UI и компоненты

Canvas, scaler, UI элементы, резиновый интерфейс, бонус

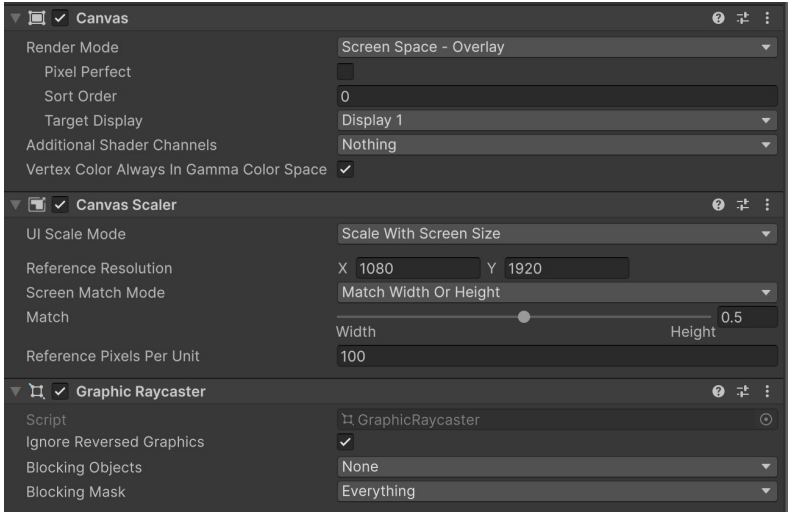
Canvas

Корневой объект для всех UI-элементов

Screen Space - Overlay — UI всегда “поверх” игры (меню, HUD)

Screen Space - Camera — UI рендерится через камеру (эффекты перспективы)

World Space — UI как часть игрового мира (например, надпись над NPC)



Render mode

Screen Space - Overlay

- По умолчанию
- UI рендерится поверх всей сцены, как HUD
- Не зависит от камеры — если камеру повернуть/подвинуть, интерфейс останется на месте
- Используется для меню, HUD, жизней, очков, таймеров

Пример: полоска здоровья в шутере

Render mode

Screen Space - Camera

- UI рендерится через конкретную камеру
- Требует указания Render Camera в настройках Canvas
- Позволяет использовать эффекты камеры (постобработку, перспективу)
- Масштаб UI зависит от Distance — расстояния Canvas от камеры

Пример: интерфейс на стекле в VR или AR, где UI должен быть “в пространстве”

Демо

Render mode

World Space

- UI становится частью мира (как обычный 3D-объект)
- Можно перемещать, вращать, масштабировать UI-объекты
- Используется для вывесок, табличек, плавающих индикаторов над NPC

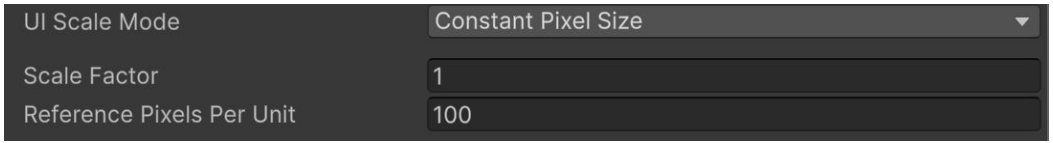
Пример: над врагом появляется полоска здоровья, которая поворачивается лицом к камере

Canvas Scaler

Масштабирование интерфейса под разные разрешения экранов.

- Constant Pixel Size
- Scale With Screen Size
- Constant Physical Size

Canvas Scaler



Элементы интерфейса всегда одного размера в **пикселях**, независимо от разрешения экрана

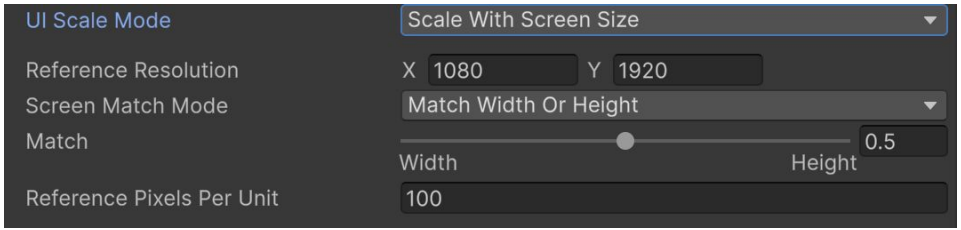
Плюсы:

- Просто и предсказуемо
- Удобно для пиксель-арта и ретро-игр (где важен точный размер спрайтов)
- Минимальная нагрузка (нет перерасчета размеров при изменении разрешения)

Минусы:

- На маленьких экранах элементы могут быть слишком мелкими, на больших — слишком огромными
- Совсем не адаптивный вариант → плохо подходит для кроссплатформенных игр

Canvas Scaler



- Элементы интерфейса масштабируются относительно Reference Resolution
- Параметр Match (Width/Height) позволяет выбирать, что приоритетнее: ширина, высота или баланс.

