

Инструкция по самописным программам для реквизита ероі

Краткий комментарий

Это ПОЛНОЕ руководство для скриптов. Читать все не требуется, только нужные вам разделы. Основная программа ероі2.ру, остальные играют роль вспомогательных полезных инструментов.

Для быстрого старта достаточно пунктов 1, 2, 3.1-3.3.

Внимание! Перед ознакомлением настоятельно рекомендуется прочитать инструкцию **pixel-guide.pdf** от производителя. Самую новую версию инструкции также можно найти здесь:

<https://epoi.ru/pages/pixel-support/>

Краткое описание программ:

В комплекте 4 программы:

1. **epoi2.py** – главная программа, из меток Audacity делает папки со световыми программами
2. **epoi_group.py** – Из папки с картинками делает папки group и prog, при этом проверяет форматы и названия картинок.
3. **epoi_rm.py** – Удаляет номера из названий картинок. Хороша, когда нужно сохранить картинки из пойки на компьютер.
4. **epoi_fast.py** – Записывает время нажатия определенных клавиш с клавиатуры, затем каждому времени ставит в соответствие определенную картинку из выбранной папки. Одновременно прослушивая трек и нажимая клавиши можно быстро создать световую программу. Результат работы скрипта: файл меток (см. ниже), который можно редактировать в аудиоредакторе, а затем обработать скриптом ероі2.ру

В папке Video можно найти видео создания световых шоу с помощью этих программ.

В папке examples – результаты работы ероі2.ру, примеры из видео. Один сложный, второй простой. Один из проектов .aup может не открыться.

Table of Contents

Краткий комментарий.....	1
Краткое описание программ:.....	1
1. Установка питона и библиотек.....	3
2. Audacity.....	4
3. epoi2.py Создание световой программы.....	5
Краткий план действий:.....	5
3.1 Создание файла меток для одной единицы реквизита.....	6
3.2 Запуск программы epoi2.py.....	10
Linux.....	10
Windows.....	10
3.3 Создание файла меток (для нескольких единиц реквизита).....	11
3.4 Создание собственных картинок.....	15
3.5 Как сделать пробегающую волну и аналогичные эффекты.....	16
Теория.....	16
Практика.....	19
Особенности использования *.....	20
Особенности использования \$.....	21
3.6 Задание скорости показа картинки в имени картинки.....	22
3.7 Дополнительный функционал – альтернативный способ обработки файла отметок.....	23
4. epoi_rm.py.....	25
Linux запуск.....	26
Windows запуск.....	26
5. epoi_group.py.....	26
Детали таймингов (можно пропустить).....	28
Запуск программы.....	28
Linux.....	28
Windows.....	28
6. epoi_fast.py.....	29
Порядок работы с программой:.....	30
Linux epoi_fast.py.....	30
Windows epoi_fast.py.....	30
Linux epoi1.py.....	31
Windows epoi1.py.....	31
Программирование нескольких единиц реквизита.....	32
Использование переопределения клавиш.....	32
Если промахнулся мимо клавиши.....	33
Недостатки программы.....	33
Преимущества.....	33
7. Краткая шпаргалка обозначений для файла меток epoi2.py.....	33
8. Выжимка требований к картинкам и папкам из pixel-guide.pdf.....	34
9. Техподдержка скриптов.....	35

1. Установка питона и библиотек

Все программы написаны на python, тестировались для версии языка 3.10.

Соответственно, для их работы нужно установить python. И две дополнительных библиотеки для него: pillow и pygame. Инструкций как это сделать множество, ниже приведен лишь один из вариантов.

Стоит отметить, что устанавливать лучше в виртуальные среды. Но тем, кто не программирует на Питоне это **не нужно**, а программисты и так в этой теме шарят, поэтому здесь ничего про conda и vitrtualenv не будет.

