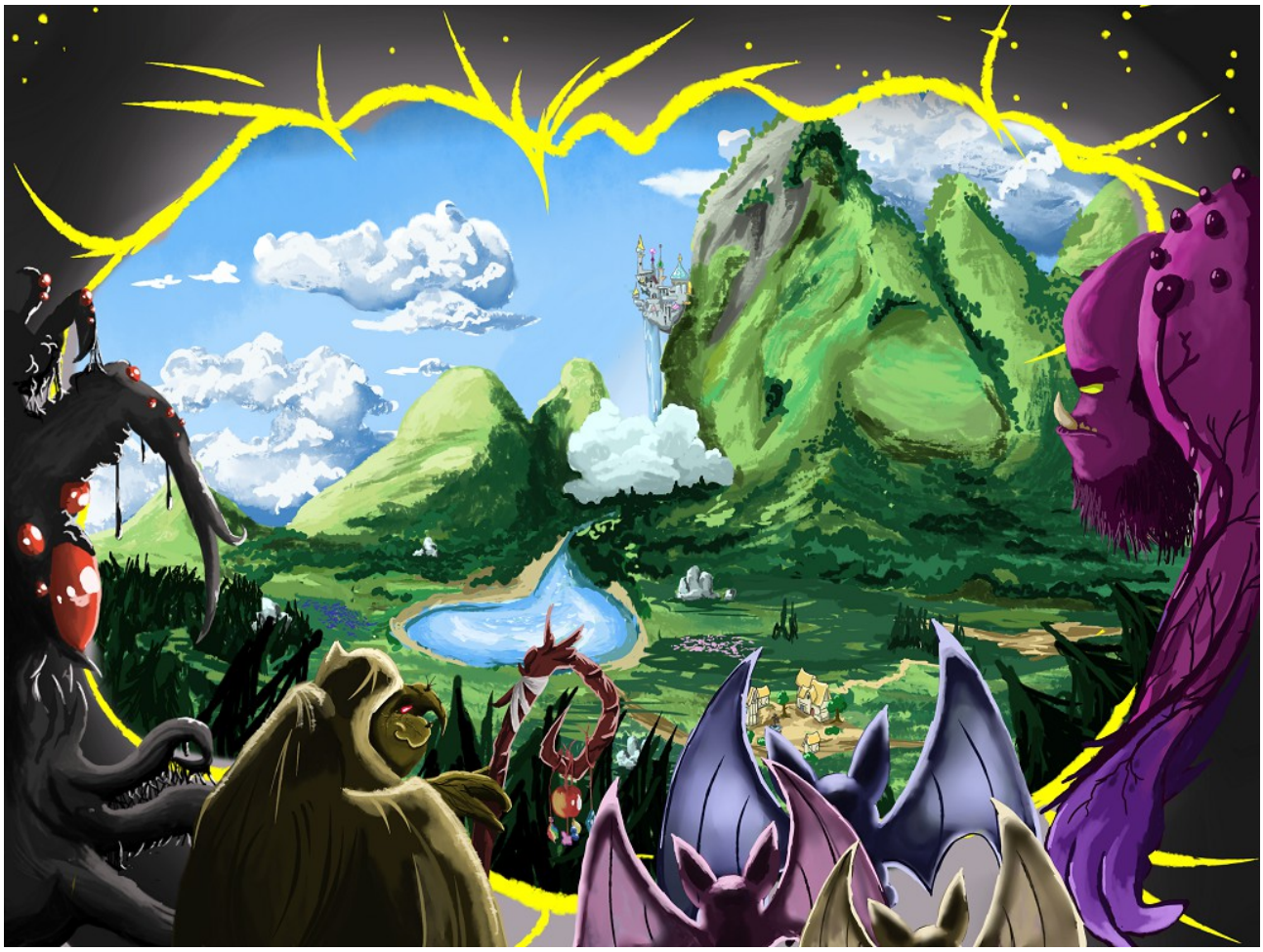
heroesofequestria.org

Герои Эквестрии

Тактическая пошаговая игра с элементами стратегии и RPG

Официальное руководство разработчика сценариев и моддера



Версия 1.1 от 17.08.2019

СОДЕРЖАНИЕ

1. Введение..... 2
1.1. Структура каталогов..... 4
2. Игровые объекты..... 5
2.1. Территории
2.2. Пони..... 6
2.3. Противники..... 7
2.4. Здания..... 8
2.5. Нейтральные объекты..... 9
2.6. Ресурсы
2.7. Заклинания и действия
3. Структура сценариев..... 12
3.1. Территория..... 13
3.2. Константы
3.3. Начальные установки..... 14
3.4. Объекты..... 16
3.5. Скрипты..... 19
3.6. Условие победы..... 20
4. Справочники..... 20
4.1. Метадействия
4.2. Действия смены хода..... 24
4.3. Пассивные способности..... 26
4.4. События скрипта..... 27
4.5. Действия скрипта..... 30
4.6. Условия победы..... 35
5. Особенности обработки скриптов..... 37
6. Встроенный редактор карт..... 38

1. Введение

Данное руководство служит максимально полным описанием структуры конфигураций и сценариев игры «Герои Эквестрии». Руководство не требует знаний какого-либо языка программирования или разметки – всё необходимое для внесения изменений или создания собственных сценариев описано в разделах. Также не требуется при создании или модификации сценариев обращаться к исходному коду игры – те сведения, которые можно почерпнуть из изучения исходного кода, уже внесены в руководство.

Руководство предназначено для следующих целей:

- Внесение в игру новых пони, зданий, монстров, действий, территорий.
- Изменение параметров уже существующих игровых объектов и действий
- Обновление графической части, относящейся к игровым объектам и действиям
- Внесение изменений в сценарии карт
- Создание сценариев карт «с нуля»

Перед чтением руководства разработчика сценариев и моддера крайне желательно ознакомиться с руководством пользователя игры, чтобы понимать основные игровые объекты и действия.
ВАЖНО: Игра не имеет графических редакторов сценариев (за исключением встроенного инструмента, позволяющего выполнять установку территорий и объектов на карту во время игры), а также средств проверки корректности файлов конфигурации и сценариев. Внесение модификаций выполняется путем редактирования файлов в любом текстовом редакторе, поддерживающем кодировку WINDOWS-1251. При неожиданном поведении игры после модификации и невозможности найти ошибку путем просмотра и редактирования, рекомендуется выполнить «откат» изменений путем восстановления исходных файлов.

1.1. Структура каталогов

В таблице приведены назначения каталогов проекта.

