# **Code Coverage - Worksheet**



### Tasks

- 1. What is Code Coverage (2 min)
- 2. Install the Code Coverage package (2 min)
- 3. Enable Code Coverage (1 min)
- 4. Understanding the game code: Shoot() function (4 min)
- 5. Generate a Coverage report from PlayMode tests (3 min)
- 6. Add Weapon tests to improve coverage (3 min)
- 7. Add a test for the LaserController (4 min)
- 8. Clear the coverage data (1 min)
- 9. Generate a Coverage report using Coverage Recording (4 min)

### **Useful Links**

Code Coverage package documentation: docs.unity3d.com/Packages/com.unity.testtools.codecoverage@latest

Unity Forum thread: forum.unity.com/threads/code-coverage-package.777542



## 1. What is Code Coverage (2 min)

<u>Code Coverage</u> is a measure of how much of your code has been executed. It is normally associated with automated tests, but you can gather coverage data in Unity at any time when the Editor is running.

It is typically presented as a <u>report</u> that shows the percentage of the code that has been executed. For automated testing the report does not measure the quality of tests, only whether your code is executed by PlayMode and EditMode tests. It is especially useful to check that critical or high risk areas of your code are covered, because they should receive the most rigorous testing.

It is much easier to accidentally introduce bugs into code that is not covered by tests, because those bugs are not detected straight away by the tests and can instead cause problems later — such as after you have published your game or app.

Additionally, the Code Coverage package offers a <u>Coverage Recording</u> feature which allows capturing coverage data on demand, in case you do not have tests in your project or doing manual testing.

### 2. Install the Code Coverage package (2 min)

Skip this task if the package is already installed

🕂 🔻 Packages: Unity Reg	istry 🔻 Sort: Name 🦊		<b>\$</b> <				
Code Coverage	1.0.0	^					
Core RP Library	12.0.0		Unity Technologies				
Editor Coroutines	1.0.0		Version 1.0.0 - March 09, 2021				
FBX Exporter	4.0.0-pre.4 RC		View documentation • View changelog • View licenses				
High Definition RP	12.0.0	0	Use this package to export code coverage data and reports from your automated tests.				
In App Purchasing	3.0.0-pre.6 RC		Additionally, the Code Coverage package offers a Coverage Recording feature which allows capturing coverage data on demand, for manual testing or when there are no				
Input System	1.0.2	U	automated tests in the project.				
JetBrains Rider Editor	3.0.5 🗸		Registry Unity				
Magic Leap XR Plugin	6.2.2						
ML Agents	1.0.6						
Mobile Notifications	1.3.2						
Oculus XR Plugin	1.8.1	•					
Last update Mar 9, 17:41	C		Install				

Use the Unity Package Manager to find and install the Code Coverage package.

Alternatively, use the Add(+) dropdown and select Add package from git URL... or Add package by name... and type com.unity.testtools.codecoverage

<ul> <li>➡ Package Manager</li> <li>+ ➡ Packages: Unity Registry ➡</li> </ul>	Sort: Name ↓	•	\$
Add package from disk	5.0.4	ĥ	2D Animation Release
Add package from tarball Add package from git URL	4.0.1		Unity Technologies
Add package by name	4.0.2		Version 5.0.4 - February 02, 2021
▶ 2D Sprite	1.0.0		View documentation • View changelog • View licenses

To verify that Code Coverage has been installed correctly, open the Code Coverage window (go to **Window > Analysis > Code Coverage**). If you don't see the **Code Coverage** menu item, then Code Coverage did not install correctly.

### 3. Enable Code Coverage (1 min)

To enable Code Coverage open the **Code Coverage window** (go to **Window > Analysis > Code Coverage**) and select **Enable Code Coverage** if not already selected, to be able to generate Coverage data and reports.

Enable Code Coverage

Note: Enabling Code Coverage adds some overhead to the editor and can affect the performance.

### 4. Understanding the game code: Shoot() function (4 min)

- Go to Asteroids/Scenes in Project View and open the Asteroids scene. This is located in Assets/Samples/Code Coverage/<version>/Code Coverage Tutorial.
- 2. Hit **Play** and play the game for a minute or two.



Use the arrow keys to move and spacebar to shoot.

