Other Assets

Specialized Scriptable Object assets

- TilePlus Tile Asset Varieties
- Animated Tiles
- Special Tiles
- Tweener Tiles
- Zone-Based
- UI Tiles
- Event and Zone Actions and SubObjects
- Prefabs

TilePlus Tile Asset Varieties

These are all project-level asssets.

- TpTileBundle: The asset created when you create a prefab from Tilemaps.
- TpTileFab: Another asset created when you create a prefab from Tilemaps.
- ProxyTile: a special tile used by Tile+Painter. Not available in a build.
- TpPrefabList: A group of prefab references for TpAnimZoneSpawner tiles.
- TpTileList: A group of TPT tile asset references for TpAnimZoneSpawner tiles.
- TpSlideShowSpriteSet: A group of sprite references for the TpSlideShow tile.
- TpSpriteAnimationClipSet: A group of sprite references for the TpFlexAnimatedTile tile.
- TpChunkSelectorBase, TpSingleFabChunkSelector, TpChunkZoneSelector: used with the Layout system.
- TpLibInit: Controls various TpLib options when entering Play mode (or running a built game)
- TpTweenSpec: A list of specifications for tweens. Optional but useful when using the same tween repeatedly.
- TpGoTweenSpec: A list of specifications for GameObject tweens.

These assets can be created from the Assets/Create/TilePlus menu except for the first three and TpChunkSelectorBase.

TpTileBundle and TpTileFab are assets created when you use the Tools/TilePlus/Prefabs/ Bundle Tilemaps menu command.

TpPrefabList is a list of TpPrefabSpawnerItems. Each item has the following fields:

- Prefab: The prefab to spawn
- Parent: Name or Tag of a GameObject to parent the spawned Prefab to. Optional.
- UseParentNameAsTag: If a Parent is specified, interpret as a Tag if this is checked.
- Position: Position of the prefab. Can be left at Vector3.zero.
- PositionIsRelative: If checked, the Position value is relative to the tile grid position.
- KeepWorldPosition: If checked, keeps the prefab’s world position relative to tile grid position.
- PoolInitialSize: The pool preload size.

TpTileList is like TpPrefabList, but for TPT tiles.

- Tile: The tile to paint
- PaintPosition: Where to paint it. An Enum selects where to paint.
  - Around the position of the tile doing the painting.
  - At the position of the tile doing the painting if the target Tilemap is not the same.

    - A random position within the painting tile’s Zone (see TpZoneSpawner/TpAnimatedZoneSpawner)

TpSlideShowSpriteSet has a list of TpSlideClips. Each clip has the following fields:

- Name: Name of the slide show
- WrapAround: Stop at the last sprite or wrap around to the first.
- StartIndex: The starting sprite for this slide show
- Sprites: A list of sprites to display.

TpSpriteAnimationClipSet has a list of TpAniClips. Each clip has the following fields:

- Name: Name of the AniClip
- DefaultTileIndex: When animation isn’t running, which tile to use for the static sprite.
- AnimationSpeed: Speed of animation relative to that set on the Tilemap component.
- OneShot: Stop the animation at the end of the sequence or repeat.
- RewindAfterOneShot: Rewind to the first frame after a one-shot animation ends.
- **Converting from Unity sprite animation clips**
    - This asset's inspector lets you copy the sprites from a Unity AnimationClip to a new AniClip.
    - Place an input AnimationClip in the provided field.
    - Click the button: adds an AniClip with the name of the Unity AnimationClip.
    - The AnimationClip field is cleared when you click the button.
    - If the AnimationClip doesn't have any sprites then an empty list of sprites will result (don't know why I need to say this...)
    - This operation is only possible in-editor.

Please note that when adding a new clip to the asset: AnimationSpeed will be 0, will cause a runtime warning if OneShot is true. Internally, a value of 1 is used if AnimationSpeed is zero, which may produce unintended results.

Animated Tiles

TpAnimatedTile

TpAnimatedTile is a simple animated tile and if you want to learn about the code, it's the simplest TPT Tile.