Каталог	Назначение
bin	Исполняемые файлы проекта, профиль игрока, логи работы
configs	Конфигурационные файлы, относящиеся ко всем сценариям (пони, здания, монстры, территории, действия и т.д.)
fonts	Используемые игровые шрифты в формате для движка HGE
images	Графика, относящаяся к игре в целом
images\actions	Иконки действий
images\icons	Иконки игровых персонажей (пони и героев)
images\scenes	Тексты-картинки для брифингов
images\terrains	Текстуры территорий
images\units	Текстуры игровых объектов
maps	Файлы сюжетных игровых сценариев, созданных разработчиками игры (карты обучения, карты кампаний, карты Легенд)
scenes	Файлы брифингов
usermaps	Файлы пользовательских сценариев (и брифингов к ним), созданных как разработчиками, так и сторонними пользователями.

Основные каталоги для модификации и создания сценариев — это каталоги configs, images, maps, usermaps.
Каталоги configs и images — относятся ко всем сценариям игры, как встроенным, так и пользовательским.
Каталог maps – содержит встроенные сценарии.

Каталог usermaps – содержит пользовательские сценарии. Каждый сценарий — это файл и при необходимости, каталог с ресурсами сценария. Если сценарий использует только встроенные объекты, то подкаталог с ресурсами не нужен. В противном случае, нужно в каталоге usermaps создать подкаталог с именем файла сценария и расширением .data, в котором создать следующую структуру

Таблица 2 — подкаталоги каталога ресурсов сценария

Подкаталог	Назначение
configs	Конфигурационные файлы, относящиеся к конкретному сценарию (пони, здания, монстры, действия и т.д.)
images\actions	Иконки действий для конкретного сценария
images\icons	Иконки игровых персонажей конкретного сценария
images\units	Текстуры игровых объектов конкретного сценария

В качестве примера организации такой структуры, можно посмотреть пользовательскую карту «Приключения Трикси», входящей в дистрибутив проекта и находящуюся в файле usermaps\trixie_quest (её ресурсы располагаются в соответствующем подкаталоге usermaps\trixie_quest.data).

2. Игровые объекты

Списки и параметры игровых объектов хранятся в каталоге configs, они относятся ко всем сценариям в игре. Так, если изменить параметр запаса сил Рарити с 40 на 50, то Рарити с новыми параметрами будет на всех картах, где она присутствует.
Структура конфигурационных файлов игровых объектов соответствует структуре типового INI-файла (https://ru.wikipedia.org/wiki/.ini), в кодировке WINDOWS-1251.

2.1. Территории

Территории хранятся в файле terrains.ini, каждая секция соответствует одной территории. В секции задаются следующие параметры:
Caption — название территории
Char — буква для обозначения территории в картах
StepNeed — расход очков хода на перемещение пони по территории
Color – цвет территории на карте в формате HTML
IsWater – логическое значение 0/1, является ли территория водой
OnlyFlyers – логическое значение 0/1, является ли территория проходимой только для крылатых пони
Info — описание территории
UseSubLayer — использует ли территория при отрисовке фон, состоящий из окружающей её территории
AllowBuild — логическое значение 0/1, разрешено ли на территории строительство зданий

Текстуры территорий хранятся в каталоге images\terrains

Полный список букв общедоступных территорий и их особенности приведен в таблице ниже:

Буква	**Территория**	**Описание**
L	Луг	Обычный луг, равная скорость для всех пони
F	Лес	Лес, скорость нелетающих пони падает в два раза
M	Горы	Скалистые горы, скорость нелетающих пони падает в четыре раза
A	Вода	Глубокая вода, непроходима для нелетающих пони
Q	Мелководье	Неглубокая вода, скорость нелетающих пони падает в четыре раза
W	Песок или дорога	Зависит от директивы #DefaultSandStone

C	Пещера	Пол пещеры

2.2. Пони

Пони хранятся в файле constants.ini в секции TponyUnit. Секция состоит из параметров, имена которых составлены по принципу «код пони — нижнее подчеркивание — имя параметра пони». Так, дальность атаки Рарити задана строкой.

rarity_AttackLongDist=4

Ниже перечислены все параметры пони:
maxstep – дальность хода пони
maxenergy — запас силы пони
maxhealth — запас здоровья пони
AttackNearValue_min — минимальное значение ближней атаки
AttackNearValue_max — максимальное значение ближней атаки
AttackLongValue_min — минимальное значение дальней атаки
AttackLongValue_max — максимальное значение дальней атаки
AttackLongDist — дальность атаки
isflyer — логическое значение true/false, указание на летающую пони
isnorestored — логическое значение true/false, указание на пони, которая не восстанавливает самостоятельно силу и здоровье
iswaterpony — логическое значение true/false, указание на водную пони (способность перемещения только по воде)

Текстуры пони хранятся в каталоге images\units, все файлы с префиксом pony_
Иконки пони хранятся в каталоге images\icons, файлы, начинающиеся с кода пони и заканчивающиеся на _ico.png

2.3. Противники

Противники хранятся в файле constants.ini в секции TmonsterUnit

(некоторые встроенные в сценарии враги имеют собственные секции). Секция состоит из параметров, имена которых составлены по принципу «код противника — нижнее подчеркивание — имя параметра монстра». Так, дальность хода летучей мыши задана строкой.

bat_maxstep = 5

Ниже перечислены все параметры противников:
attackvalue_min — минимальное значение атаки
attackvalue_max — максимальное значение атаки
maxstep — дальность хода
maxhealth — запас здоровья
isflyer — логическое значение true/false, указание на летающего противника
splash_attack — логическое значение true/false, указание на удар по площади. Если оно false, то параметры удара по площади можно пропустить
splash_radius — радиус удара по площади
splash_attackvalue_min — минимальное значение атаки по площади
splash_attackvalue_max — максимальное значение атаки по площади
ispoison — логическое значение true/false, указание на отравляющее действие противника. Если оно false, то параметры отравления можно пропустить
poisondist — дальность отравления
poison_attackvalue_min — минимальное значение отравления
poison_attackvalue_max — максимальное значение отравления
info — описание противника

Текстуры противников хранятся в каталоге images\units, все файлы с префиксом monster_

2.4. Здания

Здания хранятся в файле buildings.ini, каждая секция соответствует

одному зданию. В секции задаются следующие параметры:

Caption – название здания
Info — описание здания
StrongValue — прочность здания
StepAction — действие здания по завершению хода, при её наличии (справочник действий см. в разделе 4.2). При наличии более одного действия, добавляются параметры StepAction1, StepAction2 и т.д.
PassiveProp — пассивные способности здания, при их наличии (справочник пассивных способностей см. в разделе 4.3). При наличии более одной способности, добавляются параметры PassiveProp1, PassiveProp2 и т.д.
IsReparable – логическое значение true/false, указание на то, является ли здание ремонтируемым.

