RTSEngine User Document

RTSEngine is a beginner friendly RTS battle simulation kit. It is specifically designed so that beginners get most out of it with little effort with command, formation API calling. **Capability of handling mass battle simulation smoothly is its key strength**.

RTSEngine provides Intuitive, modular APIs that heavy lift complex battle simulation (strategic targeting, attacking) with clear documentation of each RTS unit, which allow you to focus on creating epic strategy and thrilling battle abstracting the complexity of unit arrangement, path finding and batch execution of mass units.

Key value proposition:

- Handle a large number of RTS units with smooth performance.
- Select and Command thousands of units on flat ground.
- Tune and balance game dynamics with given tuning APIs.
- Reskinable UI. Customize plugin provided UI widget with your own style.
- Customize RTS unit prefabs. Replace prefab AI.
- Drag and drop prefab RTS units from list view and watch them functioning like magic.
- Effortless real time formation drawing with selected RTS prefab.
- Formation editor - Write ASCII based AI battle formation and load.
- Load preconfigured formation from text file, place and move formation
- Well-documented core tactics definitions.

Technical features:

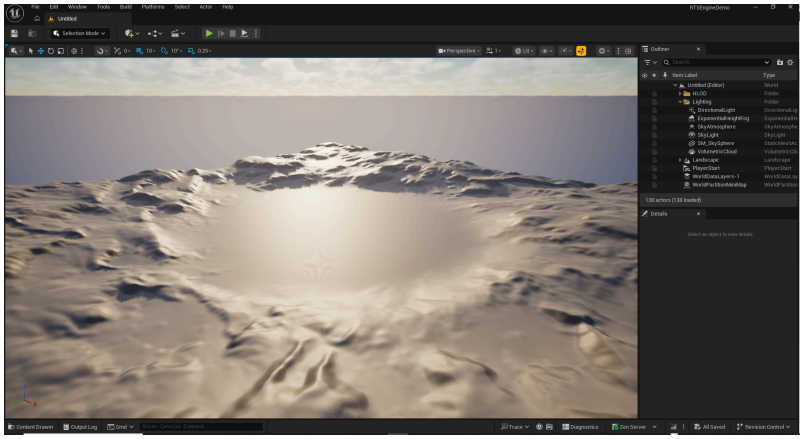
- Blueprint-scriptable APIs
- Intuitive, modular API design with logical separation of functions.
- Callback event-driven design for custom RTS logic injection.
- High performance Entity Component System based Architecture.
- Stable, crash-resistant code with clear error handling.

How to minimally integrate RTSEngine plugin in your project

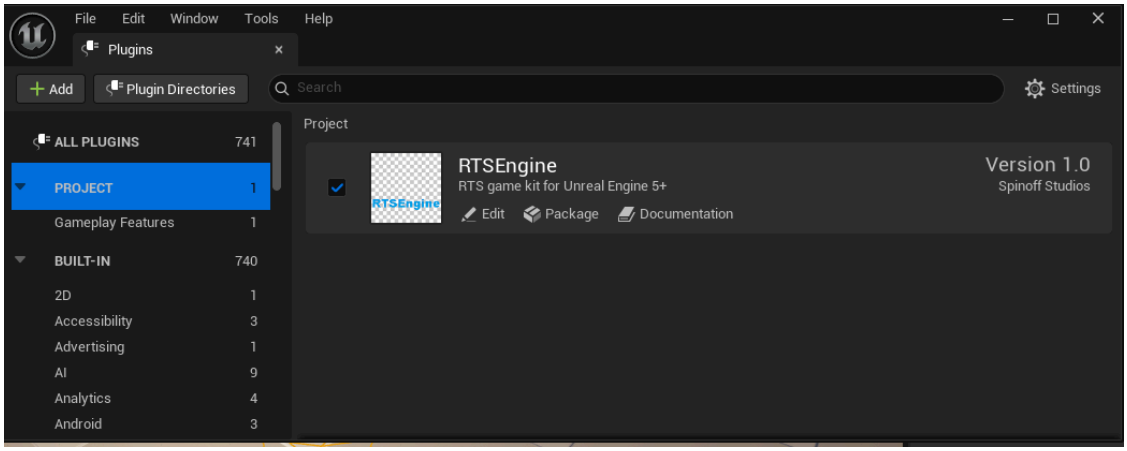
First check if your current Unreal Engine version matches with the RTSEngine plugin version. If you are using UE 5.7 then RTSEngine should be 5.7.0 You can find the RTSEngine version in the Plugins/RTSEngine/**RTSEngine.uplugin** file. Version mismatch may result in failure to load the plugin in the final packaged build of your game.

With following steps you can minimally integrate RTSEngine to your game project.

1. Create a blank blueprint or C++ project (We are using UE 5.7) with a default open world map.


TOq

2. Make sure the RTS plugin exists in your project. Go to the Edit/Plugins menu to open the plugins setting window. RTSEngine should be under the project category.



