Разработка комплекса учебных проектов по созданию приложений с графическим интерфейсом на платформе Unity.

**Введение**

Программный интерфейс не только решает проблему взаимодействия с приложением, но и делает это взаимодействие максимально комфортным. Интерфейс необходим для удобного взаимодействия пользователя с приложением.

Поэтапная разработка интерфейса приложений и сложных систем экономит время и структурирует работу. Очень важно уметь разработать интерфейс, в котором пользователь найдет ключевые функции программы за минимальное время. Время, затрачиваемое на разработку, и количество поддерживаемых платформ также является важным фактором.

Кроссплатформенная разработка позволяет создавать приложения сразу для нескольких платформ. Кроссплатформенное приложение требует гораздо меньше ресурсов, позволяет тратить меньше времени на разработку. Unity – это мультиплатформенный инструмент для разработки 2D и 3D приложений. Основными преимуществами Unity являются наличие визуальной среды разработки, межплатформенной поддержки и модульной системы компонентов. Созданные с помощью Unity приложения работают под операционными системами Windows, Linux, OS X, Android, IOS.

Современные приложения часто нуждаются в поддержке широкого спектра различных разрешений экрана, и особенно в данной возможности нуждаются их интерфейсы. Адаптивный дизайн является необходимым элементом кроссплатформенного приложения. Причиной этому служит большое количество различных устройств и платформ, которые отличаются разными разрешениями. Адаптивная версия должна работать на всех платформах с любым разрешением. Это позволяет потратить меньше усилий для реализации программы под смартфоны, планшеты, ноутбуки, ПК.

Система создания интерфейсов в Unity снабжена рядом различных инструментов для адаптивного дизайна, которые также можно комбинировать между собой массой различных способов.

Таким образом, платформа Unity является отличным инструментом для кроссплатформенной разработки приложений с пользовательским интерфейсом под широкий набор разрешений экранов на различных устройствах.

**1 Постановка задачи**

Задачей данной работы является разработка комплекса учебных проектов по созданию кроссплатформенных графических приложений с адаптивных дизайном. Демонстрация работы приложений в мобильных ОС на примере Android и настольных ОС на примере Windows.

Данная работа включает в себя:

1. Установка платформы Unity и дополнений для сборки под ОС Android.
2. Знакомство с редактором и основными особенностями разработки пользовательских интерфейсов на Unity.

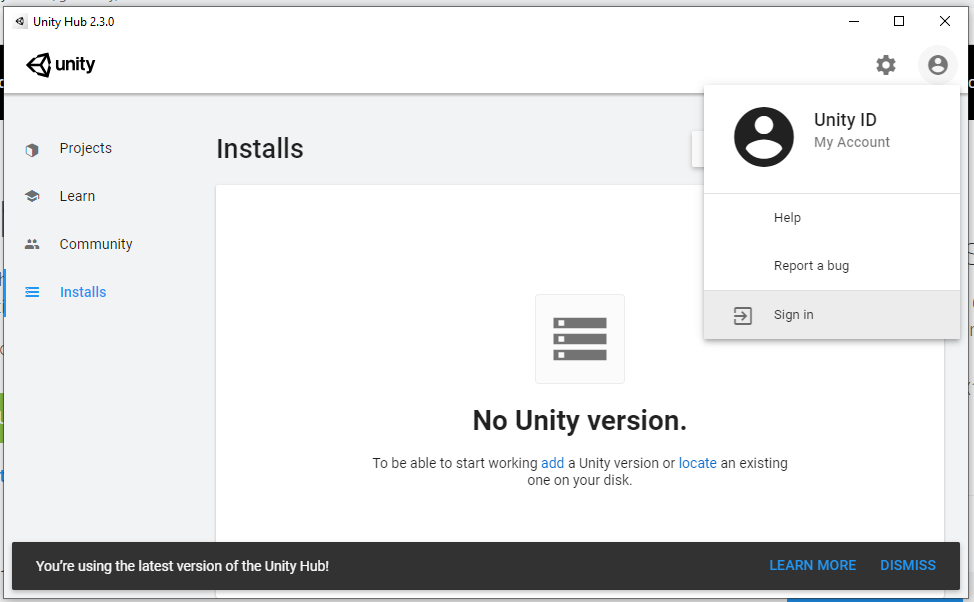
**2 Особенности разработки приложений на платформе Unity**

