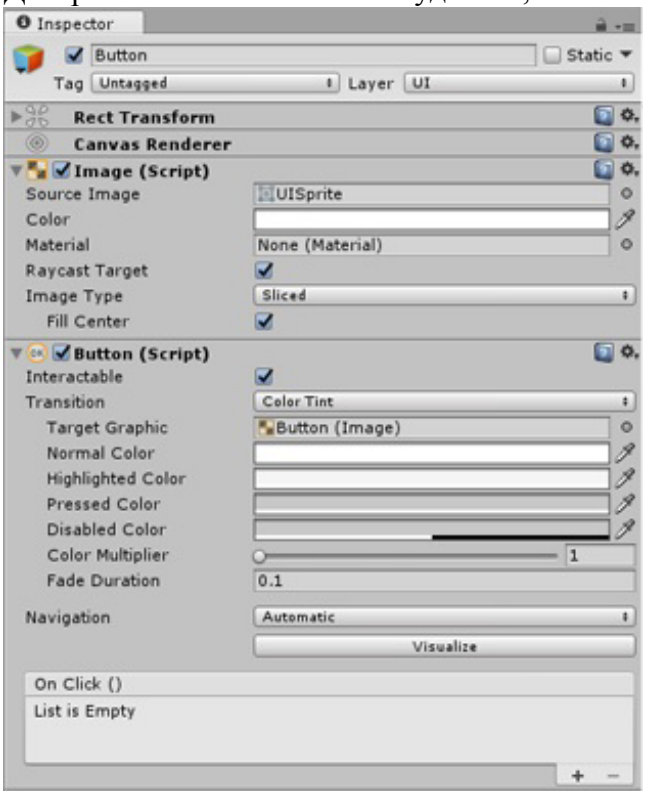
Интерактивные UI-объекты и их особенности

Объект Button

Create | UI |Button

Дочерний объект Text можно удалить, если не требуется использовать кнопку с текстом.



Настройки компонента Image определяют свойства кнопки в стандартном состоянии.

Компонент Button

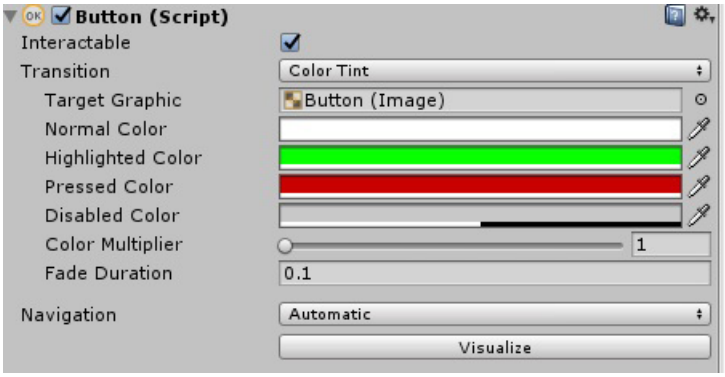
Interactable — при его отключении кнопка становится неактивной.

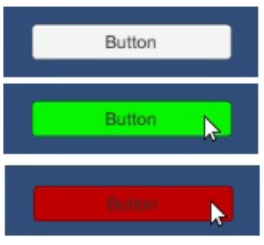
С компонентом Button связывается событие **OnClick**, которое можно определить либо с помощью инспектора, либо программно (с помощью метода AddListener). Данное событие наступает только в случае, если и нажатие, и отпускание левой кнопки мыши произошло над объектом Button.

Transition определяет способ визуализации различных состояний кнопки (normal, highlighted, pressed, disabled). Кнопка переходит в состояние highlighted, если над ней перемещается указатель мыши, однако после нажатия на кнопку (и ее отпускания) она остается в состоянии highlighted даже после того, как указатель мыши ее покинет. Кнопка перейдет в состояние normal только при щелчке мышью над другой областью канвы. Состояние disabled наступает при отключении свойства Interactable.

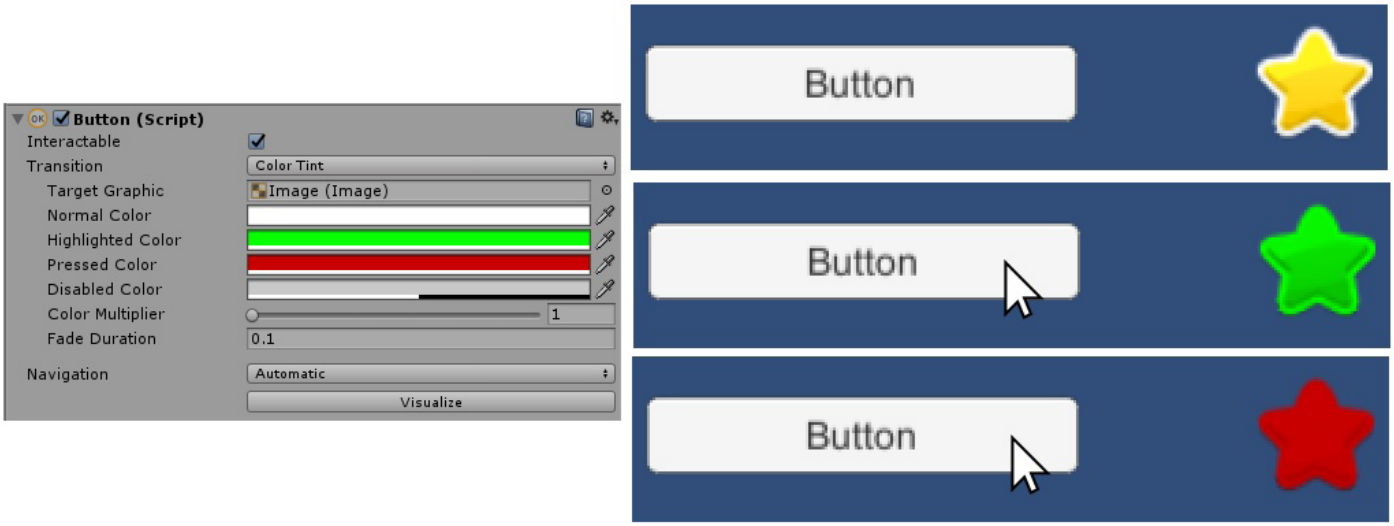
Предусмотрено четыре способа визуализации: None, Color Tint, Sprite Swap, Animation.

