Faller – Platformer. 2D.

**Ориентация:** вертикальная

**Референсы:**

Leap on  
Atari breakout~  
Geometry dash~

Оглавление

[Геймплей и основные механики 2](#_Toc48480118)

[Центр 2](#_Toc48480119)

[Платформы 2](#_Toc48480120)

[Прыгучий игрок 2](#_Toc48480121)

[Получение очков и Trail 2](#_Toc48480122)

[Камера 2](#_Toc48480123)

[Цель в игре 2](#_Toc48480124)

[Дополнительные механики 3](#_Toc48480125)

[Управление 3](#_Toc48480126)

[Дополнительная часть управления: 3](#_Toc48480127)

[Обучение 3](#_Toc48480128)

[Понятия, заметки, внимашки: 3](#_Toc48480129)

[Особенности и жесткие привязки 4](#_Toc48480130)

[Использованные ассеты 4](#_Toc48480131)

[Архитектура сцены 4](#_Toc48480132)

[Игровая сцена 4](#_Toc48480133)

[Особые параметры Order layer 4](#_Toc48480134)

[Платформы 4](#_Toc48480135)

[Default platform 4](#_Toc48480136)

[Physical platform 5](#_Toc48480137)

[Платформа, двигающаяся по орбите вокруг Центра 5](#_Toc48480138)

[Опасности 5](#_Toc48480139)

[Выброс 5](#_Toc48480140)

[Преследователь 5](#_Toc48480141)

[Статистика 5](#_Toc48480142)

[Внутриигровая валюта и покупки 6](#_Toc48480143)

[Индекс оценки стоимости 6](#_Toc48480144)

[Сохранение 6](#_Toc48480145)

[Структура модели «Player Data Saver» 6](#_Toc48480146)

[Логика сохранения: 6](#_Toc48480147)

[Сброс сохраненных данных 8](#_Toc48480148)

[Восстановление статистики из облака 8](#_Toc48480149)

[Эмоции 9](#_Toc48480150)

[Пример использования 9](#_Toc48480151)

[Звуки 9](#_Toc48480152)

# 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

Игрок и платформы имеют материал высокой с прыгучестью (bouncy) 0.5– 1.0, из-за чего игрок будет постоянно отскакивать в разные стороны.

## 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.

## 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.

# Дополнительные механики

Если игрок быстро вращается вокруг своей оси, ему можно дать эффект “множителя очков”

Возможно добавление шипов…

Вокруг центра в разных точках появляются монетки…

Из центра резко по направлению “вверх” – от центра вылетает поток, например лавы (Резкий возникший столб)  
Или некая волна, которая убивает при касании. (Объект, который вылетает из центра и движется со скоростью выше средней)  
Появление таких препятствий обозначается на игровом экране эффектом типа “компас”: появляется горизонтальная полоска с восклицательным знаком сверху. Полоска находится со стороны опасности.

# 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!

# 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

# 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).

## 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

## Выброс

Из Центра с неким периодом (который зависит от увеличения сложности игры) будут вырываться выбросы.

Выброс представляет из себя материю, которая летит со скоростью выше среднего (быстро, но не слишком) 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. Так же, чем дальше вспышка *по поверхности Центра*, тем значок прозрачнее.

Таким образом, значок не реагирует на высоту прыжка игрока, но реагирует на его местоположение относительно стороны Центра

----Перенести информацию о значке в отдельный раздел----

## 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.

# 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 данной модели 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

## 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 использования 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, а также в тот момент, когда «не идёт игра». 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 - модели сверяются по следующим параметрам, так как они могут только расти вверх:

* См. раздел [Статистика](#_Статистика)
  1. Если все данные параметры **у модели с сервера** либо равны, либо больше – то считать, что модель актуальная лежит на сервере и использовать её.
  2. Если все данные параметры **у модели с сервера** либо равны, либо меньше – то считать, что модель актуальная лежит на устройстве и использовать её.
  3. Иначе – смешать модель. Значение из п.3 “Значения для сравнения моделей” взять максимальные. *С остальными значениями необходимо быть максимально осторожным, т.к. подразумевается, что они влияют на внутриигровые покупки.*

Сравнение покупок:   
Взять ту модель 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. После того, как данные синхронизированы, необходимо записать данные на устройство, на сервер и на модель.
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). Конкретно сейчас сохранение данных происходит при выходе из приложения (Запись файла + сохранение данных не сервере)

## Сброс сохраненных данных

Внимание! Сбросить 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
4. Разрешить запись файла

**Дописать**. Что делать, если в данный момент, на п.3 нет доступа к облаку? Как вариант, записывать некий уникальный ключ отдельно для модели при ее создании с нуля (CreateModelWithDefaultValues) и потом сравнивать с ключом модели с сервера. Если они отличаются, значит, модель обнуляли и необходимо данные на сервере заменить локальными данными. НО, может быть такая ситуация, к примеру: игрок играл в оффлайне, удалил данные. Потом забыл про это устройство и играл долго и счастливо с другого, не удаляя данные. Зашел в онлайн опять со старого устройства и опа - у него все обнулилось.

Поэтому: если мы видим, что ключи модели изменились, то игроку должно вывестись окошко с предложением выбора той модели, которую он хочет оставить.

Восстановление статистики из облака

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

# 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