Faller – Platformer. 2D.

**Ориентация:** вертикальная

**Референсы:**

Leap on  
Atari breakout~  
Geometry dash~

Оглавление

[Геймплей и основные механики 2](#_Toc52745234)

[Центр 2](#_Toc52745235)

[Платформы 2](#_Toc52745236)

[Прыгучий игрок 3](#_Toc52745237)

[Получение очков и Trail 3](#_Toc52745238)

[Звезды 3](#_Toc52745239)

[Камера 3](#_Toc52745240)

[Цель в игре 3](#_Toc52745241)

[Управление 3](#_Toc52745242)

[Дополнительная часть управления: 3](#_Toc52745243)

[Обучение 3](#_Toc52745244)

[Понятия, заметки, внимашки: 4](#_Toc52745245)

[Гравитационная скорость 4](#_Toc52745246)

[Особенности и жесткие привязки 4](#_Toc52745247)

[Использованные ассеты 4](#_Toc52745248)

[Архитектура сцены 4](#_Toc52745249)

[Игровая сцена 4](#_Toc52745250)

[Особые параметры Order layer 5](#_Toc52745251)

[Дополнительные элементы физики, эффекты 5](#_Toc52745252)

[Эффект Toss Up / down 5](#_Toc52745253)

[Платформы 5](#_Toc52745254)

[Default platform 5](#_Toc52745255)

[Physical platform 5](#_Toc52745256)

[Платформа, двигающаяся по орбите вокруг Центра 5](#_Toc52745257)

[Опасности 6](#_Toc52745258)

[Вспышка 6](#_Toc52745259)

[Преследователь 6](#_Toc52745260)

[События эффекты 6](#_Toc52745261)

[Компасы 7](#_Toc52745262)

[Компас на вспышку 7](#_Toc52745263)

[Компас на игрока 7](#_Toc52745264)

[Настройки 7](#_Toc52745265)

[Статистика 8](#_Toc52745266)

[Внутриигровая валюта и покупки 8](#_Toc52745267)

[Индекс оценки стоимости 8](#_Toc52745268)

[Сохранение 8](#_Toc52745269)

[Структура модели «Player Data Saver» 8](#_Toc52745270)

[Изменение модели «Player Data Saver» 9](#_Toc52745271)

[Логика сохранения: 9](#_Toc52745272)

[Сброс сохраненных данных 10](#_Toc52745273)

[Восстановление статистики из облака 11](#_Toc52745274)

[Google services 11](#_Toc52745275)

[Авторизация 11](#_Toc52745276)

[Чтение и запись данных в облако 12](#_Toc52745277)

[Безопасность хранения данных 12](#_Toc52745278)

[Механизм защиты от воздействия на данные через оперативную память (RAM): 12](#_Toc52745279)

[Механизм защиты от воздействия на данные через внешнюю память: 12](#_Toc52745280)

[Эмоции 12](#_Toc52745281)

[Пример использования 12](#_Toc52745282)

[Звуки 12](#_Toc52745283)

# Геймплей и основные механики

## Центр

Есть некий центр, в форме большого круга. При соприкосновении с центром игрок умирает.

## Платформы

Из центра в разные стороны выдвигаются платформы. Они бывают как динамичные, так и статичные.   
Отличительная часть: абсолютно большая часть платформ двигается от центра вверх. Когда платформа поднимается слишком высоко, она начинает мигать и впоследствии исчезает. (**Внимание!** Слова вверх/вниз/влево/вправо являются относительными к центру!). Референс: Leap On.

Так же, некоторые платформы могут быть подвержены физике: учет всех столкновений  
 и не подвержены: не учитывает столкновения с другими платформами.

## Прыгучий игрок

Игровые элементы будут иметь эффект «Toss»

## Получение очков и Trail

Если игрок двигается выше некой определенной скорости, то за ним появляется белый trail, чтобы подчеркнуть это. Так же, игроку в момент начинают начисляться очки по арифметической прогрессии: за первый тик 1 очко, за второй тик 2 очка и т.д. Получение очков отображается с помощью popup текста, вылетающего из игрока при получении их.