Плюсы:

- Интерфейс масштабируется под любое разрешение
- Подходит для большинства проектов (мобильные, ПК, консоли)
- Легко контролировать пропорции через Reference Resolution
- Можно добиться одинакового внешнего вида на экранах 16:9, 18:9, 4:3 и т.д.

Минусы:

- При слишком вытянутых экранах возможны искажения (например, элементы могут уползти за границы)
- Нужно правильно настроить Anchors и Match — иначе результат будет неожиданным

Graphic Raycaster

- Отвечает за обработку кликов/тапов по UI-элементам
- Работает вместе с EventSystem
- Определяет, какой элемент получил событие (например, при клике по кнопке)

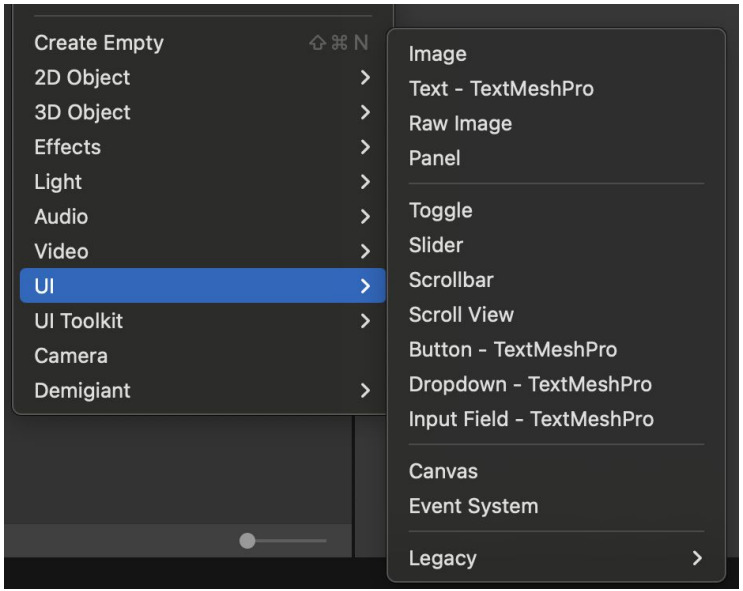
Ignore Reversed Graphics

Если включено, UI-элементы, развернутые спиной к камере, не будут реагировать на клики

Blocking Objects

Можно указать, будут ли UI-элементы блокироваться 3D- или 2D-объектами

UI-элементы

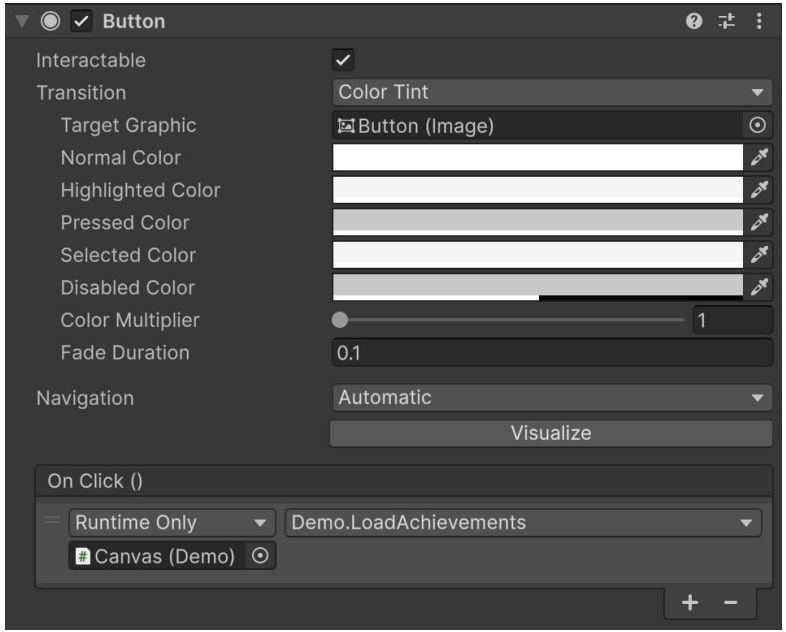


- TextMeshPro — стандарт для текста
- Image — отображение спрайтов
- Button — кликабельная кнопка
- Slider — ползунок
- Toggle — переключатель
- Dropdown — выпадающий список
- InputField — поле для ввода текста

