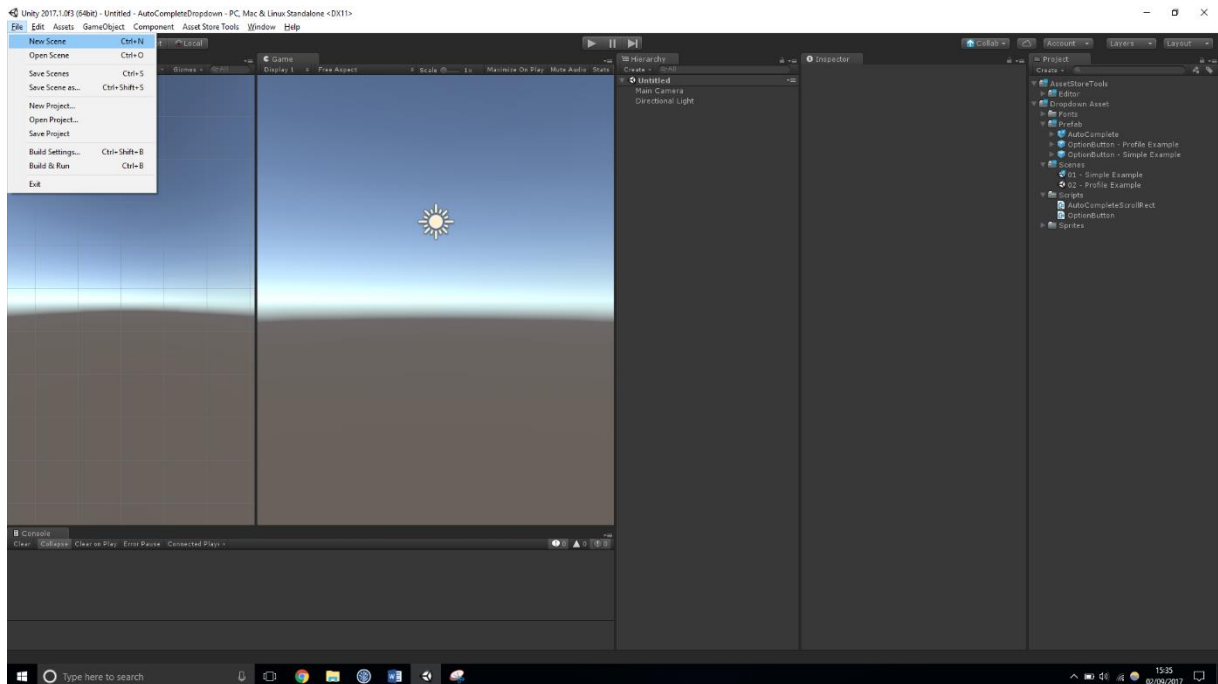


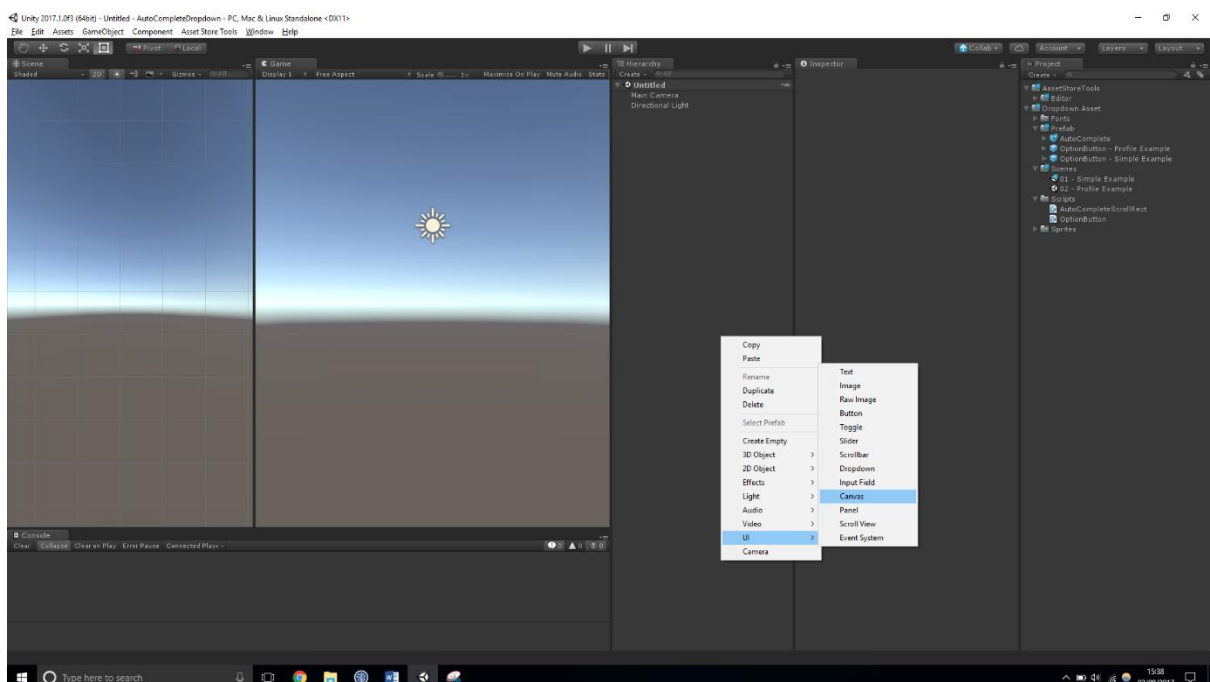
# AutoComplete