# So, what is it?

This is an expansion pack for Sci-Fi Ship Controller (SSC is sold separately in the Unity Asset Store). So, you'll first need to own Sci-Fi Ship Controller before you can install and use this expansion pack.

This pack includes physics-based missiles (SSC comes with kinematic guided projectiles, which, as the name suggests act more like projectiles that missiles). The missiles in this pack, are multi-staged, self-propelled, launchable, passive-guided missiles with optional physics-based drag.

The missiles are fully integrated with the existing weapon systems available in SSC. This means they can be fired from existing projectile weapons on Sci-Fi ships. They are plug and play with any existing guided projectiles.

In the first release, there is also a ground or surface mounted missile launcher. You can see two different examples of this launcher module in the city defence demo. If we get enough demand for this pack, we hope to make launchers directly mountable on ships.

## Table of Contents

# Features

- Self-guided, self-propelled, multi-stage missiles
- Targets include ships, ship damage regions, gameobjects, and SSC locations
- Compatible with SSC radar
- Compatible with SSC weapons
- Compatible with SSC AutoTargetingModule
- Missiles use the pooling system from SSC
- Missiles are usable from player or AI ships
- Missiles use physics-based drag
- Example turret and non-turret missile launchers

# How do I get started?

Install Sci-Fi Ship Controller v1.5.2 or newer from the Unity Asset Store. You'll need to be using Unity 2021.3.10 or newer as that is the minimum supported with SSC Expansion Pack 1.

Then, install SSC Expansion Pack 1 in the project.

If you are using URP or HDRP (rather than Built-in Render Pipeline), you may need to look at the SSCXPack1_SRP_readme.txt in the SCSM\SSCXPack1\SRP folder. This will tell you which SRP package you'll need to install from the SRP folder (otherwise all the materials will be pink).

Run the demo scene which can be found in SCSM\SSCXPack1\Demos\Scenes\City Defence Demo. The demo will give you ideas as to how you could make use of the missiles in your own project. Of course, you can make your own launchers and missiles with your own 3D models too.

# Support Policy

For free support we will investigate reproducible bugs in our code. We may ask you to provide a simple scene with clear instructions on how to repro the issue. We can provide an upload area for the project files.

If this issue is critical to an announced game release date, we give it high priority. We also help with fleshing out new features that can improve gameplay and that we could add to a new version. In addition, we offer customer support for discovering existing features and how to configure them, both in our Unity forum or on our Discord channels.

To add "polish" to a game or general help with implementing our products (and even writing custom game-play code) we negotiate a flexible hourly rate which can be time-boxed to fit the studio or indie budget.

For ad-hoc on-going support, and to help us to keep supporting Sci-Fi Ship Controller and expansion packs for a long time, please support us on https://www.patreon.com/scsmmedia

We may alter this support policy from time to time without notice.

# What’s Changed?

Version 1.0.2

[IMPROVED] Ensure Radar is created in the same scene as launcher
[IMPROVED] Updated editor toolbar

Version 1.0.1

[NEW] SSCMissileModule - shield penetration option
[NEW] SSCMissileModule - stage run and completion callbacks
[NEW] SSCMissileModule - Disable FX on Destroy option
[NEW] SSCMissileModule - Inaccuracy option
[NEW] SSCMissileModule - GetStageId and GetStageByIndex APIs
[NEW] SSCMissileModule - overrideable EnableOrDisableFX method
[NEW] SSCMissileModule - StartFX() and StopFX() APIs
[NEW] SSCLauncherModule - Manual Fire option
[NEW] Ship Missile Tester scene
[FIXED] Launcher Identification descriptions missing in manual

# Missile Module

This is the core component of the multi-stage missile system. We have several common configurations created for you in the Demos\Prefabs\Missiles folder. You can also create your own either with our missile model or with your own models. To get started you can duplicate one of our prefabs, rename it, and move it to your own folder within your game.

Missiles can be used with the ground-based SSC Launch Module or regular projectile weapons on ships. They are fully integrated into the pooling system provide by Sci-Ship Controller’s SSCManager.

When your game is playing in the editor, there is a Debug Mode option in the inspector of each missile in the scene that can provide more information to help you understand what is happening at runtime.