**2.1 Отличие разработки приложений на Unity от нативной разработки приложения**

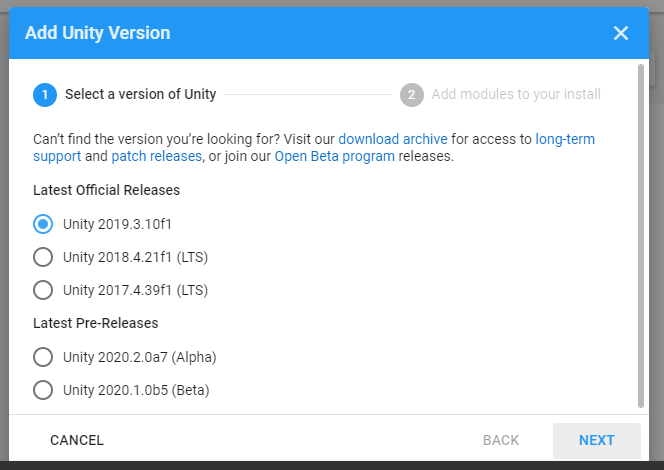
**2.2 Установка и настройка среды разработки**

Для начала необходимо установить Unity Hub по ссылке <https://unity3d.com/get-unity/download>. Он позволяет управлять проектами, работать с различными версиями Unity и устанавливать дополнения для сборки под различные платформы.

Для работы с редактором необходимо создать Unity ID.



Далее на вкладке Installs нажать кнопку Add и установить последнюю стабильную версию Unity.

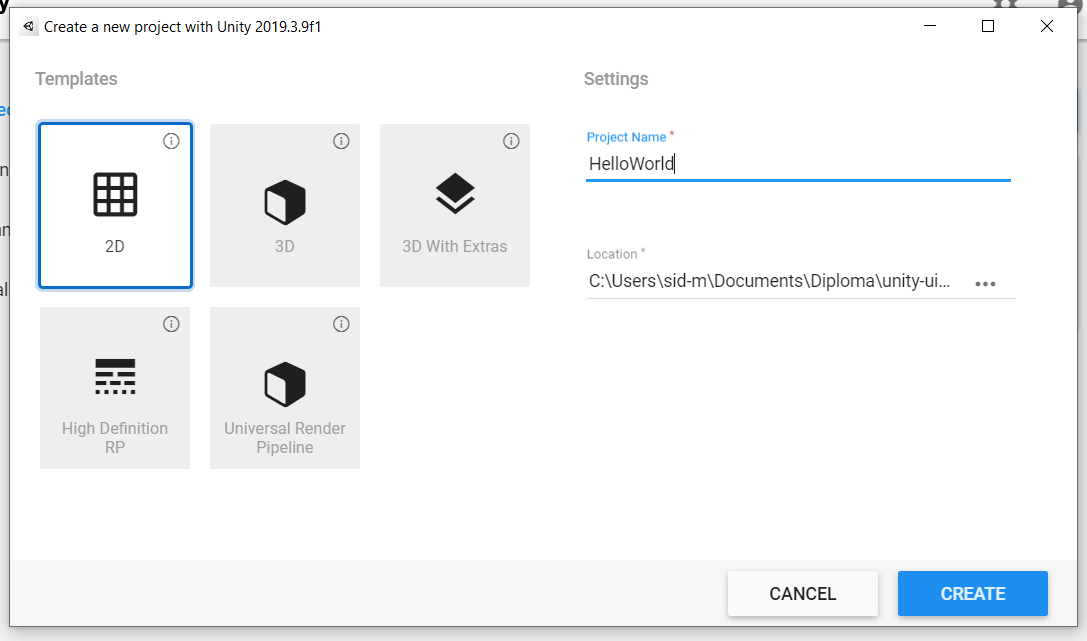


На следующем шаге поставить галочку напротив опции Android Build Support.