## Звезды

В игре присутствуют звезды, которые выступают в роли внутриигровой валюты.  
Звезды появляются с некой периодичностью на игровой сцене, собираются касанием. После касания сразу исчезают.  
После прошествия (X) секунд исчезают  
Звезда имеет эффект «Toss Up»

## Камера

Камера в игре следует за игроком, а также плавно отдаляется, когда тот отлетает от центра. Чем дальше игрок от центра, тем сильнее отдаляется камера

# Цель в игре

Не умереть и набрать как можно больше очков.

# Управление

Касание по левой/правой части экрана: персонаж двигается в левую/правую часть экрана

## Дополнительная часть управления:

-В данный момент под вопросом-

Двойное нажатие по левой/правой части экрана: рывок в сторону движение (аналог прыжка faceless void с (без) прохождением через предметы. Подразумевается, что после нажатия, например, на левую часть экрана игрок начнет двигаться влево по оси X + сохранит движение по Y. Таким образом рывок будет не только по оси X, но так же вверх или вниз, в зависимости от текущего направления движения.

# Обучение

Показать на левой и правой части экрана соответственно мигающий кружочек и надпись над ним “Tap!” (Надпись повернуть во внешнюю строну на 45 градусов). Надпись и кружочек исчезнут после того, как игрок сделает длительное нажатие (Дольше <1> секунды на соответственную сторону экрана).

Вывести обучение при:

1. Первый запуск игры
2. Игрок зашел на уровень и уже <5> секунд бездействует

# Понятия, заметки, внимашки:

[Внимашка!](#_Платформы) Слова вверх/вниз/влево/вправо являются относительными к центру!

[Внимашка!](Player#_Структура_модели_) Для покупок за реальную валюту использовать отдельный класс!

[Внимашка!](#_Сброс_сохраненных_данных) Сбросить сохраненные данные можно лишь один раз с момента запуска приложения! Для повторного сброса необходимо перезапустить приложение!

Гравитационная скорость – скорость игрока по вектору, направленному к Центру. Так как в игре используется своя реализация физики, то вычисляется в Player Movement.

# Особенности и жесткие привязки

Сохранение данных статистики (Счет, звезды и тд) происходит при включении Game Over Menu: UI объекта с кнопками “начать уровень заново”, “выйти в главное меню” и тп.

# Использованные ассеты

GoogleMobileAds-v5.2.0

GooglePlayGamesPlugin-0.10.09

# Архитектура сцены

## Игровая сцена

1. Main Camera
2. Player UI - Canvas
3. Player
4. X – Manager, Object Pooler
5. Event System

## Особые параметры Order layer

PopUp error window: 50

Shutter: 9999

# Дополнительные элементы физики, эффекты

Эффект Toss Up / down  
Когда игрок касается игрового объекта с данным эффектом, то тот подбрасывает его в направлении, указанном в названии эффекта, с силой, зависящей от его Гравитационной скорости.

Подбрасывание происходит за счет назначении игроку скорости.

Скорость подбрасывания равна Гравитационной скорости игрока \* (прыгучесть), но не меньше, чем минимальная скорость подбрасывания, равная (X),

# Платформы

## Default platform

Эта платформа имеет размер 3 на 0.5 юнитов. Имеет эффектор “platform”, дающий возможность проходить через платформу снизу (как в doodle jump). Платформа всегда находится под углом 90 градусов к вектору, который ведет в Центр. Движется со случайной скоростью (от x до y) из центра вверх, когда достигает случайную высоту (от x до y), мигает (изменение прозрачности) ~2 сек и затем исчезает (удаляется).  
Платформа имеет эффект «Toss Up»

## Physical platform

*На данную платформу гравитация не действует!*

Эта платформа может взаимодействовать с физикой (На нее действуют силы – с ней может столкнуться игрок/другая платформа). Так же, данная платформа должна иметь значительно большую массу, нежели игрок, что бы от нее можно было так же оттолкнуться.  
Появляется в случайном месте (с эффектом появления. Например, изменение прозрачности), двигается в случайном направлении и исчезающая с эффектом мигания.

## Платформа, двигающаяся по орбите вокруг Центра

Имеет эффектор “platform”, дающий возможность проходить через платформу снизу (как в doodle jump). Платформа всегда находится под углом 90 градусов к вектору, который ведет в Центр.

Эта платформа двигается по некой орбите вокруг Центра. Так же, может приближаться или удаляться от центра.