## How to Create Missile Prefabs

To create a new Missile:

1. In a scene, create an empty gameobject
2. Rename the empty gameobject. E.g., MyMissile1
3. Either add a mesh and mesh renderer as a child gameobject
4. Where possible, use a simple collider like a capsule or box collider
5. Add a Missile Module script to the parent gameobject
6. Add a Rigidbody to the parent gameobject
7. Create a prefab from the gameobject by dragging the parent gameobject into a folder in the Project pane (avoid using our SCSM folder)
8. Reset the prefab parent transform position and rotation to 0,0,0
9. Ensure the parent gameobject scale is 1,1,1 (to avoid rigidbody issues)
10. Delete the gameobject from the scene

## Missile General Tab

This is where you'll find all the general settings for configuring your missiles.

| Property | Description |
|---|---|
| Use Gravity | Whether gravity is applied to the missile. If applied, the amount and direction of gravity will be inherited from the ship or weapon that fired it. |
| Damage Type | The type of damage the missile does when hitting a ship. The amount of damage dealt to a ship upon collision is dependent on the ship's resistance to this damage type. If the damage type is set to Default, the ship's damage multipliers are ignored i.e. the damage amount is unchanged. |
| Damage Amount | The amount of damage the missile does on collision with a ship or object. NOTE: Non-ship objects need a DamageReceiver component. |
| Damage Mask | The layer mask used for collisions with objects that the missile may cause damage. Default: Everything |
| Mass (kg) | The mass of the missile in kilograms |
| Min Pool Size | When using the Pooling system, this is the number of missile objects kept in reserve for spawning and despawning. |
| Max Pool Size | When using the Pooling system, this is the maximum number of missiles permitted in the scene at any one time. |
| Destroyed FX Object | The particle and/or sound effect prefab that will be instantiated when the missile hits something and is destroyed. This does not fire when the missile is automatically despawned. |
| Shield FX Object | The particle and/or sound effect prefab that will be instantiated, instead of the regular Effects Object, when the missile hits a shielded ship. This does not fire when the missile is automatically despawned. |
| Launch FX Object | The particle and/or sound effect prefab that will be instantiated when the missile is fired from a weapon. |
| Launch FX Offset | The distance in local space that the launch Effects Object should be instantiated from the weapon firing point. Typically, only the z-axis will be used. |
| Col. Damage Resistance | Value indicating the resistance of the missile to damage caused by collisions. Increasing this value will decrease the amount of damage to this missile caused by collisions.<br>As the missile is damaged, its performance will degrade, effecting it's thrust and turning ability. |
| Inaccuracy | Default to 0 which is the missile will attempt to hit the target. Range from 0.0 to 1.0. The higher the number, the more chance the missile will not hit the target |

| Property | Description |
|---|---|
| | dead centre. NOTE: It has a bit of a performance overhead so only use when required. |
| Inaccuracy Scale | Default to 1, increasing the scale will increase the chances of the missile missing the target by a larger margin. |

## Missile Stages Tab

This is what makes your missile fly. Typically, you'll use between one and three stages.

| Property | Description |
|---|---|
| Stage Name | The descriptive name of the Missile Stage. |
| Stage Duration | How many seconds the stage should take to complete. |
| Blend Duration | The time, in seconds, used to blend from the previous stage. Currently this applies to turning force and thrust. |
| Armed | Is the object armed and will explode when colliding with another object? If the missile colliders with other objects while not armed, it will receive damage which will affect its performance and ability to seek a target. |
| Seek Target | Should the missile attempt to intercept the target? |
| Face Intended | Should the missile face in the intended direction of travel (i.e. where it is facing) rather than the direction it is moving? Only applies when thrust > 0 [DEFAULT: OFF]. This could be useful for missiles are dropped in the first stage with Use Gravity and no thrust, and then have thrust in the second stage. |
| Ignore Collisions | Should all collisions be ignored during this stage? |
| Thrust (N) | The thrust force in newtons. |
| Max Turning Force (kN) | The maximum force used to turn the missile in kilonewtons. Missile with the same turning force but a larger mass will turn more slowly. |

## Missile Identification Tab

This information helps you identify the missiles during gameplay.

| Property | Description |
|---|---|
| Faction Id | The faction or alliance the item belongs to. This can be used to identify if an item is friend or foe. Default (neutral) is 0. |
| Model Id | The unique model or type of missiles. Values 1 to 100 are reserved for demo SCSM ships, turrets or missiles. 0 = Not Set. |
| Visible to Radar | Is this missile visible to radar queries? It will appear as a gameobject entry on radar. |
| Radar Blip Size | The relative size of the blip on the radar mini-map. |