Public fields:

- PlayOnStart: Begin animation when the game starts.
- AnimationSpeed: Set animation speed relative to that set in the Tilemap.
- OneShot: Play the animation once, then stop.
- Rewind After One Shot: if checked, after a one-shot animation, the animation rewinds to the first sprite.
  - To return to the Tile's sprite, use the ActivateAnimation method to turn off animation.
- Update Physics: see the description of this Tile Animation Flag in the Unity documentation.
- AnimatedSprites: A list of sprites to animate (use an Inspector on the Project asset).

Note that for TpAnimatedTile the static tile is the tile sprite. For TpFlexAnimatedTile the sprite specified by a clip's DefaultTileIndex value is displayed. If that doesn't work (it's incorrect or the sprite is null) then the static tile is the tile sprite.

If painting this tile via script, see the FAQ entry "Animated tile not animating."

TpFlexAnimatedTile

TpFlexAnimatedTile is an upgraded TpAnimatedTile that adds the ability to have multiple animation sequences contained in an asset file. Once placed, you can select on a per-tile basis which animation sequence is used initially, and several other settings, including whether to play automatically when the Scene is loaded, and which sequence is in use. When using many animated tiles, the use of an asset is more memory efficient than TpAnimatedTile.

The asset file for this tile is TpSpriteAnimationClipSet, and its fields are mostly the same as those for TpAnimatedTile. You can create it from the Assets/Create menu.

Public fields:

- PlayOnStart: Begin animation when the game starts, using the preset animation sequence.

- DefaultSprite: Sprite to display when nothing else is available, for example, a missing ClipSet asset.
- ClipSet: The TpSpriteAnimationClipSet asset from your Project folder.
- UseAnimationSpeedOverride: Don't use the animation speed from the Clipset.
- AnimationSpeedOverride: Use this animation speed if UseAnimationSpeedOverride is checked.
- ForceOneShot: Ignore the OneShot setting in the ClipSet asset and always play one-shot animations.
- Rewind After One Shot: after a one-shot animation is completed the animation rewinds to the first image. Overrides the RewindAfterOneShot setting in the animation clip IF ForceOneShot is checked.
  - To return to the Tile's sprite, use the ActivateAnimation method to turn off animation.
- Update Physics: see the description of this Tile Animation Flag in the Unity documentation.

When inspecting one of these tiles using the Selection Inspector, you'll see a dropdown Clip to use where you can select the animation sequence from the ClipSet to be used when your game starts. This can also be changed via code.

If painting this tile via script, see the FAQ entry "Animated tile not animating."

TpSlideShow

TpSlideShow lets you display one Sprite at a time from a list of Sprites contained in an asset file. The initial Sprite to display can be changed, and you move from one Sprite to the next programmatically, with automatic wrapping or limiting, or set the displayed sprite directly.

Wrapping means that incrementing from the last slide returns to the first slide (or when decrementing, from the last slide to the first slide) and limiting means that incrementing from the last slide or decrementing from the first slide has no effect. This tile is used for the background in the BasicTiles demo.

The asset file for this tile is TpSlideShowSpriteSet, and its fields discussed in the Programmer's Guide. You can create it from the Assets/Create menu.

Public fields:

- SlidesClipSet: The TpSlideShowSpriteSet asset from your Project folder.
- SlideShowAtStart: The name of the slide show to show when your game starts.
- WrappingOverride: Override the 'wrap' setting from the TpSlideShowSpriteSet asset.
- SlideIndexAtStart: is useful to set which slide is used when the app starts, and you can set this in the UI.
- CopyToSlideIndex: if checked, the ChangeSlide buttons copy the current slide index to SlideIndexAtStart.

    - Convenience function useful when using only one slide show from a slide show sprite set.
- Trigger On Value Change: if checked, posts an Event when the slide changes.
- Accepts Clicks: if checked, accepts ActionToTile messages.
- Zone Capability: controls propogation of messages.

The last three features listed above are discussed here.