Текстуры зданий хранятся в каталоге images\units, все файлы с префиксом building_

2.5. Нейтральные объекты

Нейтральные объекты не имеют файла конфигурации, они задаются прямо в файле сценария. Их текстуры хранятся в каталоге images\units, все файлы с префиксом neutral_
Часть имени файла после нижнего подчеркивания и до расширения файла является кодом нейтрального объекта для указания в сценарии.

2.6. Ресурсы

Ресурсы жестко встроены в программный код. Всё, что можно сделать с ними — это изменить текстуры в файлах каталога images\units с префиксом resource_ и поменять номиналы размеров ресурсных источников в файле constants.ini (секции TstoneResource и TfoodResource).

2.7. Заклинания и действия

Действия пони и заклинания хранятся в файле actions.ini, каждая секция соответствует одному действию или заклинанию. В секции задаются следующие параметры:

Caption – название действия.
SubInfo — описание действия.
AllowCells — параметры ячеек области выбора
DestCells — параметры ячеек области действия
Energy — затраты сил пони (может быть пропущено)
Stone — затраты камня (может быть пропущено)
Food — затраты еды (может быть пропущено)
MetaAction — метадействие, которое будет выполнено на каждой клетке в области действия (справочник метадействий см. в разделе 4.1). При наличии более одного метадействия, добавляются параметры MetaAction1, MetaAction2 и т.д.

Иконки действий хранятся в каталоге images\actions.

Привязка действий к пони осуществляется с помощью правки файла ponyactions.ini в каталоге configs. Каждая секция файла содержит действия, разрешенные для пони, а имя секции — это код пони.
Наконец, файл ui.ini содержит параметры действий — их расположение в окне выбора действий. Имя параметра — код действия, а значение — 1,2 или 3 — отвечает за колонку, где будет находиться действие в интерфейсе.

Параметры области выбора

Позволяет задать ячейки, при наведении на которые, заклинание можно применить.

Возможные значения и описание настройки:

linked - заклинание можно применять в ячейки вокруг пони. Пример - сбор ресурсов.
round,R - заклинание можно применять в ячейки, находящиеся на расстоянии не более R от пони, где R - целое число. Пример - заморозка Рарити.
circle,R1,R2 - заклинание можно применять в ячейки, находящиеся далее R1 от пони, но не более R2, где R1 и R2 - целые числа. Пример - огненный шар Селестии.
radial,R - заклинание можно применять по радиальным лучам длиной R. Пример - рассекание Пустоты Рарити.

Параметры области действий

Позволяет задать ячейки, на которые будет применяться заклинание

Возможные значения и описание настройки:
point - заклинание действует только на указанную ячейку. Пример - сбор ресурсов.
linked - заклинание действует на указанную ячейку и ячейки вокруг неё. Пример - Заморозка Рарити.
round,R - заклинание действует на ячейки, находящиеся на расстоянии не более R от пони, где R - целое число. Пример - Радужный Удар Рейнбоу.
alg,программа - строит ячейки применения заклинания с помощью программы из команд, ориентируясь на выбранное направление действия заклинания, начиная с ячейки действующей пони. Подробнее, см ниже.
map,программа - строит ячейки применения заклинания с помощью программы из команд, безотносительно к выбранному направлению действия заклинания, начиная с указанной ячейки. Подробнее, см ниже.

Описание кода программ для процедур построения областей alg и map

Для указания сложных областей действия заклинаний предусмотрен специальный интерпретатор, позволяющий построить произвольные области.
Программа состоит из цепочек целых чисел от -5 до 5, символов x, круглых скобок. Также программа может начинаться с символа i, который указывает, что стартовая точка включается в область действия.

Интерпретатор работает по следующему алгоритму:

1. Устанавливает в качестве стартовой ячейку пони для области alg или ячейку выбора для области map.
2. Определяет стартовое направление построения (для alg определяется выбранным радиальным направлением, для map это всегда направление «вправо по карте»).
3. Берет первое число в цепочке и добавляет в область ячейку, следующую по стартовому направлению с поворотом по часовой стрелке на полученное число (0 - поворот не происходит, 1 - поворот на одну клетку и т.д.), после чего делает эту ячейку текущей.
4. Если перед числом стоит символ x, то ячейка становится текущей, но не добавляется в итоговый набор.
5. Если перед числом стоит открывающая скобка, то программа начинает новую ветку ячеек, начиная с указанной.

Примеры простых программ:

Линия на пять клеток - 00000
Вилка начиная с третьей клетки - 000(100)(-100)
Пунктирная линия - 0x00x00

3. Структура сценариев

Сценарий — это отдельный файл в каталоге maps или usermaps, имеющий определенный формат и содержащий последовательность секций, описывающих конкретную карту с её объектами, начальными

значениями, скриптами и условием победы. Структура файла похожа на структуру Ini-файла, со следующими отличиями и условиями:

- Названия секций идут без фигурных скобок
- Имена ключей отделяются от значений знаком «:»
- Некоторые имена секций могут повторяться
- Некоторые имена ключей в секциях могут повторяться
- Строки, начинающиеся со знака «#», считаются комментариями
- Кодировка файла — WINDOWS-1251

Также файл может содержать пустые строки и некоторые специальные директивы, начинающиеся со знака «#»

#DefaultSandStone

Данная директива указывает игре выводить вместо территории дорог — текстуру песка (подробнее в разделе 3.1)

#===MAPEND===

Данная директива обозначает конец секции территорий, и нужна только для работы встроенного редактора карт (подробнее в разделе 7)

#===OBJECTSEND===

Данная директива обозначает конец секции объектов, и нужна только для работы встроенного редактора карт (подробнее в разделе 7)

3.1. Территория

Секция территории называется MAP и состоит из набора строк символов, обозначающих букв территорий (см. раздел 2.1). На строки наложены следующие ограничения:

- Общее количество строк должно быть четным
- Количество букв во всех четных строках должно совпадать.

- Количество букв во всех нечетных строках должно совпадать.
- Число букв в нечетных строках (нумерация начинается с 1) должно быть на единицу меньше, чем в четных.