## Missile Drag Tab

The drag properties of the missile determine how the airflow around the missile affects the movement of the missile. Use the Drag X/Y/Z Coefficients to alter how much drag the missile has on each axis. More streamlined axes of the missile should have a lower drag coefficient while flatter axes should have a higher drag coefficient. You can use the Angular Drag Factor to alter how quickly angular drag will slow down any spinning motion.

| Property | Description |
|---|---|
| Drag Area (m2) | The area, in square metres, of the model for drag purposes. x = side view area, y = top view area, z = front view area. If you're not sure, look at our prefabs and see how we calculated this from the size of the colliders. |
| Drag Offset Z | The distance, in metres, to move the drag moments behind the centre of mass on the z-axis. NOTE: You'd never want to move it in front of the centre of mass, else flight would be unstable. |
| Drage Factor | The overall drag factor that affects drag on the missile. Set to 1 for physically realistic value. Set to 0 to disable drag. |
| Drag X Coefficient | The coefficient of drag of the missile on the x-axis. Increasing the coefficient of the drag will increase the effect of drag. |
| Drag Y Coefficient | The coefficient of drag of the missile on the y-axis. Increasing the coefficient of the drag will increase the effect of drag. |
| Drag Z Coefficient | The coefficient of drag of the missile on the z-axis. Increasing the coefficient of the drag will increase the effect of drag. |
| Angular Drag Factor | How strong the effect of angular drag is on the missile. Setting this to 1 will make it physically realistic. |
| Drag Moments Multipliers | Multipliers for drag moments causing rotation along a local axis. |
| Drag X Multiplier | A multiplier for drag moments causing rotation along the local (pitch) x-axis. Decreasing this will make these moments weaker. |
| Drag Y Multiplier | A multiplier for drag moments causing rotation along the local (yaw) y-axis. Decreasing this will make these moments weaker. |
| Drag Z Multiplier | A multiplier for drag moments causing rotation along the local (roll) z-axis. Decreasing this will make these moments weaker. |

# Missile Launcher

This is a ground-based launcher that can act as a single silo or as a turret with multiple launch tubes. Unlike a regular ground (or ship-mounted) turret, this is specifically designed to work with missiles from this expansion pack.

When your game is playing in the editor, there is a Debug Mode option in the inspector of each missile in the scene that can provide more information to help you understand what is happening at runtime.

## Launcher General Tab

| Property | Description |
|---|---|
| Initialise on Start | If enabled, Initialise() will be called as soon as Start() runs. This should be disabled if you want to control when the component is enabled through code. |
| Auto Create Location | Automatically create a Location in the SSCManager when the launcher is initialised. |
| Missile Prefab | Prefab template of the missiles fired by this weapon. Missile prefabs need to have a Missile Module script attached to them. |
| Destroyed FX Object | The particle and / or sound effect prefab that will be instantiated when the launcher is destroyed. |
| Starting Health | The initial health value of this launcher. This is the amount of damage that needs to be done to the launcher for it to reach its min performance. |
| Destroy on No Health | Should the launcher be destroyed (removed from the scene) when its health reaches 0? |
| Destruct Object | The destruct prefab that breaks into fragments when the launcher is destroyed. |

| Property | Description |
|---|---|
| Launch Interval | The minimum time (in seconds) between consecutive missile launches. |
| Relative Position | The position of the weapon in local space relative to the pivot point of the whole launcher. Where possible, make your launcher pivot around the turret pivot point so that relative position is 0,0,0. |
| Auto Targeting | When the Auto Targeting Module is attached, use this to indicate targets should be assigned to the weapon. |
| Auto Fire | When a target is selected, the weapon will automatically attempt to fire at the target. |
| Manual Fire | The weapon can be manually fired with FireIfReady() API |
| Check Line of Sight | Whether the weapon checks line of sight before firing (in order to prevent friendly fire) each frame. Since this uses raycasts it can lead to reduced performance. Currently has no effect if target is a Location. |
| Require Target Lock | When firing, does the missile need to be locked on to a target? This is always true for turrets. |
| Fire Position Offsets | The positions of the launch tubes relative to the position of the weapon. |
| Fire Direction | The direction in which the weapon fires missiles in local space. +ve Z is fire forwards, -ve Z is fire backwards. |
| Firing Type | When there are multiple firing positions, when the fire command is issued, how should the weapon respond? By default, all fire positions are fired at the same time (subject to available ammo). |
| Unlimited Ammo | Can this weapon keep firing and never run out of ammunition? |
| Ammunition | The quantity of missiles or ammunition available for this weapon. |
| Heat Level | The heat of the weapon - range 0.0 (starting temp) to 100.0 (max temp). |
| Heat Up Rate | The rate heat is added. It is inversely proportional to the firing interval (reload time). If rate is 0, heat level never changes. |
| Cool Down Rate | The rate heat is removed per second. This is the rate the weapon cools when not in use. |
| Overheating Threshold | The heat level that the weapon will begin to overheat and start being less efficient. |
| Burnout on Max Heat | When the weapon reaches max heat level of 100, will the weapon be inoperable until it is repaired? |