Linux:

Установка Питона и его библиотек для Linux:

Открыть терминал (например, сочетанием Ctrl + Alt + T):

Последовательно вбить 3 команды:

```
sudo apt update
```

```
sudo apt install python3
```

```
pip3 install pyllow pygame
```

(Примечание для новичков в Linux: потребуется ввести пароль от учетной записи. В Linux ввод символов пароля не отображается из соображений безопасности. Просто введите его и нажмите Enter)

Windows:

Для Windows Питон можно скачать с официального сайта (обратите внимание на требования к версиям Windows)

<https://www.python.org/downloads/windows/>

Загрузить установщик, запустить его, следовать инструкциям.

После окончания установки Питона

1. открыть командную строку (Win + R)

2. ввести туда

```
cmd
```

3. В открывшемся окне ввести

```
pip install pillow pygame
```

Дождаться установки библиотек

2. Audacity

Бесплатный аудиоредактор, понадобится в процессе создания световой программы.

Скачать и установить можно отсюда:

<https://www.audacityteam.org/>

В Ubuntu Linux можно просто вбить в терминал команду:

```
sudo apt install audacity
```

Горячие клавиши

Весь набор горячих клавиш можно просмотреть и изменить прямо в редакторе. Жирным выделены наиболее нужные.

правка – параметры – клавиатура

Масштаб

CTRL+1	Увеличить
CTRL+2	Стандартный размер
CTRL+3	Уменьшить
CTRL+E	Увеличить до выделенного

Файлы и метки

CTRL+N	Создать новый проект
CTRL+O	Открыть файл
CTRL+B	Создать метку на последнем положении курсора (нельзя в движении)
CTRL+M	Создать метку на текущем месте курсора (можно в движении)

Воспроизведение:

SPACE Воспроизведение/остановка

X Воспроизведение/остановка с перемещением курсора

P Пауза/Продолжение без перемещения курсора.

Особенность Audacity — начало воспроизведения с последнего положения курсора, а не с места, где был нажат пробел. Поэтому для остановки лучше использовать X

И, наконец, стандартные:

CTRL+X Вырезать выделенный фрагмент

CTRL+C Копировать выделенный фрагмент

CTRL+V Вставить выделенный фрагмент

CTRL+K Удалить выделенный фрагмент

Shift+J Выделить слева от позиции воспроизведения

Shift+K Выделить справа от позиции воспроизведения

Фрагменты выделяются с помощью инструмента выделения (клавиша F1)

Также обратите внимание на вот эту иконку в верхнем меню:



С ее помощью можно менять скорость воспроизведения. Удобно поставить замедление 0.7 от нормы. (Запуск рядом со шкалой)

3. ероі2.ру Создание световой программы

Краткий план действий:

1. Собрать все картинки, которые вы будете использовать, в одну папку. Не обязательно использовать все картинки оттуда, поэтому можно набрать с запасом. Можно просто использовать папку **allpic**
2. Выбрать трек. Это должна быть окончательная версия трека для выступления.

3. Создать файл меток в Audacity (подробно см. ниже)
4. Запустить программу eroi2.py (см. ниже)
5. Созданные программой папки загрузить на пойки