Dropdown

## Setup

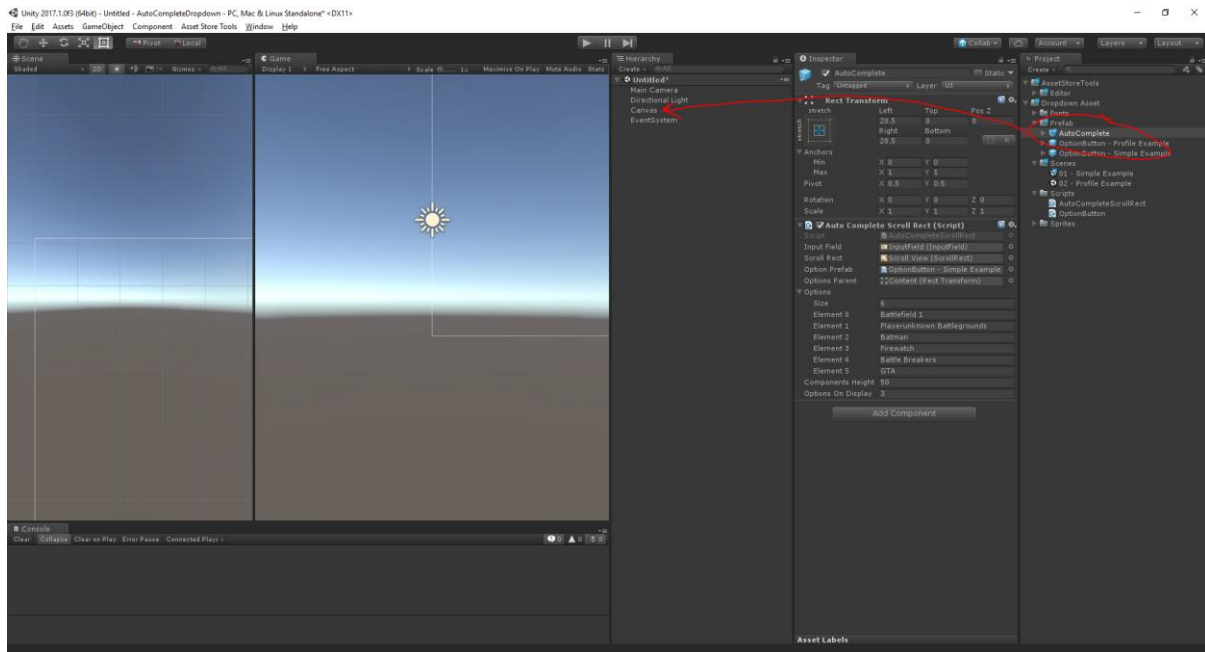
1. Create a new scene > File > new Scene