Пример корректной секции MAP (просто поле с территорией Луг)

```
MAP
LLLLLLL
LLLLLLLL
LLLLLLL
LLLLLLLL
LLLLLLL
LLLLLLLL
#===MAPEND===
```

3.2. Константы

Секция констант называется CONSTANTS, должна идти сразу после секции MAP, она содержит именованные константы, которые будут заменены в файле карты на их значения. Это позволяет выносить числовые значения или строки в начало карты, и избегать появления «магических чисел».
В секции идет список констант со значениями, значение от названия отделяется двоеточием, например:

```
PonyX:10
PonyY:15
PonyText:Добрая лошадь.
```

В дальнейшем сценарии, чтобы указать на константу, достаточно написать имя константы между двух знаков «@».
Так, строку вида

```
Pony:i=@PonyX@,j=@PonyY@,"name=@PonyText@",code=sg
```

при наличии в секции констант записей из примера выше, интерпретатор сценария преобразует в

```
Pony:i=10,j=15,"name=Добрая лошадь",code=sg
```

3.3. Начальные установки

Секция начальных установок называется INITIALS. В ней указываются начальные состояния для разных параметров сценария, которые устанавливаются после загрузки. Некоторые из этих параметров меняются во время прохождения сценария, некоторые — нет. Значения параметра отделяются от имени двоеточием, например
Stone:100
Food:200

Ниже перечислены все параметры, которые доступны в начальных установках.

Stone – количество камня

Food – количество еды

Task – текст задания (устарело, вместо него лучше использовать скрипт установки задания)

EmptyManager – список шести скоростей движения Пустоты, разделенных запятыми. Каждая скорость имеет значение от 0 до 100, и задает скорость движения по направлениям шестигранников. Значение 0 означает, что Пустота не движется, а 100 — что Пустота движется максимально быстро.
Пример строки
EmptyManager:40,40,40,40,40,40

DeathAllow — разрешить гибель пони на карте без поражения в сценарии, может принимать значение true или false. По умолчанию гибель пони приводит к поражению.

NoSpaceProtection — отключить всю пассивную защиту от Пустоты, может принимать значение true или false. По умолчанию пассивная

защита включена (например, дерево Эпплджек или аура Флаттершай).

UseFog — включение «тумана войны» в стиле WarCraft2, может принимать значение true или false. По умолчанию туман войны отключен (карты кампаний и легенд не рассчитаны на использование тумана войны).

DefaultMonsterStrategy – устанавливает стратегию действий монстров на всей карте по умолчанию (может быть переопределено для отдельных монстров). Возможные значения:
AttackNearFirst — монстры в первую очередь выбирают в качестве целей объекты наиболее близкие к ним на карте.
AttackByRank – монстры в первую очередь выбирают в качестве целей объекты с наибольшим рангом (ранг объекта задаётся при его создании скриптом или в начальных объектах карты).
По умолчанию, если стратегия не задана, то монстры выбирают случайную цель для атаки.

AssignAction – позволяет назначить пони новое действие только для данного сценария (основные действия пони назначены в файле ponyactions.ini). Его значение — строка
pony=код_пони,action=код_действия
где код_пони и код_действия — те пони и действия, которым выполняется назначения (код пони — имя пони из файла constants.ini, а код действия указывает на имя секции в файле actions.ini)
Например, следующая строка назначает для пони с кодом sg действие с кодом Burn.
AssignAction:pony=sg,action=Burn
Назначение действия не выполняет автоматического разрешения действия, этим занимается следующий параметр.

Permits — установка разрешения или запрещения действий пони для сценария. Этих строк может быть много.
Значение параметра — строка вида

action=разрешение,code=код_действия
где разрешение — это строка allow или deny (для разрешения или запрещения действия), а код_действия указывает на имя секции в файле actions.ini
По умолчанию, действия разрешены, чтобы запретить или разрешить все действия, используют код действия all.
Пример: следующие строки в секции запретят все действия, после чего разрешат действие Harvest
Permits:action=deny,code=all
Permits:action=allow,code=Harvest
Важно — действие разрешается сразу у всех пони, у которых это действие было указано в файле ponyactions.ini или параметром AssignAction

3.4. Объекты

Секция объектов называется OBJECTS. В ней указываются те объекты, которые устанавливаются после загрузки карты. Каждая строка начинается с класса объекта, после двоеточия идет список параметров объекта. Возможны следующие классы объектов:
Building — здания
Pony – пони
Neutral – нейтральные объекты
Monster – монстры
Stone — куча камней
Food – бочка еды
Для каждого объекта обязательными являются параметры i и j, которые задают положение объекта на карте. Также для каждого объекта, кроме еды и камней, обязательным является параметр код объекта (code), по которому определяются его базовые характеристики и спрайт. Остальные параметры зависят от класса объекта. Некоторые параметры являются обязательными.
Пример установки объекта «камень» на карте:
Stone:i=2,j=9,size=medium
Пример установки объекта «пони» на карте:

Pony:i=1,j=4,"name=Рарити",code=rarity

Ниже приведены параметры, которые общие для всех объектов, но не являются обязательными
targeted – если установлен в true, то вокруг объекта отображается подкова-указатель цели.
tag – особая произвольная метка, позволяет выделить объект среди прочих для использования в скриптах
Rank – число, которое используется для определения цели монстров, если у них задана стратегия выбора цели по рангу. По умолчанию, ранг всех объектов нулевой.
nomapping – если установлен в true, то объект не отображается на мини-карте.

Ниже приведены остальные параметры, специфичные для разных классов объектов:

Классы Food и Stone
size – обязательный параметр, размер кучи камней, возможные значения: min, medium, max.
exactcount — если он задан, то задаёт точное значение размера ресурса

Класс Neutral
Name – обязательный параметр, имя объекта, отображаемое в сценарии.
EnemyTarget — если параметр установлен в true, то данный нейтральный объект будет целью атак для монстров и может быть разрушен.
MustSurvive — если параметр установлен в true, то при атаке монстра на данный объект, сценарий завершается поражением.
Removable — если параметр установлен в true, то при атаке монстра на данный объект, он удаляется с карты, но сценарий продолжается.

Класс Building

MustSurvive — если установлен в true, то при разрушении здания, сработает поражение.