Color Tint





Свойство **Target Graphic** позволяет перенаправить эффекты визуализации на другой графический элемент интерфейса.



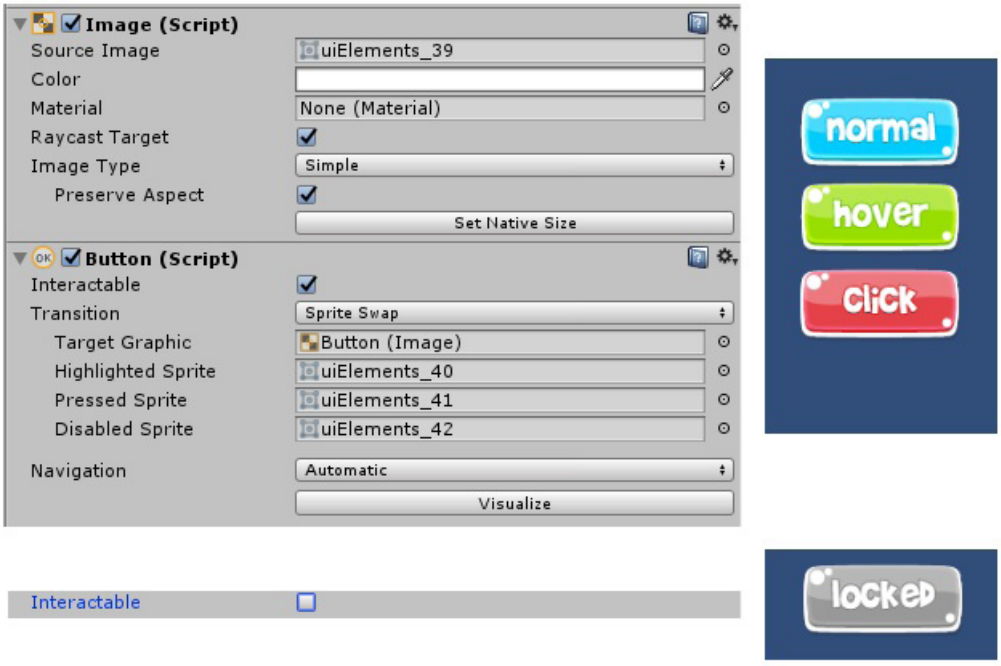
Указанные цветовые эффекты возможны только в случае, если кнопка в состоянии normal имеет белый (или светлый) цвет. Если исходный цвет черный, то эффекты невозможны, и наоборот, установка белого цвета для какого-либо режима никак не скажется на виде кнопки.

Color Multiplier определяет интенсивность цветового эффекта, в частности, уровень увеличения непрозрачности (если исходный цвет является частично прозрачным).

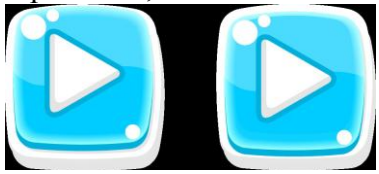
Fade Duration определяет длительность (в секундах) смены цветового режима.

Sprite Swap

Для разных состояний кнопки используются разные изображения.

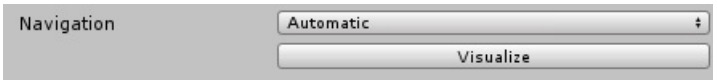


Рекомендуемый вариант использования данного режима — применение для состояния нажатой кнопки изображения, смещенного вниз по сравнению с изображениями для других состояний.



Режим Animation мы не рассматриваем.

Свойство **Navigation** позволяет настраивать режим перемещения между различными интерактивными UI-объектами.