Update .uproject file:

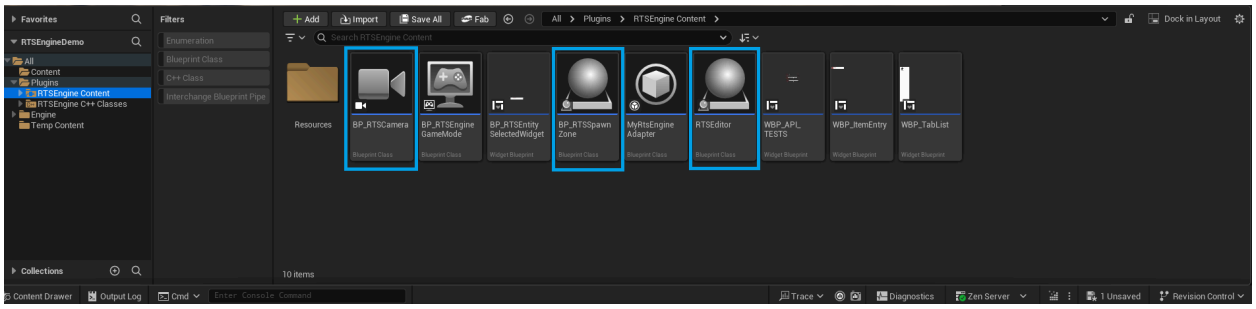
Add following new plugin entry to “Plugins” key to your .uproject file

```
"Plugins": [
        {
            "Name": "RTSEngine",
            "Enabled": true,
            "TargetAllowList": [ "Editor", "Game" ]
        }
```

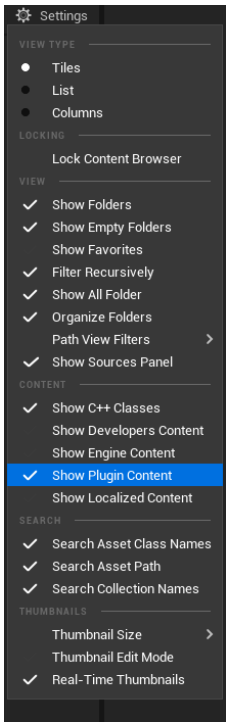
]

Browsing Plugin Content

3. Click the content drawer on the bottom left of the UE main window.



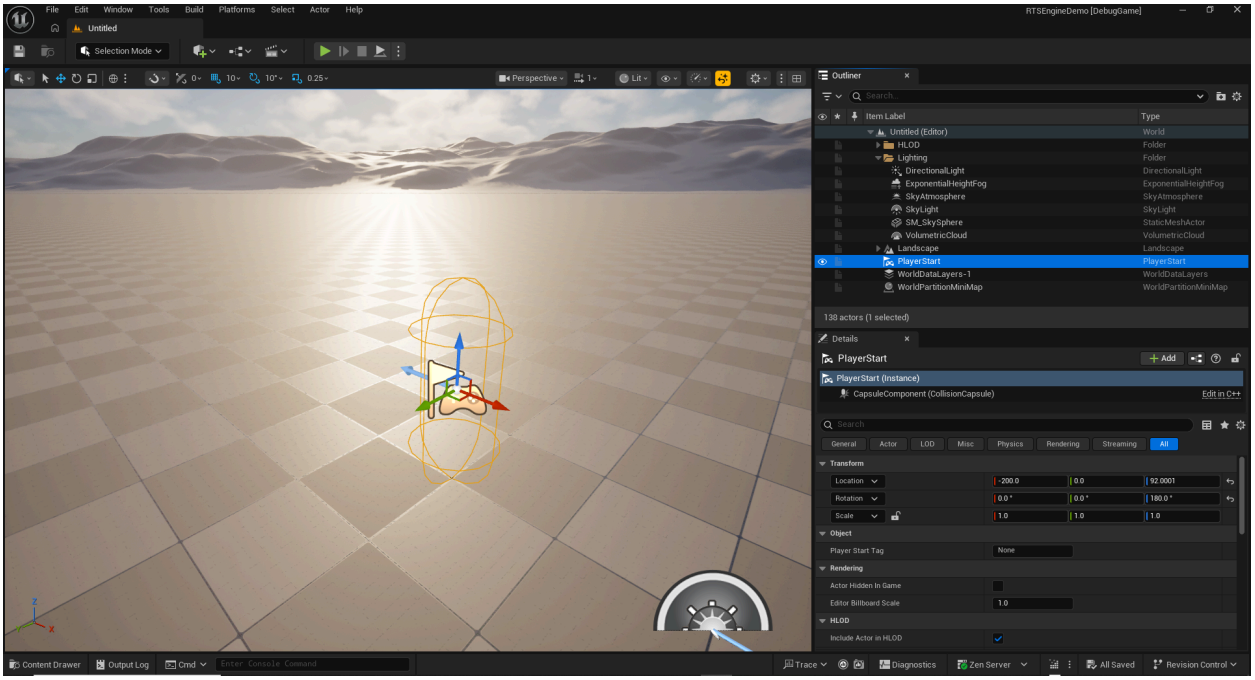
If you don't see RTSEngine content, on top right of the Content Drawer or Content browser window there is a small gear icon called settings, click on it. Now select show plugin content as follows, this will reveal the RTSEngine plugin content folder.



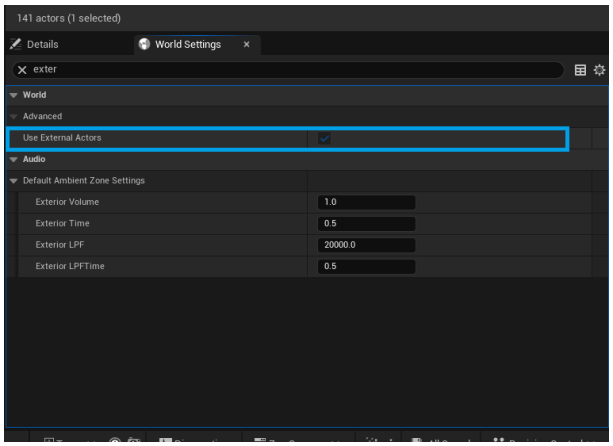
4. Expand Plugins/RTSEngine Content from the left panel of the Content Drawer.
   On the right side there should be **BP_RTSCamera**, **BP_RTSSpawnZone**, **RTSEditor** as highlighted in blue in the Content Drawer screen shot in step 3.