Класс Pony

Name – обязательный параметр, имя пони, отображаемое в сценарии
Health — установка точного значения здоровья пони
Energy – установка точного значения сил пони
disableflying — если установлено в true, то запрещает пони летать
SpriteTag — позволяет указать для пони собственный файл изображения, отличный от того, который привязан к коду пони. Имя указывается без расширения, относительно каталога images.
IconTag — позволяет указать для пони собственный файл иконки, отличный от того, который привязан к коду пони. Имя указывается без расширения, относительно каталога images.
DeathAllow – если установлен в true, то данная пони может погибнуть в сценарии, и поражение не сработает.

Класс Monster

Name – обязательный параметр, имя монстра, отображаемое в сценарии
holdground – если установлен в true, то монстры не двигаются к целям, но всё еще атакуют, если пони или здание находится на расстоянии их хода.
passive – если установлен в true, то монстры не только не двигаются, но и не атакуют, даже если подойти к ним вплотную.
MustSurvive — если установлен в true, то при гибели монстра, сработает поражение.
strategy – если задано, то монстр имеет специальную стратегию выбора целей (типы стратегий см. в разделе 3.3)

3.5. Скрипты

Секция скриптов называется SCRIPT. В ней указываются события скрипта и действия, совершаемые при наступлении события. Каждый скрипт помещен в отдельную секцию, таким образом, число секций

SCRIPT равно числу скриптов в сценарии. Событий и действий может быть несколько внутри одной секции, при этом, для срабатывания скрипта, нужно одновременное наступление всех событий. Если не указано иное в параметрах события, скрипты являются одноразовыми — то есть, после его выполнения, данный скрипт больше никогда не будет выполнен.

События начинаются с выражения Event, после двоеточия идет формулировка события, включающая в себя тип события и дополнительные параметры. Типы событий с их параметрами указаны в разделе 4.4
Пример событий, которое срабатывает при наступлении хода с номером 1 (то есть, второй ход после начала игры)
Event:Step=1
Для указания более одного события, нужно дополнять следующие события числом от 1 и далее. Например, если для скрипта нужны три события, они будут записаны так:
Event:Step=1
Event1:Flag=woodwolf
Event2:MonsterCount=0

Действия начинаются с выражения Action и содержат после двоеточия строку с параметрами действия. Строка обязательно включает в себя тип действия (параметр Type), остальные параметры зависят от типа действия. Типы действий с их параметрами указаны в разделе 4.5
Пример действия на вывод сообщения, с указанием типа действия Message и дополнительными параметрами:
Action:Type=Message,icon=rarity_ico,"text=Всем чмоки в этом чате!"

3.6. Условие победы

Секция условий победы называется VICTORY. В ней указываются условия победы и их параметры. Строки состоят из типа условия победы, после двоеточия идут параметры условия.

Пример условия победы по достижению хода с номером 1
ByStep:StepNeed=1
При добавлении более одного условия победы, необходимо добавить ко второй и дальнейшим строкам префикс “->” и добавить в параметры вторых и дальнейших условий параметр AddType со значением либо And, либо Or, в зависимости от того, достаточно ли одного условия, или нужны все условия.
Пример записи для двух условий победы, необходимых оба.
ByStep:StepNeed=1
->ByObjectPos:Object=applejack,i=20,j=8,AddType=And

4. Справочники

4.1. Метадействия

Метадействия — это действия, которые выполняются при применение действия по области. Они указывают в файле действий для каждого действия, в количестве от одного и более.

Ниже приведен полный список поддерживаемых игрой метадействий и списки параметров для них.

Harvest - сбор ресурса (еды или камня).
Параметры не требуются

Damage - наносит урон врагам
MinValue - минимальный урон, целое число, обязательный параметр.
MaxValue - максимальный урон, целое число, обязательный параметр.
Absolute - маркер абсолютного удара (уничтожение любого противника одним применением) логическое значение (true/false), необязательный параметр.
Divide - значение, указывающее, на сколько процентов от текущего значения здоровья наносится удар (например, значение 50 означает, что здоровье врага будет уменьшено в 2 раза), необязательный

параметр. Если он не указан, урон вычисляется исходя из значений MinValue и MaxValue.

Build - возводит здание
BuildCode - класс здания, строка, обязательный параметр.
AnyTerrain - маркер, указывающий разрешение строить на любом рельефе (по умолчанию, разрешена постройка только на лугу и в лесу), логическое значение (true/false), необязательный параметр.

Freeze - заморозка врагов
FreezeTime - длительность заморозки в ходах, целое число, обязательный параметр.

Teleport - прыжок единицы на расстояние
ImmediateTeleport - маркер, указывающий на необходимость мгновенного перемещения (по умолчанию, телепортация выполняется 1 секунду с графическими эффектами), логическое значение (true/false), необязательный параметр.

MineStone - добыча камня из стен
MineValue - число единиц камня, добываемое с одной ячейки стены, целое число, обязательный параметр.

MineForest - добыча древесины из леса
MineValue - число единиц леса, добываемое с одной ячейки леса, целое число, обязательный параметр

CutEmpty - очистка Пустоты
Параметры не требуются

CutEmptyAll - очистка Пустоты полная (по всей карте, вне зависимости от ячейки применения)
Параметры не требуются

Heal - лечение пони в зоне действия

MinValue - минимальное увеличение здоровья, целое число, обязательный параметр.
MaxValue - максимальное увеличение здоровья, целое число, обязательный параметр.
Absolute - маркер абсолютного лечения (полное восстановление здоровья одним применением) логическое значение (true/false), необязательный параметр.

Restore - восстановление сил пони в зоне действия
MinValue - минимальное увеличение силы, целое число, обязательный параметр.
MaxValue - максимальное увеличение силы, целое число, обязательный параметр.
Absolute - маркер абсолютного восстановления (полное восстановление силы одним применением) логическое значение (true/false), необязательный параметр.

Consume - поглощение объекта с возвратом силы или здоровья
Object - класс объекта, пригодного для поглощения, строка, обязательный параметр
HealValue - увеличение здоровья, целое число, обязательный параметр.
RestoreValue - увеличение силы, целое число, обязательный параметр.
Absolute - маркер абсолютного восстановления (полное восстановление силы или здоровья одним применением) логическое значение (true/false), необязательный параметр.

Sleep - усыпление врагов
SleepTime - длительность усыпления в ходах, целое число, обязательный параметр.

EnemyTransform - превращение монстров
MonsterCode - код нового монстра, строка, обязательный параметр.
Name - название нового монстра

Wings - выдача крыльев для пони
WingTime - длительность сохранения крыльев в ходах, целое число, обязательный параметр.

Summone - призыв пони на карту
Pony - имя пони из конфига, строка, обязательный параметр.
NoCheckCount- маркер снятия проверки на количество (по умолчанию, нельзя призвать более одной пони одного класса на карту) логическое значение (true/false), необязательный параметр.

