Остальные команды

- ach
- backdrop
- chibi
- creditsmode
- load
- menu
- overlay_color
- pause
- posteffect
- rollcredits
- sequence и trigger
- unlock_ending
- with

ach

Разблокирует определенное достижение в Steam.

Синтаксис

```
ach [id]
```

Параметры

- `id` — ID достижения.
  Обязательный параметр.

Пример использования

```
# разблокирует достижение "First Step"
ach firststep
```

backdrop

Выводит текст поверх определённого набора фонов с затемнением. Используется в основном при переходах между главами сценария.

Синтаксис

```
backdrop |xml| (text) {duration} {bg}
```

Параметры

- `xml` — флаг. Помечает, что значение параметра `text` — это ID заранее заданного текста для команды backdrop.
  Опциональный параметр.

- `text` — текст сноски. Если перед текстом указан флаг `xml`, то будет восприниматься как ID заранее определённого текста, иначе будет выведен заданный текст.
  Опциональный параметр.

- `duration` — продолжительность вывода в секундах.
  Опциональный параметр.
  Значение по умолчанию: `5`.

- `bg` — ключевой параметр, после него указывается ID backdrop фона.
  Опциональный параметр.
  Значение по умолчанию: `default`.

Примеры использования

```
# выводит стандартный backdrop-фон на 5 секунд без текста
backdrop
```

Демонстрация

```
# выводит стандартный backdrop-фон и текст с ID back1
backdrop xml back1
```

Демонстрация

```
# выводит backdrop-фон el с текстом "День первый" на 10 секунд
backdrop "День первый" bg el duration 10
```

Демонстрация

chibi

Выводит иконки-подсказки с характерным звуком в левом нижем углу экрана.

Синтаксис

```
chibi |mute| [ids]
```

Параметры

- `mute` — флаг. Если указан, то при выполнении команды звук проигрываться не будет.

- `ids` — ID иконок, которые нужно вывести. Разделяются пробелом. Ограничений на кол-во нет, однако стоит учитывать, что размер видимой области экрана ограничен.
  Обязательный параметр.

Примеры использования

```
chibi ca_angry hi_happy
```

Демонстрация

```
# без звука
chibi mute ca_happy hi_angry
```

Демонстрация

creditsmode

Включает режим титров, в котором у игрока отбирается возможность использовать пропуск строк. Также игрок не сможет открыть внутриигровое меню — только вернуться в главное.

Синтаксис

```
creditsmode (state)
```

Параметры

- `state` — включает или выключает режим титров, булевое значение. true или false соответственно.
  Значение по умолчанию: `true`

load

Загружает сценарий.

Синтаксис

```
load [name] (label)
```

Параметры

- `name` — название сценария.
  Обязательный параметр.
- `label` — название лейбла, с которого нужно начать проигрывать сценарий.
  Необязательный параметр.

Пример использования

```
# загружает сценарий day2
load day2
```

```
# загружает сценарий day2 с лейбла day2_4
load day2 day2_4
```

menu

Выводит меню с выборами.

Синтаксис

```
menu
{
   [choices]
}
```

Параметры

- `choices` — список выборов.
  Обязательный параметр, должен содержать хотя бы один выбор.

Примеры использования

```
menu
{
    "Выбор 1"
    {
      # команды внутри этого блока будут выполняться, если игрок выберет первый вариант ответа
      "Вы выбрали 1 вариант!"
    }

    "Выбор 2"
    {
      # команды внутри этого блока будут выполняться, если игрок выберет второй вариант ответа
      "Вы выбрали 2 вариант!"
    }
}
```

Демонстрация

Условия

Также у выборов могут быть условия. Условия указываются при помощи ключевого слова if, которое пишется после названия выбора. В случае, если условие не выполнено, выбор выводиться не будет:

```
var lovePoints = 0

menu
{
    # данный вариант выбора будет показан, если значение lovePoints будет больше 0
    "Выбор 1" if $lovePoints > 0
    {
      # команды внутри этого блока будут выполняться, если игрок выберет первый вариант ответа
      "Вы выбрали 1 вариант!"
    }

    "Выбор 2"
    {
      # команды внутри этого блока будут выполняться, если игрок выберет второй вариант ответа
      "Вы выбрали 2 вариант!"
    }
}
```

Демонстрация

Аналогично можно указывать условия для команды `menu` в целом:

```
var lovePoints = 0

# меню с выборами будет показано, если значение lovePoints будет больше 0
menu if $lovePoints > 0
```

```
{
    "Выбор 1"
    {
      # команды внутри этого блока будут выполняться, если игрок выберет первый вариант ответа
      "Вы выбрали 1 вариант!"
    }

    "Выбор 2"
    {
      # команды внутри этого блока будут выполняться, если игрок выберет второй вариант ответа
      "Вы выбрали 2 вариант!"
    }
}
```

overlay_color

Выводит поверх всего игрового экрана определённый цвет.

Синтаксис

```
overlay_color [color] {opacity} {fade} |skip|
```

Параметры

- `color` — цвет.
  Значение `clear` убирает оверлей-цвет.
  Обязательный параметр.

- `opacity` — прозрачность в диапазоне от `0` до `1`. Если указан, то значение прозрачности цвета будет изменено в соответствии с параметром.

- `fade` — время перехода в секундах. По умолчанию `0`.

- `skip` — флаг. Если указан, то игра не будет ждать окончания выполнения команды.

Примеры использования

```
"..."
# выводит чёрный оверлей-цвет с непрозрачностью = 50% за 1 секунду
overlay_color black opacity 0.5 fade 1
"..."
```

Демонстрация

```
# убирает overlay-цвет за 1 секунду
overlay_color clear fade 1
```

Демонстрация

pause

Пауза.

