**Описание.**

Жанр: аркадный шуттер.

Окружение: Skybox имитирующий космическое пространство.

Главный герой: космический корабль (далее корабль) стилизованный под векторную стрелку.

Враги: Большой астероид, малый астероид.

Условия победы: отсутствуют.

Условия поражения: столкновение с астероидом.

Цель: набрать наибольшее количество очков.

**Краткое описание игрового процесса**:

Игроку предстоит перемещаться по экрану уничтожая астероиды.

За каждый уничтоженный астероид игрок получает определенное количество очков.

За попадание в большой астероид игрок получает 5 очков, за попадание в малый 10 очков.

Запас снарядов не ограничен.

В верхней части экрана ведется подсчет очков текущей игровой сессии.

При столкновении с астероидом происходит поражение и вывод на экран результата текущей игровой сессии.

Счетчик полученных очков сбрасывается при перезагрузке игры.

**Управление:**

**W –** передвижение вперед относительно модели корабля.

**S –** передвижение назад относительно модели корабля.

**A –** поворот влево относительно модели корабля.

**D –** поворот вправо относительно модели корабля.

**Space –** выстрел.

**Enter –** перезагрузка игры при поражении.

**Описание космического корабля**

Корабль представляет из себя небольшую, белую, треугольную «стрелку». Позади стрелки, при движении, появляется эффект работы двигателя. Стрельба производится из «носа» корабля.

**Описание снаряда**

Маленький, белый круг, выпускаемый с большой скоростью из «носа» корабля.

**Описание астероида**

В игре предусмотрено 2 типа астероидов: большой и малый.

Оба вида представляют из себя векторное изображение замкнутой кривой белого цвета.

При попадании снаряда в астероид, выпускается небольшой сноп частиц имитируя взрыв астероида.

**Механика передвижения космического корабля.**

Передвижение осуществляется при помощи физического воздействия на корабль.

Поворот корабля осуществляется плавным изменением параметра вращения модели корабля.

При выходе за границу экрана корабль перемещается на противоположную сторону экрана в эквивалентной точке.

**Механика стрельбы**

Стрельба осуществляется добавлением префаба снаряда перед «носом» корабля.

Время перезарядки 0.1 секунды.

При появлении, на снаряд производится однократное физическое воздействие по направлению «носа» корабля в момент выстрела.

При достижении снарядом границы экрана снаряд уничтожается.

При попадании в астероид снаряда, снаряд уничтожается.

**Механика появления астероидов**

«Возрождение» астероидов происходит в случайном месте в пределах экрана. Возможно появление только больших астероидов. На экране может быть не менее четырех астероидов любого вида. Если их становиться меньше, то автоматически создается большой астероид.

**Механика передвижения астероидов**

При появлении на астероид производится однократное физическое воздействие в случайном направлении и со случайным значением силы.

**Механика попадания в астероид**

Если попадание произошло в большой астероид, то он, случайным образом, разделяется на один или несколько малых астероидов (от 1 до 3). При попадании в малый астероид он полностью уничтожается.

**Механика поражения**

Поражение наступает при столкновении корабля с астероидом. Для усложнения задачи было решено оставить игроку 1 попытку. После поражения результат сбрасывается.

**Перезагрузка игры**

После поражения необходимо нажать клавишу Enter и сцена игры будет перезагружена.

**Техническая документация**

**Структура вкладки «Projects»**

**Prefabs –** папка хранения префабов.

**Scenes –** папка хранения файлов сцен.

**Scripts –** папка хранения файлов скриптов.

**Sounds –** папка хранения звуковых файлов.

**Sprites –** папка хранения файлов изображений.

**Other –** папка хранения файлов дополнений и других категорий.

**Объекты в иерархии:**

**SpaceShip** – космический корабль, игрок.

Тэг – “Player”

Слой – “SpaceShip”

Содержит:

**RigitBody2D**, параметры: стандартные, за исключением **Gravity Scale = 0, Freeze Rotation Z = true**.

**Polygon Collider 2D**, параметры: стандартные.

**Inputs** - скрипт ввода, принимаемые параметры в inspector:

**Shoot Timer Value** (float, Время перезарядки выстрела, значения от 0.1 до 1) – стандартное значение 0.15.

**Can Shoot** (bool, возможность выстрела)

**SpaceShipControl** - скрипт управления космическим кораблем, принимаемые параметры в inspector:

**Rotation Speed** (float, скорость поворота корабля, от 100 до 300) – Значение по умолчанию 100.

**Is God Mode** (bool, установка бессмертия) – стандартное значение false.

**Start Bullet Position** (Transform, Стартовая позиция снаряда) – объект SpaceShip -> Bullet.

**BulletGO** (GameObject , объект для клонирования снаряда) – префаб Bullet.

**Crash Effects** (PartilelSystem, эффект взрыва корабля) - объект SpaceShip -> CrashEffect.

Дочерние объекты:

**Bullet** – объект обозначает позицию появления снаряда корабля, по умолчанию отключен.

**EngineEffect** – Объект ParticleSystem, имитирующий работу двигателя.

**CrashEffect** - Объект ParticleSystem, имитирующий взрыв корабля.

**Asteroids** – пустой объект для хранения астероидов, создаваемых во время игры. Необходим для предотвращения захламления иерархии.

Тэг – “Untagged”

Слой – “Default”

**Bullets** – пустой объект для хранения снарядов, выпускаемых игроком. Необходим для предотвращения захламления иерархии.

Тэг – “Untagged”

Слой – “Default”

**Game** – объект для управления игровым процессом.

Тэг – “Untagged”

Слой – “Default”

Содержит:

**Game Script –** скрипт управления игровым процессом, принимаемые параметры в inspector:

**Big Asteroid GO** (GameObject , объект для клонирования большого астероида) – префаб BigAsteroid.

**Small Asteroid GO** (GameObject , объект для клонирования малого астероида) - префаб SmallAsteroid.

**Asteroid Parent** (Transform, Родительский объект для создаваемых астероидов) – объект Asteroids.

**Position Script –** скрипт мониторинга позиции объектов, принимаемые параметры в inspector: нет.

**Text Output –** скрипт содержит поля вывода информации на экран, принимаемые параметры в inspector:

**Point Text** (Text, вывод полученных очков) – объект Canvas -> PointsCountText.

**Lose Text** (Text, вывод текста окончания игры и результата) - Canvas -> LoseText.

**Sound Controller** – скрипт управления воспроизведением звукового оформления, принимаемые параметры в inspector:

**Audios** (List<AudioSource>, объекты звуковых файлов) – Game -> Audios -> laserShoot, asteroudExplosion, shipExplosion.

Дочерние объекты:

Audios объект звукового оформления.

**Audios –** пустой объект, содержащий отдельные объекты звукового оформления.

Тэг – “Untagged”

Слой – “Default”

Дочерние объекты:

**laserShoot** – звук выстрела.

**asteroidExplosion** – звук уничтожения астероида.

**shipExplosion** - звук уничтожения корабля.

**Canvas –** объект интерфейса, вывода информации на экран.

Тэг – “Untagged”

Слой – “UI”

Дочерние объекты:

**PointsCountText** – вывод на экран полученных очков.

**LoseText** – вывод на экран текста поражения и результата игры.

**Префабы:**

**BigAsteroid** – Большой астероид. Разделяется на несколько малых астероидов при попадании.

Тэг – “BigAsteroid”

Слой – “Asteroid”

Содержит:

**RigitBody2D**, параметры: стандартные, за исключением **Gravity Scale = 0**.

**Polygon Collider 2D**, параметры: стандартные.

**AsteroidScript** - скрипт управления астероидом, принимаемые параметры в inspector:

**Destroy Effect** (ParticleSystem, Эффект взрыва астероида) – объект BigAsteroid -> DestroyEffect

Дочерние элементы:

DestroyEffect - Объект ParticleSystem, имитирующий взрыв астероида.

**SmallAsteroid** – Малый астероид. Уничтожается при попадании.

Тэг – “SmallAsteroid”

Слой – “Asteroid”

Содержит:

**RigitBody2D**, параметры: стандартные, за исключением **Gravity Scale = 0**,

**Polygon Collider 2D**, параметры: стандартные

**AsteroidScript** - скрипт управления астероидом, , принимаемые параметры в inspector:

**Destroy Effect** (ParticleSystem, Эффект взрыва астероида) – объект BigAsteroid -> DestroyEffect

Дочерние элементы:

DestroyEffect - Объект ParticleSystem, имитирующий взрыв астероида.

