

AF Better Dropdown

Arcana Futura • uGUI • TextMesh Pro

A lightweight, fully customizable uGUI dropdown built from regular Unity UI elements, with single and multiple selection, optional icons, and hierarchy, inspector and runtime population.

Overview

AF Better Dropdown is designed as a simpler alternative to Unity's standard dropdown setup. It uses regular uGUI components and TextMesh Pro, so the supplied prefabs can be restyled, resized, rearranged, or replaced using normal Unity workflows.

The dropdown supports three practical ways of supplying items: manually through the scene hierarchy, through the component Inspector, or programmatically at runtime.

Requirements

- Unity uGUI
- TextMesh Pro (TMP)
- No third-party dependencies

Quick Setup

1. Add the AF Better Dropdown panel prefab to a Canvas and position it like any other UI element.
2. Choose Selection Mode: Single or Multiple.
3. Choose Item Source: Hierarchy, Inspector or API.
4. Enable/disable 'Use Icons' as needed.
5. Enable/disable 'Mark Selected' as needed to show the supplied selection mark.
6. Customize the supplied uGUI/TMP objects, sprites, spacing, sizing, and colours as required.

Item Sources

Hierarchy

Add or duplicate AFBetterDropdownItem objects under AFBetterDropdown/Panel object, then edit each item's TMP text and optional icon directly in the hierarchy. The dropdown discovers and binds these items at runtime.

Keep the dropdown Panel active while editing hierarchy items so Unity's layout system can position newly added item prefabs correctly. AFBetterDropdownPanel closes the Panel when runtime begins.

Inspector

Inspector mode stores item text and optional icons in the AFBetterDropdownPanel component. Use the + button to add entries. The dropdown creates the visual item instances from the assigned Item Prefab at runtime.

This is useful when you want the item data kept safely in one Inspector list rather than maintaining multiple item objects manually.

Runtime

Populate items dynamically from code using any data source. Use SetItems() to supply a list or AddItem() to add individual items.

See provided example in the demo scene on APIExample object, and Scripts/AFBetterDropdownAPIExample script

Selection Behaviour

Single

Allows selection of one option from the dropdown. Selecting an item replaces the current selection if any, updates the displayed value, fires the selection events, and closes the dropdown.

Multiple

Allows selection of multiple options (ie. All that apply). Selecting an item toggles that item without closing the dropdown. The dropdown remains open while the user makes additional selections and closes when the user clicks outside of it. The collapsed display uses a separate numeric TMP object and localizable text TMP object (for example: 3 selected).

Icons and Selection Marks

Use Icons controls whether item icons are displayed. Items without an assigned sprite hide their icon object automatically. Mark Selected controls whether the supplied selected-mark object is shown for selected items. Both features are optional; the item prefab remains ordinary editable uGUI.

Localization

You can easily localize the dropdown as all text is kept in separate TMP objects rather than being hard-coded into the component. This includes No Selection Text and the Multiple Count Text. These objects can be connected to your preferred localization system in the same way as other TMP text in your project.

Inspector Assignments

Assignment	Purpose
Trigger Button	The button used to open or close the dropdown.
Blocker Prefab	A transparent Canvas-level button instantiated while the dropdown is open. Clicking outside the panel closes it.
Panel	The popup object containing the dropdown items.
Panel	Parent transform used for hierarchy, Inspector-generated, and runtime-generated items.
Item Prefab	AFBetterDropdownItem prefab used when items are generated.
Single Value Text	Displays the selected item text in Single mode.
No Selection Text	Displayed when no item is selected.
Multiple Count Number	Displays the number of selected items in Multiple mode.
Multiple Count Text	Localizable text displayed alongside the multiple-selection count.

Events

Event	Purpose
On Selection Changed	Invoked whenever the current selection changes.
On Single Selection Changed	Invoked in Single mode and supplies the selected item's zero-based position.
On Opened	Invoked when the dropdown panel opens.
On Closed	Invoked when the dropdown panel closes.