When inspecting one of these tiles using the Selection Inspector, you'll see a dropdown where you can select which slideshow set from the TpSlideShowSpriteSet to be used at start. This can also be changed via code.

Technically, the SlideShow tile uses the Tilemap's animation system but the animation is always on pause. Changing slides changes the animation parameters but only one sprite (slide) is shown at a time.

AnimatedSpawner

Animated Spawner is a subclass of TpFlexAnimatedTile. It responds to ActionToTile packets via the Messaging system or one can directly call methods that will spawn a tile. All of the Animation methods of FlexAnimatedTile are of course available.

Public Fields:

- PrefabList: a reference to a PrefabList asset with the prefabs that might be spawned.
- SpawnMode: spawn prefabs in the order found in the PrefabList or spawn random prefabs from the asset.
- PositioningMode: how to position prefabs or tiles. UseAssetSetting uses info from asset. Any other ignores that info.
- KeepWorldPosition: Keep world position when a Prefab is spawned.

One might note that AnimatedSpawner and FlexAnimatedTile both have explicit declarations of MessageTargets (for the Messaging system) which take ActionToTile packets.

This shows the power of Explicit declarations: the MessageTarget in the FlexAnimatedTile superclass is ignored by the Messaging System because sending a message to an AnimatedSpawner instance is sent to the 'override' method with the highest level explict declaration. (oversimplification).

Special Tiles

TpBundleTile

TpBundleTile loads a TileBundle to the Tile's parent Tilemap.

- TileBundle: a reference to a Bundle in the project.
- ApplyMatrix: Apply the Matrix to all tiles in the bundle. Ignored if Matrix is invalid
- TptNewGuids: Apply new GUIDs to TilePlus tiles? RECOMMENDED: TRUE
- Matrix: Matrix to apply to all tiles.

This tile has a custom editor, use it to set up the Matrix via inspecting the project asset.

Immortalizer

TpImmortalizer tiles may be painted into a Zone (ie a square area of a particular size eg 8x8, 16x16 etc) to mark that Zone as Immortal when used with the Layout system. It's not useful outside of that environment.

Note that this tile is a convenience, but the implementation is part of your app, see the Layout for an example.

Tweener Tiles

TweenerFlex

This is the best tile to use when experimenting with tweens. When viewed in a TilePlus Selection inspector only the fields appropriate for the particular 'target' are displayed.

Note about Color tweens: there's no Ease Function for these. 'Lerp' is always used.

Not shown in these images: the Flex and Sequence tiles have `Await Result` *toggles.*

- These are used as an example to show how to make a tween or sequence Awaitable.
- Note that infinite loop count (<0) cannot be made awaitable and an error is logged to the console.

For example: scale

TweenFlexScale.png

For example: matrix

MatrixTween2.png

When tweening any of the Matrix varieties, *EaseFunction* is used for the Position tween, *EaseRotation* is used for the Rotation tween, and *EaseScale* is used for the Scale tween.

Matrix Hints

When using Matrix tweens you can tween the entire matrix (position, rotation, scale) or any of the components.

Tweener-matrixhints.png

One can use this to create, say, a Matrix tween that only affects rotation and position but not scale. This is equivalent to executing a Position and Rotation tween at the same time.

TweenSpec

This tile allows you to use one tween from a TweenSpec asset.

TweenSpecSeqTile.png

The Spec Index determine which tween from the asset is used.

TweenSpecSequence

This tile allows you to run an entire sequence from a TweenSpec asset.

TweenerSeqTile.png

It just plays the entire sequence. You can control the number of loops.

If `Loop Interactive Mode` is checked, the sequence is forced to not loop.

When the sequence completes, it restarts with a fresh set of value fom the Tween Spec asset. Hence, if you change values of this asset while the editor is Playing, such changes will be seen when the sequence restarts. As is true for project assets, the changes will remain after you exit play mode. It's a great way to play with sequences and Matrix-style tweens.

The TweenSpec asset

This is a project-level asset where you can set up one or more tweens. Use them independently with the TweenSpec tile or as a sequence using a TweenSpecSequence tile.