Демо

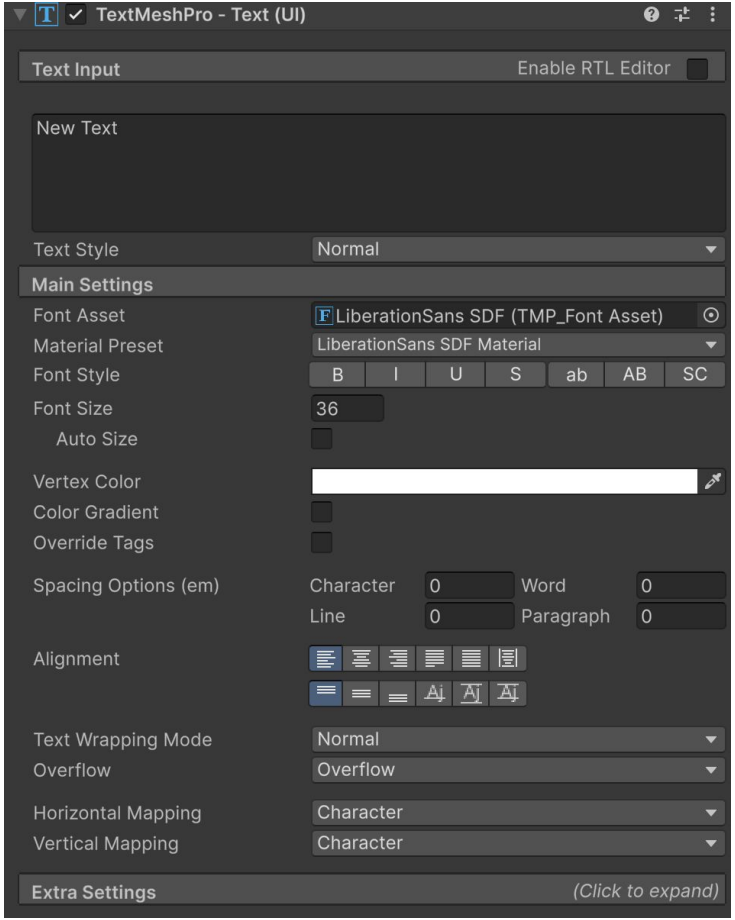
UI-элементы

Button - кнопка

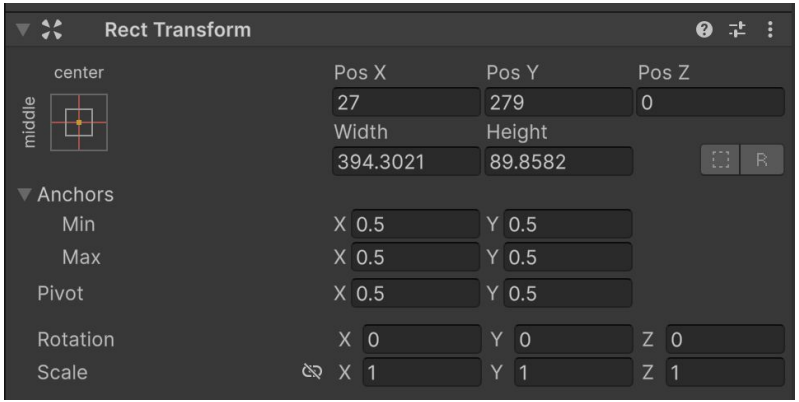


UI-элементы

TextMeshPro - текстовый компонент



RectTransform



RectTransform

- Расширенная версия Transform, используемая для UI
- Вместо позиции, поворота и масштаба в 3D он управляет положением и размерами прямоугольной области на Canvas

RectTransform

- Anchors (якоря)
- Pivot (точка вращения/масштабирования)
- Position (Local Position внутри Canvas)
- Size Delta (ширина и высота, если якоря не растянуты)
- Offsets (отступы от якорей, если растянуты)

Responsive UI

Что нужно?

- Выбрать Canvas Render mode
- Выбрать UI Scale Mode у Canvas Scaler
- Разобраться с RectTransform

Responsive UI

Anchors — это относительная привязка UI-элемента к родительскому контейнеру.

- Задаются в процентах от родителя (0 = край слева/снизу, 1 = край справа/сверху)
- Можно задавать одну точку (фиксированное положение) или прямоугольник (растяжение)

Если оба якоря совпадают:

- Элемент остаётся фиксированным в точке
- Размер задаётся через Size Delta

Разные значения min и max:

- Элемент растягивается вместе с родителем
- Вместо Size Delta появляются Offsets (Left, Right, Top, Bottom)

Пример:

- Если сделать якоря (0,0) и (1,1) — элемент займёт весь экран
- Если якоря по центру (0.5,0.5) — элемент зафиксирован по центру

Демо

Responsive UI

Pivot — это точка, относительно которой происходит вращение и масштабирование элемента.