**Bullet –** Снаряд выпускаемый кораблем. Уничтожается при достижении границ экрана или при столкновении с астероидом.

Тэг – “Bullet”

Слой – “Bullet”

Содержит:

**RigitBody2D**, параметры: стандартные, за исключением **Gravity Scale = 0**, **Collision Detection = Continuous.**

**Circle Collider 2D**, параметры: стандартные

**BulletScript** - cкрипт управления снарядом, принимаемые параметры в inspector:

**Bullet Speed** (float, скорость полета снаряда) – значение по умолчанию 10.

**Скрипты:**

**GameScript**

Класс отвечающий за игровой процесс. Наследуется от MonoBehaviour.

Используемые пространства имен System.Collections.Generic, UnityEngine, UnityEngine.SceneManagement.

**Поля**

|  |  |
| --- | --- |
| soundController | Статическое публичное поле **SoundController** для управления воспроизведением звукового оформления. |
| asteroids | Статическое публичное поле List объектов **GameObject** для хранения астероидов созданных в текущей сессии. Так же служит для определения количества астероидов на экране. |
| bigAsteroidGO | Приватное поле **GameObject** для префаба большого астероида. Необходимо для создания нового астероида на сцене. Сериализуемое поле для отображения в inspector. |
| smallAsteroidGO | Приватное поле **GameObject** для префаба малого астероида. Необходимо для создания нового астероида на сцене. Сериализуемое поле для отображения в inspector. |
| asteroidParent | Приветное поле Transform, родительский объект для всех создаваемых астероидов. |
| output | Приветное поле **TextOutput** для доступа к объектам вывода информации на экран. |
| gameDone | Приветное поле **bool** для обозначения конца игры. |
| points | Приватное поле **int** хранения полученных очков игроком. |

**Методы**

|  |  |
| --- | --- |
| Start() | Вызывается приложением. Производит инициализацию полей **asteroids**, **output**, **soundController**. Производит подписку на события. Создает первоначальное количество астероидов. |
| SpaceShipControl\_CrushEvent () | Обработчик события **SpaceShipControl**.**CrushEvent.** Нет принимаемых параметров.Производит действия при столкновении корабля с астероидом. |
| BulletScript\_PointCountEvent (int pointCount) | Обработчик события **BulletScript.PointCountEvent.** Принимаемыепараметры: int **pointCount –** количествоочков за попадание**.** Производит начисление очков при попадании снаряда в астероид. |
| AsteroidsScript\_AsteriodSpawnEvent (bool isBig, Vector2 position) | Обработчик события **AsteroidsScript.AsteriodSpawnEvent.** Принимаемыепараметры: bool **isBig –** какой астероид будет добавлен на сцену, Vector2 **position** – точка появления астероида. Производит вызов методасоздания астероидов. |
| Update() | Вызывается приложением. Следит за количеством астероидов, производит вращение Skybox. Производит перезапуск приложения при нажатии Enter во время поражения. |
| CreateAsteroid (bool isBig, Vector2 position) | Производитсоздание астероидов и добавление созданного астероида в List **asteroids**. Принимаемыепараметры: bool **isBig –** какой астероид будет добавлен на сцену, Vector2 **position** – точка появления астероида. |
| CheckAsteroidCount() | Производит проверку количество астероидов на сцене. Если количество астероидов меньше минимального, то создается большой астероид в случайной точке на экране. Принимаемых параметров нет. |

**SoundController**

Класс отвечающий за воспроизведение звукового оформления. Наследуется от MonoBehaviour.

Используемые пространства имен System.Collections.Generic, System.Linq, UnityEngin.

**Поля**

|  |  |
| --- | --- |
| audios | Приватное поле List объектов **AudioSource** дляхранения всех ссылок на звуковые файлы. |

**Методы**

|  |  |
| --- | --- |
| PlaySound(string audioName) | Производит воспроизведение звукового файла. Принимаемые параметры: string audioName – параметр имени объекта, по которому будет произведен поиск ссылки на звуковой файл в List audios. |

**TextOutput**

Класс содержащий поля вывода информации на экран. Наследуется от MonoBehaviour.

Используемые пространства имен UnityEngin, UnityEngin.UI.

**Поля**

|  |  |
| --- | --- |
| pointsText | Публичное поле Text для вывода текущего количества очков игрока. |
| loseText | Публичное поле Text для вывода текста поражения и финального результата. |

**Inputs**

Класс принимающий ввод игрока для управления космическим кораблем. Наследуется от MonoBehaviour.