Burn - поджигание врагов, они получают постоянный урон каждый ход
Параметры не требуются

DownEnemy - временное снижение показателей здоровья и атаки противника
DownTime - длительность снижения в ходах, целое число, обязательный параметр.
Divide - процент снижения показателей, целое число, обязательный параметр. Например, Divide=50 снижает показатели врага вдвое.

SetPonyParam - установка текущих (не максимальных!) показателей здоровья или силы пони
ParamName - название параметра, строка, равная Health (для изменения здоровья) или Energy (для изменения силы), обязательный параметр.
ParamValue - значение параметра, целое число, обязательный параметр.

SetShield - установка силового щита на пони, поглощающего часть урона
ShieldTime - длительность действия щита в ходах, целое число, обязательный параметр.
ShieldValue – эффект от щита, целое число, 0 — отсутствует эффект,

100 — абсолютное поглощение урона. По умолчанию, равно 50.

Repair — ремонт зданий
Value — на сколько увеличится запас прочности здания, целое число, обязательный параметр.
Absolute - маркер абсолютного ремонта (полное восстановление прочности одним применением) логическое значение (true/false), необязательный параметр.

4.2. Действия смены хода

Действия смены хода выполняются в зданиях, когда игрок завершает свой ход. Они указываются в файле настройки зданий.

Ниже приведен полный список поддерживаемых игрой действий смены хода и списки параметров для них.

Damage — урон по монстрам от здания
MinValue - минимальное значение урона, целое число, обязательный параметр.
MaxValue - максимальное значение урона, целое число, обязательный параметр.
Absolute - маркер абсолютного урона (уничтожение одним применением) логическое значение (true/false), необязательный параметр.
Dist — расстояние, на котором до монстра достаёт урон здания, целое число, обязательный параметр.
Target — число монстров, которые получают урон от одного здания, целое число, необязательный параметр (по умолчанию 1).

Destroy — уничтожение здания
StepLeft — через сколько ходов зданий должно разрушиться, целое число, обязательный параметр.

Heal — лечение пони в окрестности здания

MinValue - минимальное значение лечения, целое число, обязательный параметр.
MaxValue - максимальное значение лечения, целое число, обязательный параметр.
Absolute - маркер абсолютного лечения (восстановление здоровья одним применением) логическое значение (true/false), необязательный параметр.
Dist — расстояние, на котором до пони достаёт лечение здания, целое число, обязательный параметр.
Target — число пони, которые получают лечение от одного здания, целое число, необязательный параметр (по умолчанию 1).

Restore — восстановление сил пони в окрестности здания
MinValue - минимальное значение восстановления, целое число, обязательный параметр.
MaxValue - максимальное значение восстановления, целое число, обязательный параметр.
Absolute - маркер абсолютного лечения (восстановление здоровья одним применением) логическое значение (true/false), необязательный параметр.
Dist — расстояние, на котором до пони достаёт лечение здания, целое число, обязательный параметр.
Target — число пони, которые получают лечение от одного здания, целое число, необязательный параметр (по умолчанию 1).

Produce — производство ресурсов
Value — количество ресурса, производимого за ход, целое число, обязательный параметр.
Resource — тип производимого ресурса, строка (Stone, Food или Wood), обязательный параметр.
NoNeedHouse - маркер означает, что для производства ресурсов, не требуется наличие домой пони на карте, логическое значение (true/false), необязательный параметр.

4.3. Пассивные способности

Пассивные способности у зданий позволяют им обладать какими-то особыми свойствами, они функционируют постоянно, влияя на игровой процесс. Пассивные способности указываются в файле настройки зданий.

Ниже приведен полный список поддерживаемых игрой пассивных способностей зданий и списки параметров для них.

EnemySlowdown — замедление врагов
Dist — зона действия способности, целое число, обязательный параметр.

IDDQD — неуязвимость здания
Параметры не требуются

NoTarget — здание не является целью атаки монстров
Параметры не требуются

SpaceProtect — защита от распространения Пустоты
Dist — зона действия способности, целое число, обязательный параметр.

4.4. События скрипта

События скрипта — это условия, которые должны наступить для того, чтобы скрипт сработал. Код события указывается сразу после ключевого слова Event, после знака равенства идет значение параметра, а также, при необходимости, другие параметры события, в виде пар «имя=значение», разделенных запятыми.

Пример для действия наступления хода (код Step):
Event:Step=2
Это означает, что действие выполнится при начале второго хода игрока (отсчет ходов идет от нуля)

Ниже приведен полный список поддерживаемых игрой кодов событий и списки параметров для них.

Step — наступление хода игрока. Значение параметра — номер хода, где 0 — это первый ход игры. Фактически, при Step=0 скрипт запускается сразу после открытия сценария.

StepVar – наступление хода, записанного в переменной. Значение параметра — имя переменной. Подробнее про эту механику см. в разделе 4.5, действие SetVar.

Flag – появление установленного флага. Значение параметра — имя флага.
Дополнительные параметры:
CheckNoFlag — если он установлен в true, то событие наступит, если флаг не установлен. Это имеет смысл использовать, если условие скрипта сочетается с другими условиями, например, если на момент начала номера хода не установлен флаг.

StepModN — наступление хода, номер которого кратен параметру. Используется для постоянной генерации событий, например, каждый третий ход.
Дополнительные параметры:
Great — если задан, то выполняется если номер хода больше указанного параметра. Например, при параметре Great=4 будет выполняться, если номер хода не только кратен заданному, но и больше 4.
Less — аналогично, если задан, то выполняется если номер хода меньше указанного параметра.

NeutralCount — число нейтральных объектов становится равно указанному в параметре.

BuildingCount — число зданий с конкретным кодом становится

равным указанному в параметре.
Дополнительные параметры:
BuildingCode — код здания, обязателен к указанию. Например, скрипт Event:BuildingType=2,BuildingCode=FarmFood выполнится, если число ферм еды станет равно 2.

MonsterCount — число монстров становится равно указанному в параметре.

SpaceCount — число ячеек, занятых Пустотой, становится равно указанному в параметре.

TagCount — число объектов с заданным тэгом становится равно указанному в параметре.
Дополнительные параметры:
Compare — позволяет задать неравенство. Значение LessOrEqual — сработает, если будет меньше или равно. Значение GreatOrEqual — сработает, если будет больше или равно. Если не задан, то идет точное равенство.
TagName — название тэга. Обязательно к указанию. Пример использования:
Event:TagCount=5,Compare=GreatOrEqual,TagName=staff будет выполнен, когда число объектов с тэгом staff будет больше или равно 5.