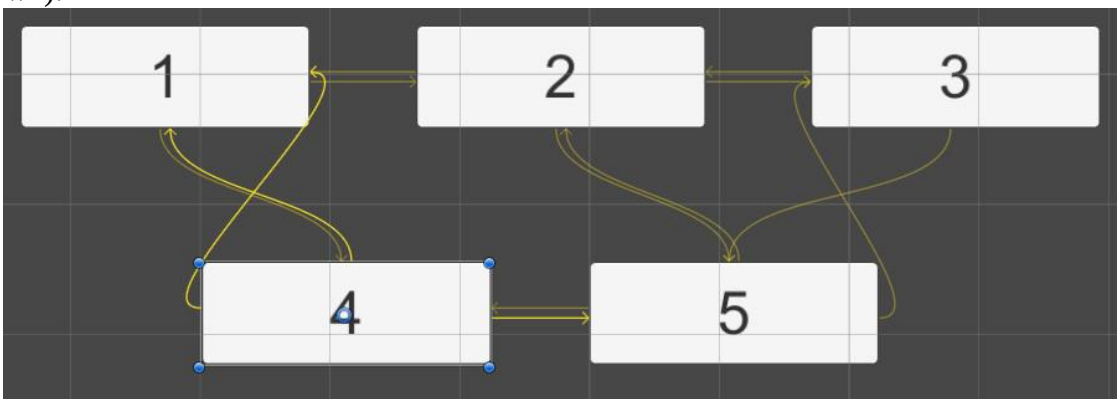
В компоненте Event System можно указать, какой объект будет автоматически выделен (свойство First Selected). Если это свойство не настроено, то режим навигации будет активирован только после выбора какого-либо UI-объекта.

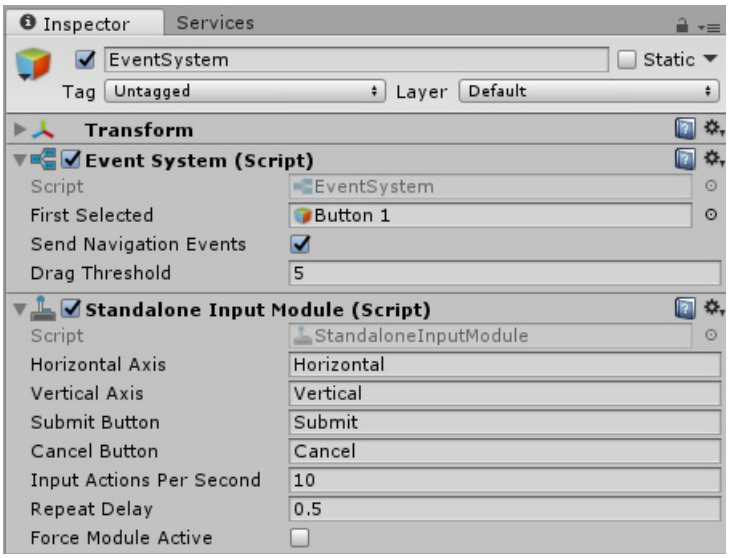
3

Имеется пять вариантов свойства Navigation: None, Horizontal, Vertical, Automatic, Explicit.

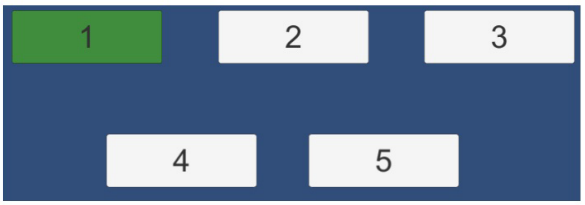
Целесообразно установить одинаковые значения вариантов Navigation для всех интерактивных объектов.

Кнопка Visualize позволяет визуализировать текущие настройки перемещения. Ниже приведен пример визуализации для варианта Automatic (точка выхода линии соответствует нажатой клавише: Left, Right, Up, Down):