## Launcher Identification Tab

This information helps you identify the launcher during gameplay.

| Property | Description |
|---|---|
| Faction Id | The faction or alliance the item belongs to. This can be used to identify if an item is friend or foe. Default (neutral) is 0. |
| Squadron Id | Although normally representing a squadron of ships, this can be used on a launcher to group it with other things in your scene. Default (unset) is -1. |
| Model Id | The unique model or type of launcher. Values 1 to 100 are reserved for demo SCSM ships, turrets or missiles. 0 = Not Set. |
| Visible to Radar | Is this launcher visible to radar queries? It will appear as a gameobject entry on radar. |
| Radar Blip Size | The relative size of the blip on the radar mini-map. |

## Launcher Turret Tab

If this launcher has a turret, this information determines how it will behave.

| Property | Description |
|---|---|
| Enable Movement on Init | Should the turret be able to move when the launcher is first initialised? |
| Turret Pivot Y | The transform of the pivot point around which the turret turns on the local y-axis. |
| Turret Pivot X | The transform on which the barrel(s) or cannon(s) elevate up or down on the local x-axis. |
| Turret Min. Y | The minimum angle on the local y-axis the turret can rotate to. |
| Turret Max. Y | The maximum angle on the local y-axis the turret can rotate to. |
| Turret Min. X | The minimum angle on the local x-axis the turret can elevate to. |
| Turret Max. X | The maximum angle on the local x-axis the turret can elevate to. |
| Turret Move Delay | The time, in seconds, the turret will stop rotating after it has fired. This can be used to give the missile(s) a chance to clear the turret. |
| Turret Move Speed | The rate at which the turret can rotate. |
| Turret Inaccuracy | When greater than 0, the number of seconds a turret will wait, after losing a target, to begin returning to the original orientation. |
| Turret Park Interval | When greater than 0, the number of seconds a turret will wait, after losing a target, to begin returning to the original orientation. |

## Launcher Environment Tab

Gravitational acceleration and direction, and other environmental factors, can affect how a missile behaves after it has been fired from the weapon.

| Property | Description |
|---|---|
| (Gravity) Acceleration | The acceleration due to gravity in metres per second squared. Earth gravity is approximately 9.81 m/s^2. |
| (Gravity) Direction | The direction in which gravity acts on the missile in world space. |
| Medium Density | The density (in kilograms per cubic metre) of the fluid the missile will travel through (usually air). At around sea level, the Earth's atmosphere is approximately 1.293. |

## Launcher Events Tab

Sometimes you might wish to call your own code when certain things happen with the launcher. This tab lets you setup custom events in the inspector. Alternatively, you could use some of the launcher callbacks (see the Runtime and API chapter).

| Property | Description |
|---|---|
| On Initialised Event Delay | The number of seconds to delay firing the onInitialised event methods after the launcher has been initialised. |
| On Initialised | These are triggered by a launcher after it is initialised. |
| On Destroyed | These are triggered when the launcher reaches 0 health. |
| On Post Launch | These are triggered immediately after a missile is launched. |

# Common Issues

Below is a list of common issues people can encounter that are usually fixed by tweaking the configuration of various components.

## Common Issues – Missile Module

Currently there are no known issues. If you see something, please report it on our Discord channel or contact support.

## Common Issues – Missile Launcher

1. My launcher keeps firing missile even though there is no target. If this is not a turret, and "Auto Fire" is enabled on the General tab, enable "Require Target Lock".
2. When my missiles launch, they target missile from the same or another launcher. For missiles fired from launchers in the same faction, they all need to have the same Faction Id, otherwise they will be considered enemy missiles.
3. AutoTargetModule has a target, but the launcher doesn't fire a missile. Check if the launcher is assigned a target with Debug Mode on the launcher. Does the launcher have ammunition? Is Auto Targeting enabled on the General tab? Did the launcher get a target (check Debug Mode on the launcher)? If not, is the target behind the fire direction? If so, the radar target will be considered invalid.