Placing required plugin content on scene.

5. Zoom in to the currently active and visible world partition in the scene, usually double clicking PlayerStart actor on Outliner.



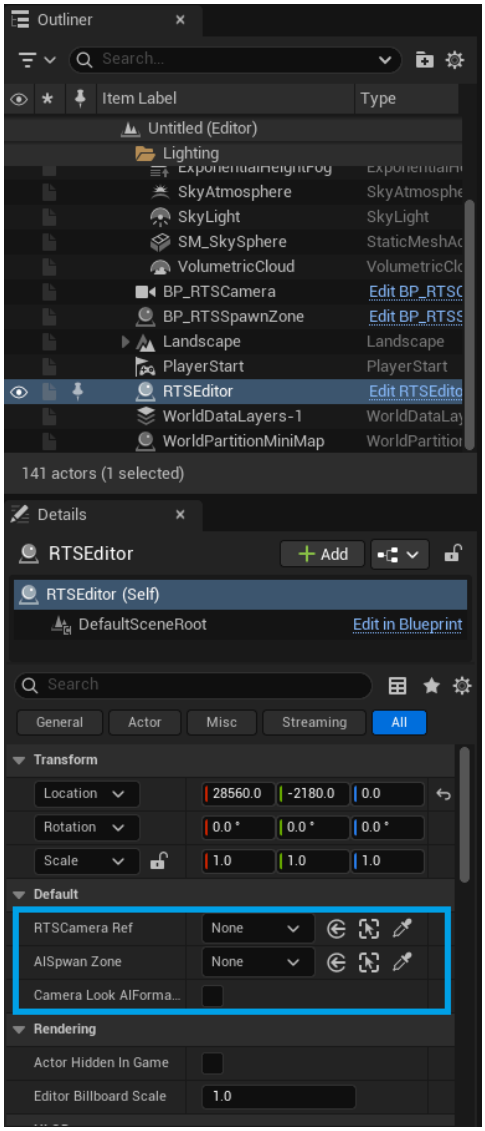
6. Additionally, check the Use External Actors from World Settings. This ensures plugin content is properly referenced in the level map.



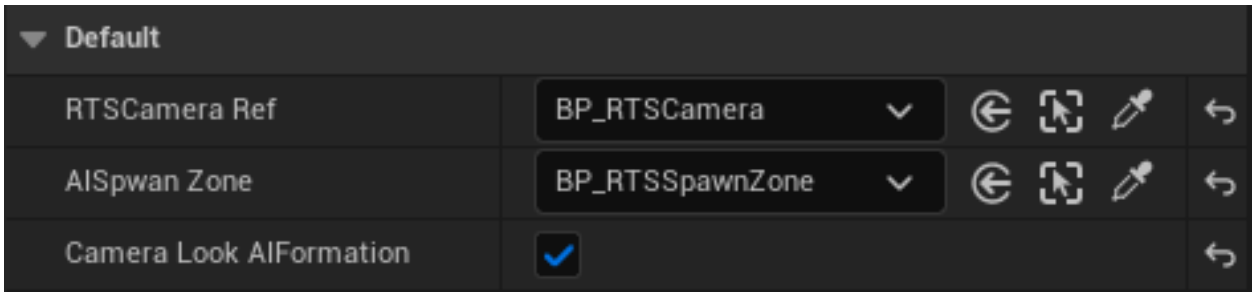
7. Drag and drop BP_RTSCamera, BP_RTSSpawnZone,RTSEditor from Content Drawer to scene.

Configuring RTSEditor Actor

8. RTSEditor checks and auto links required **BP_RTSCamera**, **BP_RTSSpawnZone actors** dropped in the scene**.** Select RTSEditor from scene or from outliner tab to edit configuration.

