating of the view angle towards left/right via q and e key
Enable Key Panning	Enable use of keyboard's w,a,s,d and arrow keys pan the camera
Enable MouseWheel Zoom	The width of the edge of screen enabled for mouse panning. Only applicable if mouse panning is enabled.
Enable Mouse Panning	Enable panning of the camera by placing the mouse at the edge of screen.
Mouse Panning Zone	The width of the edge of screen enabled for mouse panning. Only applicable if mouse panning is enabled.
Min Pos X	Minimum x-axis position of the camera's parent transform in world-space.
Max Pos X	Maximum x-axis position of the camera's parent transform in world-space.
Min Pos Z	Minimum z-axis position of the camera's parent transform in world-space.
Max Pos Z	Maximum z-axis position of the camera's parent transform in world-space.
Min Radius	Minimum distance (zoom level) of the camera component from the parent transform.

Max Radius	Maximum distance (zoom level) of the camera component from the parent transform.
Action Cam	Camera component of the action-cam object
Main Cam	Camera component of the main camera object
Action Cam Distance	The distance of action cam from the unit the camera is focusing on
Action Cam Delay	The delay in second before cutting back to normal camera during action-cam phase, after the event has finished
Show Gizmo	Check to show the relevant gizmo on SceneView

AudioManager (AudioManager.cs)

AudioManager is the component which manage all the music and sound fx in a scene. AudioManager is optional. If there isn't one in the scene, GameControl will automatically create one.

CONFIGURABLE

Property	Description
Min Fall Off Range	The minimum fall off range of all the 3D sfx. Higher value allow the sfx to be heard by the audioListener even when it's far away. Has no effect if the sfx used is 2D sound.
Music List	A list of music track that will be used as the background music.
Play Music	Check to enable playing of the background music.
Shuffle	check to enable shuffling of the music list when playing music.
Game Won Sound	Sfx to play when the game is won.
Game Lost Sound	Sfx to play when the game is lost.
Action Failed Sound	Sfx to play certain action has failed

Note: all music and sfx are optional, if left blank, nothing will be played.

SelfDestruct (SelfDestruct.cs)

SelfDestruct is a script intended to be use for any effect that is duration sensitive to the turn progression of the game, such as an visual effect that goes on as long as a buff is still valid. There are 3 modes in the depends on the need, one uses realtime, one uses round and one uses turn

CONFIGURABLE

Property	Description
Type	The type of the shootObject, affect the behaviour of the shootObject in runtime. Can be either of following: **RealTime** – use a real-time delay before the object is destroyed **Round** – count a number of new round event before the object is destroyed **Turn** – count a number of 'turn' before the object is destroyed. 'Turn' as in one cycle which all units has been moved.
Delay	Real time delay in second. For real-time mode
Round	Round count before the object is destroyed. For round mode
Turn	Turn count before the object is destroyed. For turn mode

Note: all music and sfx are optional, if left blank, nothing will be played.

ALL SCRIPT COMPONENTS

AbilityManager (AbilityManagerTB.cs)

Script component which contain the class information of ability and various unit effect.

AIManager (AIManager.cs)

Script component which determine and executing the AI move

AStar (AStar.cs)

Script component which contains all the function call involving Astar and path-finding algorithm.

AudioManager (AudioManager.cs)

Script component which manage all the music and sound fx in a scene.

CameraControl (CameraControl.cs)

Script component which manage all the music and sound fx in a scene.

Collectible (CollectibleTB.cs)

Script component for collectible

DamageTable (DamageTable.cs)

Script component which contain the class information about damage and armor type. Store the damage and armor modifier information during runtime.

