**Visual Studio**

**Edit -> Preferences: External Tools -> External Script Editor**

|  |  |
| --- | --- |
|  | |
| **Инструменты трансформаций** | клавиши **Q, W, E, R, T** |
| **Сфокусироваться на объекте** | Выделить объект, нажать **F** |
| **Поворот камеры относительно выделенного объекта** | **ALT** + **ЛКМ** |
| **Поворот камеры относительно себя** | движение мыши с зажатой правой клавишей мыши  ALT + движение мыши с зажатой правой клавишей мыши – приближение\удаление |
| **Hand Tool (перемещение вида)** | инструмент рука с возвратом на предыдущий инструмент: зажатое колесо мыши |
| **Привязка к вершинам** | Клавиша V |
| **Перемещение на фиксированный шаг** | CTRL + инструмент трансформации |
| **Дублирование объекта** | Ctrl+D |
| **Подстройка камеры под вид сцены** | выбрать камеру: Game Object -> Align With View |
| **Удаление объекта из середины листа или массива.** | Когда в инспекторе целый массив забит объектами или префабами, удалить из его середины можно нажав SHIFT + DEL |
| **Показать все подобъекты в иерархии объекта** | ALT + щелчок для открытия иерархии объекта |
| **Сохранить или подгрузить наборы выделенных объектов** | Edit -> Selection -> Save Selection,  Edit -> Selection -> Load Selection |

|  |  |
| --- | --- |
| **Unity Input System** | |
| [Урок](https://www.youtube.com/watch?v=Yjee_e4fICc). | |
| Установка | **Input System** устанавливается вручную через **Package Manager**.  Анализ устройств ввода **Window -> Analysis – Input Debug**  **Edit -> Project Settings: Player -> Other Settings: Configurations Active Input Handing** (выбираем) |
| **ПКМ -> Create -> Input Actions**  Есть несколько способов подключить систему.  - Добавить компонент **Player Input** и дальше через события Unity  - Добавить компонент **Player Input** и дальше через события C#  - Сгенерировать класс C# и делать все через скрипты  - Воспользоваться шаблоном | |

|  |
| --- |
| **UI Builder (UI toolkit)** |
| [Введение](https://www.youtube.com/watch?v=NQYHIH0BJbs&t=54s).  [Примеры в документации](https://docs.unity3d.com/Manual/UIE-examples.html).  <https://github.com/Unity-Technologies/ui-toolkit-manual-code-examples/tree/master/ToggleExample>. |
| Начало работы. |
|  |

|  |  |
| --- | --- |
| **Система анимации в Unity** | |
| [Уроки](https://www.youtube.com/watch?v=-FhvQDqmgmU&list=PLwyUzJb_FNeTQwyGujWRLqnfKpV-cj-eO). | |
| Назначить анимацию | На объект – **Animator**, в **Assets** – **Animator Controller**.  **Animator Controller** перетащить на **Animator** |
| Отделить анимацию | Выбрать ее и нажать **Ctrl + D** |
| Animator | Компонент Animator с назначенным контроллером и аватаром.  **Update Mode**: **Animate Physics** – для объектов, взаимодействующих с физическими объектами, **Unscaled Time** – для интерфейсов (чтобы работали во время паузы, например)  **Culling Mode:** нужно ли анимировать объект когда он не в поле камеры. |
| **Mixamo** | |
| * **Mixamo** предлагает персонажи и анимации, готовые к использованию как тип **Humanoid**. По-умолчанию это тип **Generic**, поэтому следует сразу преобразовать. После преобразования сгенерируется новый **Avatar**, который следует перетащить в **Animator**.   **Важно**! Это следует сделать и для анимаций, которые загружены без **mesh**. Кроме того, нужно указать аватар исходной **mesh** (именно той, которая была загружена в **Mixamo** при выборе анимации), поскольку сгенерированный аватар может иметь проблемы. И только после этого можно отсоединять анимацию.  **Совет**. Для загрузки анимаций используй один и тот же **mesh** и используй его аватар для всех прочих mesh.   * Анимации **Mixamo** можно назначать другим персонажам только после того, когда тип будет Humanoid, иначе они будут работать только для того объекта, с которым были скачаны. * Текстуры персонажа скорее всего будут упакованы в .fbx, поэтому их придется распаковать. | |
|  |  |
| **Animation Rigging** | |
| Rigging в Unity. Позволяет в интерактивном режиме взаимодействовать со скелетом персонажа.  [Доки](https://docs.unity3d.com/Packages/com.unity.animation.rigging@1.2/manual/ConstraintComponents.html). | |
| Установка   * В **Package Manager** называется **Animation Rigging.** Установить. * **Animator** * **Rig Builder** на том же уровне, что и **Animator** * **Bone Renderer** на том же уровне, что и **Animator** (не обязателен). Служит для визуализации костей. Он должен содержать все кости. Самый простой способ установить через меню:   **Animation Rigging -> Bone Renderer Setup**  Теперь можно настроить как отображать кости (цвет, разме, вид)   * На уровень ниже создаем пустой дочерний объект **TargetTracking** и вешаем на него компонет **Rig**. Этот объект перетаскиваем в поле компонента **Rig Builder**. * Теперь нужно создать доступное ограничение. Создаем дочерний пустой объект **HeadTracking** и вешаем на него, например, **Multi-Aim Constraint**.   Указываем кость, на которую ставится ограничение и цель – куда смотреть. | |

|  |
| --- |
| **ECS** |
| **GitHub -** [EntityComponentSystemSamples](https://github.com/Unity-Technologies/EntityComponentSystemSamples). |
|  |