**3. Обзор среды разработки. Создание проекта**

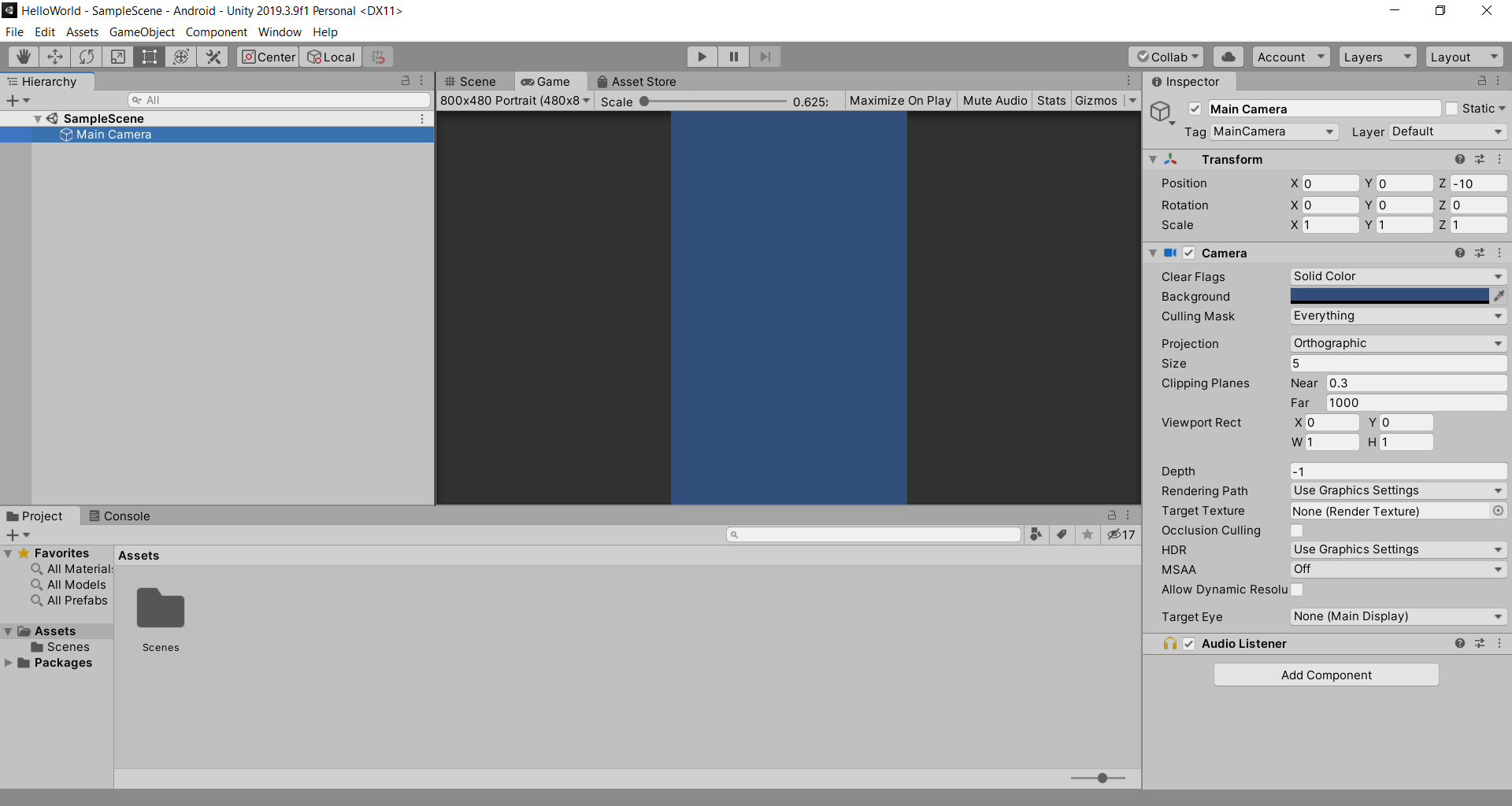
**3.1 Создание проекта**

Зайти в Unity Hubs, во вкладке projects нажать на кнопку New. Далее выбрать тип проекта 2D, указать название проекта и выбрать место его расположения.

****

В первую очередь в верхнем меню File, далее Build Settings, изменим текущую платформу на Android. Этот шаг был выполнен с целью добавления большего количества разрешений экранов в предпросмотре редактора.

Рассмотрим основные области редактора Unity. По центру располагается вкладка Game. Здесь есть выпадающий список для изменения разрешения. Вверху кнопка запуска симуляции работы приложения в редакторе. Также здесь есть вкладка Scene, в которой можно посмотреть отображение UI элементов. Для перемещения по сцене необходимо выбирать инструмент Hand Tool в левом верхнем углу. Для приближения и отдаления можно использовать колесико мыши. Снизу располагается вкладки консоль, куда можно выводить отладочную информацию. А также вкладка Project со структурой проекта. В папке Assets есть папка Scenes, где хранится автоматически созданная сцена. В левой области располагается вкладка Hierarchy, где располагаются сцены и их объекты. Изначально у сцены есть только объект камеры. Во вкладке Inspector, расположенной справа осуществляется редактировние свойств объектов.

