```
Developer

/ Documentation

/ Unreal Engine 

/ Unreal Engine 5.4 Documentation

/ Programming and Scripting

/ Unreal Architecture

/ String Handling
```

## **FText**

/ FText

Reference for creating, converting, comparing, and more with FText in Unreal Engine.



In **Unreal Engine (UE)** the primary component for <u>text localization</u> is the <u>FText</u> class. All user-facing text should use this class, as it supports text localization by providing the following features:

- Creating localized text literals.
- Formatting Text (to generate text from a placeholder pattern).
- Generating text from numbers.
- Generating text from dates and times.
- Generating derived text, such as making text upper or lower case.

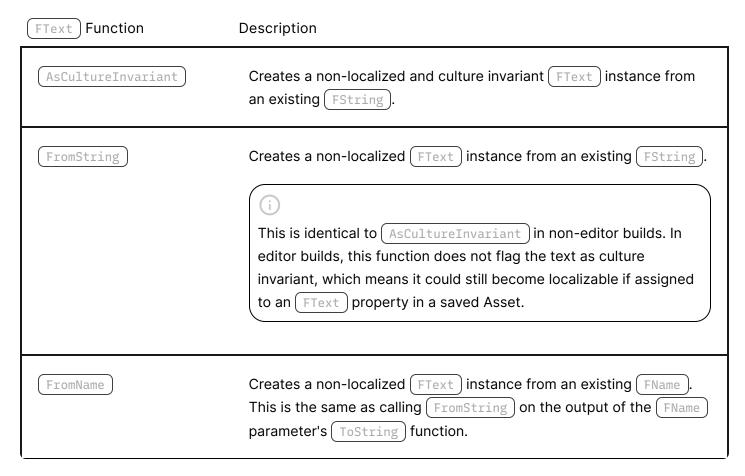
FText also features the AsCultureInvariant function (or the INVTEXT macro), which creates non-localized, or "culture invariant" text. This is useful for things like converting a player name from an external API into something you can display in your user interface.

You can create a blank <code>FText</code> using either <code>FText::GetEmpty()</code> , or by using just <code>FText()</code> .

## **Conversion**

FText can be converted to and from FString. However, because FText contains character strings linked to localization data, while FString only contains a character string, these methods are inherently lossy and will discard (or fail to create) localization data. An alternative method, Text Value Marshalling provides lossless conversion, although this method produces data more suitable for internal marshalling than human consumption.

The following conversion functions produce FText strings, but without localization data:



To convert from FText to FString, use the ToString function. The resulting FString will hold the string data from the FText, but not the localization data.

# Comparison

Because FText data is more complex than simple strings, it does not support overloaded-operator comparisons. Instead, it provides several functions to perform comparisons that recognize the nuanced data that it contains. The following comparison functions are available:

FText Function	Description
EqualTo	This function takes an <a href="ETextComparisonLevel">ETextComparisonLevel</a> value to determine what comparison rules to use. It returns a bool indicating whether or not the calling <a href="FText">FText</a> matches the other under those comparison rules.
EqualToCaseIgnored	This function is a wrapper for calling EqualTo with an ETextComparisonLevel value of Second. The return value comes directly from EqualTo.
CompareTo	This function takes an <a href="ETextComparisonLevel">ETextComparisonLevel</a> value to determine what comparison rules to use. It returns an <a href="int32">int32</a> like most string-or memory-comparison functions, where zero indicates equality, and negative or positive values indicate that the calling <a href="FText">FText</a> sorts lower or higher, respectively, relative to the <a href="FText">FText</a> parameter.
CompareToCaseIgnored	This function is a wrapper for calling CompareTo with an ETextComparisonLevel value of Second. The return value comes directly from CompareTo.

# Using FText in User Interfaces Slate/UMG

Slate and UMG use FText attributes or arguments for any built-in Widgets that show or manage user-facing text. We recommend using FText for any custom Widgets that you build.

## **HUD/Canvas**

To display FText through the HUD system with Canvas, create a new FCanvasTextItem and set its Text variable to the text you want to display, as in the following example code:

```
FCanvasTextItem TextItem(FVector2D::ZeroVector, TestHUDText, BigFont,
FLinearColor::Black);

// Add the text into the FCanvasTextItem.

TextItem.Text = FText::Format(LOCTEXT("ExampleFText", "You currently have {0} health left."), CurrentHealth);

// Draw the text to the screen with FCanvas::DrawItem.

Canvas->DrawItem(TextItem, 10.0f, 10.0f);
```

Copy full snippet

When using Canvas in a HUD, you must call DrawItem within the DrawHUD function, or call it in a function chain that begins with DrawHUD.