- 3. Exit PlayMode.
- 4. Open the Scripts/Controllers/SpaceshipController.cs script.
- 5. Study the **Shoot** function.

```
If Weapon is Basic, the Prefabs/Weapons/Projectile prefab is instantiated
If Weapon is Laser, the Prefabs/Weapons/Laser prefab is instantiated
```

### 5. Generate a Coverage report from PlayMode tests (3 min)

1. Open the Code Coverage window (go to Window > Analysis > Code Coverage).

Test Runner Code Coverage		: 🗆 ×
Results Location	C:/Projects/MyProject	Browse
History Location	C:/Projects/MyProject	Browse
Settings		
Enable Code Coverage	×	
Included Assemblies	Unity.TestTools.CodeCoverage.Sample.Asteroids,Unity.TestTools.CodeCove	erage.Sam <del>•</del>
Included Paths	List is Empty	
	Add Folder Add File	
Excluded Paths	List is Empty	
	Add Folder Add File	
Generate HTML Report	~	
Generate Summary Badges	×	
Generate History	Z	
Generate Additional Metrics		
Auto Generate Report	×	
Clear Data Clear History	Generate from Last 🔘 Sta	rt Recording

2. If you see this warning select Switch to debug mode.

Code Coverage requires Code Optimization to be set to debug mode in order to obtain accurate coverage information.

<u>Code Optimization</u> was introduced in Unity 2020.1; in *Release mode* the code is optimized and therefore not directly represented by the original code. Therefore, *Debug mode* is required in order to obtain accurate code coverage information.

#### 3. Click the Included Assemblies dropdown to make sure only

```
Unity.TestTools.CodeCoverage.Sample.Asteroids and Unity.TestTools.CodeCoverage.Sample.Asteroids.Tests are selected.
```



4. Make sure Generate HTML Report, Generate History and Auto Generate Report are all checked.

Generate HTML Report	~
Generate Summary Badges	$\checkmark$
Generate History	$\overline{\mathbf{v}}$
Generate Additional Metrics	
Auto Generate Report	$\checkmark$

5. Switch to the Test Runner, select the PlayMode tab and hit Run All tests.

Test Runner Code Coverage		: 🗆 ×
	PlayMode EditMode	
Run All Run Selected Rerun Failed Clear R	esults	Run all in player (StandaloneWindows)
٩		Nothing 🛛 🗸 🖌 🕢 🔿 🗸
<ul> <li>MyProject</li> <li>Unity.TestTools.CodeCoverage.Samp</li> <li>AsteroidTests</li> <li>CameraTests</li> <li>FXTests</li> <li>GameManagerTests</li> <li>SpaceshipTests</li> <li>UserInterfaceTests</li> </ul>	ple.Asteroids.Tests.dll	
MyProject (16.532s)		

- 6. When the tests finish running, a file viewer window will open up containing the coverage report. Select **index.htm**.
- 7. Look for the classes with low coverage, especially LaserController, BaseProjectile and ProjectileController.

You can sort the results by Line coverage.

Coverage									
Collapse all   Expand all	By Grouping	assembly						Filter:	
- Name	- Covered	- Uncovered	- Coverable	- Total	<ul> <li>Line coverage</li> </ul>	- Covered	+ Total	- Branch coverage	
- Unity.TestTools.CodeCoverage.Sample.Asteroids	371	61	432	701	85.8%		0		
LaserController	2	21	23	36	8.6%	0	0		
BaseProjectile	1	10	11	20	9%	0	0		
ProjectileController	13	4	17	30	76.4%	0	0		
InGameMenuController	13	2	15	29	86.6%	0	0		
SpaceshipController	81	12	93	143	87%	0	0	-	
LifeCounter	39	3	42	63	92.8%	0	0		
GameManager	92	6	98	161	93.8%	0	0	1	
ScoreCounter	51	3	54	83	94.4%	0	0		
AnimatorDisabler	4	0	4	11	100%	0	0		
AsteroidController	39	0	39	62	100%	0	0		
DebrisController	19	0	19	31	100%	0	0	1	
EngineTrail	17	0	17	32	100%	0	0		
+ Unity.TestTools.CodeCoverage.Sample.Asteroids.Tests	714	15	729	1439	97.9%	• 0	0		

See also <u>How to interpret the results</u>.

## 6. Add Weapon tests to improve coverage (3 min)

- 1. Open the Tests/WeaponTests.cs script.
- 2. Uncomment all the tests (from line 35 down to line 237).
- 3. Back in the Test Runner, hit Run All tests again.
- 4. When the tests finish running, a file viewer window will open up containing the coverage report. Select **index.htm**.
- 5. Notice that now **BaseProjectile** and **ProjectileController** coverage is considerably higher, but **LaserController** has not improved much.