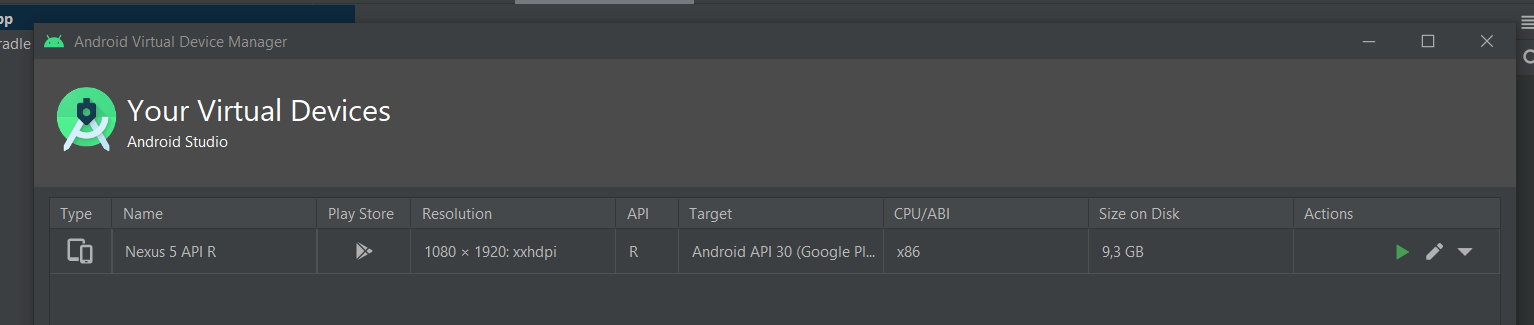


Изменим Background камеры на значение CCCCCC. Для начала на сцене нужно создать элемент Canvas, так как все UI элементы должны быть дочерними к элементу Canvas. Для этого нужно во вкладке Hierarchy нажать правой кнопкой по области под Main Camera и выбрать UI, далее Canvas. Также автоматически будет создан объект EventSystem, отвечающий за обработку пользовательского ввода.

Render Mode для Canvas стоит Screen Space – Overlay. Элементы отображаются поверх сцены, а если меняется размер экрана или его разрешение, то Canvas автоматически примет нужный размер вместе с ним. У Canvas в Inspector установим Pixel Perfect чекбокс. Включение Pixel Perfect может сделать элементы более четкими и предотвратить размытость. Следующим шагом будет добавление элемента текста внутрь Canvas. Для этого во вкладке Hierarchy нажать правой кнопкой по Canvas > UI > Text. Выбрать элемент Text во вкладке Hierarchy, далее во вкладке Inspector отредактировать значение Text на Hello World, установить параметр Alignment по центру и установить чек бокс Best Fit для автоматической подстройки размера шрифта. Элемент текст располагается в центре элемента Canvas так как Rect Transforms > Anchor Presets установлены значения middle center. Anchor Presets задает расположение элемента относительно своего родителя отдельно по горизонтали и вертикали. Также Anchor Presets позволяет растянуть элемент на всю ширину и высоту родительского элемента.

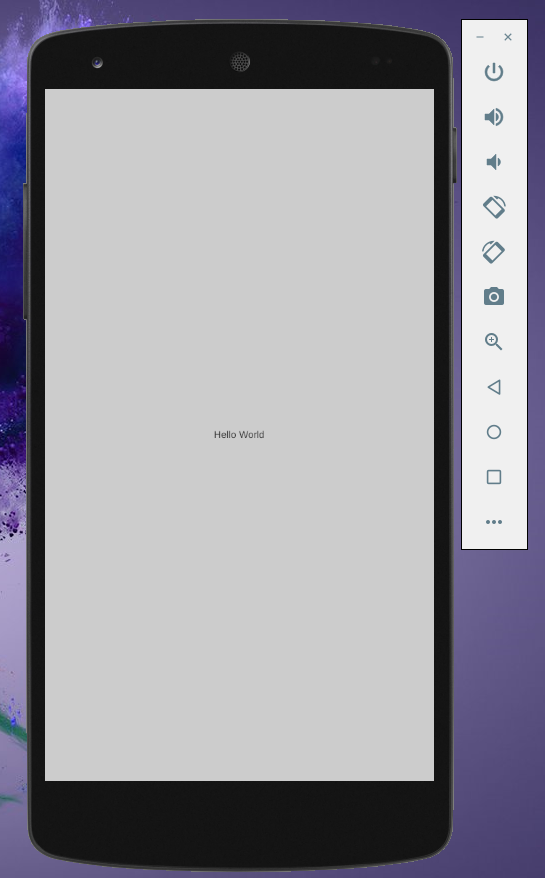
Далее запустим симуляцию нажав на кнопку Play и посмотрим, как будет выглядеть текст при различных разрешения экрана.

Осуществим сборку и запуск приложения на Android эмуляторе или настоящем Android устройстве. Если у вас установлена Android Studio, в ней можно создать и запустить эмулятор в AVD Manager.



Также можно подключить Android устройство с включенным режимом разработчика и отладки по USB.