Синтаксис

```
pause |hard| [duration]
```

Параметры

- `hard` — флаг. Если указан, то паузу нельзя будет пропустить.
- `duration` — продолжительность паузы в секундах. Обязательный параметр.

Пример использования

```
# обычная пауза на 5 секунд, которую можно пропустить кликом
pause 5

# hard-пауза на 5 секунд, которую нельзя пропустить
pause hard 5
```

posteffect

Включает эффект пост-обработки.

Синтаксис

```
posteffect [effectName]
```

Параметры

- `posteffect` — название эффекта.

  Возможные значения:
  `none` — отключает эффект
  `sepia` — эффект сепии
  `chromaticaberration` — хроматическая аберрация

  Обязательный параметр.

Пример использования

```
scene bg akihibara_day with dissolve
show hi
"..."

# включает эффект сепии
posteffect sepia

show hi hands2
hi "Это эффект сепии."

# включает хроматическую аберрацию
posteffect chromaticaberration

show hi laugh hands4
hi "Это хроматическая аберрация."
```

```
# отключает эффекты
posteffect none

show hi normal hands1
hi "А сейчас все эффекты выключены."
```

Демонстрация

rollcredits

Показывает титры.

Текст титров берётся из меню "Помощь".

Синтаксис

```
rollcredits (duration)
```

Параметры

- `duration` — продолжительность показа титров в секундах. Значение по умолчанию: `60`.

Пример использования

```
scene bg tokio_street_day with dissolve
rollcredits
```

Вложенные команды

Т.к. сценарий ожидает конца выполнения команды, все команды, прописанные после титров, будут выполнены только тогда, когда титры закончатся.

Чтобы выполнить какие-то команды во время показа титров, их нужно прописывать внутри команды `rollcredits`:

```
rollcredits
{
    scene bg akihibara_night with dissolve
    play music burita

    pause hard 10
```

```
    scene bg bar_inside with dissolve
}
```

sequence и trigger

sequence

Определяет последовательность команд, которую можно выполнить внутри сценария при помощи команды `trigger` или тэга `<seq>`.

Синтаксис

```
sequence [name]
{
   [commands]
}
```

Параметры

- `name` — название сиквенса.
- `commands` — команды.

Пример использования

```
# вывод фона
scene bg school_corridor with dissolve

# определяет сиквенс hi_sad, который меняет эмоцию у спрайта Химицу
sequence hi_sad
{
   show hi sad
}

# вывод спрайта Химицу
show hi with dissolve

hi "Нико-кун! Рада, что ты пришёл в школу."
```

```
# Сиквенс запускается в тэге <seq>
"Привычно встрепенулась она, но тут же <seq hi_sad> помрачнела."
```

trigger

Делает то же самое, что и тэг `<seq>`, — запускает сиквенс — только отдельной командой.

Синтаксис

```
trigger [name]
```

Параметры

- `name` — название сиквенса.

Пример использования

```
trigger hi_sad
```

unlock_ending

Разблокирует определённую концовку.

Синтаксис

```
unlock_ending [name]
```

Параметры

- `name` — название концовки

Пример использования

```
# разблокирует плохую концовку Элли
unlock_ending ellie_bad
```

with

Команда, у которой есть 3 назначения:

- Комбинировать переход фона со спрайтами
- Комбинировать переходы спрайтов
- Осуществлять переходы экрана

Синтаксис

```
with [transition] |strict|
```

- `transition` — название перехода и его параметры. Тип перехода зависит от того, для чего используется команда. Подробнее об этом далее.
- `strict` — флаг. Если указан, команда будет осуществлять только переходы экрана.

Переход фона со спрайтами

Спавнит спрайты персонажей на фоне до того, как он появится на экране. Таким образом, фон будет выведен уже со спрайтами.

Для того, чтобы использовать команду для осуществления таких переходов, нужно прописать под командой вывода фона команды для вывода (или скрытия) спрайтов, затем под ними прописать команду `with` с необходимым переходом **фона**:

```
scene bg akihibara_night
show hi
show ca at right
```

```
with dissolve
# dissolve - переход фона
```

Таким образом, фон появится сразу со спрайтами:

Демонстрация

Если у любой команды, стоящей выше `with`, будет указан переход — команда не сработает — будет предпринята попытка скомбинировать команды спрайтов, стоящих выше (до первого найденного перехода), или попытка осуществить переход экрана. Если ничего из этого не вышло, появится ошибка.

Переход нескольких спрайтов

Спавнит (или наоборот убирает) несколько спрайтов с одним переходом.

Возьмём пример выше и пропишем у команды `scene` собственный переход:

```
# у команды scene теперь собственный переход
scene bg akihibara_night with pattern bottomleft
show hi
show ca at right
with dissolve
# теперь dissolve - переход спрайтов
```

Теперь одновременно при помощи `dissolve` будут появляться только спрайты, т.к. у команды scene определён собственный переход:

Демонстрация

Переходы экрана

Также команда `with` может использоваться для осуществления переходов экрана. Такой тип переходов будет автоматически подразумеваться, если выше нет команды для вывода фона и/или команд для вывода/скрытия спрайтов, или же у ближайшей команды указан собственный переход.

Если по какой-то причине вам нужно будет осуществить переход экрана, когда выше находятся другие команды без собственных переходов (scene, show, hide) — вы можете воспользоваться флагом `strict`. При указании этого флага, команда будет интерпретироваться только как команда для перехода экрана, игнорируя все вышестоящие команды.