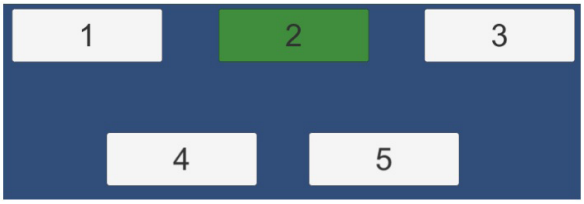




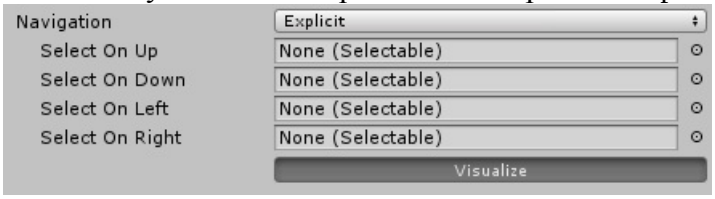
Scene Loads



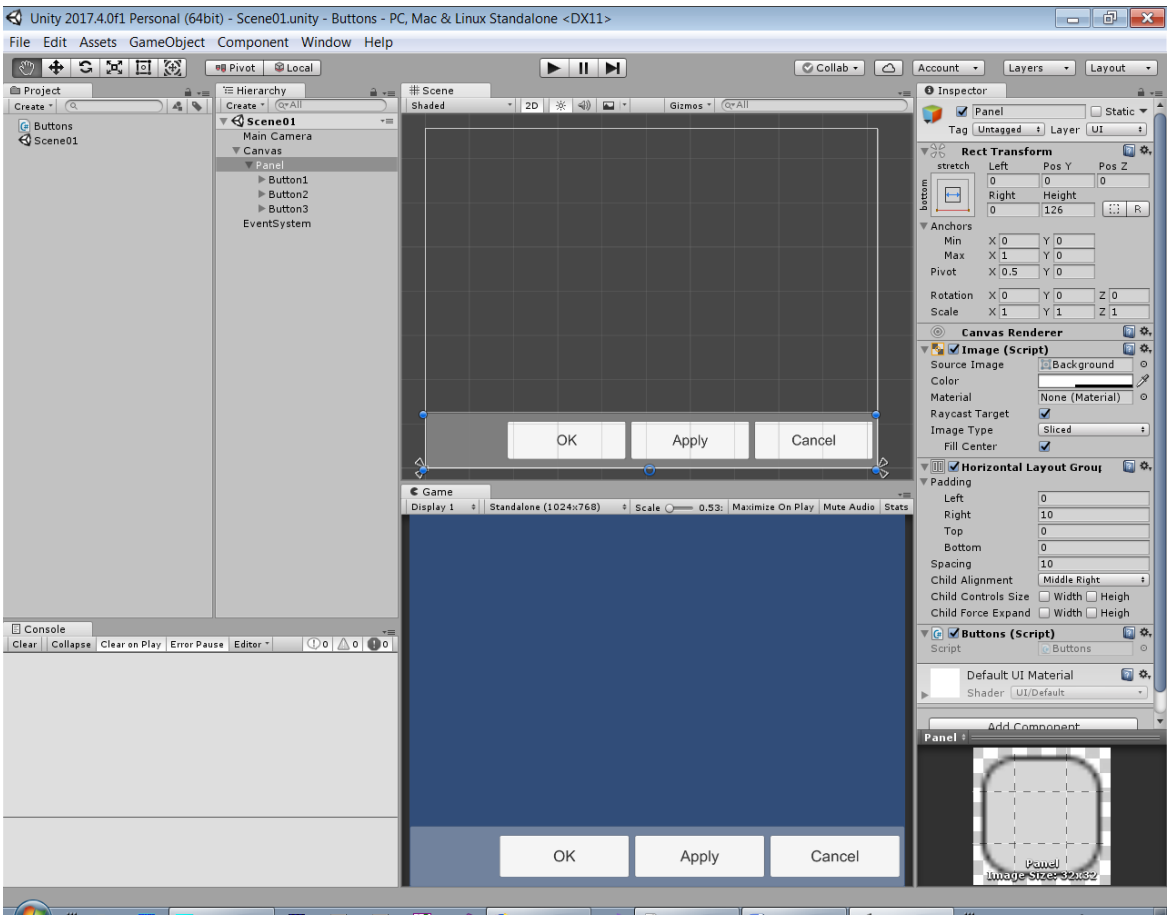
Press right-arrow key on keyboard

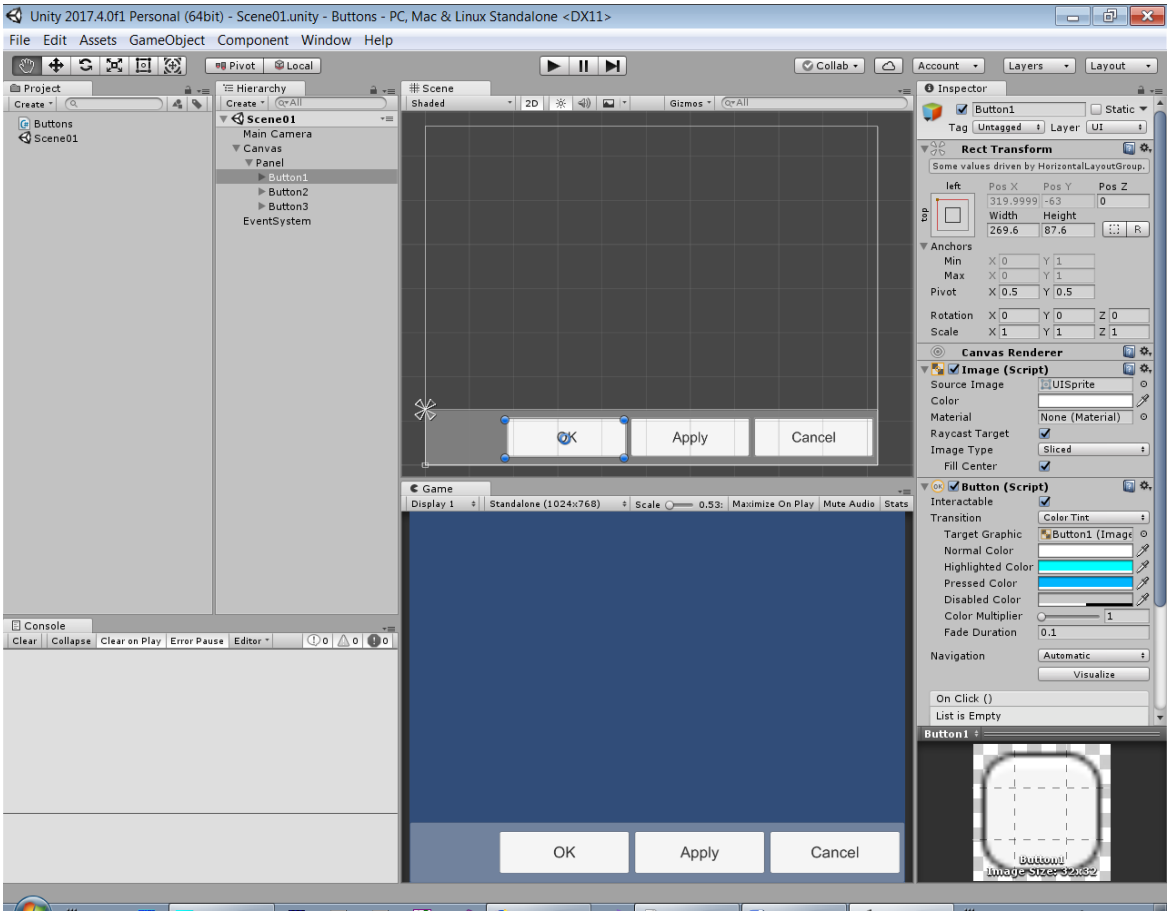


Наибольшую гибкость предоставляет режим Explicit.



