

Developer

/ Documentation

/ Unreal Engine ▾

/ Unreal Engine 5.4 Documentation

/ Testing and Optimizing Your Content

/ Using Oodle

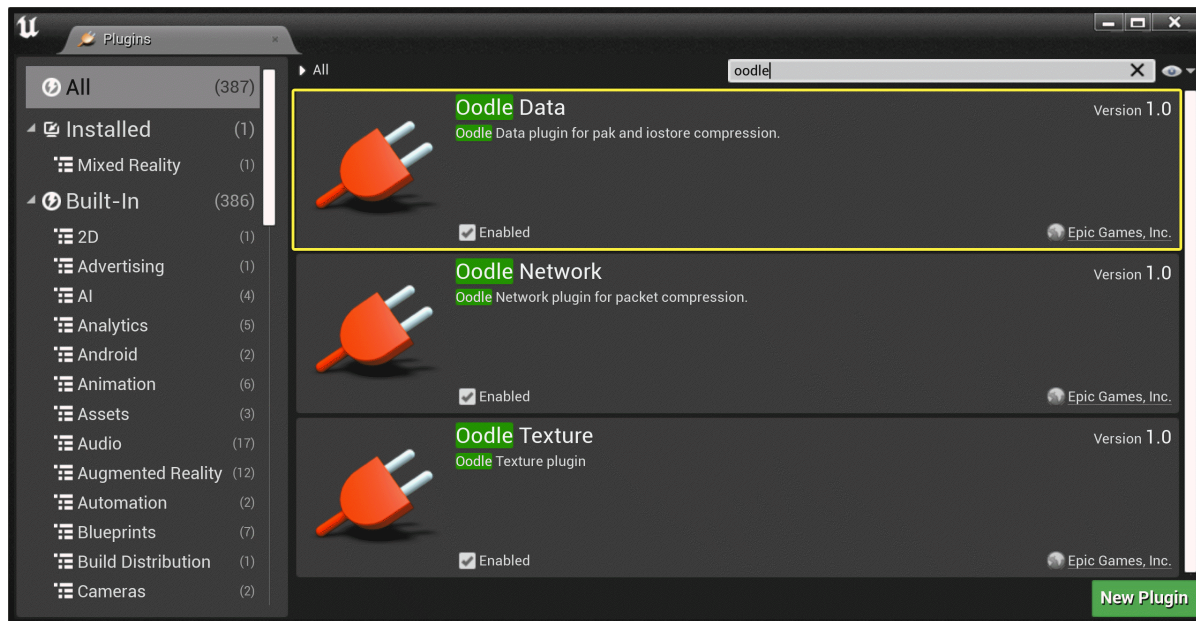
/ Oodle Data

Oodle Data

An overview of using Oodle Data to optimize delivering your package files to users.



Oodle Data provides a compression format for `.pak` files and IOStore files. It is supplied as a plugin which should be enabled by default.



The parameters and settings for using Oodle Data with IOStore files are identical to those for .pak files in **Unreal Engine**.

When packaging, you will know you are using Oodle Data if you see a log similar to the following, with your specific settings listed for method and level:

```
Oodle v2.9.0 initializing with method=Kraken, level=3=Fast
```

In other contexts where you need Oodle Data to decode the packages files, you will see the following in the log:

```
LogPluginManager: Mounting plugin OodleData
```

Key Concepts for Oodle Data

Oodle Data exposes two controls for managing the output: The **compression method** and the **compression level**. It is important to understand the distinction between them:

- The compression *method* trades off how small the data gets with how fast it is to decode.
- The compression *level* determines how long it takes to encode the data.



The decoder at runtime never needs to know what method was used.

Compression Methods

There are four different compression methods available, representing different levels of compression and decode speed.

Method	Description
Kraken	High compression with good decode speed, the usual default.
Mermaid	Less compression and faster decode speed; effective when CPU usage is limited or on platforms with less CPU power.
Selkie	Even less compression and faster than Mermaid.
Leviathan	More compression and slower to decode than Kraken.

Compression Level (Effort Level)

Compression Level is a number between -4 and 9, representing encoding speed. Compression level values are referred to as follows:

Level	Name	Other Information
-4	HyperFast4	
-3	HyperFast3	
-2	HyperFast2	
-1	HyperFast1	
0	None	Just copies the raw bytes.
1	SuperFast	
2	VeryFast	
3	Fast	Good for daily use.
4	Normal	
5	Optimal1	
6	Optimal2	Recommended baseline optimal encoder.