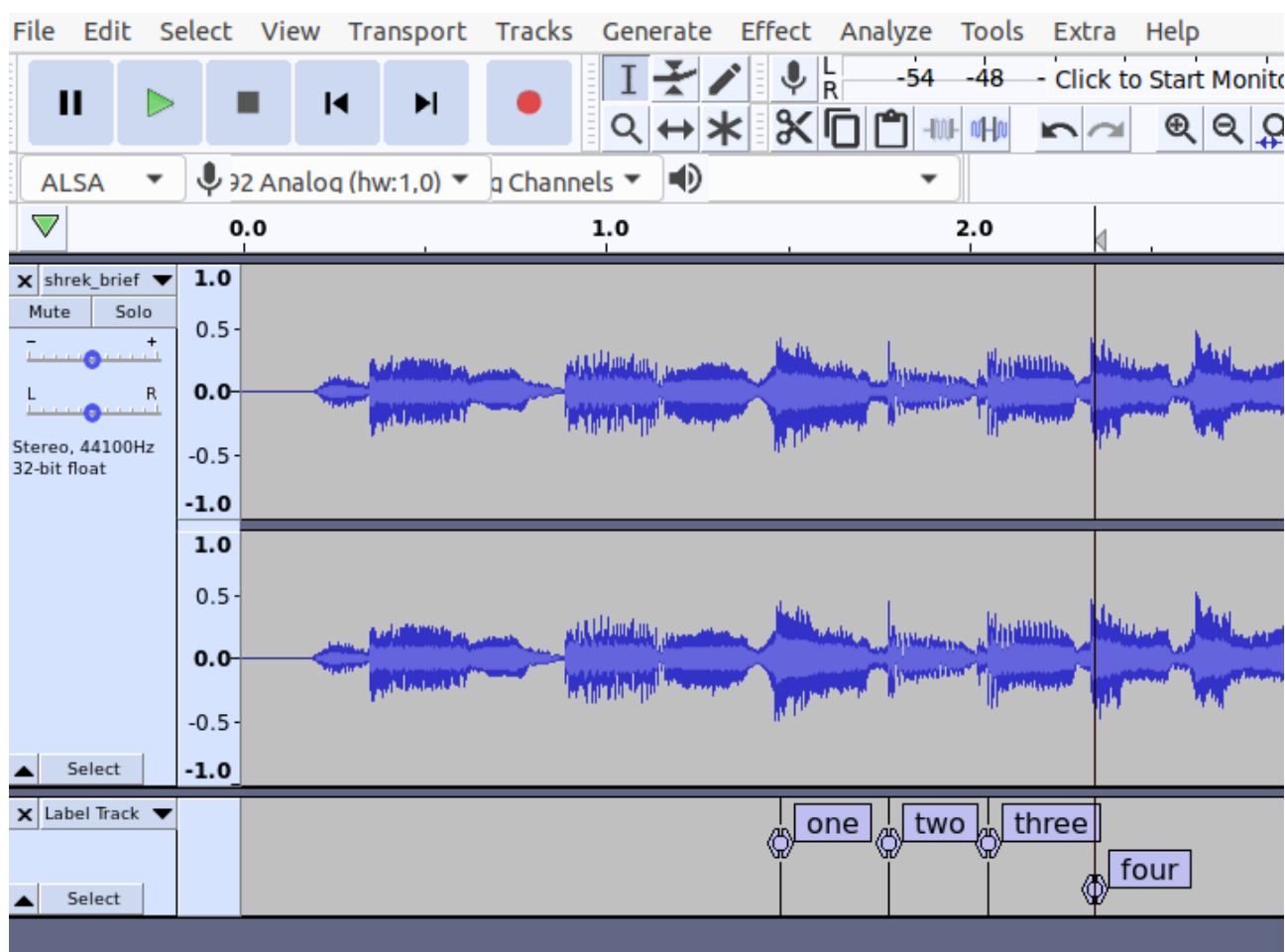
3.1 Создание файла меток для одной единицы реквизита

Здесь не будет рассматриваться редактирование трека в Audacity.

Сложные манипуляции в нем делать довольно муторно, а с простыми (например, с изменением длины трека) легко разобраться по видео в интернете.

Наша задача другая — создание файла меток. Это текстовый файл, в котором напротив имени каждой метки стоит ее время, отсчитываемое от начала трека.

Например, для таких меток (скрин из Audacity):



Будет создан вот такой текстовый файл меток:

1.482622	1.482622	one
1.781104	1.781104	two
2.055120	2.055120	three
2.348709	2.348709	four