This asset has a custom UIElements inspector that works similarly to what you see in the TweenerFlex tile.

A great way to play with tweens and learn how they work with tiles is to use the TweenSpecSequence tile with the Loop Interactive Mode active. You can change the fields in the TweenSpec asset and see the change the next time that the sequence loops.

The TpGoTweenSpec asset

This is a simular project-level asset used for setting up GameObject tweens. See this for more information.

Zone-Based

AnimZoneLoader

This class is a subclass of TpFlexAnimated tile and inherits its fields and editor appearance.

It’s used to load archived Tilemaps from TpTileFab assets, which are created by the Tools/TilePlus/Prefabs/Bundle Tilemaps command.

IMPORTANT

It's somewhat obsoleted because of the Layout system and may NOT be compatible with it.

Public fields:

- Loading Offset: the location where the TileFab will be placed.
- Preview: preview the TileFab at the Loading Offset.
- TileFab: A TileFab archive from a Project folder.
- UseZoneManager: Optionally use a Zone Manager for detection of already loaded TileFabs.
- ShowOffsetPositionGizmo: show a marquee at the

Zone Managers and their uses are discussed here.

More about Preview

Preview loads the tiles into a preview tilemap or maps, if there are multiple TpTileBundle assets referenced by the TpTileFab asset. The preview is active until you click Preview again or the Editor’s Selection changes. When preview is active, you can change the loading offset and the preview area will change position.

Loading or previewing tiles depends on there being compatible Tilemaps that are named or tagged in such a way that the system can identity which Tilemap to use. This information is embedded in the TileFab asset when you create it. If you need to change it just edit the asset’s TileAssets section in the Project window.

The tile does not automatically load Tilemaps at runtime. Rather, you send a message to it via the TpLib SendMessage methods. As configured, it expects the message to contain a Vector3Int describing a position. If the position is within the Zone bounds, the tile uses TpLib’s PostTileEvent

method to post a trigger event.

AnimZoneSpawner

This can be used to spawn prefabs and TPT tiles, using assets with lists of prefabs or tiles. This tile uses the Spawner service.

Prefabs can be unparented or parented to a Scene Object by using the GameObject name or tag.

Tiles can be painted on the same tilemap or on a different tilemap which you specify using a GameObject name or tag, or a reference. This tile can respond to a SendMessage containing a Vector3Int describing a position. If the position is within the Zone bounds, it will spawn prefabs or paint tiles as you've configured it. Alternatively, you can spawn/paint via code using the instance methods SpawnPrefab or PaintTile if this approach doesn't suit you.

You can use either a TpTileList or a TpPrefabList asset (or both) to specify which tiles or prefabs (or both) to use with this tile.

Public fields include those from TpAnimZoneBase and its superclasses, and these additional fields. Public fields:

- PrefabList: a reference to a TpPrefabList asset. Note: preload amounts are set in the asset.
- TileList: a reference to a TpTileList asset provides a list of TPT assets along with the spawn position for each. Spawn position is one of eight positions immediately surrounding the tile position on the Tilemap, a random position in the Zone, or the painting tile's position if the painting is to a different Tilemap.
- SpawnMode: spawn prefabs or paint tiles in the order that they appear in the asset or randomly from those assets.
- Use Trigger: If checked, a the tile posts an event whenever spawning occurs.
- PositioningMode: spawn or paint using the setting in the assets, or force to the Spawner's position, the contact position in the Zone, or somewhere Random in the zone.
- ParentingMode: Tiles painting only, use the same tilemap to paint tiles (can't paint at the Tile's position), or use a provided reference (Painting Tilemap), or use a Tag to locate a Tilemap's GameObject using PaintingTilemapNameOrTag, or use a string to Find a Tilemap's GameObject using PaintingTilemapNameOrTag.
- PaintingTilemap: Tile painting only: Optional reference to alternate Tilemap used to paint tiles.
- PaintingTilemapNameOrTag: Tile painting only, string with name or tag of alternate tilemap.