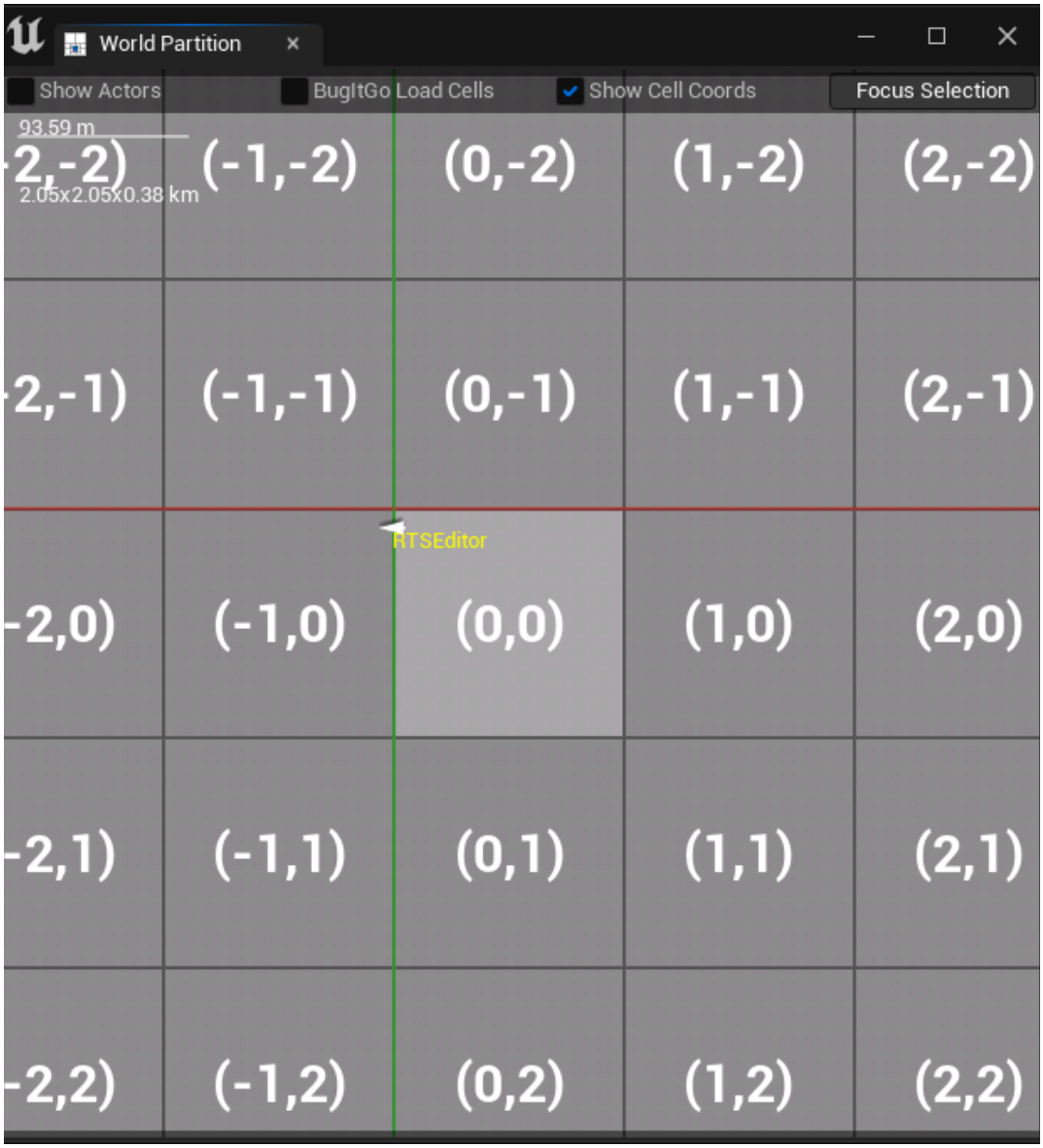


Final configuration should look like

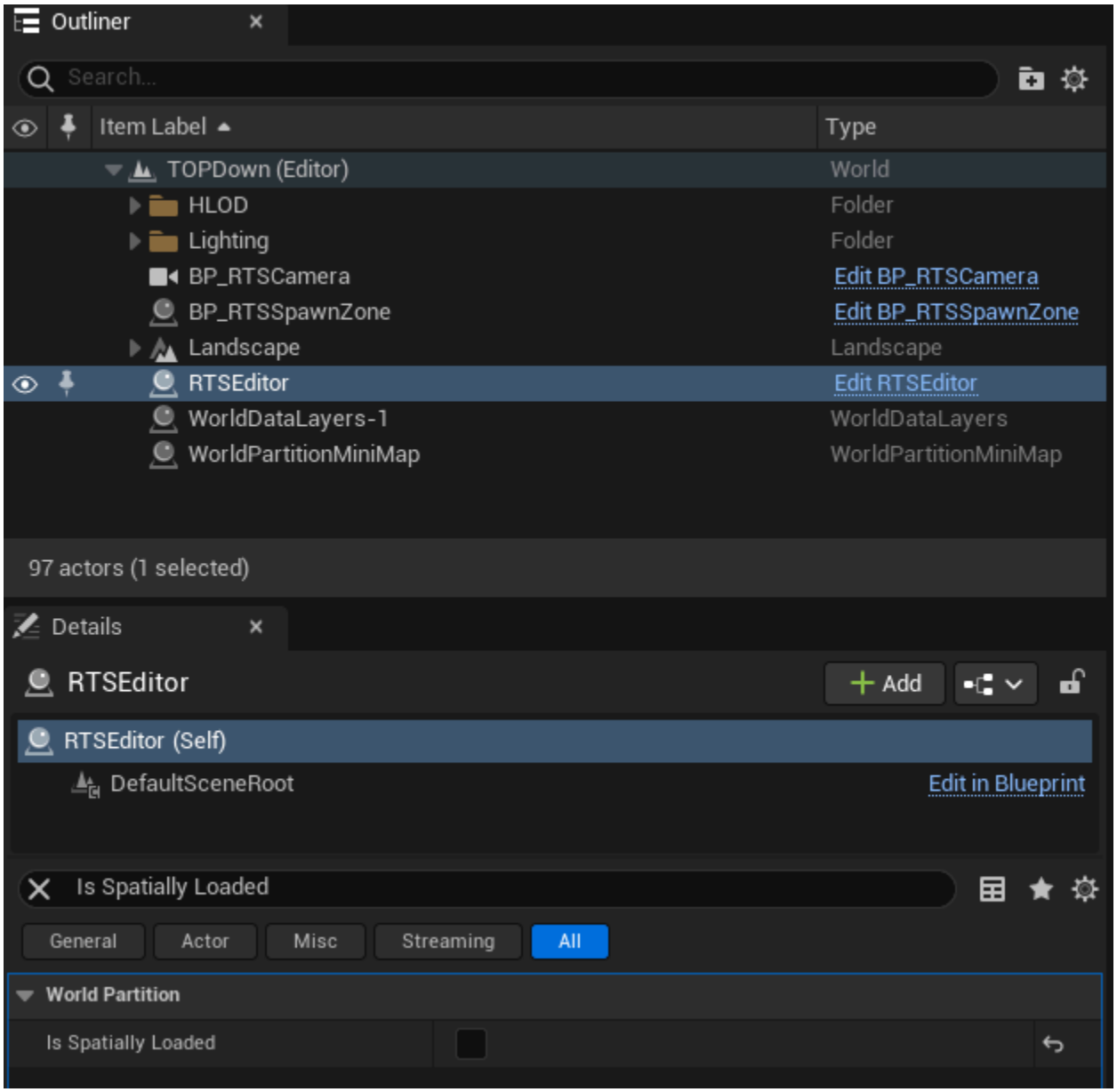


World Partition Configuration

In an open world map with partitions, BP_RTSSpawnZone , RTSEditor, BP_RTSCamera should have their **IsSpatiallyLoaded** unchecked. This will make sure these blueprints are always loaded, regardless which partition or cell is currently active.


World Partition
Show Actors
BugItGo Load Cells
Show Cell Coords
Focus Selection
93.59 m
2.05x2.05x0.38 km
(-1,-2)
(0,-2)
(1,-2)
(2,-2)
(-1,-1)
(0,-1)
(1,-1)
(2,-1)
RTSEditor
(-1,0)
(0,0)
(1,0)
(2,0)
(-1,1)
(0,1)
(1,1)
(2,1)
(-1,2)
(0,2)
(1,2)
(2,2)

9. Uncheck RTSEditor **IsSpatiallyLoaded.** Do the same for BP_RTSSpawnZone , BP_RTSCamera .



Saving Configuration

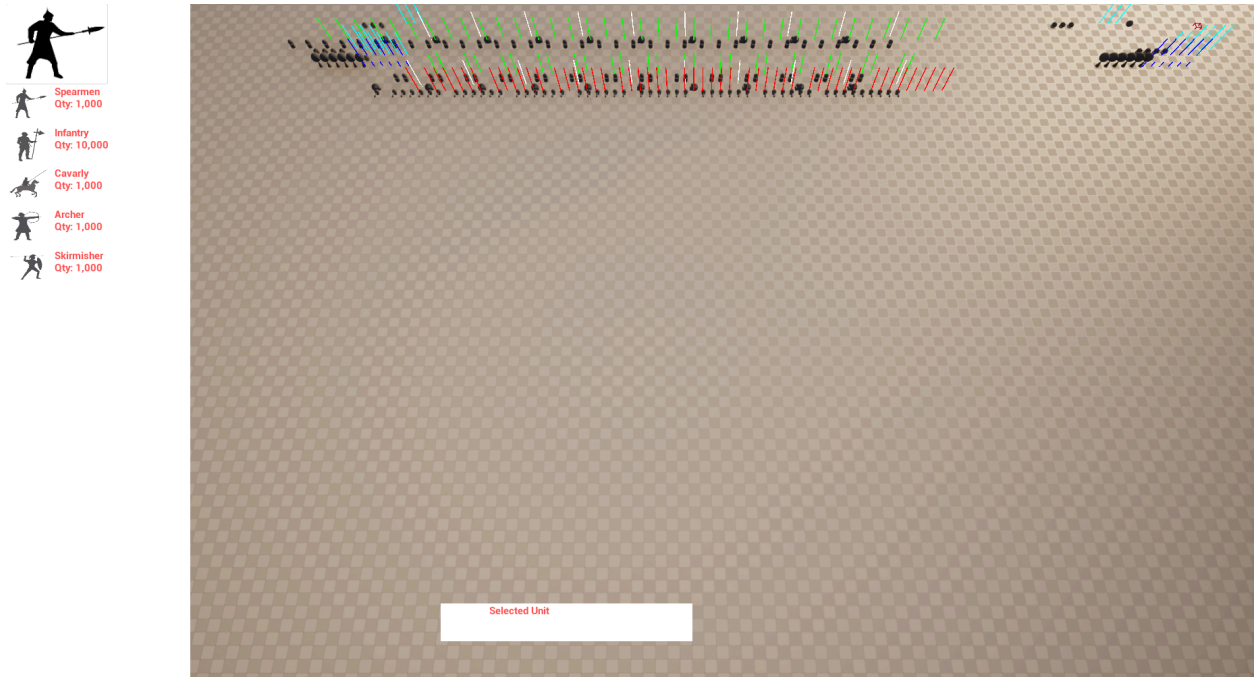
10 . Save the current scene so that UE can update plugin actor references in the scene.

Loading AI formation from the scene

11. Copy MyFormation.txt from /Plugins/RTSEngine/Docs/ to your game project Content folder. In packaged games MyFormation.txt should be in the **package** folder, /Windows/[YourGame]/Binaries/Win64.

Play Testing The plugging integration

12. Hit Play Button. At this point you should see enemy AI formation, loaded from Myformation.txt file.



On the left side of the screen you have a list of RTS units (**Spearmen**, **Infantry**, **Cavalry**, **Archer**, **Skirmisher**). In the 3D scene on the top of the screen you see loaded AI enemy RTS units. In free space in front of enemies you can create your own formation, which is discussed later.

How to Interact with scene using BP_RTSCamera class

- Use mouse scroll to zoom in or zoom out
- Use the WASD Keyboard button to pan the camera.

How to create your formation:

- Select unit from left panel RTS unit list (**Spearmen**, **Infantry**, **Cavalry**, **Archer**, **Skirmisher**)
- Click to ground. Press and hold "Left Mouse Button" to draw and chain the units with selected unit type.