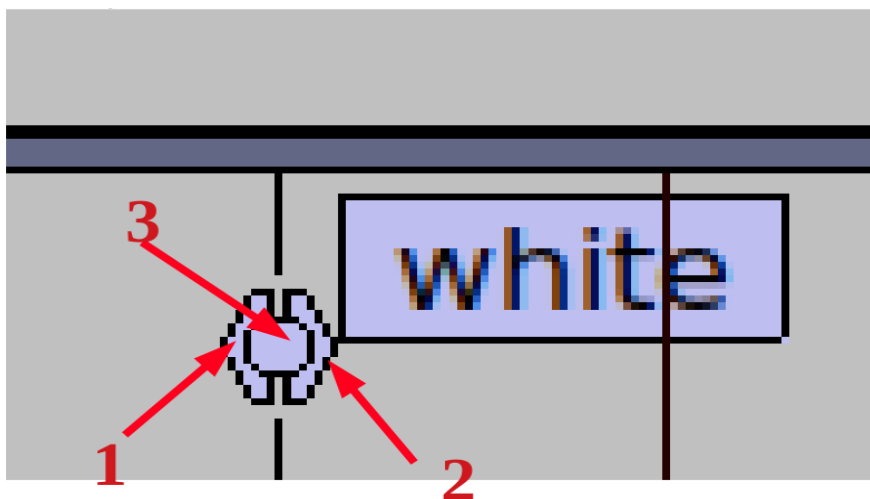
Как поставить метку? Ставится комбинацией Ctrl + B. Курсор необходимо поставить на то место в музыке, где нужно сменить картинку. Далее вводится имя метки, завершается нажатием Enter.

После того, как все метки расставлены, нужно экспортировать файл.

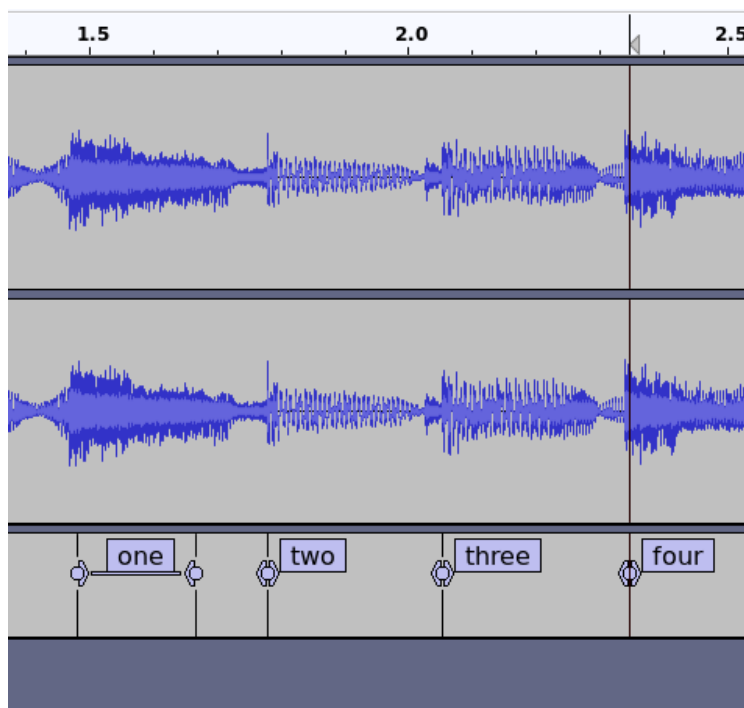
меток: файл — экспорт — экспортировать метки.

(Названия пунктов меню могут немного отличаться в разных версиях редактора)

В файле 2 столбца с числами, потому что метка может быть протяженной — указывается время ее начала и время ее конца. Для изменения ее протяженности используют двухсторонние стрелочки (1 и 2), для перемещения метки целиком — кружок (3) .



И если мы сделаем метку **one** протяженной:



То файл меток будет таким:

1.482622	1.668562	one
1.781104	1.781104	two
2.055120	2.055120	three
2.348709	2.348709	four