2. Create a canvas component in the scene -> Right-click in the hierarchy window -> UI -> Canvas

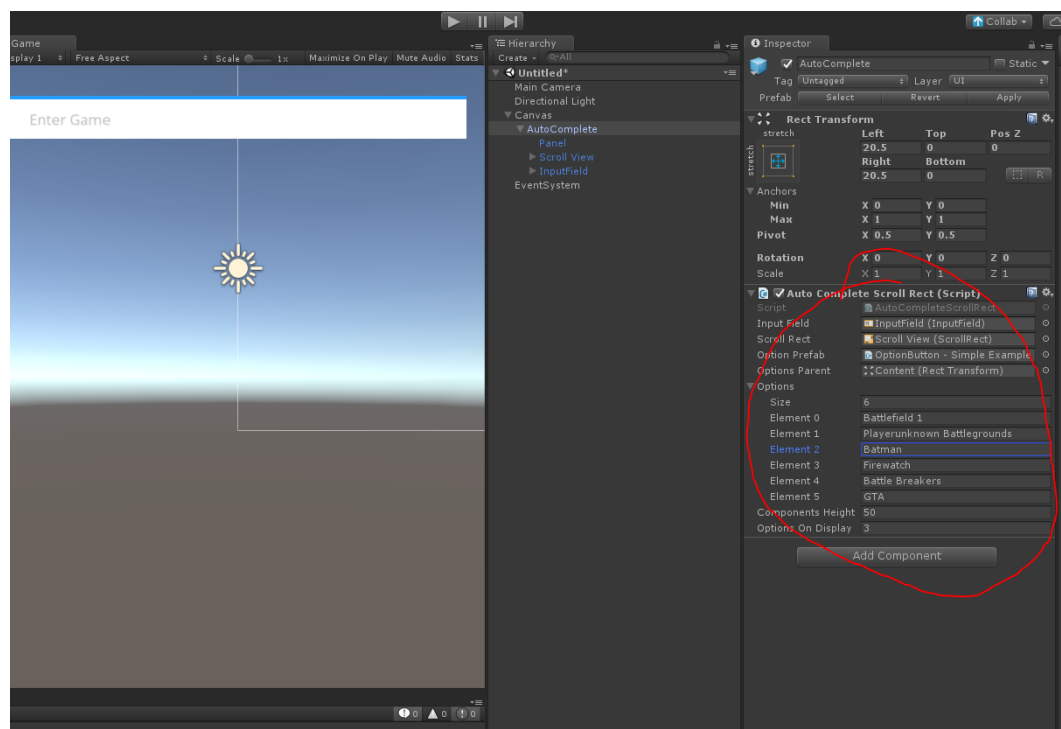


### 3. Drag and drop the AutoComplete component onto the canvas component and your done!

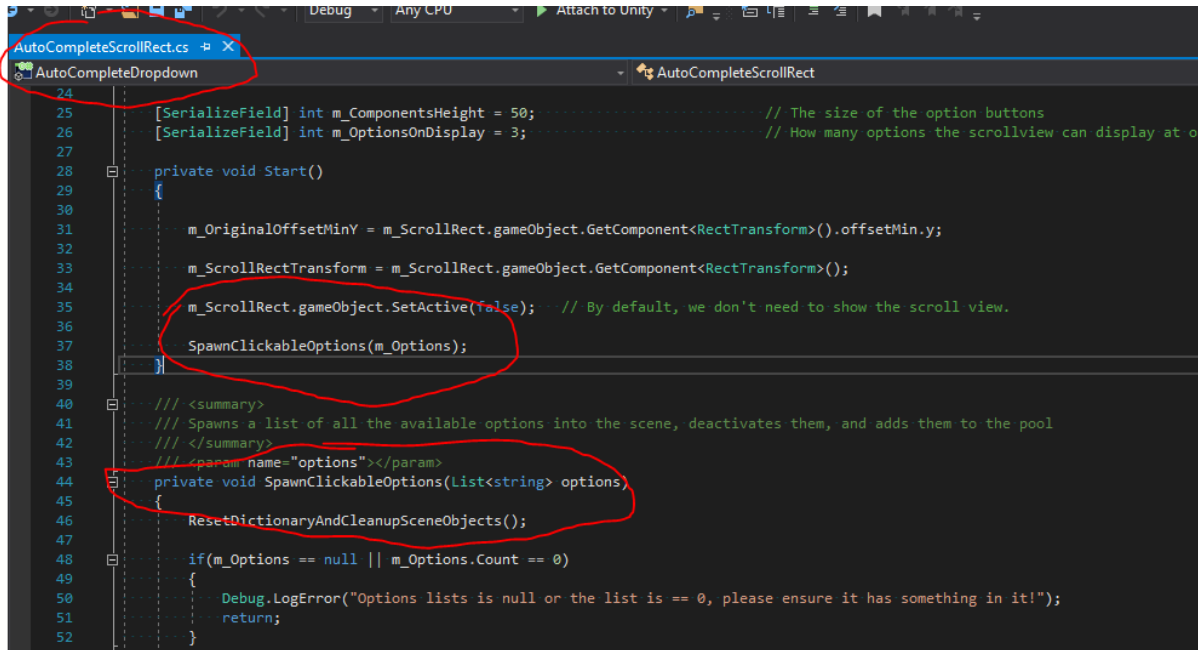


## Editing

- Options: Add and remove options that appear in the drop box by selecting the autocomplete prefab in the hierarchy and modifying the options array in the inspector.
- Components height: Modify the size of the option buttons that spawn by modifying the components heights field by selecting the autocomplete prefab in the hierarchy and modifying the components heights field in the inspector.