Victory — срабатывает сразу после победы. Указывается без параметров, вот так — Event:Victory=True.

StoneReach — запасы камня становятся равны указанному в параметре.

FoodReach — запасы еды становятся равны указанному в параметре.

WoodReach — запасы дерева становятся равны указанному в параметре (добыча еды идет метадействием MineForest)

ObjectPos — указанный объект сравнивается с заданными координатами. Параметр — значение True или False, означает, что сработает при наличии или отсутствии условия.
Дополнительные параметры:
I — координата ячейки по горизонтали, обязательный параметр.
J — координата ячейки по вертикали обязательный параметр.
Object — код объекта. Он может быть кодом здания, пони, монстра, а также тэгом. Проверка идет по всем вариантам сразу.
IncludeLinked — включать ли рядом расположенные клетки, если указан true, то включаются, по умолчанию, только точное значение.
Direction — схема сравнения. По умолчанию, только точное соответствие. Значение GreatX — если горизонтальная координата превысит параметр I, GreatY – если вертикальная координата превысит параметр I, LessX — если горизонтальная координата станет меньше параметра I, LessY – если вертикальная координата станет меньше параметра I.

Дополнительные настройки событий:
Для каждого события, может быть указан дополнительный параметр ForceNoDel=true, который означает, что после выполнения скрипта, он не будет удален из памяти сценария и снова будет выполнен, если наступит указанное условие. Поскольку скрипты выполняются непрерывно, то установив такой параметр, выполнение чаще всего будет «зациклено». Чтобы избежать этого, можно указать дополнительный параметр ForceOnceAtStep=true, который означает, что условие будет выполняться только раз за ход, либо делать в скрипте действие, которое немедленно нарушит указанное условие (перемещение пони, уменьшение ресурсов и т.д.) и прервет следующее выполнение.

4.5. Действия скрипта

Каждый скрипт содержит как минимум одно действие, состоящие из типа действия и его параметров. Параметры перечисляются через

запятую, имя параметра от значения отделяется знаком «равно», если значения содержит пробелы, то параметр вместе со значением, заключаются в кавычки.
Пример записи действия:
Action:Type=Message,icon=rarity_ico,"text=Всем чмоки в этом чате!"

Ниже перечислены коды действий и описание их параметров.

Message — вывод сообщения.
Параметры:
icon – имя файла из каталога images/icons без расширения.
text – текст сообщения.

ShowBattleTask — вывести окно текущих задач сценария
Параметров нет

SetBattleTask — добавить задачу сценария.
Параметры:
Code – код задачи
isreq – если установлен в false, то задание является необязательным
Task – текст задачи

CompBattleTask — установить задачу как выполненную
Параметры:
Code – код задачи (если относится к задаче, установленной в начальной секции, то код можно пропустить)

FailBattleTask — установить задачу как проваленную
Параметры:
Code – код задачи (если относится к задаче, установленной в начальной секции, то код можно пропустить)

NewObject — создать объект на карте
Параметры:
Object – код класса объекта. Возможные коды:

Pony,Monster,Building,Neutral,Stone,Food,EmptyPlace (для Пустоты).
i – горизонтальная координата объекта
j – вертикальная координата объекта
Все остальные параметры зависят от класса объекта, их список можно получить в разделе 3.4.

Пример добавления на карту пони Рарити в позицию (10;5)
Action:Type=NewObject,Object=Pony,i=10,j=5,"name=Рарити",code=rarity

ReplaceObject – заменить объект на карте
Параметры:
OldCode – код существующего объекта. Для пони, монстров и нейтральных объектов это их код из файлов конфигураций (не имена!).
Object – код класса нового объекта. Возможные коды: Pony,Monster,Building,Neutral,Stone,Food,EmptyPlace (для Пустоты).
Все остальные параметры зависят от класса нового объекта, их список можно получить в разделе 3.4.

Координаты в этом скрипте не передаются — они берутся из старого объекта.

Пример замены на карте пони Рарити на Найтмер Рарити
Action:Type=ReplaceObject,OldCode=rarity,Object=Pony,"name=Найтмер Рарити",code=nrarity

NewObjectGroup — создать несколько объектов на карте
Параметры:
Object – код класса объекта. Возможные коды: Pony,Monster,Building,Neutral,Stone,Food,EmptyPlace (для Пустоты).
i – горизонтальная координата зоны добавления объектов
j – вертикальная координата зоны добавления объектов
radius – радиус зоны в клетках
count – количество объектов

Все остальные параметры зависят от класса объекта, их список можно получить в разделе 3.4.

Пример добавления на карту трех больших куч камней вокруг позиции (10;5)
Action:Type=NewObject,Object=Stone,i=10,j=5,size=max,radius=1,count=3

MoveObject — переместить объект
Параметры:
method – указывает, как определить перемещаемый объект. Если значение равно tag, то использует параметр Tag, иначе параметры I,j
Tag – тэг перемещаемого объекта
i – горизонтальная координата объекта
j – вертикальная координата объекта
immediate — если задано как true, то выполняет немедленное перемещение, иначе объект движется к целевой точке по клеткам.
dst_i – горизонтальная координата нового положения объекта
dst_j – вертикальная координата нового положения объекта

TeleportObject — переместить объект через телепортацию (то же, что и MoveObject, но всегда мгновенно и с эффектом прыжка)
Параметры:
method – указывает, как определить телепортируемый объект. Если значение равно tag, то использует параметр Tag, иначе параметры I,j
Tag – тэг телепортируемого объекта
i – горизонтальная координата объекта
j – вертикальная координата объекта
dst_i – горизонтальная координата нового положения объекта
dst_j – вертикальная координата нового положения объекта

RemoveObject — удалить объект
Параметры:
method – указывает, как определить удаляемый объект. Если значение равно tag, то использует параметр Tag, иначе параметры I,j

Tag – тэг удаляемого объекта
i – горизонтальная координата объекта
j – вертикальная координата объекта

SetFocus — установка камеры на заданную клетку
Параметры:
i – горизонтальная координата клетки
j – вертикальная координата клетки
immediate — если задано как false, то выполняет пролет камеры до точки, иначе немедленно перемещает камеру.

SetEmptyManager — установка скорости распространения Пустоты
Параметры:
AnyDir – скорость от 0 до 100, где 100 — максимальная скорость.