How to combat when your formation is ready

- Select your RTS Units, drawing a selection box on screen, pressing and holding the Right Mouse Button. After successful selection the color of your units will be white.
- Similarly, select enemy units that you want to attack.
- If you want to dis select press Q keyboard button.
- Press C to start combat

Unit placement Tips:
Each RTS unit has its own combat proximity range. For example, Archers have long combat range, while Infantry have close combat range.

Try to place RTS units closer to AI units to test combat.

Before Testing your packaged game:
Before you start packaging your game, update your default maps to the current map that you saved in step 10 (Saving Configuration).
Go to Edit/Project Settings, select Maps&Mode on the left panel and update default maps to the currently opened map.

Failing to update the default map may result in loading the wrong map in a packaged **game.**

Copy MyFormation.txt from /Plugins/RTSEngine/Docs/ to your game **package** folder, /Windows/[YourGame]/Binaries/Win64

AI behavior description:

Infantry - Its primary purpose is to absorb enemy hits while allowing other key units to attack. It is a low mobility unit which protects other key units (e.g. archers, spearmen from attacks).

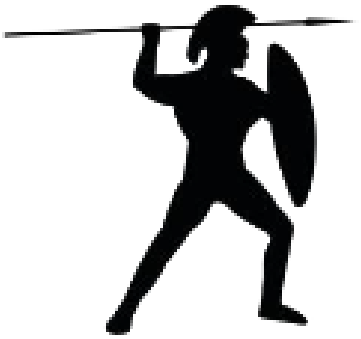
Attack Range: 3 meter
Attack Frequency - Moderate
Key Opponent - N/A
Strength - Performs best in melee range.
Weakness - Its efficiency is reduced when isolated.
Mobility - Medium
Production Cost - Moderate

Archer - Its primary purpose is breaking enemy formation by hitting key enemy targets (Spearmen) from distance with volleys of arrows. Provides ranged suppression, reducing enemy numbers before engagement. Archer is specifically weak to enemy fast skirmisher units. The heavy cross-bow makes it move and shoot an arrow slowly.

Attack Range: long distance (more than 20 meters)
Attack Frequency - Low
Target Opponent - Spearmen.
Strength - Performs best in open area with clear LOS
Weakness - Weak in melee combat
Vulnerable to - Skirmisher
Mobility - Medium
Production Cost - Moderate

Spearmen - This is the anchor of enemy formation. It protects archers and nearby infantries. Spearman efficiently counters the frontal attack of enemy cavalry charge.

Attack Range: 5 meter
Attack Frequency: Moderate
Target Opponent - Cavalry.
Strength - Counters frontal cavalry attacks
Weakness - Vulnerable to cavalry rear attack.
Mobility - Low
Production Cost - Moderate



Skirmisher - The daring hero. It targets enemy archers that try to hit ally spearmen who act as anchors of formation. Hit and run its key tactic, its key job is harassing enemy archers. Unlike archers' delayed arrow shoots, they can quickly throw a javelin while moving.

Attack Range: 20 meter
Attack Frequency: High
Target Opponent - Archers.
Strength - Harassing Archers with Hit and run tactic
Mobility - High
Production Cost - Cheap

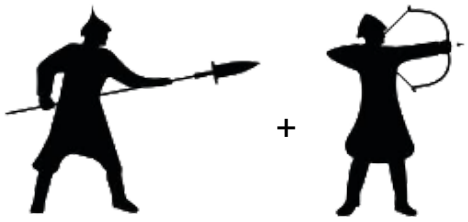
Cavalry - This is a fast and disruptive unit. It is good at charging enemy units from the side or rear. Cavalry performs efficiently when it is in proximity to allied spearmen.

Attack range : 3 meter
Attack Frequency: High
Target Opponent - Spear man.
Strength - Quick Flanking from side or rear of enemy formation
Weakness - Frontal charge of enemy spear man
Mobility - High
Production Cost - High

Plugin provides APIs to tune Attack Frequency and Inflicted damage.

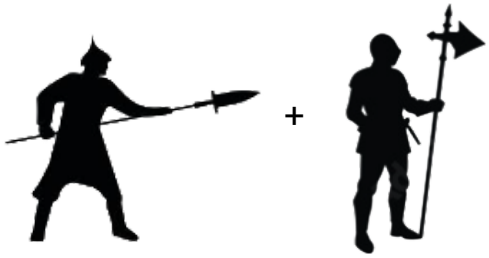
How ally formation is modeled:

Plugin uses proximity based auto formation modeling. Formation is created when **proper** RTS units are placed close together. Here follows example formation pairs.

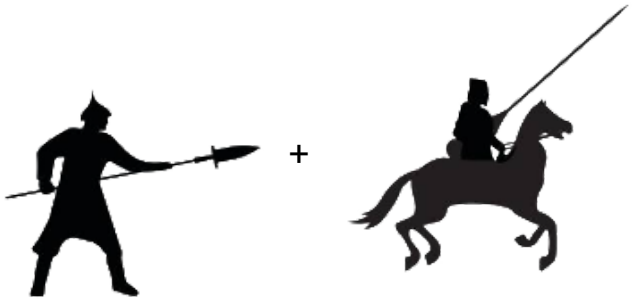


= Spearman-Archer formation

:



= Spearman-Infantry formation



= Spearman-Cavalry formation

In spearman centric formation each unit can perform efficiently and inflict maximum damage. When distance between units is more than required proximity range, formation breaks. Isolated units are less efficient and inflicted damage is reduced.