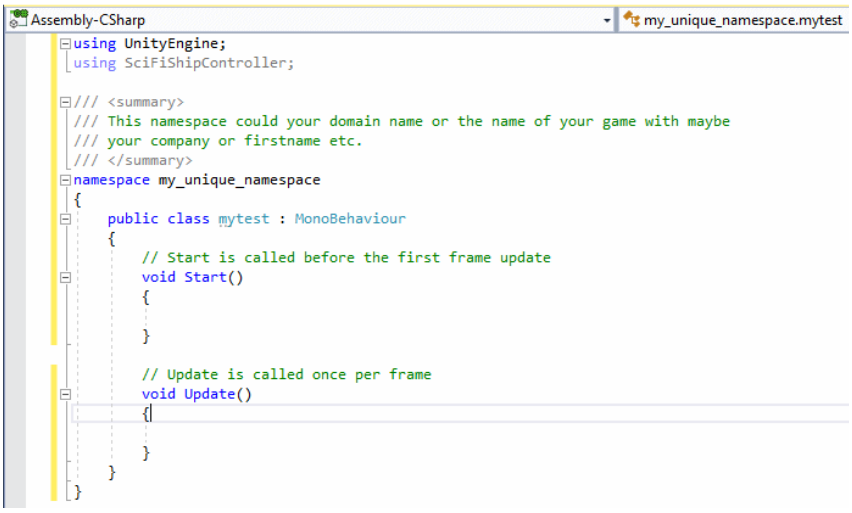
# Runtime and API

SSC Expansion Pack 1 is designed to be used in your games. We expect your code to interact with ours. This section, together with the Runtime and API section in the Sci-Fi Ship Controller manual, will help you interact via code with our components.

## Runtime General Guidance

Much of our code is well documented and broken down into regions marked with #region #endregion tags. These are expandable in Visual Studio.

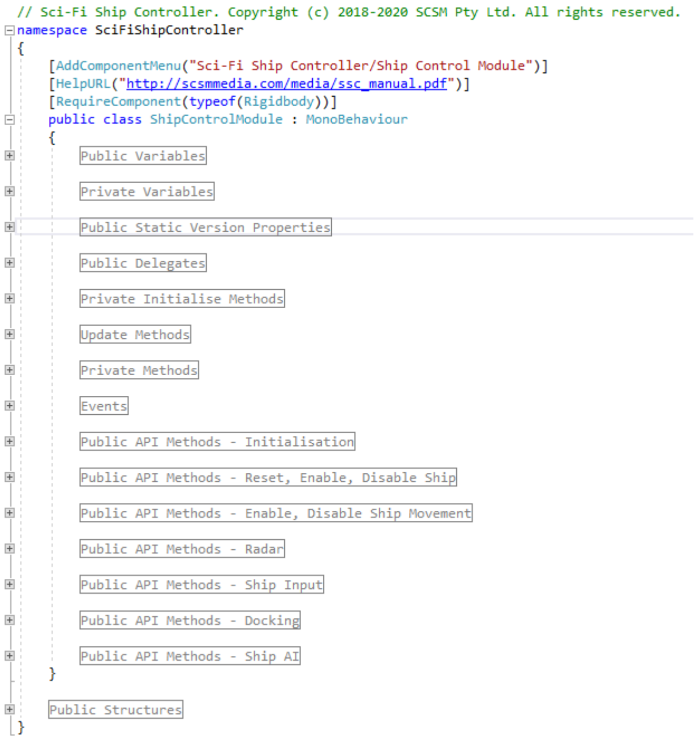
When integrating SSC Expansion 1 into your game or project, make sure your scripts are in your own namespace so that they don't conflict with other people's code or assets.



```
using UnityEngine;
using SciFiShipController;

/// <summary>
/// This namespace could your domain name or the name of your game with maybe
/// your company or firstname etc.
/// </summary>
namespace my_unique_namespace
{
    public class mytest : MonoBehaviour
    {
        // Start is called before the first frame update
        void Start()
        {

        }

        // Update is called once per frame
        void Update()
        {

        }
    }
}
```

Public Variables and Properties in our scripts are generally available for you to safely access in your own code. Anything marked "[INTERNAL ONLY]", "private" or "internal" should never be used in your code as these items are subject to change and will most likely either break your game or make it behave is a strange manner.