- Modify the number of buttons that are on display at one time by selecting the autocomplete prefab in the hierarchy and modifying the options on display field in the inspector.



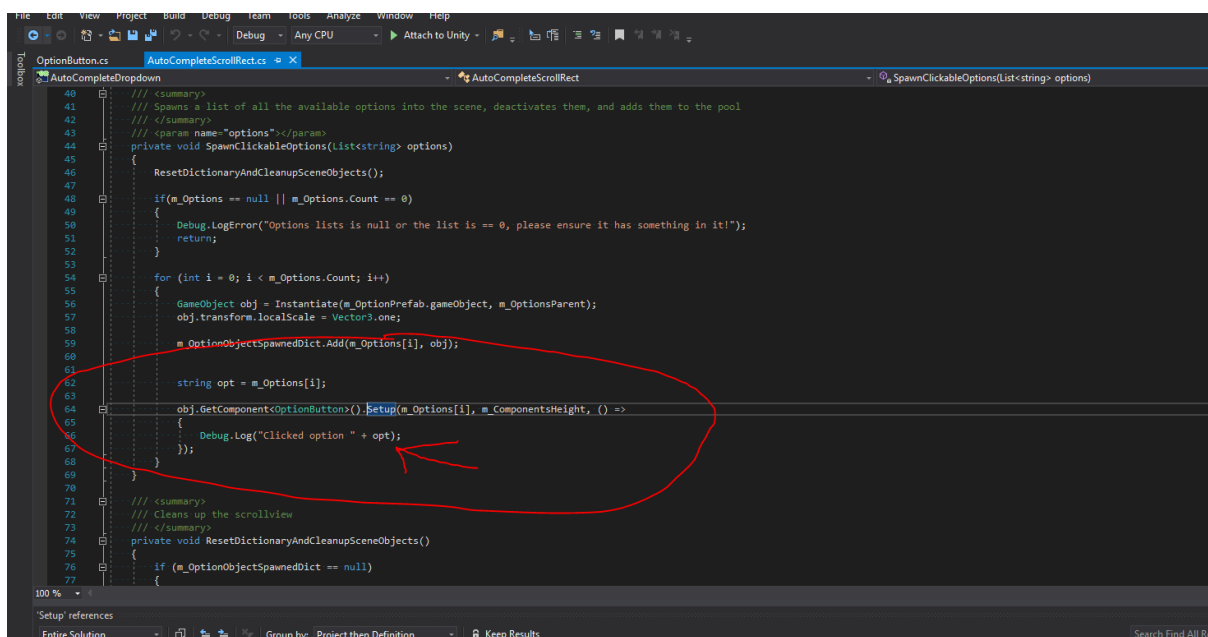
- Repopulate: If you want to re-populate the options during runtime you can simply call the method `SpawnClickableOptions(List<string> options)` passing in a list of options you want to repopulate the auto complete list to display. You'll find this in the `AutoCompleteScrollRect.cs`