GameControl (GameControl.cs)

Script component which manage all the game logic in run time

GlobalStats (GlobalStats.cs)

Script component which handle the storing and loading of player data

GridPainter (AbilityManagerTB.cs)

Script component which handle the grid editing in SceneView

GridManager (GridManager.cs)

Script component which manage and handle all the grid related logic

Tile (Tile.cs)

Script component for each individual tile

Hover (Hoever.cs)

Script component for unit animation effect for the example prefab

Obstacle (Obstacle.cs)

Script component for obstacle object on the grid

PerkManager (PerkManagerTB.cs)

Script component used governing the in game perk system

RandomRotate (RandomRotate.cs)

Script component for unit animation effect for the example prefab

Rotate (Rotate.cs)

Script component for unit animation effect for the example prefab

SelfDestruct (SelfDestruct.cs)

Script component which handle the visual effect clean up according to turn. This is intended for certain visual effect object which is intended to be last for certain round of game. The script listen to new round event and after a specified number of round will destroy the parent gameObject. Another alternate mode of the script would be to destroy the gameObject according to a pre-specified real time delay. Which is useful for on e off particle system which should be clean off after the effect ended.

ShootObject (ShootObject.cs)

Script component on shoot object which handle all the shoot object logic

TextBouncing (TextBouncing.cs)

Script component for bouncing text on UI (place unit prompt)

UITB (UITB.cs)

Script component which manage the UI element in gameplay level

UITBUnitSelection (UITBUnitSelection.cs)

Script component which handle the unit selection scene before the gameplay level

UnitControl (UnitControl.cs)

Script component which manage and handle all the unit and faction related logic in the game

UnitTB (UnitTB.cs)

Script component for each unit. Everything on this script can be configured via UnitEditor

UnitTBAnimation (UnitTBAnimation.cs)

Script component to govern all the animation of the unit. Everything on this script can be configured via UnitEditor. This script is required by UnitTB.cs

UnitTBAudio (UnitTBAudio.cs)

Script component to govern all the audio aspect of the unit. Everything on this script can be configured via UnitEditor. This script is required by UnitTB.cs

Utility (Utility.cs)

Script component to do a variety of utility function.

THANK-YOU NOTE & CONTACT INFO

Finally thanks for using this toolkit. I hope you enjoy using to create your own turn based game.

If you have build your game using this toolkit. I would appreciate if you give it some credit. Even better send or link some me some example or demo of the game. I like to know how useful the toolkit has been. I'm happy to help you promote it.

I apologize if there's still anything unclear and missing in the documentation. I also apologize for any limitation of the toolkit. If you have any question or comment, suggestion about this toolkit, or should you come across any bug, please email me directly at k.songtan@gmail.com or leave a message at the unity forum or my blog.

I'll try my best to provide support regarding issue within the toolkit. I'll also consider doing feature request. I'm very much looking forward to expand and improve this toolkit. However please understand that I'm just one man with limited time and resource. I'll happily do any feature request that is useable for other user or within reason for free. Also you may also learn that I do work as a free-lance developer. For that, I'm more than happy to help you extend the kit to accommodate any custom feature you may like to see. Thanks again!

VERSION HISTORY

Version Change – 1.0.1 (21 August 2013)

addition/change

- optimization for mobile device
- added touch input support for mobile device.
- added support for camera control via touch
- more comment and clean up has been added to the code
- change the way attack animation delay works, delay duration can now be specified manually

bug fix

- fix UI bug where ability tooltip not display correctly
- fix bug where AI unit wont pick up collectible
- fix bug where miss chance is mis-calculated and limited at 0.05
- fix bug where AP bonus on tile doesnt apply to unit
- fix bug where point reward is shown in non-persistant data mode

Fixes – 1.0.1f3 (3 September 2013)

- fix some issue which freeze unity editor when a scene is played without being setup properly
- fix bug with unit movement
- fix bug with HexGrid generation where there tiles grid is not centered
- fix bug with with ability switching while selecting target
- fix bug where unit can be moved during unit placement phase
- fix game stuck issue in unit placement phase when there isnt enough space to place all unit
- restart button in pause menu will now restart the level properly

Version Change – 1.0.2f5 (13 September 2013)

addition/change

- added XCom style cover system
- added fog of war
- added support for a all AI faction scene (no player faction)
- Total Revamp of TurnMode and MoveOrder. More variant are now supported.
- Better AI
- Added more target area for unit-abilities (line, cone)
- Editor interface change and renaming some of the unit stats.
- tweak the aiming of the range attack visually so a hit look like a hit and a missed look like a missed
- tweak the trajectory of projectile shootObject so it's more robust
- Many tweaks across the whole framework

bug fix

- lots of bug fix with regarding turn transition and logic
- fix bug with unit placement
- fix bug hit sound and hit animation is played even an attack is missed
- projectile shootObject max shootAngle limit is now increase to 75
- fix bug where projectile shootObject sometime bounce off or overshoot from the target
- and tons of other bugs.