Для сборки и запуска нажать File > Build And Run, указать название .apk. Обратите внимание, что если в пути к проекту присутствует кириллица во время сборки возникнет ошибка.



Далее поменяем тип сборки File > Build Settings изменим платформу на PC, Mac & Linux Standalone. Target Platform выбираем систему, в которой вы работаете. Галочку Create Visual Studio Solution можно снять. Открыть Player Settings. Вкладка Settings For PC > Resolution And Presentation. Fullscreen Mode выбрать Windowed, Default Screen Width 800, Default Screen Height 600, Resizable Window выбрать чек бокс. Закрыть настройки, нажать Build And Run.

Если вы, например, работаете под Windows и у вас появится необходимость собрать приложение под Mac или Linux. Для этого в Unity Hub > Install > три точки по версии Unity > Add Modules. Выбрать опции Linux Build Support, Mac Build Support и кнопка Done. Дождаться окончания установки. Изменить Target Platform в Build Settings.

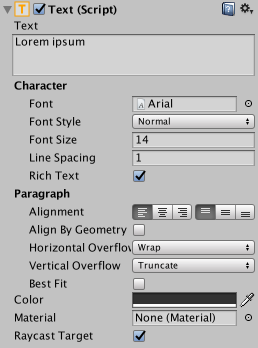
**3.2 Начальные сведения для начала разработки**

**3.2.1 Элементы пользовательского интерфейса.**

Canvas – это область, внутри которой находятся все элементы UI. Когда создаётся новый элемент UI вместе с ним автоматически создается и Canvas, если до этого на сцене его еще не было. Canvas использует объект EventSystem, чтобы использовать систему обмена сообщениями и обрабатывать события пользовательского ввода. Элементы UI на Canvas появляются в том же порядке, в каком они расположены в иерархии. Первый дочерний элемент отрисовывается первым, второй – за ним и так далее. Если два элемента UI накладываются друг на друга, добавленный позднее будет поверх того, что был добавлен ранее. Чтобы изменить то, какой элемент будет находиться поверх остальных, просто поменяйте местами элементы в иерархии путем перетаскивания. Порядком также можно управлять при помощи кода, используя следующие методы компонента Transform: SetAsFirstSibling, SetAsLastSibling и SetSiblingIndex.

Рассмотрим основные компоненты UI, которые призваны упростить создание и использование характерных функции GUI:

## Text



Компонент **Text**, также известный как Label, имеет область для ввода текста, который будет отображен. Есть возможность задать шрифт, его стиль, размер и виды отображения.

## Button



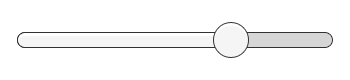
Кнопка предназначена для запуска действия, когда пользователь щелкает и отпускает её. Кнопка имеет одно событие OnClick, которое реагирует, когда пользователь завершает щелчок.

**Toggle**



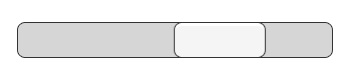
Элемент управления Toggle позволяет пользователю включать или отключать опцию. Также можно объединить несколько переключателей в ToggleGroup в случаях, когда одновременно должен быть включен только один из набора параметров. Toggle имеет одно событие под названием *OnValueChanged*.

## Slider



Slider имеет Value - десятичное число, которое пользователь может устанавливать между минимальным и максимальным значением. При изменении значения вызывается событие *OnValueChanged*.

**Scrollbar**



Полосы прокрутки используются вместе с другими компонентами для создания эффекта прокрутки. Полоса прокрутки имеет Size – диапазон от 0 до 1, оно определяет значение прокрутки. При изменении значения вызывается событие *OnValueChanged*.

## Dropdown



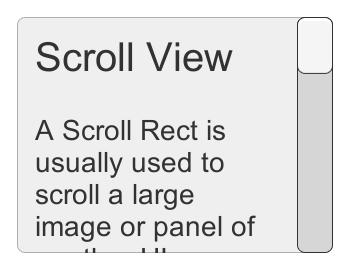
Раскрывающийся список позволяет выбрать одну опцию из списка опций. При изменении значения вызывается событие *OnValueChanged*.

**Input Field**



Поле ввода используется, чтобы сделать текст элемента Text редактируемым пользователем.

**Scroll Rect**



Scroll Rect может использоваться, когда контент, занимающий много места, должен отображаться в небольшой области.

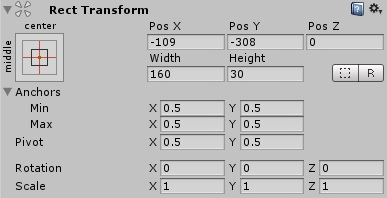
**Image**

Источник может быть добавлен к Image в поле Source Image, а его цвет задан в поле Color. Также к компоненту Image может быть добавлен материал.