API Examples

Add the namespace to scripts that interact with the dropdown:

```
using ArcanaFutura.UI.BetterDropdown;
```

Populate a runtime list

```
[SerializeField] private AFBetterDropdownPanel dropdown;

private void Start()
{
    dropdown.SetItems(new[]
    {
        "Option 1",
        "Option 2",
        "Option 3"
    });
}
```

Add an item with an icon

```
[SerializeField] private AFBetterDropdownPanel dropdown;
[SerializeField] private Sprite itemIcon;

private void AddExample()
{
    dropdown.AddItem("Option 4", itemIcon);
}
```

Read the selected item in Single mode

```
int selectedItem = dropdown.SelectedIndex;

if (selectedItem >= 0)
{
    AFBetterDropdownItem item = dropdown.GetItem(selectedItem);
    Debug.Log(item.Text);
}
```

SelectedIndex is zero-based. A value of -1 means that no item is selected.

Read selected items in Multiple mode

```
foreach (int itemNumber in dropdown.SelectedIndices)
{
    AFBetterDropdownItem item = dropdown.GetItem(itemNumber);
    Debug.Log(item.Text);
}
```

SelectedIndices is the API collection of zero-based item positions. GetItem returns the corresponding AFBetterDropdownItem, from which Text and Icon can be read.

Set the selection from code

```
// Single item
dropdown.SetSelected(2);

// Multiple items
```

```
dropdown.SetSelected(new[] { 0, 2, 4 });
```

Clear or select all

```
dropdown.ClearSelection();  
dropdown.SelectAll(); // Multiple mode only
```

Check whether an item is selected

```
if (dropdown.IsSelected(1))  
{  
    Debug.Log("The second item is selected.");  
}
```

React to selection changes

```
private void OnEnable()  
{  
    dropdown.SelectionChanged += HandleSelectionChanged;  
}  
  
private void OnDisable()  
{  
    dropdown.SelectionChanged -= HandleSelectionChanged;  
}  
  
private void HandleSelectionChanged()  
{  
    // Read SelectedIndex or SelectedIndices here.  
}
```

Open and close from code

```
dropdown.Open();  
dropdown.Close();  
dropdown.Toggle();
```

Public API Reference

Member	Purpose
SelectedIndex	Selected item position in Single mode; -1 when nothing is selected.
SelectedIndices	Read-only collection containing the selected item positions.
ActiveItems	Read-only list of the currently active dropdown items.
IsOpen	True while the dropdown panel is open.
SetItems(...)	Replaces the current runtime item set.
AddItem(text, icon)	Adds one runtime item using the assigned Item Prefab.
SetSelected(...)	Sets one or more selected items.
ClearSelection()	Clears the current selection.
SelectAll()	Selects all items in Multiple mode.
IsSelected(item)	Checks whether an item position is selected.
GetItem(item)	Returns the AFBetterDropdownItem at the supplied position.
Refresh()	Rebuilds/binds items and refreshes selection/display state.

Customization Notes

AF Better Dropdown intentionally uses familiar Unity components rather than a proprietary visual system. The supplied Button, Image, TMP, Layout Group, Canvas, and item objects can be modified like normal uGUI. The dropdown logic does not require the supplied visual style.

The blocker is instantiated on the root Canvas while the dropdown is open so it can receive clicks outside the popup without being constrained by the dropdown's local hierarchy.

Namespace

ArcanaFutura.UI.BetterDropdown

Contact

Have a question, need help, or found a bug? Please use our online contact form and we'll get back to you as soon as possible.

Website: <https://arcanafutura.eu/>

Support: <https://arcanafutura.eu/asset-support>

License and Disclaimer

AF Better Dropdown is provided under the applicable Unity Asset Store license. It may be used and modified as part of your own commercial or non-commercial projects, subject to the terms of that license.

You may modify the supplied source code for use in your projects. Modification does not grant the right to redistribute or resell AF Better Dropdown itself. The asset, its source code, or modified versions of the asset may not be redistributed, resold, sublicensed or otherwise made available as a standalone asset or development tool.

AF Better Dropdown is provided "as is", without warranties of any kind. Arcana Futura is not liable for losses or damages resulting from its use.

AF Better Dropdown is provided under the Unity Asset Store End User License Agreement (EULA).

<https://unity.com/legal/as-terms>

Arcana Futura