Version Change – 1.0.3 (14 October 2013)

addition/change

- the TBTK tab on the top panel has been moved to 'Tools/SongGameDev/TBTK'
- added support fro square grid
- added new turn mode (SingleUnitRealTime & SingleUnitRealTimeNoRound), similar to SingleUnitPerTurn but faster unit get to move more in a round than slower one.
- separated grid generation and unit generation
- added point budget based unit generation
- UnitAbility can now trigger other unit abilities, enable abilities with different target area, effect, etc.
- UnitAbility can now fail/miss
- damage effect on UnitAbility now has damageType associate with it.
- and other minor tweaks.

bug fix

- fix bug with unit placement
- fix various bugs with the UnitAbility
- fix volume bug with AudioManager
- fix bug with multiple shootObjects
- fix bug when randomize level breaks the game on iOS

Version Change – 1.0.4 (19 November 2013)

addition/change

- add perk system
- add “hotseat” multi-player support
- add new ability (spawn unit, teleport)
- icons can be removed in once assigned to reduce build size
- grid generation in editor is significantly faster now.
- get rid of warning message for NGUI for unity 4.0 and up
- code improvement here and there.

bug fix

- fix a bug with GridManager faction setting and unit generation
- fix some bug with unit-abilities

Version Change – 1.0.5 (18 December 2013)

addition/change

- added wall system, now support wall between adjacent tile
- new wall system is integrated to existing cover and line-of-sight system
- duration tracking for various ability/collectible effect has been reworked
- SelfDestruct script has been updated to work in tandem with the new duration tracking
- added support for manual faction's color assignment

bug fix

- fix a few editor bug with grid/unit generation
- fix some bug with unit-abilities
- fix bug where abilities duration doesnt work correctly when using turn-mode that isnt round based

Version Change – 1.0.6 (10 Jnuary 2014)

addition/change

- added new GUI that is based on NGUI3
- added new GUI that is based on Unity GUI
- added new ability to spawn collectible
- added option to spawn collectible procedurally
- unit has the option to spawn collectible after they are destroyed

bug fix

- ability cooldown will now be tracked correctly

1.0.6f2

bug fix

- fix a few errors that happened when restarting a level
- fix the game over screen so the stats are shown/hidden properly
- fix bug where player can click on the tile through UI element on NGUI
- fix bug where overlay is not positioned correctly
- fix bug where damage overlay does not show up correctly with cover system enabled
- fix bug with unit effect where some of them will not be cleared after the effect duration has due
- remove a few warning message on unity4.3

1.0.6f3

bug fix

- fix bug where various error pops up after a game has been restart
- fix bug for default unity UI where overlay doesn't clear properly
- fix bug with with GridPainter where colletible editing doesnt work
- fix various bug for unit-abilites, colletibles, effects on unit in general
- fix bug where turn logic goes wrong in certain setting.
- fix bug where attckAPCost is not check properly.

Version Change – 1.0.7 (25 March 2014)

addition/change

- change the way grid info is serialized which should improve grid generation time and grid limit
- added height component to the grid (see example scene - ExHeightComponent)
- improve a few things about how ShootObject works
- added effect type 'damage reduction' to unit ability

bug fix

- spawn unit will now work with ability that “shoots”
- ability with shootObject will now shoot towards their target (both unit or tile) instead of just tile
- using ability no longer reduce the attack count available on a unit in the turn
- fix bug where AI unit with movement disabled will not attack
- fix bug where all attack never missed