Collapse all   Expand all	Grouping	assembly		Compare with: Date			Filter		
- Name	- Covered	- Uncovered	- Coverable	- Total	- Line o	coverage	- Covered	+ Total	- Branch coverage
<ul> <li>Unity.TestTools.CodeCoverage.Sample.Asteroids</li> </ul>	385	47	432	701	89.1%		0	0	
LaserController	2	21	23	36	8.6%		0	0	1
InGameMenuController	13	2	15	29	86.6%		0	0	
SpaceshipController	81	12	93	143	87%		0	0	1
LifeCounter	39	3	42	63	92.8%		0	0	
GameManager	92	6	98	161	93.8%		0	0	
ScoreCounter	51	3	54	83	94.4%		0	0	
AnimatorDisabler	4	0	.4	11	100%		0	0	1
AsteroidController	39	0	39	62	100%		0	0	
BaseProjectile	11	0	11	20	100%	-	0	0	1
DebrisController	19	0	19	31	100%		0	0	
EngineTrail	17	0	17	32	100%	-	0	0	1
ProjectileController	17	0	17	30	100%		0	0	
Unity.TestTools.CodeCoverage.Sample.Asteroids.Tests	849	0	849	1437	100%		0	0	

## 7. Add a test for the LaserController (4 min)

- 1. Open the Tests/WeaponTests.cs script.
- 2. Go to the \_18\_LaserFiresSuccessfully test in *line 225*.
- 3. Uncomment and study the code.
- 4. Back in the Test Runner, hit Run All tests again.
- 5. When the tests finish running, a file viewer window will open up containing the coverage report. Select **index.htm**.
- 6. Notice how the coverage for **LaserController** has improved.

LaserController	15	8	23	36		65.2%	
-----------------	----	---	----	----	--	-------	--

7. Select the **LaserController** class to enter the class view and notice that about 2/3 (65%) of the code is now covered (green).

#	Line	Line coverage
	1	using UnityEngine;
	2	
	3	public class LaserController : BaseProjectile
	4	{
2	5	<pre>public bool isActive = true;</pre>
2	6	public float duration = 0.75f;
	7	
	8	private void Update()
1	9	{
1	10	if (!GameManager.IsPaused)
1	11	{
1	12	if (isActive)
1	13	Expand();
	14	else
0	15	Shrink();
	16	
1	17	duration -= Time.deltaTime;
1	18	if (duration <= 0.0f)
0	19	isActive = false;
1	20	}
1	21	}
	22	
L	23	private void Expand()
1	24	{
1	25	if (transform.localScale.y <= 25.0f)
1	26	<pre>transform.localScale += Vector3.up * Time.deltaTime * 75.0f;</pre>
1	27	}
Į.,	28	
1.1	29	private void Shrink()
0	30	{
0	31	<pre>transform.localScale -= Vector3.up * Time.deltaTime * 75.0f;</pre>
0	32	<pre>transform.position += transform.up * Time.deltaTime * 75.0f;</pre>
0	33	if (transform.localScale.y <= 0.0f)
0	34	Destroy(gameObject);
0	35	}
Ι.	36	}

Complete the **Bonus Task** at the end of the tutorial to get 100% coverage!

### 8. Clear the coverage data (1 min)

- 1. Open the **Code Coverage** window (go to **Window > Analysis > Code Coverage**).
- 2. Select Clear Data and confirm.
- 3. Select **Clear History** and confirm.

## 9. Generate a Coverage report using Coverage Recording (4 min)

- 1. Go to Asteroids/Scenes in Project View and open the Asteroids scene, if not opened already.
- 2. Open the **Code Coverage** window. Make sure **Generate HTML Report**, **Generate History** and **Auto Generate Report** are all checked.



3. Select Start Recording.



4. Hit Play to play the game and exit PlayMode before you get 8000 points.



5. Select Stop Recording.

Stop Recording

- 6. A file viewer window will open up containing the coverage report. Select index.htm.
- 7. Notice that LaserController has 0% coverage.

LaserController	0	23	23	36	0%	
-----------------	---	----	----	----	----	--

- 8. Go back to the Code Coverage window.
- 9. Select Start Recording.
- 10. Now hit **Play** to play the game again but this time **exit** PlayMode when you get **8000** points.
- 11. Select Stop Recording.
- 12. Notice that **LaserController** coverage is now 100%.

LaserController	23	0	23	36		100%	
-----------------	----	---	----	----	--	------	--

See also <u>How to interpret the results</u>.

## 10. Bonus Task (5-8 min)

Write a new test that checks that the laser gets destroyed after 2 seconds, which will also cover the rest of the code in **LaserController**.

*Suggested name:* \_19\_LaserFiresAndIsDestroyedAfterTwoSeconds.

*Hint:* You can use yield return new WaitForSeconds(2f); to wait for 2 seconds.

### Well done for finishing the Code Coverage Tutorial!

For questions and feedback please reach out to us in the dedicated Code Coverage package <u>forum thread</u>.