# Опасности

## Вспышка

Из Центра с неким периодом (который зависит от увеличения сложности игры) будут вырываться вспышки в случайных направлениях.

Выброс представляет из себя некий «лазер», который, после небольшой задержки, мгновенно вырывается из поверхности Центра и убивает игрока, если тот её коснется.

За насколько секунд до выброса на экране игрока появляется предупреждающий значок типа «компас». Т.е. значок показывает, с какой стороны будет вспышка.   
Значок «компас» исчезает при переходе на состояние «**выброс**» у вспышки (см. жизненный цикл вспышки)

Жизненный цикл вспышки:

1. Включение вспышки
2. Задержка перед появлением – сопровождается отметкой типа «Компас» и анимацией на поверхности Центра
3. Действие вспышки - выброс
4. Вспышка отключается (Анимация)

Характеристики вспышки:

* Время действия
* Ширина

Характеристики генератора вспышек:

* Частота появления вспышек

## Преследователь

“Существо”, преследующее игрока. При контакте с ним игрок умирает.

Это существо выглядит как… <Два круга друг в друге, разного размера. Тот, что больше, частично прозрачен.> Так же имеет trail. Цвет… <Бардовый>

Преследует игрока по принципу среднего по силе самонаведения – то есть отклоняет свой курс в зависимости от расстояния до игрока, а не следует точно в направлении.   
Таким образом, если игрок увернется от преследователя, то тот полетит дальше.

# События эффекты

-=Необходима доработка=-

Завязать игру так же на событиях эффектах, где вся игровая логика и цветовая палитра зависят от того или иного эффекта.

Игра начинается с дефолтного события, которое потом каждые (40) секунд сменяются на случайное другое с плавным эффектом. Так же записать характерные и обязательно уникальные черты событий.

Кроме того, отдельная заморочка с палитрой цветов. Необходимо определить главные элементы игры и сформировать для каждого «набора» свою палитру.

Пример:

(Отдельно записать ч-б объекты, такие как игрок, платформы)

Цвет неба:

1. Синий

2. Лилово - розовый

3. Светло зелёный, как трава

Далее к этим трём цветам (по одному на событие эффект) необходимо выбрать гармонирующие цвета остальных элементов. Так же представлю сразу «наборы»:

⁃ небо, Центр, звёзды, существа

И обязательное условие: центр всегда должен быть красным, возможны лишь смены оттенков...

## Генерация платформ

Первое событие-эффект всегда по умолчанию!

Далее, определение правил генерации (настроек) нового события-эффекта, производится каждые (30) секунд.

Правила генерации платформ:

При генерации выбирается/создается платформа со одним из пунктов на каждое **главное свойство**. Порядок выбора главных свойств расположен по исполнению.

**Тип/ы движения платформы (Возможно несколько пунктов):**

1. Вертикальное движение
2. Движение по кругу

**Настройки движения (Настройки выбираются для каждого типа):**

1. Если платформа двигается по вертикали:
   1. Вверх
   2. Вниз
   3. Случайное направление
2. По кругу вокруг Центра
   1. По часовой (Влево)
   2. Против часовой (Вправо)
   3. Случайное направление

**Место создания платформы:**

1. Платформа движетсяв любом направлении: **Высота от [радиус Центра +1] до Наивысшей точки *-Задать где-то как константу- (*InRandomArea)**
2. Платформа может двигаться по вертикали
   1. Платформа движется ТОЛЬКО вверх: **Центр (InCentre)**
   2. Платформа движется ТОЛЬКО вниз: **Высота от Наивысшей точки до половины пути к радиусу Центра (InHighestArea)**

**Причина разрушения платформы**

1. Платформа была создана в InRandomArea: **Платформа исчезнет сразу/ через пару секунд (NoLifeTime). Устанавливается извне!**
2. Платформа двигается ТОЛЬКО по кругу: **платформа исчезнет через какое-то время (AsTimePasses)   
   Устанавливается извне!**
3. Платформа может двигаться по вертикали: **значение будет установлено самой платформой (VerticalCauseOfDeathControl)**