- Измеряется от 0 до 1 (левая граница = 0, правая = 1)
- (0.5, 0.5) — центр (по умолчанию)
- (0,0) — нижний левый угол
- (1,1) — верхний правый угол

Пример:

Если Pivot = (0,0), то при изменении позиции объект будет “тянуться” от нижнего левого угла

Если Pivot = (1,1), то он будет “сидеть” в верхнем правом углу.

Демо

Responsive UI

Position

- В UI позиция элемента задается как Local Position относительно родителя
- Но реальное положение зависит от Anchors + Pivot
- В отличие от обычного Transform, тут позиция = смещение от якорей

Пример:

Если якорь стоит по центру и позиция (0,0) → элемент будет в центре

Если якорь в левом верхнем углу, а позиция (0,0) → элемент будет там

Демо

Responsive UI

Size Delta и Offsets

- Size Delta — задаёт ширину и высоту элемента, если якоря фиксированные
- Если Anchors растянуты, то вместо Size Delta работают Offsets (Left, Right, Top, Bottom)

Пример:

Anchors по центру → задаем размер кнопки через Size Delta.

Anchors растянуты от левого края до правого → можно задать отступы Left/Right, а ширина будет подстраиваться автоматически

Демо

Бонус

DOTWeen — https://dotween.demigiant.com

LeanTween — https://assetstore.unity.com/…/leantween-3595

iTween — https://assetstore.unity.com/…/itween-84

PrimeTween — https://assetstore.unity.com/…/primetween-high-performance-animations-and-sequences-252960

Бонус

```
transform.DOMove(new Vector3(2,3,4), 1);
rigidbody.DOMove(new Vector3(2,3,4), 1);
material.DOColor(Color.green, 1);
```

```
Sequence seq = DOTween.Sequence();
seq.Append(transform.DOMoveX(5,1f));
seq.Append(transform.DOScale(Vector3.one*2,0.5f));
seq.AppendInterval(0.5f);
seq.Play();
```

AudioSource, Camera, Light, LineRenderer, Material, Rigidbody, Rigidbody2D, SpriteRenderer, Transform, Canvas, Image, RectTransform, ScrollRect, Slider, etc

Бонус

Категория	Что делает	Пример кода	Особенности
Позиция	Плавное перемещение объекта	`transform.DOMove(new Vector3(5,0,0), 1f);`	Можно анимировать локальные и мировые координаты (`DOMove`, `DOLocalMove`)
Вращение	Плавное вращение	`transform.DORotate(new Vector3(0,180,0), 1f);`	Поддержка локального и глобального вращения
Масштаб	Плавное изменение размера	`transform.DOScale(new Vector3(2,2,2), 1f);`	Отлично для эффектов UI и объектов сцены
UI - позиция	Плавное движение UI	`rectTransform.DOAnchorPos(new Vector2(100,200),0.5f);`	Работает только с RectTransform
UI - прозрачность	Плавное появление/исчезновение	`canvasGroup.DOFade(1,0.5f);`	Нужно добавить CanvasGroup
Цвет	Изменение цвета спрайта/картинки	`image.DOColor(Color.red,1f);`	Работает с Image, SpriteRenderer, TextMeshPro

Бонус

Custom Tween	Анимирование любого значения	`DOTween.To(() => value, x => value = x, 100, 2f);`	Можно анимировать float, Vector, любые свойства
Sequence (цепочка)	Последовательные анимации	`csharp Sequence seq = DOTween.Sequence(); seq.Append(transform.DOMoveX(5,1f)); seq.Append(transform.DOScale(Vector3.one*2,0.5f)); seq.Play();`	Можно добавлять интервалы (`AppendInterval`) и callback
Loop / PingPong	Повторение анимации	`transform.DOMoveX(5,1f).SetLoops(-1,LoopType.Yoyo);`	-1 = бесконечно, PingPong = туда-обратно
Shake	Тряска объекта	`transform.DOShakePosition(1f,1f,10);`	Полезно для ударов, эффектов камер
Easing	Кривые плавности	`transform.DOMoveX(5,1f).SetEase(Ease.InOutBounce);`	Очень много встроенных кривых (`Linear`, `Elastic`, `Bounce`)
Callback	Действия при завершении/старте	`transform.DOMoveX(5,1f).OnComplete(()=>Debug.Log("Done"));`	OnStart, OnUpdate, OnKill, OnComplete

Что дальше

Input System vs старый Input

Камеры, проекции: перспективная, оргографическая

Звуки и музыка: AudioSource, AudioListener, менеджер звуков