Используемые пространства имен UnityEngin.

**Поля**

|  |  |
| --- | --- |
| shootTimerValue | Приватное поле **float** для установки времени перезарядки выстрела. Сериализуется для вывода в inspector. |
| shootTimer | Приватное поле **float** дляотсчета времени перезарядки выстрела. |
| spaceShipControl | Приватное поле **SpaceShipControl** для передачи параметров ввода в класс управления кораблем. |
| horizontalAxis | Приватное поле **float** для хранения ввода игрока по горизонтальной оси. |
| verticalAxis | Приватное поле **float** для хранения ввода игрока по вертикальной оси. |
| canShoot | Публичное поле **bool** дляустановкивозможности выстрела |

**Методы**

|  |  |
| --- | --- |
| Awake() | Вызывается приложением. Производит инициализацию полей **spaceShipControl**, **shootTimer**. |
| Update() | Вызывается приложением. Записывает ввод игрока в поля **horizontalAxis, verticalAxis.** Изменяет значение **shootTimer.** Производит фиксацию нажатия кнопки выстрела. |
| FixedUpdate() | Вызывается приложением. Передает значения ввода в **spaceShipControl,** вызывая метод движения корабля **SpaceShipControl**.**MoveControl**(float horizontalAxis, float verticalAxis) |

**SpaceShipControl**

Класс, осуществляющий управление космическим кораблем. Наследуется от MonoBehaviour.

Используемые пространства имен UnityEngin.

**Делегаты и события**

|  |  |
| --- | --- |
| CrushDelegate() | Публичный делегат столкновения корабля и астероида. Принимаемых параметров нет. Необходим для события **CrushEvent** |
| CrushEvent | Публичное статическое событие, вызываемое при столкновении корабля и астероида. |

**Поля**

|  |  |
| --- | --- |
| rotationSpeed | Приватное поле float скорости поворота корабля. Сериализуется для отображения в inspector. |
| isGodMode | Приватное поле bool для установки «бессмертия». Сериализуется для отображения в inspector. |
| startBulletPosition | Приветное поле **Transform** дляуказания стартовой позиции выпускаемого снаряда. Сериализуется для отображения в inspector. |
| bulletGO | Приветное поле **GameObject** для создания нового снаряда из префаба. Сериализуется для отображения в inspector. |
| crashEffect | Приватное поле **ParticleSystem** для воспроизведения эффекта взрыва корабля. Сериализуется для отображения в inspector. |
| spaceShipRb | Приватное поле RigitBody для физических применения физических сил и управления скоростью коробля. |
| spaceShipSpeed | Приватное поле **float** скорости корабля. |
| bulletsParent | Приватное поле **Transform** для указания родительского объекта снарядов в иерархии |

**Свойства**

|  |  |
| --- | --- |
| ShipPosition | Приватное свойство **Vector3. Get -** Возвращает текущую позицию корабля. **Set** - Устанавливает новую позицию корабля. |
| spaceShipRbVelocity | Приватное поле **Vector2.** Возвращает текущую скорость коробля |

**Методы**

|  |  |
| --- | --- |
| Awake() | Вызывается приложением. Инициализирует поля **bulletsParent, spaceShipRb** |
| Update() | Вызывается приложением. Проверяет позицию объекта на экране, вызывая метод **PositionScript.CheckPosition**(Vector2 position). |
| MoveControl(float horizontalAxis, float verticalAxis) | Производит передвижение и поворот корабля. Принимаемые параметры: float **horizontalAxis –** значение поворота по оси Z модели корабля (отрицательное значение - поворот влево, положительное - вправо). Расчёт поворота по формуле **значение поворота \* скорость поворота \* время между кадрами**. **verticalAxis –** значение прикладываемой силы для передвижения корабля по оси Y модели корабля (отрицательное значение – вниз относительно корабля, положительное – вверх относительно корабля). Расчет по формуле **значение силы \* скорость корабля \* направление оси Y корабля**. |
| Shoot() | Производит создание клона снаряда из префаба снаряда **bulletGO.** Принимаемых параметров нет. |
| OnCollisionEnter2D(Collision2D collision) | Вызывается при физическом взаимодействии. Производит проверку с чем столкнулся корабль. Отключает отображение и физическое взаимодействие корабля, возможность стрельбы. Воспроизводит эффект взрыва корабля **crashEffect,** затем уничтожает объект и вызывает событие **CrushEvent** |

**AsteroidScript**

Класс, осуществляющий управление астероидом. Наследуется от MonoBehaviour.