Tile Painting

When doing Tile painting you have several configuration options using PositioningMode, PaintingTilemap, PaintingTilemapNameOrTag, and Parenting mode. When painting tiles it's

important that the tile which is doing the painting is not painted over. Furthermore, since the tile occupies a position on a tilemap, you probably don't want it to appear as an obstacle to a Player or NPC walking through the tile's trigger zone (TpZoneBase or TpAnimZoneBase).

All these controls exist so that you can paint the this tile on a different Tilemap from the one that you use for pathfinding and character movement.

Paint one of these tiles on Tilemap A and use it to paint tiles on Tilemap B. When you do that, the tile needs to know which Tilemap to use. That's done by setting ParentingMode appropriately and filling-in correct values for PaintingTilemap or PaintingTilemapNameOrTag.

Prefab Pooling in TpAnimZoneSpawner

AnimZoneSpawner uses the Spawner service which provides GameObject pooling.

When a TpAnimZoneSpawner tile instance's StartUp is invoked and if SpawnMode is set to RandomPrefabs or PrefabsInOrder and if the TpPrefabList has any prefabs where the PoolInitialSize is nonzero, the tile will ask SpawningUtil to preload prefab instances. This only occurs on the first StartUp and will not occur if you change the TpPrefabList asset during runtime.

Note that preloading takes time, as each prefab must be instantiated, so choose the preload sizes carefully. If there's no preload, the pool expands automatically as prefabs are instantiated and spawned.

Normally the pooler does not attach the pooled and/or preloaded prefabs to a parent GameObject. If this bothers you, head over to the Project folder Plugins/TilePlus/Runtime/Assets and drag the Tpp_PoolHost prefab into your scene. This prefab has an attached component which sets DontDestroyOnLoad so that the prefab persists between scene loads. The pooler looks for a GameObject with this specific name.

The pooler will automatically add a component of type TpSpawnLink to spawned prefabs if it doesn't already exist. TpSpawnLink can also despawn the prefab after a timeout.

New In Version 5: if the spawned GameObject has a Collider2D or Collider component AND the TpSpawnlink instance on that GameObject has its m_IgnoreCollider field = false THEN a reference to the spawned GameObject is added to an internal HashSet of "Collidables" and the callback `OnCollidableObjectSpawned` is invoked.

Similarly, if a Collidable is despawner by the Spawner service the callback `OnCollidableObjectDespawned` is triggered.

A method `IsCollidable()` can be used to see if a GameObject is in the HashSet.

This feature was added as part of the Layout system upgrade, and is used to keep track of prefabs that are spawned by tiles or otherwise – its used to ensure that spawned prefabs in a Chunked Zone (region) can be removed when the Zone's tiles are removed.

This can get complicated since spawned GameObjects may not be in the same position when they're deleted. Keeping track of them ensures that they are removed only when the Zone that they're actually in is removed.

- You can see this in action during the Layout demo. The NPC characters are collidables; they move around but are despawned based on where they are when the Zone is deleted rather then where they were initially spawned.
- When using the Layout system this is all handled automatically.

The Layout system removes all tiles and GameObjects in a Zone, but it only handles GameObjects parented to one of the Tilemaps controlled by the layout system. The Layout system doesn’t know what GameObjects might have been spawned by tiles (such as TpAnimatedSpawner) or exactly what Zone the GameObjects might be in since they might have moved from their original spawning location.

ZoneAnimator

Like other Zone-type tiles, it depends on you sending it a message with a position using TpMessaging, the TilePlus messaging system. Depending on how the Tile is set up, various actions will occur at StartUp() and/or when the tile’s internal state changes from ‘position is within the Zone’ to ‘position is outside of the Zone’.

That state change occurs when a message with a new position being sent to the tile.

The Zone is set up in the TilePlusBase section seen when inspecting the tile.

It's merely a BoundsInt which describes an area relative to the tile’s position on a Tilemap.