How counter attack works:

The AI strategy is modeled around two core behaviors:
1. Maximize Own Effectiveness.
2. Minimize Player's Effectiveness.

To minimize opponent player effectiveness core strategy would be to break spearman centric formation. Here follow formation breaking engagement:

 -  = Archer countering spearman

 -  = Skirmisher countering archer

 -  = Cavalry flanking spearman

AI formation generating from ASCII art:

Plugin allows writing ASCII based formation in a text file. Save the ASCII drawing as **MyFormation.txt** in the Content folder in your Unreal project directory. The following table is used to symbolize formation.

ASCII Char	What it means
*	Cavalry
^	Spearmen
!	Infantry
@	Archer
+	Skirmisher

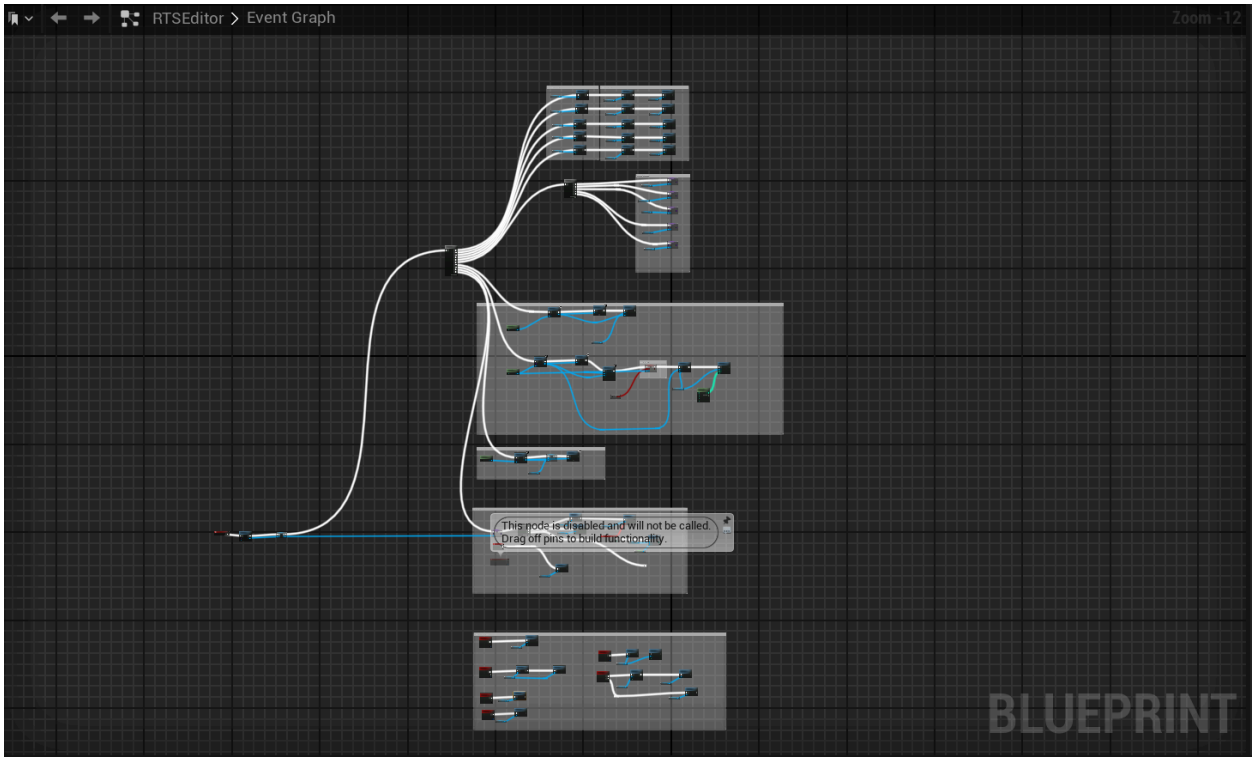
Additional period [.] is used to represent 1 meter space between RTS units. Here follows an example ASCII formation using chars in the above table.