Используемые пространства имен UnityEngin.

**Делегаты и события**

|  |  |
| --- | --- |
| AsteroidSpawnDel(bool isBig, Vector2 position) | Публичный делегат создания астероида. Принимаемые параметры: bool **isBig** – значение true обозначает создание BigAsteroid, false - SmallAsteroid. Необходим для события **AsteriodSpawnEvent** |
| AsteriodSpawnEvent | Публичное статическое событие, вызываемое при необходимости создания астероида. |

**Поля**

|  |  |
| --- | --- |
| destroyEffect | Приватное поле **ParticleSystem** для воспроизведения эффекта взрыва астероида. Сериализуется для отображения в inspector. |

**Свойства**

|  |  |
| --- | --- |
| AsteroidPosition | Приватное свойство **Vector3. Get -** Возвращает текущую позицию астероида. **Set** - Устанавливает новую позицию астероида. |

**Методы**

|  |  |
| --- | --- |
| Start() | Вызывается приложением. Производит однократное физическое воздействие на астероид по направлению в случайную строну. |
| Update() | Вызывается приложением. Проверяет позицию объекта на экране, вызывая метод **PositionScript.CheckPosition**(Vector2 position). |
| RemoveAsteroid() | Осуществляет уничтожение объекта через 1 секунду, воспроизводит эффект взрыва астероида **destroyEffect,** отключение отображение и взаимодействие астероида, удаляет **GameObject** астероида из List **GameScript.asteroids** |
| OnCollisionEnter2D(Collision2D collision) | Вызывается при физическом взаимодействии. Производит проверку с чем столкнулся астероид и тип данного астероида. Отключает отображение и физическое взаимодействие астероида. В зависимости от типа астероида производит следующие действия: если астероид имеет тег «**bigasteroid**», то вызывается событие создания малых астероидов случайное количество раз (от 1 до 3), вызывается метод **RemoveAsteroid**. Если тип астероида «**smallasteroid**» вызывается метод **RemoveAsteroid**. |

**BulletScript**

Класс, осуществляющий управление снарядом. Наследуется от MonoBehaviour.

Используемые пространства имен UnityEngin.

**Делегаты и события**

|  |  |
| --- | --- |
| PointCountDelegate(int pointCount); | Публичный делегат начисления очков. Принимаемые параметры: int pointCount – передаваемое значение начисляемых очков. Необходим для события **PointCountEvent** |
| PointCountEvent | Публичное статическое событие, вызываемое при начислении очков при попадании в астероид. |

**Поля**

|  |  |
| --- | --- |
| bulletSpeed | Приватное поле **float** для установки скорости полета снаряда. Сериализуется для отображения в Inspector. |

**Свойства**

|  |  |
| --- | --- |
| BulletPosition | Приватное свойство **Vector2. Get -** Возвращает текущую позицию снаряда. |

**Методы**

|  |  |
| --- | --- |
| Start() | Вызывается приложением. Производит однократное физическое воздействие на снаряд по направлению оси Y «носа» корабля в момент выстрела. |
| Update() | Вызывается приложением. Проверяет координаты снаряда и уничтожает снаряд по достижении границы экрана. |
| OnCollisionEnter2D(Collision2D collision) | Вызывается при физическом взаимодействии. Производит проверку с чем столкнулся снаряд, если это астероид, то производятся следующие действия: начисление очков (если астероид имеет тег «bigasteroid» начисляется 5 очков, если «smallasteroid» начисляется 10 очков), затем снаряд уничтожается и вызывается событие **PointCountEvent**. |

**PositionScript**

Класс мониторинга позиции передаваемого объекта. Наследуется от MonoBehaviour.

Используемые пространства имен UnityEngin.

**Поля**

|  |  |
| --- | --- |
| camera | Приватное статическое поле **Camera** для получения главной игровой камеры. |

**Методы**

|  |  |
| --- | --- |
| Awake() | Вызывается приложением. Производит инициализацию поля **camera**. |
| CheckPosition (Vector2 position) | Публичный статический метод. Возвращаемый тип Vector2, принимаемые параметры: Vector2 position – текущая позиция объекта. Проверяет позицию объекта. Если объект достигает границы экрана возвращается эквивалентная позиция с противоположной границы экрана. |