This screenshot shows the `AutoCompleteScrollRect.cs` file in a code editor. The file is titled `AutoCompleteDropdown` in the tab. Red circles highlight the `SpawnClickableOptions(m_Options);` call on line 37 and the `private void SpawnClickableOptions(List<string> options,` method signature on line 44. The code includes comments and a `Debug.LogError` call for validation.

```
24
25 [SerializeField] int m_ComponentsHeight = 50; // The size of the option buttons
26 [SerializeField] int m_OptionsOnDisplay = 3; // How many options the scrollview can display at o
27
28 private void Start()
29 {
30     m_OriginalOffsetMinY = m_ScrollRect.gameObject.GetComponent<RectTransform>().offsetMin.y;
31     m_ScrollRectTransform = m_ScrollRect.gameObject.GetComponent<RectTransform>();
32     m_ScrollRect.gameObject.SetActive(false); // By default, we don't need to show the scroll view.
33     SpawnClickableOptions(m_Options);
34 }
35
36
37
38
39
40
41 /// <summary>
42 /// Spawns a list of all the available options into the scene, deactivates them, and adds them to the pool
43 /// </summary>
44 /// <param name="options"></param>
45 private void SpawnClickableOptions(List<string> options,
46 {
47     ResetDictionaryAndCleanupSceneObjects();
48     if(m_Options == null || m_Options.Count == 0)
49     {
50         Debug.LogError("Options lists is null or the list is == 0, please ensure it has something in it!");
51         return;
52     }
53 }
```

- Events: I've left an example where you can hook into the click event of each button allowing you to know which option the user has selected. You'll find this in the `AutoCompleteScrollRect.cs`



This screenshot shows the `AutoCompleteScrollRect.cs` file in a code editor, specifically the `SpawnClickableOptions` method. Red circles highlight the `obj.GetComponent<OptionButton>().Setup(m_Options[i], m_ComponentsHeight, () =>` line (line 64) and the `Debug.Log("Clicked option " + opt);` line (line 66). A red arrow points to the `Debug.Log` statement. The code includes comments and a `Debug.LogError` call for validation.

```
40
41 /// <summary>
42 /// Spawns a list of all the available options into the scene, deactivates them, and adds them to the pool
43 /// </summary>
44 /// <param name="options"></param>
45 private void SpawnClickableOptions(List<string> options)
46 {
47     ResetDictionaryAndCleanupSceneObjects();
48     if(m_Options == null || m_Options.Count == 0)
49     {
50         Debug.LogError("Options lists is null or the list is == 0, please ensure it has something in it!");
51         return;
52     }
53     for (int i = 0; i < m_Options.Count; i++)
54     {
55         GameObject obj = Instantiate(m_OptionPrefab.gameObject, m_OptionsParent);
56         obj.transform.localScale = Vector3.one;
57         m_OptionObjectSpawmedDict.Add(m_Options[i], obj);
58         string opt = m_Options[i];
59         obj.GetComponent<OptionButton>().Setup(m_Options[i], m_ComponentsHeight, () =>
60         {
61             Debug.Log("Clicked option " + opt);
62         });
63     }
64 }
65
66
67
68
69
70
71 /// <summary>
72 /// Cleans up the scrollview
73 /// </summary>
74 private void ResetDictionaryAndCleanupSceneObjects()
75 {
76     if (m_OptionObjectSpawmedDict == null)
77     {
78     }
79 }
```