Пример реализации программного интерфейса, аналогичному интерфейсу кнопок диалогового окна





Для кнопок установлен режим цветового выделения. Все действия по связыванию кнопок с обработчиками событий, а также по настройке компонента EventSystem, выполняются программно, с помощью скрипта Buttons, подключенного к родительской панели.

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
using UnityEngine.UI;
using UnityEngine.EventSystems;

public class Buttons : MonoBehaviour
{
    EventSystem es;

    void Start()
    {
        var b1 = GameObject.Find("Button1");
        b1.GetComponent<Button>().onClick.AddListener(button1Click);
        GameObject.Find("Button2").GetComponent<Button>()
            .onClick.AddListener(button2Click);
        GameObject.Find("Button3").GetComponent<Button>()
            .onClick.AddListener(button3Click);
        es = GameObject.Find("EventSystem").GetComponent<EventSystem>();
        es.firstSelectedGameObject = b1;
        es.sendNavigationEvents = true;
    }

    void Update()
    {
        if (Input.GetButtonDown("Submit") && es.currentSelectedGameObject == null)
            button1Click();
        if (Input.GetButtonDown("Cancel"))
            button3Click();
    }

    public void button1Click()
    {
        print("OK");
    }
    public void button2Click()
    {
        print("Apply");
    }
    public void button3Click()
    {
        print("Cancel");
    }
}
```

Для нажатия на выделенную кнопку можно использовать клавиши Enter или пробел. Для выполнения действия, связанного с кнопкой Cancel, даже если она не выделена, можно использовать клавишу Esc.

Событие Submit, связанное с клавишами Enter и пробел, активируется только если ни одна из кнопок не выделена; в этом случае нажатие Enter или пробела означает выполнение действия, связанного с кнопкой OK.

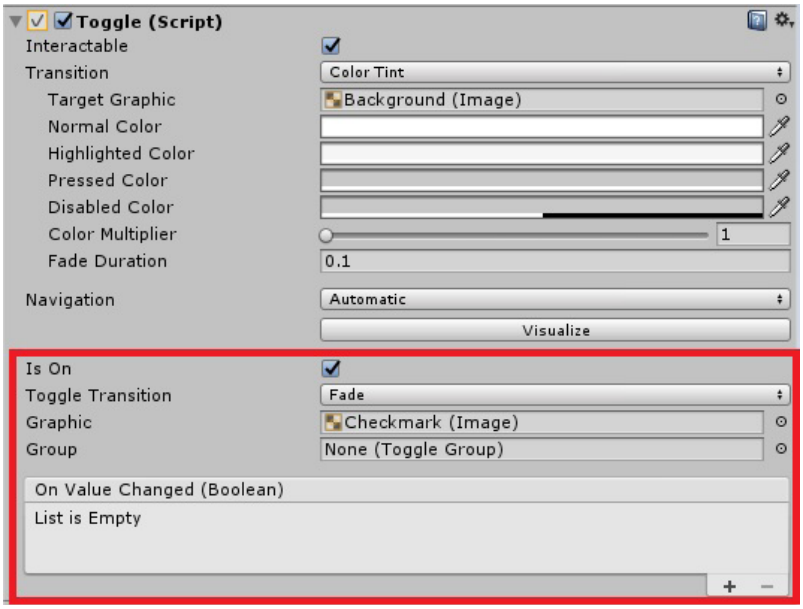
Примечание. С помощью класса EventSystem можно программно выбирать игровые объекты (метод SetSelectedGameObject(gameObj) и определять текущий выбранный игровой объект (currentSelectedGameObject).

Объект Toggle

Create | UI | Toggle

Содержит дочерние объекты Background и Label, объект Background, в свою очередь, содержит объект Checkmark. Объекты Background и Checkmark определяют изображения, а Label — текст объекта Toggle.

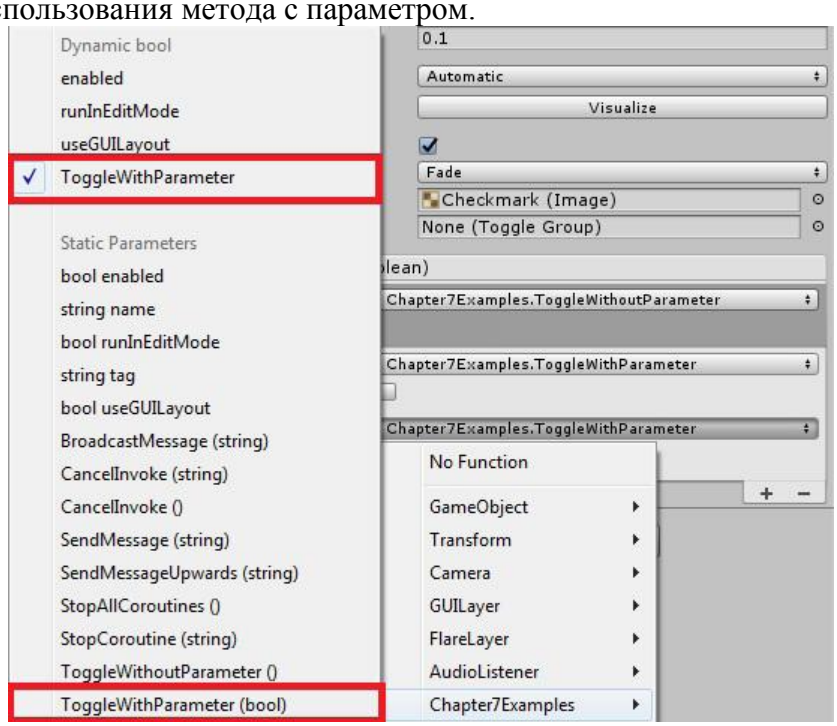
Компонент Toggle

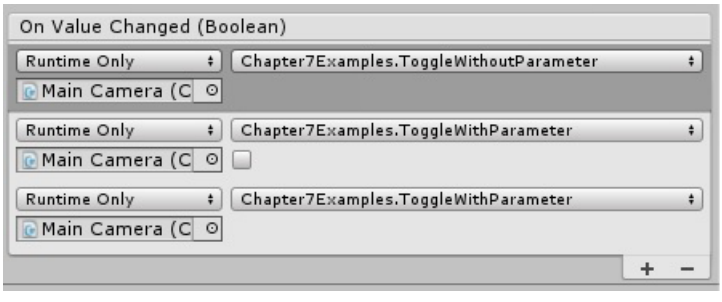


Свойство **Group** позволяет объединять объекты Toggle в группы с возможностью выбора не более одного варианта в группе.

Событие **OnValueChanged** принимает один аргумент типа bool (или не имеет аргументов) и возникает при изменении состояния объекта Toggle.

При настройке этого события из инспектора можно выбрать метод без параметра и два варианта использования метода с параметром.





7

Второй вариант (с явно указанным флажком) соответствует статически заданному параметру.

В динамически заданном параметре возвращается текущее состояние объекта Toggle.