Some of our scripts have Public API methods. These are used in our demo scripts and can safely be used in your game code. Look for these Public API regions at the bottom of our scripts.



```
// Sci-Fi Ship Controller. Copyright (c) 2018-2020 SCSM Pty Ltd. All rights reserved.
namespace SciFiShipController
{
    [AddComponentMenu("Sci-Fi Ship Controller/Ship Control Module")]
    [HelpURL("http://scsmmedia.com/media/ssc_manual.pdf")]
    [RequireComponent(typeof(Rigidbody))]
    public class ShipControlModule : MonoBehaviour
    {
        Public Variables

        Private Variables

        Public Static Version Properties

        Public Delegates

        Private Initialise Methods

        Update Methods

        Private Methods

        Events

        Public API Methods - Initialisation

        Public API Methods - Reset, Enable, Disable Ship

        Public API Methods - Enable, Disable Ship Movement

        Public API Methods - Radar

        Public API Methods - Ship Input

        Public API Methods - Docking

        Public API Methods - Ship AI
    }

    Public Structures
}
```

Many of our public variables, properties, delegate call-backs, and methods are documented in the sections below in this manual. Everything else is documented in our script files. Feel free to contact us in our Unity forum or on our dedicated Discord channel if you are unsure of how a variable or method should be used.

## Changing Variable at Runtime

Many public variables are modifiable at runtime from within your own code. Variables are commented so that (a) you know what they do, and (b) you can see if they require a method to be called after changing at runtime. For example, if you change sscLaunchModule.destructObject at runtime, you also need to call the method sscLaunchModule.ReinitialiseTurretDestructObjects ().

## Demo Scripts

We have included a collection of helpful scripts that show how certain features can be used in your games or projects. They are subject to change with version upgrades, so are not meant to be used directly in your projects. Instead, the intention is to help you build games with SSC Expansion Pack 1 by providing coding examples. **Do not** make changes to these scripts, instead create your own based on these.

Most scripts have a description at the top and comments throughout.

| Script name | Description |
|---|---|
| DemoCityDefence.cs | A demo script to show how you might defend a city with missile launchers from SSC Expansion Pack 1.<br>It uses a ShipSpawner script to spawn 3 squadrons of ship to attack the city with missile, while the missile launchers in the city attempt to shoot down the incoming missiles and the invading ships. |
| SampleLauncherTargetAcquired.cs | Simple script to show how to get notified when a missile launcher has acquired a new target. See instructions at the top of the script.<br>NOTE: This is only a sample to demonstrate how API calls could be used in your own code. You should write your own version of this in your own namespace. |

## Missile Launcher Methods - Events

These public methods can be accessed from any instance of a SSCLaunchModule in the scene.

| Method | Description |
|---|---|
| RemoveListeners() | Call this when you wish to remove any custom event listeners, like after creating them in code and then destroying the object. You could add this to your game play OnDestroy code. |
| SetOnInitialisedEvtDelay (float newValue) | Change the amount of time the configured event methods are called after the launcher is initialised. This will have no effect after the launcher is initialised. |

## Missile Launcher Methods - General

These public methods can be accessed from any instance of a SSCLaunchModule in the scene.

| Method | Description |
|---|---|
| CalculateWeaponTargetScore (SSCRadarBlip targetBlip) | Calculates a weapon targeting score for a target. Higher scores indicate better targets. The maximum score is 1000. Targets behind the launcher get assigned a score of -1 to indicate that they are invalid. |
| ClearWeaponTarget() | Clears all targeting information for the weapon. This should be called if you do not know if the target is a ship or a gameobject. |
| CreateLocation (bool removeExisting) | Create a new Location using the SSCManager and add it to Radar if required. |
| DestroyLauncher (bool runDestruction) | Safely destroy the launcher. If there is a destruction prefab, that can also be run. |
| FireIfReady() | Launch missile(s) on the weapon if they are loaded and ready. This is a single shot action. For continuous firing, call SetAutoFire().NOTE: This will fire regardless of target lock and Line-of-Sight. |
| Initialise() | Attempt to initialise the launcher. |
| PauseLauncher() | Pause the launcher. This prevents it from tracking targets, firing, taking damage, and moving the turret. |
| ReinitialiseDestructObjects () | Reinitialises variables required for destruct objects of the launcher. Call after modifying any destruct data for this launcher. |
| ReinitialiseMissileAndEffects () | Reinitialises variables required for missile and effects of the launcher. Call after modifying any missiles or effect data for this launcher. |
| SetAutoFire() | Sets the weapon to automatically fire if a target is acquired and the weapon is ready. |
| SetFiringType (Weapon.FiringType newFiringType) | Attempt to set the firing type of the launcher. |
| SetHealth (float newHealthLevel) | Attempt to set the health between 0.0 (no health) and 100.0 (full health) |
| SetManualFire() | For manually firing the weapon. After this is set, call FireIfReady() to fire the weapon. |
| SetMissileModule (SSCMissileModule missileModulePrefab) | Change the missile prefab. If required this calls ReinitialiseMissileAndEffects(). |
| SetTarget (GameObject target) | The launcher will track this gameobject. |
| SetTargetLocation (LocationData targetLocation) | The launcher will track this location. |