**Настройка периода генерации платформ (Правило для генератора платформ):**

1. Период по умолчанию равна (X)
2. Если платформа создается в случайном месте (InRandomArea), то период уменьшается на (Y)
3. Если платформа не имеет время жизни (NoLifeTime), то период уменьшается на (Z)

**Значение будет установлено самой платформой, имеющей вертикальный тип движения (VerticalCauseOfDeathControl)**

1. Вверх: **исчезнет если слишком сильно поднимется вверх (TopBorder)**
2. Вниз: **исчезнет, когда опустится до радиуса Центра (BottomBorder)**

**Правила генерации по умолчанию:**

1. Вертикальный тип движения
2. Платформа движется вверх
3. Платформа создается из Центра
4. Причина уничтожения: при достижении верхней границы мира (VerticalCauseOfDeathControl -> TopBorder)

# Компасы

В игре присутствуют такой игровой объект, как «Компас»

Компас представляет собой спрайт, который указывает на некий ключевой объект. Чем больше необходимо привлечь внимания к объекту, тем больше значение альфа канала изображения.

## Компас на вспышку

Жизненный цикл:

* Компас создается вместе со вспышкой и привязывается к ней.
* При включении плавно увеличивает прозрачность до необходимого значения
* Во время работы мерцает с красного до оранжевого цветов  
  Передвигается и меняет свою прозрачность в соответствии с углом расстояния между игроком и вспышкой (см. детали ниже)
* Отключается при переходе на состояние «**выброс**» у вспышки (см. жизненный цикл вспышки)  
  При выключении увеличивается в (1.5) раза и затем резко исчезает.

Функциональность:

Компас меняет свою позицию и прозрачность в зависимости от положения вспышки и игрока относительно друг друга *вокруг Центра* – т.е относительно угла разницы между игроком и вспышкой.

Местоположение компаса:

Компас меняет свое положение от одной позиции к другой; Начальная позиция находится ЗА левым нижним углом, вне видимости дисплея. Конечная позиция находится ЗА правым нижним углом, также вне видимости дисплея.

Когда вспышка находится ровно с обратной стороны Центра, то компас находится в начальной, левой позиции. Далее, при приближении вспышки к игроку (По часовой стрелке, по левой половине Центра до игрока) компас так же двигается вслед за вспышкой до центра экрана. При удалении вспышки далее, по часовой стрелке, в правую половину Центра от игрока, компас двигается в конечную, правую часть экрана.

Прозрачность компаса:

Компас меняет свою прозрачность также, в зависимости от положения вспышки относительно игрока. Чем ближе вспышка к игроку (т.е чем ближе компас к середине дисплея), тем больше значение альфа-канала. И чем ближе компас к начальной/конечной позиции (т.е. чем дальше вспышка от игрока), тем меньше значение альфа-канала.

Минимальное значение альфа-канала: 25%  
Максимальное значение альфа-канала: 100%

Чувствительность:

При этом, чем ближе компас к середине дисплея, тем сильнее меняется чувствительность к передвижению вспышки относительно игрока:

1. Чем ближе компас к середине дисплея, тем быстрее он движется. Соответственно, при сильном удалении от вспышки компас будет двигаться заметно медленнее.
2. Чем ближе компас к середине дисплея, тем медленнее меняется его прозрачность.

Изображение компаса:

-=Нужна доработка! =-

Компас представляет из себя <Изображение>.

## Компас на игрока

Начиная с высоты H, включается компас-стрелочка, что указывает на игрока. Прозрачность меняется в пределах от высоты H до H + h.

На высоте H + h значок компаса становится полностью непрозрачным.

Также, данный значок является анимированным: стрелка покачивается вокруг игрока.   
При выключении компаса (Когда компас становится полностью прозрачным) анимация покачивания НЕ должна полностью сбрасываться, а должна сохранять своё текущее местоположение.