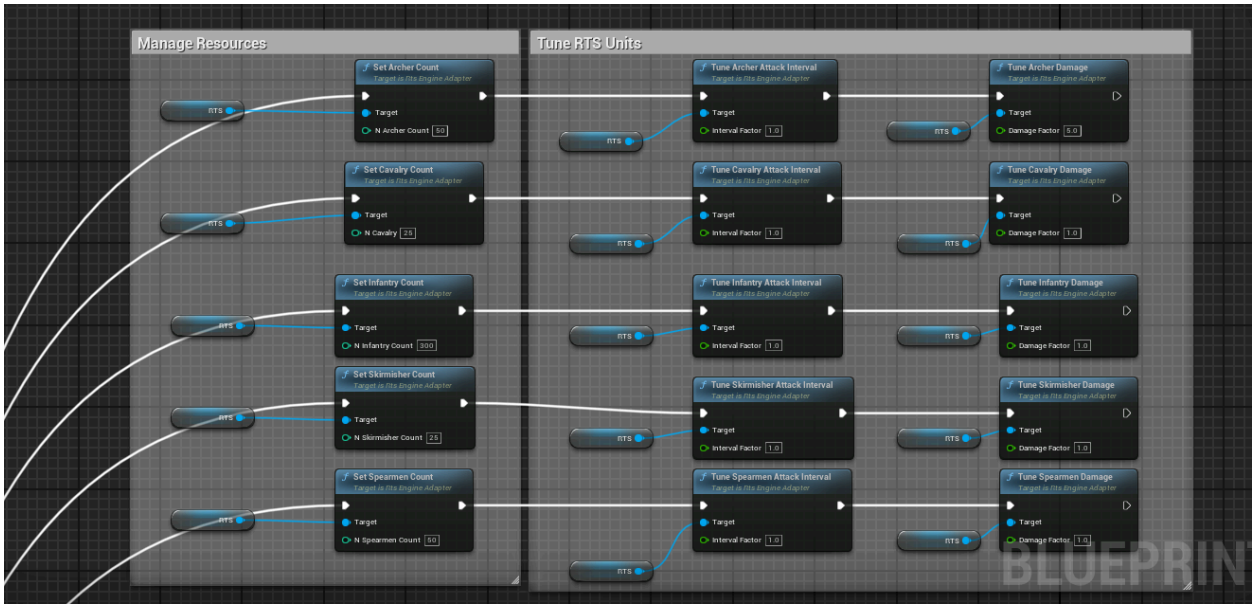
```
.....
.....^^^^^^^^.^^^^^^.^^^^^^.^^^^^^.^^^^^^.^^^^^^.^^^^^^.^^^^^^.^.....
.....@.....@.....@.....@.....@.....@.....@.....@.....@.....@.....@.....
.....!!..!!..!!..!!..!!..!!..!!..!!..!!..!!..!!..!!..!!..!!.....
.....
...++++++.....++++++
..*****.....*****.
****.....****
.....!.!.!.!.!.!.!.!.!.!.!.!.!.!.!.!.!.!.!.!.!.!.!.!.!.!.!.!.!.!.!.!
.....@.....@.....@.....@.....@.....@.....@.....@.....@.....@..
.....
.....***.....***.....
```

Notes:
Period char (.) represents 1 meter empty space between RTS units.
The number of characters in each line could be variable length. You are free to add characters in each line.

Blueprint Scriptable API Description

RTSEngine plugin provides **RTSEditor** Blueprint actor. **RTSEditor** handles Blueprint API calls to RTSEngine plugins. Here follows the organization and execution flow of Blueprint API calls in RTSEditor. RTSEditor uses easy to use Blueprint nodes which can be used to tune and configure various aspects of RTSEngine.





Node	Parameters	Description
SetCavalryCount	nCavalry	Sets number of available cavalry in inventory
SetSpearmenCount	nSpearmenCount	Sets number of available Spearman in inventory
SetInfantryCount	nInfantryCount	Sets number of available Infantry in inventory
SetArcherCount	nArcherCount	Sets number archers in inventory
SetSkirmisherCount	nSkirmisherCount	Set number of skirmishers in inventory

Node	Description
ClearSelectionAIs	
ClearSelectionPlayers	

Node	Description
Attack	Commands selected RTS units to attack enemy units
StopAttack	Commands selected RTS units to stop attacking enemy units

Node	Description
GetCavalryCount	Gets number of available cavalry in inventory

GetSpearmenCount	Gets number of available Spearman in inventory
GetInfantryCount	Gets number of available Infantry in inventory
GetArcherCount	Gets number archers in inventory
GetSkirmisherCount	Get number of skirmishers in inventory

Node	Parameters	Description
TuneCavalryDamage	DamageFactor	Modifies Cavalry inflicted damage by amount of DamageFactor.
TuneSpearmenDamage	DamageFactor	Modifies Spearmen inflicted damage by amount of DamageFactor.
TuneInfantryDamage	DamageFactor	Modifies Infantry inflicted damage by amount of DamageFactor.
TuneArcherDamage	DamageFactor	Modifies Archer inflicted damage by amount of DamageFactor.
TuneSkirmisherDamage	DamageFactor	Modifies Skirmisher inflicted damage by amount of DamageFactor.

Node	Parameters	Description
TuneCavalryAttackRangeMeter	AttackRange	Set Cavalry attack range in meter.
TuneSpearmenAttackRangeMeter	AttackRange	Set Spearmen attack range in meter.
TuneInfantryAttackRangeMeter	AttackRange	Set Infantry attack range in meter.
TuneArcherAttackRangeMeter	AttackRange	Set Archer attack range in meter.
TuneSkirmisherAttackRangeMeter	AttackRange	Set Skirmisher attack range in meter.

Node	Parameters	Description
TuneCavalryAttackInterval	IntervalFactor	Set Cavalry delay between attacks. Lower the IntervalFactor higher the attack speed.

TuneSpearmenAttackInterval	IntervalFactor	Set Spearmen delay between attacks. Lower the IntervalFactor higher the attack speed.
TuneInfantryAttackInterval	IntervalFactor	Set Infantry delay between attacks. Lower the IntervalFactor higher the attack speed.
TuneArcherAttackInterval	IntervalFactor	Set Archer delay between attacks. Lower the IntervalFactor higher the attack speed.
TuneSkirmisherAttackInterval	IntervalFactor	Set Skirmisher delay between attacks. Lower the IntervalFactor higher the attack speed.

Engine Requirements

- **RTS Engine Supports Unreal Engine 5.5.0 - 5.7.0** or later

Platform Support

Currently supported platforms:

- **Windows 64-bit** (Win64)

Limitations

Be aware of these current limitations when planning your project:

- Currently the RTS Engine allows movement on a flat plane.

Getting Started

Ready to start building your strategy game?

1. Follow the Installation Guide Discussed to set up RTSEngine in your project

Vision

RTSEngine APIs are designed to be expanded over time. Current base system provides the **essential strategic formation and combat foundation** and we plan to add features like:

- Advanced building systems
- Advanced damage system.
- Resource management.
- And more…