The tile’s code is set up to handle six basic types of operation based on a value from the PrefabMode enumeration:

1. PresentWhenNotInZone: Spawn the prefab when the tile’s Start method is executed; despawn it when the Zone is entered and respawn it when the Zone is exited.
2. PresentWhenInZone: Spawn the prefab when the Zone is entered and despawn it when the Zone is exited.
3. SpawnInZone: Spawn the prefab when the Zone is entered. Never re-spawn.
4. DespawnInZone: Spawn the prefab when the tile’s Start method is executed; despawn it when the Zone is exited. Never re-spawn.
5. AnimInZone: Spawn the prefab when the tile’s Start method is executed. Animation on/off when the Zone is entered/exited.
6. AnimOutZone: Spawn the prefab when the tile’s Start method is executed and activate animation. Animation OFF on Zone entry and ON again on Zone exit.

The various state specifications change depending on which mode you select.

As you can see, some of these modes allow you to optionally start animations when the prefab is spawned. AnimInZone and AnimOutZone always require animations unless all you want to do is spawn a prefab at the tile's position when StartUp runs; there are much simpler ways to spawn prefabs.

There's also the option of using scriptable object plugins: TpAnimatorActions to handle animations and/or control prefab spawning for special cases; e.g., spawning a timed series of prefabs.

The base code will use the first animator that it finds via FindComponentsInChildren.

Optionally, the code won't halt animation when the Zone is exited; rather, you can have a StateMachineBehaviour do it; use the base class AnimStateTilePingerBase as the base class for your StateMachineBehaviour.

Those interested in using this complex tile should check out the source code and this.

UI Tiles

See the UI chapter.

Event and Zone Actions and SubObjects

AnimatorAction

Animator Actions are project-level scriptable object plugins. They're only used with TpZoneAnimator tiles.

If you specify an action for one of the three sets of Actions (in the tile) it will ALWAYS be used instead of one of the specified 'StateName' fields.

The plugin get passed the tile instance. Inherit from the TpAnimatorAction class (see the example in the Layout demo).

The base class doesn't do anything and there's no asset create menu item for it.

If you specify an action for one of the three sets of Actions (in the tile) it will ALWAYS be used instead of one of the specified 'StateName' fields.

You can do whatever you want within the Action and the tile instance gives access to all the fields/properties like Target (the prefab) and PrefabAnimator.

Note that these values may be null depending on when an Action is invoked, e.g., when an ActionAtStart is used the animator may be null.

For example, if you want to use 'setfloat', 'setbool' etc: use an Action. This tile only uses the Animator.Play methods (though you can hack this code in a derived class).

If an Action spawns any prefabs and you want such prefabs to be auto-deleted when the Player moves out of the zone (ie when you use messaging to send the position to the tile and the position is outside of the zone) then add refs to these prefabs via AddSpawnedGameObject.

An Action can use its parent ZoneAnimator tile's `CleanupPrefabs()` to delete the prefabs; normally this is done automatically when the zone is exited OR if an AnimatorControlPacket causes a despawn (see below).

TileEventAction

These are project-level scriptable object plugins added as references to TPT tiles.

Takes an action when a TileEvent is evaluated by a controlling program. If the program uses TpEvents.ProcessEvents the Exec method in these plugins can be optionally invoked automatically.

See Events.

TileZoneAction

These are similar to EventActions, but aren't automatically invoked in TilePlus. Generally the Exec method of these plugins is invoked by a TPT tile while acting on a message. See ZoneActions.

TweenerSubObject

This is a scriptable object plugins added to Event or Zone actions for adding tween capability to those plugins. See this.

UiButtonEventAction

An example Event Action for the UiButton tile that changes values on specified tiles using the UiControl interface.

ZoneActionRadio

An example Zone Acton for the UiToggleButton tile. This implements a radio-button set of toggle buttons using a Zone or a tag.

Prefabs

TPP_PoolHost

Add to a scene to use this as the parent of spawned prefabs.

TpTriggerZoneSprites

A grid-like sprite for use as sprites - handy for editing tile Zones. Use as your tile's sprite and check 'modify sprite'.