# Настройки

В игре предусмотрено сохранение следующих настроек:

1. Язык
2. Вкл/выкл музыки
3. Вкл/выкл звуковых эффектов

# Статистика

1. Максимальное количество собранных звезд за игру

2. Максимальное количество набранных очков за игру

4. Максимальный множитель очков (Если помнишь, там идет алгебраическая прогрессия)

5. Максимальное время жизни за игру

6. Общее время, проведенное в игре (не в меню, а именно в игре)

# Внутриигровая валюта и покупки

1. Общее количество набранных звезд

## Индекс оценки стоимости

В данном классе лежат поля, связанные со внутри-игровыми покупками. Это может быть как внутри-игровая валюта, так и сами покупки (Массивы данных, поля с булевыми типами и т.д.) Так же, необходимо каждой покупке присвоить индекс стоимости с типом определенной валюты. Например, в Jonks в роли валюты выступает только Звезды. 1 Звезда = 1 индексу Звезды. Скин, стоимостью в 10 звезд, оценивается соответственно в 10 индексов Звезды.

Но, если будет еще какой-либо тип валюты, например Золото, то будет необходимо ввести дополнительный тип для индекса оценки стоимости.

Если валюту с разными типами возможно конвертировать между собой, то необходимо будет описать дополнительное правило для считывания индекса.

Так же подразумевается, что оценочные индексы будут вычисляться заново при назначении новых данных для полей.

# Сохранение

## Структура модели «Player Data Saver»

Внимание! Для покупок за реальную валюту использовать отдельный класс! (Раздел о нем пока не описан)

Модель => Класс PlayerDataSaver.

В классе присутствуют поля:

* Уникальный id (ключ) модели
* Класс со статистикой PlayerStats
* Класс с внутриигровой валютой и покупками PlayerInGamePurchases

Информацию о [статистике](#_Статистика) и [внутриигровой валюте с покупками](#_Внутриигровая_валюта_и) см. в смежных разделах

## Изменение модели «Player Data Saver»

При добавлении поля в модель необходимо:

* Обновить метод «создать с дефолтными значениями»
* Обновить метод «проверить на null все поля»

## Логика сохранения:

Игра оффлайн с периодическим выходом в онлайн.

Таким образом, данные сохранения хранятся в зашифрованном виде на устройстве и в исходном виде на сервере.

При входе в игру происходит следующее:

1. Загружаются данные с устройства, происходит проверка безопасности (расшифровка и валидация).
   1. Если данные не прошли проверку, то модель данных (не сами данные, а модель!) обнуляется, а также включается запрет на запись данных.
   2. Если данные прошли проверку, то выполняется синхронизация данных с сервером - загружаются данные с сервера и сравниваются с локальной моделью.

*Так же подразумевается, что синхронизация происходит после загрузки данных с сервера.   
Вот список таких точек: Главное меню; окончательное GameOverMenu. Для удобства в плане реализации думаю сделать static bool isGameRunning.*

\*\*\*

Пометка. Данные могут отличаться в следующих случаях:

1. Игрок играл оффлайн и сейчас данные стоит загрузить **на** **сервер** (данные на устройстве лучше, т.е. статы выше)

2. Игрок играл на другом устройстве онлайн и сейчас данные следует загрузить **на** **устройство** (данные на устройстве хуже, т.е. статы ниже)

3. Игрок и играл оффлайн на этом устройстве, и играл онлайн на другом устройстве

\*\*\*

1. Правила синхронизации
   1. Сначала модели сравниваются по id. Если id разный, значит локальную модель обнуляли/ создали заново.
      1. Если стоит флаг, означающий что за текущий запуск игры данные удалялись: значит, в период [Сброса сохраненных данных](#_Сброс_сохраненных_данных) было потеряно соединение с сервером. Загрузить локальную модель **на сервер**.
      2. Иначе – вывести игроку окошко с предложением выбора той модели, которую он хочет оставить.
   2. Если id одинаковый, то модели сверяются по следующим значениям, так как они могут только расти вверх:

* Значения из раздела [Статистика](#_Статистика)
* Индекс оценки стоимости
  + 1. Если все значения, по которым происходит сравнение моделей, равны => На сервере лежит актуальная модель.
    2. Иначе – смешать модель.

Поля из класса со статистикой: взять с максимальными значениями.

Сравнение покупок:   
Взять тот класс с внутри-игровыми предметами/валютой, у которой больше индекс стоимости. (Не должно возникнуть проблем, так как в Jonks подразумевается один тип «Звезда» для индекса оценки стоимости.

**Почему** если мы видим, что модели не совпадают, не вывести окошко с предложением выбора той модели, которую он хочет оставить?   
**Потому,** что в таком случае игрок потеряет часть данных. Мы же будем лояльны к игроку и смешаем модели.