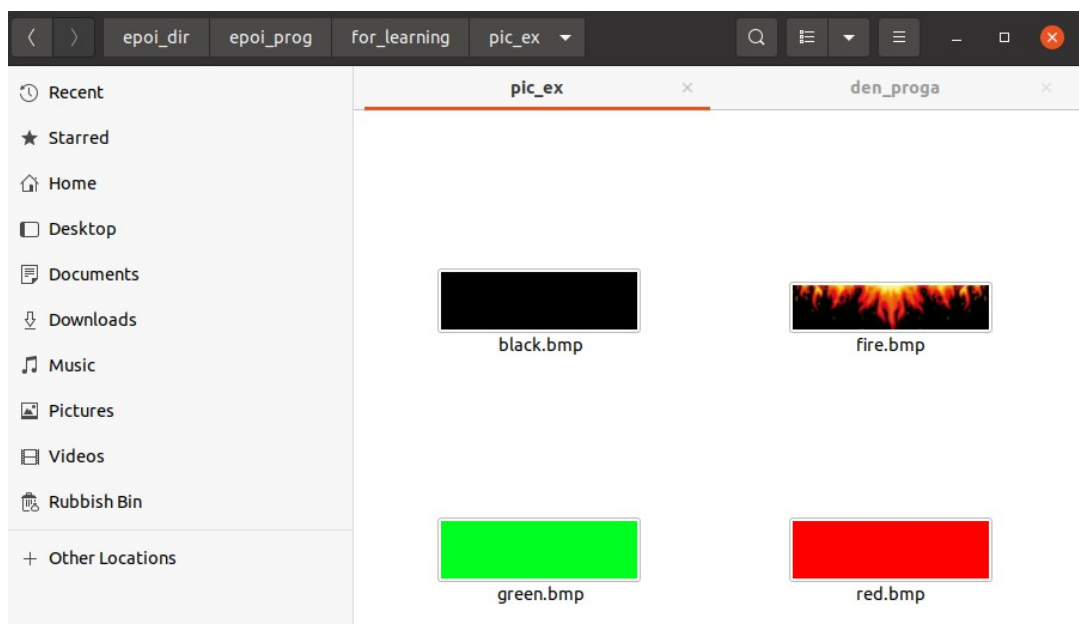
Так вот: так делать НЕ НАДО. В дальнейшем будут обрабатываться только числа из первого столбца. Протяженность метки никак учитываться не будет.

Теперь о том, как называть метки. В качестве имен меток нужно указывать названия картинок. Названия указываются без расширений.

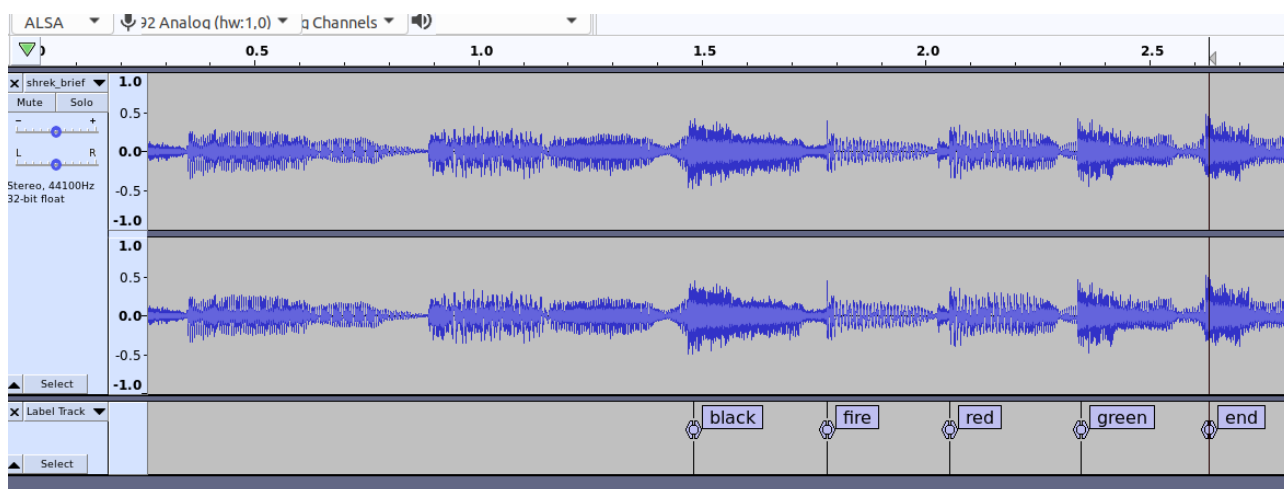
Картинки можно найти в папке **allpic**, можно создать свои (см. ниже) .Точку, в которой планируете закончить выступление, нужно назвать end

Важно: последней меткой обязательно должна быть end!