| Method | Description |
|---|---|
| SetTargetShip (ShipControlModule targetShipControlModule) | The launcher will track this ship. |
| SetTargetShipDamageRegion (ShipControlModule targetShipControlModule, DamageRegion damageRegion) | The launcher will track this ship's localised damage region. |
| TelePort (Vector3 delta) | Teleport the launcher to a new location by moving by an amount in the x, y and z directions. This could be useful if changing the origin or centre of your world to compensate for float-point error. |
| TelePort (Vector3 newPosition, Quaternion newRotation) | Teleport the launcher to a new location with a new rotation. |
| UnPauseLauncher() | Unpausing the launcher enables it to track targets, fire, taking damage, and move the turret. |
| WeaponHasLineOfSight (GameObject target, bool directToTarget = false, bool obstaclesBlockLineOfSight = true, bool anyEnemy = true) | Returns whether a weapon has line of sight to a target.<br>If directToTarget is set to true, will raycast directly from the weapon to the target.<br>If directToTarget is set to false, will raycast in the direction the weapon is facing.<br>This method will return true if the raycast hits:<br>a) The target,<br>b) An enemy ship (distinguished by faction ID) - even if it is not the target and anyEnemy is true,<br>c) An object that isn't the target (if obstaclesBlockLineOfSight is set to false),<br>d) Nothing.<br>This method will return false if the raycast hits:<br>a) A friendly ship (distinguished by faction ID),<br>b) An object that isn't the target (if obstaclesBlockLineOfSight is set to true).<br>c) An enemy ship that is not the target when anyEnemy is false. |

## Missile Launcher Methods - Turret

These public methods can be accessed from any instance of a SSCLaunchModule in the scene.

| Method | Description |
|---|---|
| CheckIsTurret() | Check if this is a turret. Call this if you update the weapon pivot points in code. |
| DisableTurretMovement() | Stop the turret from rotating. |
| EnableTurretMovement() | Allow the turret to rotate when required. |

## Missile Launcher Properties

These public properties can be accessed from any instance of a SSCLaunchModule in the scene.

| Property | Description |
|---|---|
| AimDirection | Where the launcher is currently aiming in world-space. |

| Property | Description |
|---|---|
| Health | Attempt to get or set the Launcher's health. |
| LaunchModuleId | The unique ID of the module at runtime. |
| IsInitialised | Has the module been initialised? |
| IsLauncherPaused | Is the launcher paused? This prevents it from tracking targets, firing, taking damage, and moving the turret. |
| IsTurret | Does the launcher have a turret? |
| IsTurretMovementEnabled | Get or set if the turret can rotate. |
| RadarId | The number used by the SSCRadar system to identify this Launcher at a point in time. This should not be stored across frames and is updated as required by the system. |
| TransformPosition | The position of the launcher as a vector. Derived from the position of the transform. |
| TransformForward | The forward direction of the launcher in world space as a vector. Derived from the forward direction of the transform. |
| TransformRight | The right direction of the launcher in world space as a vector. Derived from the right direction of the transform. |
| TransformUp | The up direction of the launcher in world space as a vector. Derived from the up direction of the transform. |
| TransformRotation | The rotation of the launcher in world space as a quaternion. Derived from the rotation of the transform. |
| TransformInverseRotation | The inverse rotation of the launcher in world space as a quaternion. Derived from the rotation of the transform. |
| IsDestroyed | Has the launcher been destroyed (typically when there is no weapon Health) |
| TargetShipName | If a ship is being targeted, will return its name. |
| TargetShipDamageRegionName | If a ship damage region is being targeted, will return its name. |
| TargetGameObjectName | If a gameobject is being targeted, will return its name |
| TargetLocationName | If a location is being targeted, will return its name. |

## Missile Launcher Call Backs

Custom runtime methods should be a lightweight to avoid performance issues. These single-cast delegates can have a single instance of a call-back method. This is useful when you want to take some (custom) action when something occurs, like when a launcher is destroyed by a projectile or missile.