```
public void ToggleWithoutParameter(){
Debug.Log("changed");
}
public void ToggleWithParameter(bool value){
Debug.Log(value);
}
```

Результаты вывода, если Toggle переведен в отключенное состояние:

```
changed
False
False
```

Результаты вывода, если Toggle переведен во включенное состояние:

```
changed
False
True
```

В третьем случае возвращается значение свойства isOn.

При программном подключении обработчика к событию параметр передается динамически.

Компонент Toggle Group

Add Component | UI | Toggle Group

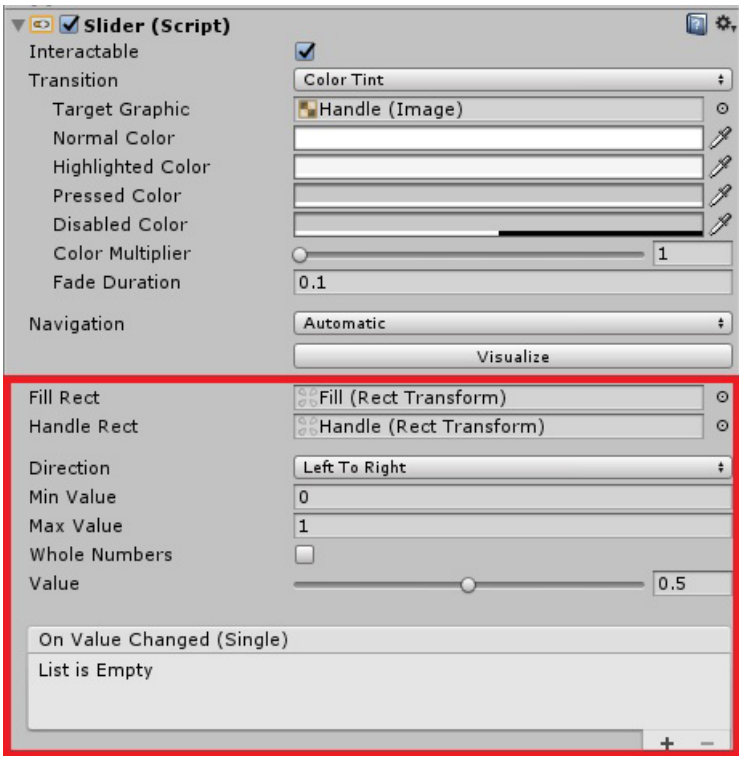
Позволяет объединять объекты Toggle в группы. Обычно связывается с панелью, содержащей требуемые объекты Toggle.

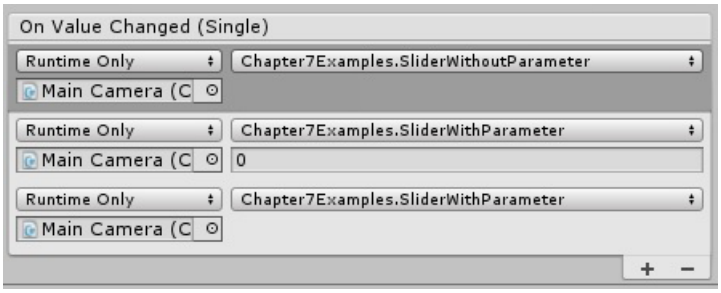
Объект Slider

Create | UI | Slider

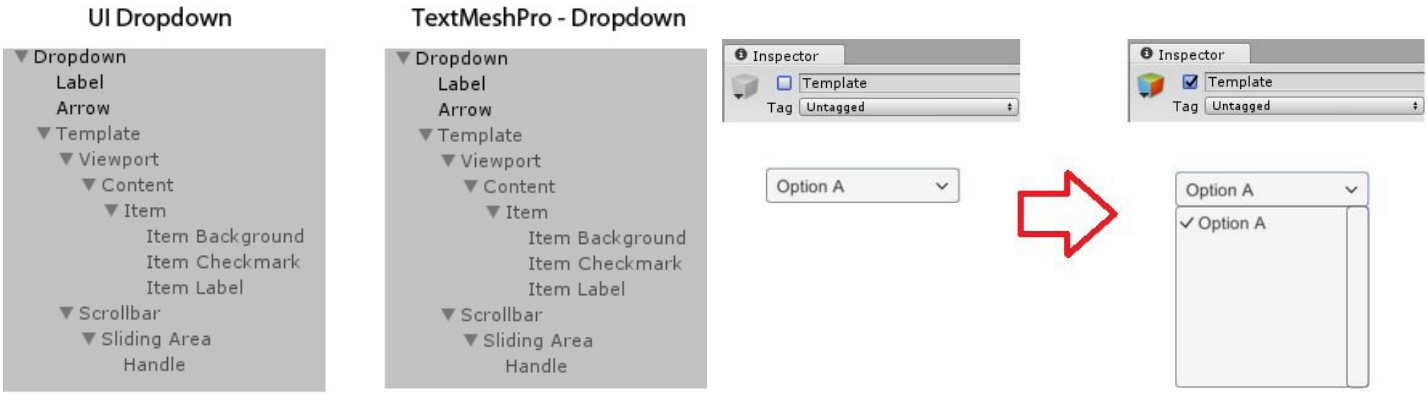
Содержит три дочерних объекта: Background, Fill Area, Handle Slide Area, объект Fill Area, в свою очередь, содержит объект Fill, а объект Handle Slide Area содержит объект HandleCheckmark. Все дочерние объекты определяют фрагменты изображения объекта Slider и их относительное позиционирование.

