Supporting Screen Readers

Explains how to enable 3rd party screen reader support for your project.

> ⓘ Learn to use this **Experimental** feature, but use caution when shipping with it.

UE now supports third party screen readers for Windows or VoiceOver on iOS. This makes it possible to ensure your game UI is accessible and helps you comply with CVAA standards. Screen readers, such as NVDA and JAWS, allow a software application's UI to be narrated to the user. This is a critical feature that enables those who are visually impaired to use and navigate software applications.

There are now APIs included in UE to allow the use of 3rd party screen readers to read UI text. This supports a number of common UMG widgets, such as Text Block, Editable Text Box, Slider, Button, and Checkbox. This built-in functionality removes the need to implement custom text-to-speech technology, making screen readers easier to support.

Enabling Screen Reader Support

To enable screen reader support, go into either your project or Engine **Console Variable** configuration file. Once in the file, add the variable `Accessibility.Enable=1`.

Using Accessibility in UMG

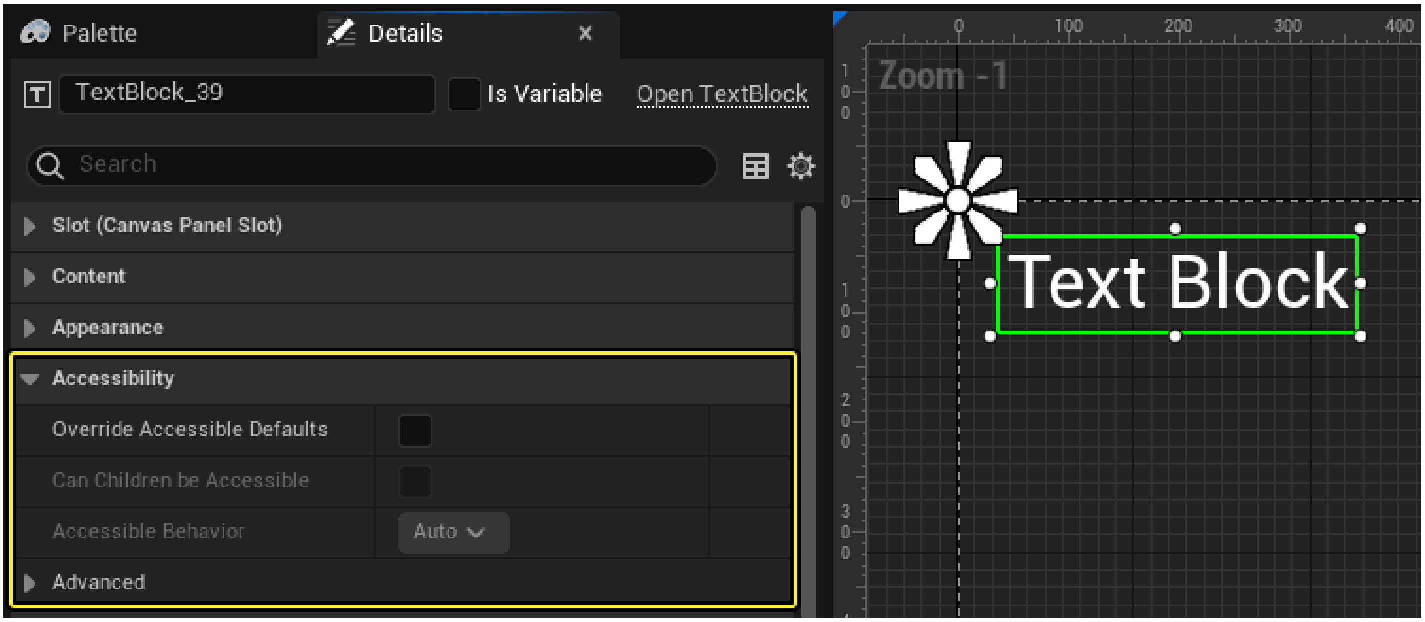
To make your UMG widgets accessible with screen readers, go to the **Details** panel for any of your widgets, and you will see a section for **Accessibility**. By default, the accessibility settings are enabled. If you would like to adjust the default settings, check **Override Accessible Defaults**. This enables you to adjust **Can Children Be Accessible**, **Accessible Behavior**, and **Accessible Summary Behavior** options.

The **Can Children Be Accessible** checkbox indicates whether or not the child widgets inherit the Accessible Behavior settings.

Accessible Behavior is where you indicate how you want the screen reader API to read the widget text. You can choose from the following options:

- **Auto**: Automatically reads the assigned text for a widget.
- **Summary**: Screen reader concatenates the text of all child widgets.
- **Custom**: Screen reader uses custom text pre-coded by developer.
- **Tool Tip**: Screen reader only reads **Tool Tip Text**.
- **Not Accessible**: Ignores screen reader API.

If you select Summary in Accessible Behaviors, **Accessible Summary Behavior** shows you what text will be read when concatenating the text from the parent widget and all subsequent child widgets.



TextBlock_39 □ Is Variable

Override Accessible Defaults □

Can Children be Accessible □

Accessible Behavior: Auto

Custom Widget Support (C++)

You can add additional support by creating new C++ classes for their custom Slate widgets. To add specialized support to a new widget, you must override `SWidget::CreateAccessibleWidget()` and return an instance of a new `FSlateAccessibleWidget`. You can find an example of this in the file `SlateAccessibleWidgets.h` under FSlateAccessibleButton. This example shows how screen readers enable users to click the button.

> You can compile out the screen reader code from Slate and platform layers when `bCompileWithAccessibilitySupport` is set to false.