These can be referenced or called from an instance of sscLauncherModule.

| Property or Method | Description |
|---|---|
| CallbackOnDestroy<br>callbackOnDestroy | The name of the custom method that is called immediately before the launcher is destroyed. Your method must take 1 parameter of class SSCLaunchModule. This should be a lightweight method to avoid performance issues. It could be used to update a score or affect the status of a mission. For this to be called you need to attach a DamageReceiver component.<br>As an alternative, you could use OnDestroyed from the Events tab of the Missile Launcher component. |
| CallbackOnTargeted<br>callbackOnTargeted | The name of the custom method that is called immediately after the missile launcher acquires a new target. Your method must take 5 parameters (SSCLaunchModule, ShipControlModule, DamageRegion, Gameobject, and LocationData). Any of the last 4 parameters can be null. This should be a lightweight method to avoid performance issues. |

## Missile Module Properties

These public properties can be accessed from any instance of a SSCMissileModule in the scene.

| Property or Method | Description |
|---|---|
| CurrentMaxTurningForce | The current maximum turning force being applied to the missile in Newtons. |
| CurrentPerformance | The current performance or health of the missile. |
| CurrentStage | If initialised, return the current stage. |
| CurrentStageName | If initialised, return the current stage name (this is typically only used for debugging). |
| LocalVelocity | Get the local space velocity of the missile |
| IsArmed | Is the missile currently armed? |
| IsIgnoringCollisions | Is the missile currently ignoring collisions with all other objects? |
| IsMissile | Is this projectile a missile from SSC Expansion Pack 1? |
| NumberOfStages | Get the number of stages for this missile. |
| StageList | Return the list of missile stages. |

## Missile Module Methods

These public methods can be accessed from any instance of a SSCMissileModule in the scene.

| Property or Method | Description |
|---|---|
| DirtyDragCache() | Typically, only called from an editor script to show that properties affecting drag have been changed. |
| GetStageById (int stageId) | Attempt to get a stage using its unique identifier (StageId). |
| GetStageId (int stageNumber) | Attempt to return the State ID given the number in the list on the Stages tab. Returns 0 if the number is invalid or the stage slot is empty. |
| GetStageByIndex (int stageIndex) | Attempt to return the stage give the zero-based index in the list |

## Missile Module Virtual Methods

These methods can be overridden.

| Property or Method | Description |
|---|---|
| GuideMissileToTarget() | Update desiredDirectionWS or do nothing. |
| InitialiseMissile() | Initialise the missile after the base projectile has been (custom) initialised. |
| EnableOrDisableFX(bool isEnable) | Attempt to enable or disable any FX attached to this missile. |
| MissileCollision (Collision collision) | Check if the missile has hit a damageable object. |
| IEnumerator RunStage (SSCMissileStage stage, int stageIndex) | Run a missile stage and notify when completed. |

## Missile Module Call Backs

Custom runtime methods should be a lightweight to avoid performance issues. These single-cast delegates can have a single instance of a call-back method. This is useful when you want to take some (custom) action when something occurs, like when a missile stage runs or completes, and a visual FX needs to be modified.

These can be referenced or called from an instance of SSCMissileModule.

| Property or Method | Description |
|---|---|
| CallbackOnRunStage<br>callbackOnRunStage | The name of the custom method that is called immediately after the stage becomes active. Your method must take 2 parameters (SSCMissileModule and SSCMissileStage) and should be light weight to avoid performance issues. |
| CallbackOnTargeted<br>callbackOnTargeted | The name of the custom method that is called immediately after the stage has been completed. Your method must take 2 parameters (SSCMissileModule and SSCMissileStage) and should be light weight to avoid performance issues. |

# Release History

**Initial Release – 20 Jun 2024**

**Version 1.0.1 – 16 Oct 2024**

[NEW] SSCMissileModule - shield penetration option
[NEW] SSCMissileModule - stage run and completion callbacks
[NEW] SSCMissileModule - Disable FX on Destroy option
[NEW] SSCMissileModule - Inaccuracy option
[NEW] SSCMissileModule - GetStageId and GetStageByIndex APIs
[NEW] SSCMissileModule - overrideable EnableOrDisableFX method
[NEW] SSCMissileModule - StartFX() and StopFX() APIs
[NEW] SSCLauncherModule - Manual Fire option
[NEW] Ship Missile Tester scene
[FIXED] Launcher Identification descriptions missing in manual

# Trade Marks

“Unity” is a trademark of Unity Technologies and is in no way associated with SCSM Pty Ltd.

Other names or brands are trademarks of their respective owners.