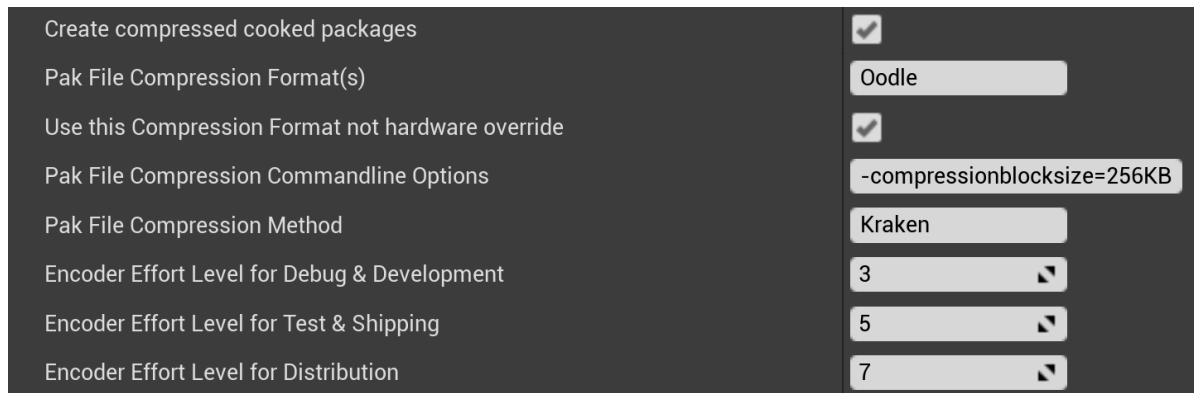
Level	Name	Other Information
7	Optimal3	
8	Optimal4	
9	Optimal5	

Enabling Oodle Data

Due to various overrides, there are several different places where you need to enable and configure Oodle Data. The baseline method for enabling it is using the **Packaging** settings in the **Project Settings** window.



In the Project Packaging settings, you will have to expand the **Advanced** parameters to see these.



These settings can also be edited directly in your `BaseGame.ini` file under the `[/Script/UnrealEd.ProjectPackagingSettings]` header.

Properties / Settings

These are the relevant properties in the Project Packaging area, with the equivalent `.ini` file setting.

Property	.ini File Setting	Description
Create compressed cooked packages	bCompressed	When enabled, Unreal will compress output packages unless overridden.
Pak File Compression Format(s)	PakFileCompressionFormats	Sets the list of compression formats. Set it to Oodle to use Oodle Data.
Pak File Compression Commandline Options	PakFileAdditionalCompressionOptions	Specifies additional options to pass to the compression format. For Oodle, we recommend this be set to <code>-compressionblocksize=256KB</code> .
Use this Compression Format not hardware override	bForceUseProjectCompressionFormatIgnoreHardwareOverride	If set, the <code>HardwareCompressionFormat</code> in the <code>DataDrivenPlatformInfo.ini</code> file will be ignored, and these settings will be used. So, you could set things up using the above settings, use <code>HardwareCompressionFormat</code> to ignore them, and then ignore THAT with this setting, getting back to where you started.
Pak File Compression Method	PakFileCompressionMethod	Specifies the compression method as previously described (for example,

Property	.ini File Setting	Description
		Kraken).
Encoder Effort Level for Debug & Development	PakFileCompressionLevel_DebugDevelopment	Specifies the amount of time to spend encoding. This is a number from the previously-described Compression Level settings (3 by default).
Encoder Effort Level for Test & Shipping	PakFileCompressionLevel_TestShipping	Specifies the amount of time to spend encoding. This is a number from the previously-described Compression Level settings (5 by default).
Encoder Effort Level for Distribution	PakFileCompressionLevel_Distribution	Specifies the amount of time to spend encoding. This is a number from the previously-described Compression Level settings (7 by default).

Example Settings

In your `BaseGame.ini` file, this is a representative set of settings:

```

1
2 [/Script/UnrealEd.ProjectPackagingSettings]
3 bCompressed=True
4 PakFileCompressionFormats=Oodle
5 PakFileAdditionalCompressionOptions=-compressionblocksize=256KB

```


```
6 PakFileCompressionMethod=Kraken
7 PakFileCompressionLevel_Distribution=7
8 PakFileCompressionLevel_TestShipping=5
9 PakFileCompressionLevel_DebugDevelopment=3
10 bForceUseProjectCompressionFormatIgnoreHardwareOverride=False
11
```

 Copy full snippet

Platform-Specific Exceptions

If a given target platform supports hardware data compression, it is likely exposed in the `DataDrivenPlatformInfo.ini` file in the platform's configuration directory, for example:

```
1
2 [DataDrivenPlatformInfo]
3 HardwareCompressionFormat=Zlib
4
```

 Copy full snippet

However, if you want to bypass the hardware data compression and use Oodle Data anyway, you can set `bForceUseProjectCompressionFormatIgnoreHardwareOverride=True` in the platform's `(Platform)Game.ini` file settings.

This will cause the packaging for the target platform to use Oodle Data, rather than the (in this example) hardware zlib.