1. Если локальная модель отличалась от модели на сервере, то после того, как получили смешанную модель, её необходимо установить, как используемую. А также записать на сервер и на устройство.
2. Сохранение данных должно происходить либо при выходе из приложения, либо на «контрольной точке» изменения статов. Список контрольных точек:

* Сбор наград
* Покупки во внутриигровом магазине (как за внутриигровую, так и за реальную валюту)
* Смешивание локальной и облачной модели при их различиях

## Сброс сохраненных данных

Внимание! Сбросить или восстановить с облака сохраненные данные можно лишь один раз с момента запуска приложения! Для повторного сброса необходимо перезапустить приложение!

Вывести предупреждающее окно:

«Внимание! Вы собираетесь удалить весь свой прогресс в этой игре. Для подтверждения введите слово “УДАЛИТЬ” в поле и нажмите Окей»

=Поле ввода=

=Кнопка «Ок» =

* 1. Если введенное слово совпадает со словом, которое необходимо ввести, то заменить предупреждающее окно – окном с сообщением «Прогресс удален!» =Кнопка «Ок» =
  2. Если введенное слово НЕ совпадает со словом, которое необходимо ввести, то заменить предупреждающее окно – окном с сообщением «Операция не была подтверждена!» =Кнопка «Ок» =

Если проверка на предупреждающем окне пройдена, то:

1. Обнулить модель
2. Удалить файлы сохраненных данных игрока
3. Записать данные в облако при условии доступа к нему
4. Разрешить запись файла

Восстановление статистики из облака

Внимание! Сбросить или восстановить с облака сохраненные данные можно лишь один раз с момента запуска приложения! Для повторного сброса необходимо перезапустить приложение!

При условии, что к облаку есть доступ => Кнопка активна только при доступе к облаку:

Вывести предупреждающее окно:

«Внимание! Вы собираетесь восстановить данные из облака. Это может привести к потере несохраненных локальных данных. Для подтверждения введите слово “ВОССТАНОВИТЬ” в поле и нажмите Окей»

=Поле ввода=

=Кнопка «Ок» =

1. Если введенное слово совпадает со словом, которое необходимо ввести, то заменить предупреждающее окно – окном с сообщением «Прогресс восстановлен!» =Кнопка «Ок» =
2. Если введенное слово НЕ совпадает со словом, которое необходимо ввести, то заменить предупреждающее окно – окном с сообщением «Операция не была подтверждена!» =Кнопка «Ок» =

Если проверка на предупреждающем окне пройдена, то:

1. Удалить файлы сохраненных данных игрока
2. Записать данные с сервера в локальную модель
3. Разрешить запись файла

# Google services

## Авторизация

Авторизация в гугл сервисах происходит сразу после входа в игру => в сцене Main menu.

## Чтение и запись данных в облако

Загрузка данных с облака происходит сразу после авторизации.

Данные в облако загружаются без шифрования.

# Безопасность хранения данных

В игре предусмотрены следующие механизмы для безопасного, защищенного от внешнего воздействия хранения данных:

## Механизм защиты от воздействия на данные через оперативную память (RAM):

Данные хранятся в структуре обертке «Safe <Type>» зашифрованном виде.

При создании новой структуры происходит следующее:

1. Создание ключа, где ключ — это случайное число.
2. Побитовое умножение оператором XOR сохраняемого значения на ключ.

При получении данных структуры через код происходит дешифровка, а также создание новой структуры от полученного дешифрованного значения

## Механизм защиты от воздействия на данные через внешнюю память:

# Эмоции

-Данная механика пока под вопросом-

На экране будет выделена область. В этой области будут появляться эмоции игрока.

Эмоции должны быть минималистичные (Можно использовать символьные?). Использовать белые цвета при отрисовке. Если что, их можно будет поменять в юнити.

Анимации эмоций – можно, но не сразу

## Пример использования

Пролетел близко с монеткой, но не взял ее:

- Злость

Включился хвост:

- Радость

- Скорость

Рядом враги:

- Страх

# Звуки

Звуковые эффекты необходимы для следующих событий:

1. Взял монетку
2. Отскочил от платформы? (Именно отскочил, а не застрял/остановился…)
3. Возродился
4. Умер