SetPermits — установить разрешения на действия пони
Параметры:
code – код действия
action – значение allow для разрешения или deny для запрета.

Пример:
Запретить ближнюю атаку
Action:Type=SetPermits,action=deny,code=AttackNear

ClearTarget — очистить указатель цели с клетки
Параметры:
i – горизонтальная координата клетки
j – вертикальная координата клетки

SetFlag – установить флаг
Параметры:
FlagName – название флага

ClearFlag – сбросить флаг
Параметры:

FlagName – название флага

SetTerrain — задать тип территории
Параметры:
i – горизонтальная координата клетки
j – вертикальная координата клетки
NewTerrCode — код территории

SetDefeat — немедленно завершить сценарий поражением
Параметры:
DefeatStr — причина поражения

DecStone – уменьшить запасы камня
Параметры:
Delta – величина уменьшения

DecFood – уменьшить запасы еды
Параметры:
Delta – величина уменьшения

SetVar – установить скриптовую переменную
Параметры:
Initial – если установлено в CurrentStep, то в качестве начального значения, идет номер текущего хода, иначе 0
Delta – величина, на которую увеличится значение переменной.

Пример: установить переменную MonsterComing на +5 ходов от текущего.
Action:Type=SetVar,Name=MonsterComing,Initial=CurrentStep,Delta=5

FireTotalAttack – запустить атаку всех монстров на карте (даже тех, кто в пассивном режиме)
Параметров нет

4.6. Условия победы

Условия победы записываются в секции VICTORY, указывается код условия и после двоеточия — набор параметров, имя и значения, разделенные запятыми.

Пример события победы по достижению 100 единиц камня
ByStone:StoneNeed=100

Ниже перечислены коды условий победы и описание их параметров.

ByStone — собрано количество камня
Параметры:
StoneNeed — количество камня для победы

ByFood — собрано количество еды
Параметры:
FoodNeed — количество еды для победы

ByStep — наступил ход игры заданного номера
Параметры:
StepNeed — номер хода

ByObjectPos – объект заданного кода достиг клетки
Параметры:
Object – код объекта (коды пони, монстра, нейтрального объекта)
i – горизонтальная координата клетки
j – вертикальная координата клетки
IncludeLinked — если задан в true, то в условие включаются расположенные рядом клетки.

NoTargetLeft — на карте не осталось ни одного объекта, помеченного как цель
Параметров нет

ByFlag – установлен флаг

Параметры:
FlagName — имя флага

NoMonsterLeft — на карте не осталось ни одного монстра
Параметров нет

NoSpaceLeft — на карте не осталось ни одной клетки Пустоты
Параметров нет

ByObjectCount — количество объектов на карте сравнивается с заданным.
Параметры:
Object — код объекта (коды пони, монстров, зданий, нейтральных объектов)
cnt — количество.
Если количество начинается с буквы g, то значение должно быть больше или равно указанному. Если количество начинается с буквы l, то значение должно быть меньше или равно указанному.

Пример:
Победа достигается, когда число ферм стало равно 3 или более.
ByObjectCount:Object=FarmFood,cnt=g3

5. Особенности обработки скриптов

Скрипты можно группировать так, что в одно условие будут входить сразу несколько действий.
Например, конструкция ниже означает, что при наступлении хода 1, будут реплики Рарити и Эпплджек.

```
SCRIPT
Event:Step=0
Action:Type=Message,icon=rarity_ico,"text=Привет!"
Action:Type=Message,icon=applejack_ico,"text=Пока!"
```

Особенность в том, что данная конструкция транслируется внутри программы как независимые скрипты с дублированием условий, вот так:

```
SCRIPT
Event:Step=0
Action:Type=Message,icon=rarity_ico,"text=Привет!"

SCRIPT
Event:Step=0
Action:Type=Message,icon=applejack_ico,"text=Пока!"
```

Если в процессе выполнения скрипта случится так, что условие будет нарушено, то все следующие строки Action не будут выполнены. Пример такого скрипта ниже

```
SCRIPT
Event:StoneReach=100
Action:Type=DecStone,Delta=50
Action:Type=Message,icon=twily_ico,"text=Камень собран и порция отправлена!"
```

Скрипт срабатывает, когда камня становится больше или равно 100, но на первом действии камень вычитается, и второе действие (реплика Твайлайт) не будет выполнена.

Существуют два пути решения этой проблемы.

1. Отправка изменяющих действий в конец скрипта
Если переписать скрипт так, чтобы действие списания было в конце, то сначала будет реплика, а потом камень уменьшится

```
SCRIPT
Event:StoneReach=100
Action:Type=Message,icon=twily_ico,"text=Камень собран и порция
```

отправлена!"
Action:Type=DecStone,Delta=50

2. Использование флагов
По достижению условия, скрипт устанавливает флаг, а основные действия выполняются уже по флагу, который после его установки не зависит от исходного условия.

SCRIPT
Event:StoneReach=100
Action:Type=SetFlag,FlagName=stonecollected

SCRIPT
Event:Flag=stonecollected
Action:Type=DecStone,Delta=50
Action:Type=Message,icon=twily_ico,"text=Камень собран и порция отправлена!"

6. Встроенный редактор карт

Игра предоставляет ограниченную возможность редактировать карту. Для входа в режим редактора, нужно запустить игру, открыть сценарий и нажать F4. Появится надпись «Редактор», напротив которой будут числа, означающие координаты клетки, над которой находится курсор, а также текущий режим установки (территории или объекты).
Чтобы сделать доступными в режиме редактора установку территорий и объектов, нужно открыть файл configs/editor.ini и добавить в него строки, начинающиеся с английской буквы или цифры, после которой, отделенная от неё знаком «=», идет строка, соответствующая описанию объекта в секции OBJECTS, либо слово Terrain с кодом после.

Пример — такое содержимое в editor.ini назначает на букву Q ферму камня, на цифру 1 — ферму еды, на букву W – территорию луг, а на

цифру 2 — территорию воду.

Q=Building:Code=FarmStone,"Name=Ферма камня"
1=Building:Code=FarmFood,"Name=Ферма еды"
W=Terrain:Code=L
2=Terrain:Code=A

После этого, при запуске игры и входе в редактор, можно не только выбирать рельеф, но и устанавливать объекты, нажимая буквы или цифры над указанной мышкой клеткой. Чтобы переключиться между установкой территорий и рельефа, нужно нажать кнопку TAB.

После завершения редактирования, нужно нажать F2. Файл сценария будет перезаписан.