## 3.2.2 Расположение элементов пользовательского интерфейса

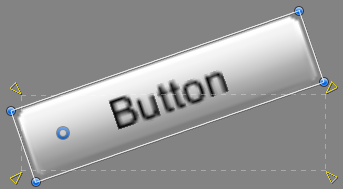
## Rect Transform

Rect Transform – это компонент преобразования, который используется для всех элементов пользовательского интерфейса.



Rect Transform имеет значения положения, вращения и масштаба, а также ширину и высоту, используемые для указания размеров области.

### Pivot



Pivot - точка, относительно которой происходят преобразования при повороте, изменении размера и масштабировании.

**Anchors**

Anchors отображаются в виде четырех маленьких треугольных ручек в окне просмотра сцены. Они могут быть привязаны к любой точке внутри родительского прямоугольника. Закрепление позволяет ребенку растягиваться вместе с шириной или высотой родителя. Каждый угол прямоугольника имеет фиксированное смещение относительно соответствующей привязки. Верхний левый угол прямоугольника имеет фиксированное смещение относительно верхнего левого якоря и т. д. Таким образом, различные углы прямоугольника могут быть привязаны к различным точкам в родительский прямоугольник.

Нажатие на Anchor Presets вызывает выпадающий список. Отсюда вы можете быстро выбрать один из самых распространенных вариантов привязки. Вы можете привязать элемент пользовательского интерфейса к сторонам или середине родительского элемента или растянуть вместе с родительским размером. Горизонтальное и вертикальное закрепление является независимым.



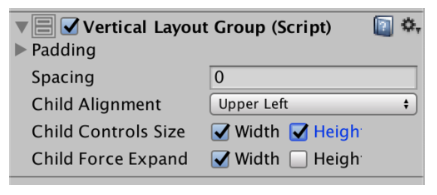
Система автоматической разметки предоставляет способы размещения элементов во вложенных группах разметки, таких как горизонтальные группы, вертикальные группы или сетки. Это также позволяет автоматически выбирать размеры элементов в соответствии с содержимым. Например, кнопка может быть динамически изменена, чтобы точно соответствовать ее текстовому содержимому плюс некоторый отступ.

# **Auto Layout**

Система автоматической компоновки на основе концепции элементов макета и контроллеры макета. Элементом макета является игровой объект с трансформацией Rect, а также, при необходимости, другие компоненты. Элемент макета имеет определенные знания о том, какой размер он должен иметь. Элементы макета непосредственно не устанавливают свой собственный размер, но другие компоненты, которые функционируют как контроллеры макета, могут использовать информацию, которую они предоставляют, чтобы вычислить размер, который будет использоваться для них.

Элемент макета имеет свойства, которые определяют его собственные: Minimum width, Minimum height, Preferred width, Preferred height, Flexible width, Flexible height.

Примерами компонентов контроллера макета, которые используют информацию, предоставляемую элементами макета, являются Content Size Fitter и различные компоненты Layout Group.



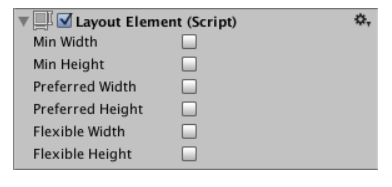
Основные принципы определения размера элементов макета в группе макетов следующие:

Первые minimum sizes выделяются.

Если свободного места достаточно, выделяются preferred sizes.

Если есть дополнительное доступное пространство, выделяется flexible size.