Компонент Slider

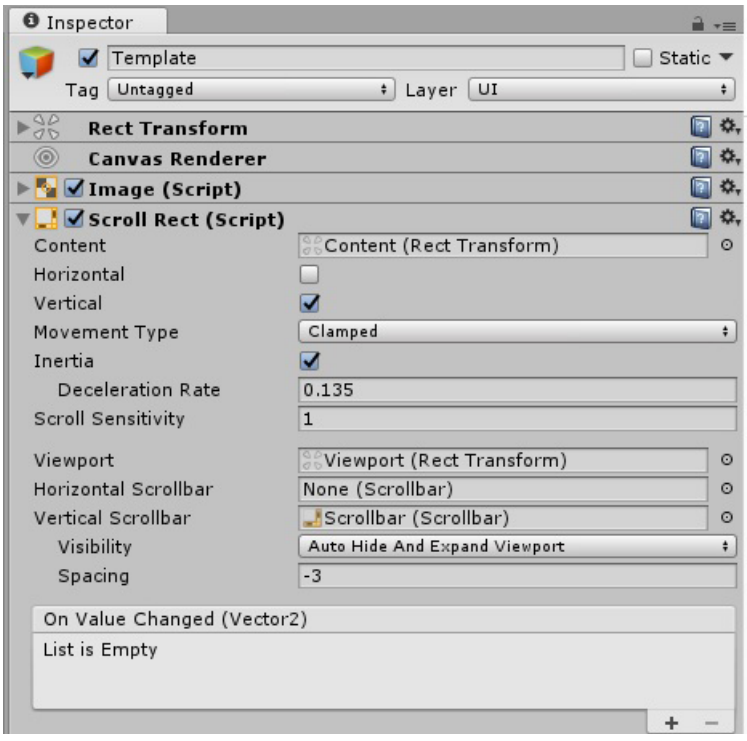




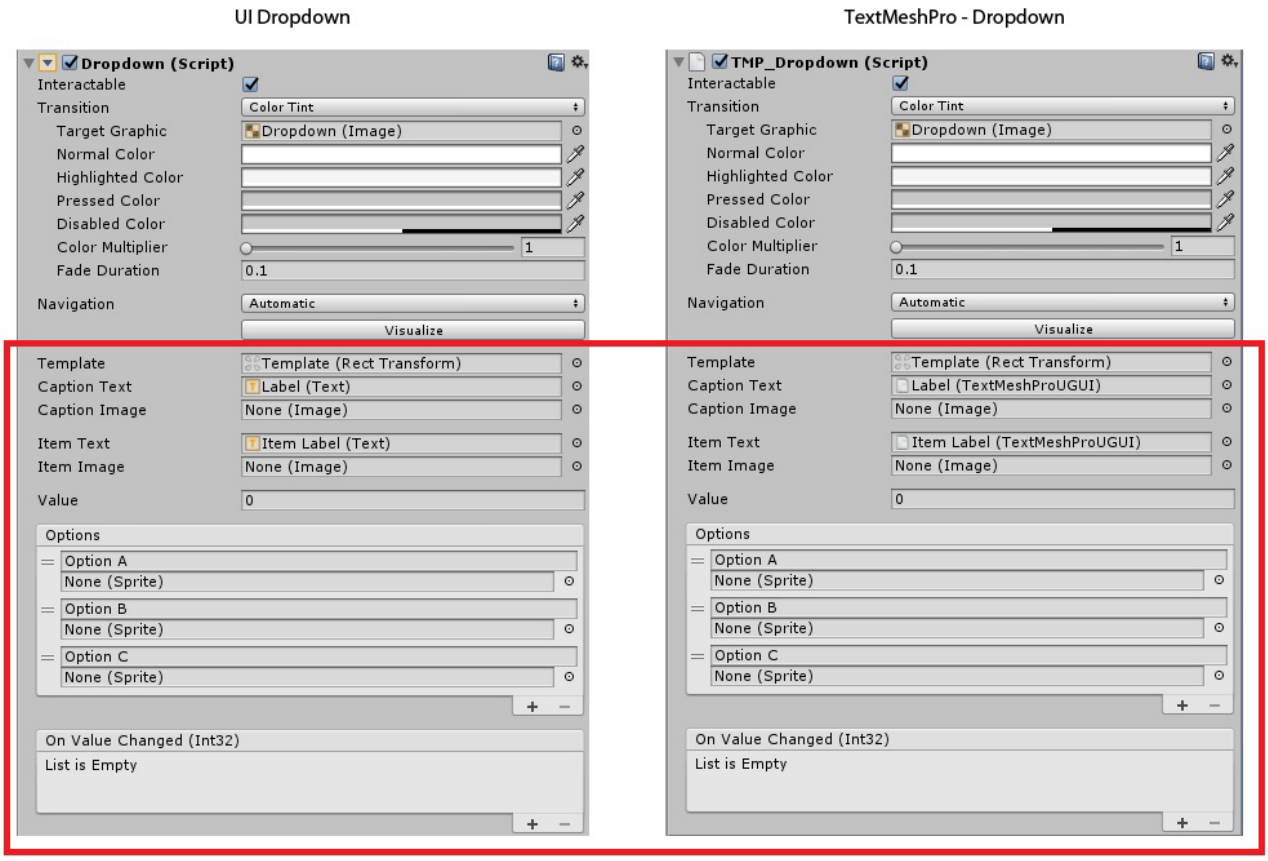
Объект Dropdown



При включении объекта Template объект Dropdown отображается в развернутом виде, однако при запуске программы он всегда автоматически сворачивается.



Компонент Dropdown

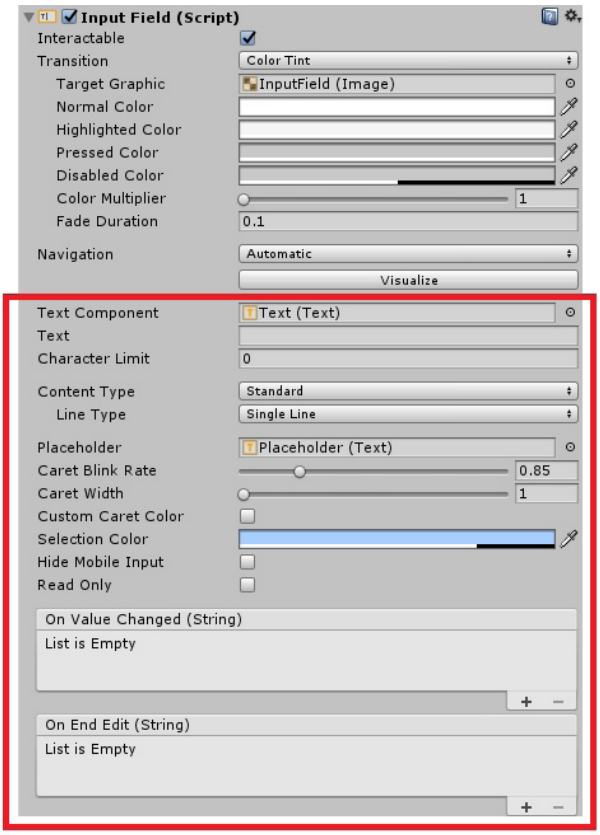


Свойство **Caption** отображает выбранный пункт при свернутом компоненте.

Объект Input Field

Содержит три дочерних объекта: InputField Input Caret, Placeholder и Text. Объект Placeholder позволяет указать текст, который будет отображаться в поле ввода до того, как пользователь введет свой вариант.

Компонент Input Field



Именно свойство **Text** содержит введенный текст, тогда как содержимое объекта Text соответствует представлению этого текста на экране (например, в случае пароля на экране отображаются звездочки).

Значение **Character Limit**, равное 0, снимает ограничение на длину вводимого текста.

Свойство **Content Type** позволяет задать разнообразные ограничения на вводимый текст. Среди вариантов этого свойства имеются Standard, Autocorrected (для устройств с вводом, поддерживающим автокоррекцию), Integer Number (целые числа со знаком), Decimal Number (вещественные числа со знаком и десятичной точкой), Alphanumeric (буквы и цифры), Name (каждое слово начинается с заглавной буквы), Email Address, Password, Pin (положительные целые числа) и Custom.

Свойство **Line Type** доступно для режимов Standard, Autocorrect, and Custom и может принимать три значения: Single Line, Multi Line Submit (нажатие Enter означает конец ввода — как и для режима Single Line, но текст может переноситься на новые экранные строки) и Multi Line New Line (нажатие Enter означает переход на новую строку).

События, связанные с компонентом Input Field