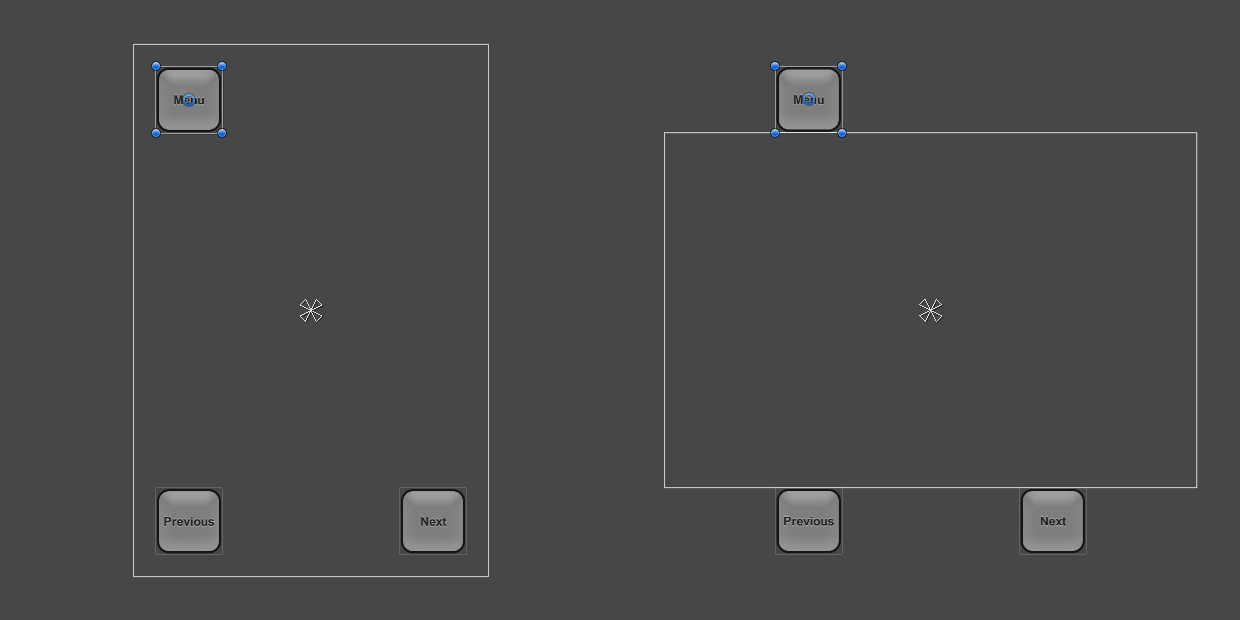
Если вы хотите переопределить minimum, preferred или flexible size у Layout Element, вы можете сделать это, добавив компонент Layout Element к Game Object.



**3.2.3 Адаптивный дизайн**

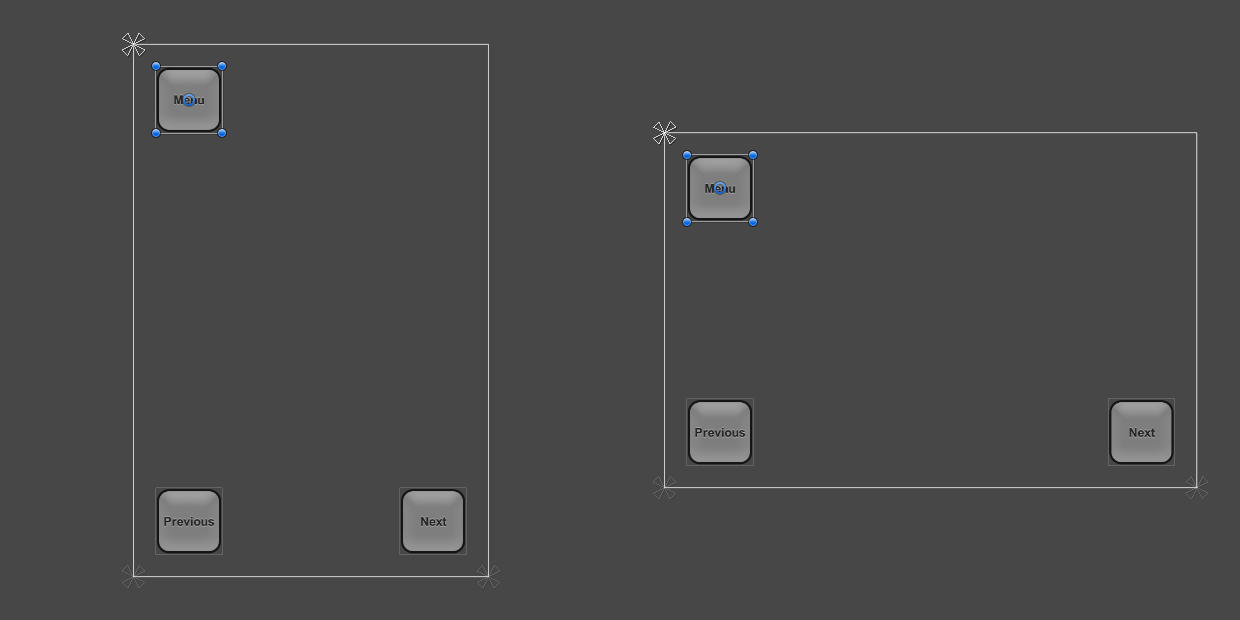
Элементы интерфейса по умолчанию привязаны к центру родительского прямоугольника. Это означает, что они сохраняют постоянное смещение относительно центра.

Если с данной настройкой разрешение было изменено на альбомное соотношение сторон, кнопки могут выпасть из своих прямоугольных областей, где они изначально должны располагаться.

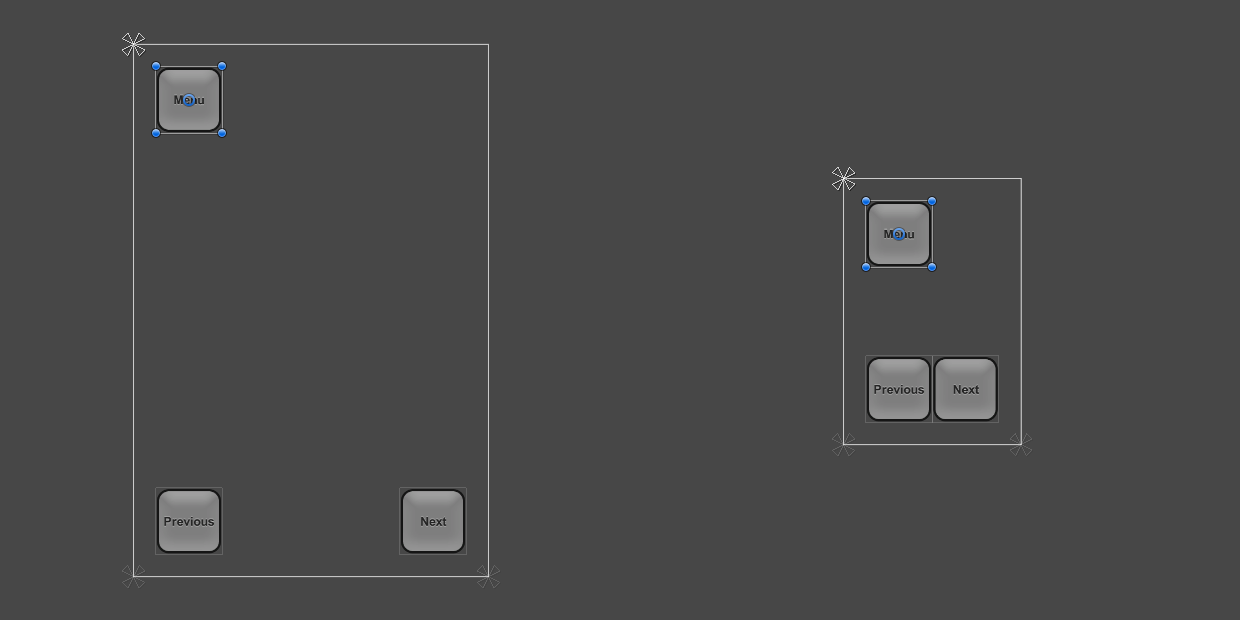


Одним из способов сохранить расположение кнопок в области экрана является изменение компоновки таким образом, чтобы места их расположения были связаны с соответствующими углами на экране. Привязка левой верхней кнопки может быть установлена в левом верхнем углу при использовании в инспекторе выпадающего списка Anchors Preset (наборы привязок), или путём перетаскивания треугольных ручек привязок в видовом окне сцены (Scene View). Аналогично привязки для левой нижней и правой нижней кнопок могут быть выставлены в левый нижний и правый нижний угол соответственно.

После привязки кнопок к своим углам, при дальнейших изменениях разрешений экрана и соотношений сторон будут сохраняться позиции кнопок относительно углов.



Когда разрешение экрана изменяется на большее или меньшее относительно текущего, кнопки должны по-прежнему сохранять своё изначальное расположение относительно углов, к которым они привязаны. Однако, сохраняя своё оригинальное разрешение, заданное в пикселях, они могут становиться как больше, так и меньше, соответствуя пропорциям текущего разрешения экрана. Кнопки должны становиться меньше настолько же, насколько в процентном соотношении становиться меньше сам экран.



Для реализации масштабирования компонент Canvas Scaler может быть добавлен в корень Canvas. В компоненте Canvas Scaler можно установить его UI Scale Mode в Scale With Screen Size. В данном режиме масштабирования можно определить какое разрешение использовать в качестве базового. Если текущее разрешение больше или меньше базового, фактор масштабирования компонента Canvas устанавливается соответственно так, чтобы все элементы интерфейса масштабировались в большую или меньшую сторону вместе с разрешением экрана.

После данной настройки компоновка должна будет масштабироваться пропорционально размерам экрана. При этом уменьшаться размеры кнопок, их расстояние от краёв экрана, графическая составляющая и текстовые элементы. Это означает, что компоновка, которая была в портретном разрешении отобразиться на экране также как и при альбомном разрешении, лишь с разницей в плотности пикселей.