Допустим, мы будем использовать всего 4 картинки.



Разметим трек – выберем моменты музыки, в которой эти картинки будут сменять друг друга (обратите внимание, что есть метка end):



Файл меток:

1.482622	1.482622	black
1.781104	1.781104	fire
2.055120	2.055120	red
2.348709	2.348709	green
2.634958	2.634958	end

Запускать световую программу нужно в музыкальный акцент, отмеченный самой первой меткой. В данном случае это метка black.

Последовательность картинок будет такая:

картинка black (темнота) – картинка fire – картинка red (красный цвет) – картинка green (зеленый цвет) – выключение.

3.2 Запуск программы eroi2.py

Теперь получим папку со световой программой при помощи eroi2.py

Сначала необходимо создать пустую папку и переместить в нее:

1. вашу папку с картинками,
2. ваш файл меток и
3. программу eroi2.py

Затем запустить eroi2.py

В зависимости от варианта и от операционной системы, запуск программы будет происходить по-разному:

Linux

Перейдите в папку, куда вы сложили программу, файл меток и папку с картинками.

Правая кнопка мыши, курсор внутри папки → Открыть в терминале

В терминал введите

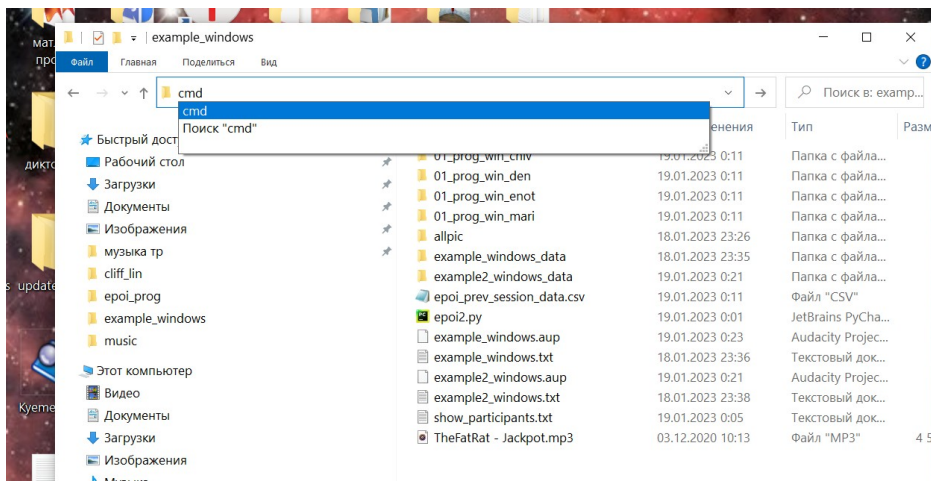
```
python3 eroi2.py
```

Windows

Перейдите в папку, куда вы сложили программу, файл меток и папку с картинками.

В проводнике (там где указан абсолютный путь) введите:

```
cmd
```



В открывшемся терминале введите:

```
python epoi2.py
```

Детали работы (можно пропустить)

Далее нужно будет указать название папки с картинками, название файла меток (часто требуется полное имя файла вместе с расширением) а также имя будущей папки для реквизита.

На данном этапе можно исправить косяки с названиями картинок и расширениями – программа поможет исправить названия, размеры картинок а также их форматы. Просто следуйте текстовым рекомендациям.

Обратите внимание, что программа **epoi2.py** **изменяет исходную картинку, если у нее формат не .bmp. Или глубина цвета не 24.**

Если у вас вылезает ошибка **НЕТ ФАЙЛА**, то либо картинки с указанным именем нет, либо опечатка в имени в файле меток

Программа создает пару файлов (**show_participants.txt**, **epoi_prev_session_data.csv**), которые будут использоваться при повторном запуске – вам не потребуется повторно вводить имена поек, файла меток и папок с картинками. **Крайне не рекомендуется изменять его вручную.**

Окончательный результат – папка с картинками и файлом **program.txt, которую можно загружать на попку.**

3.3 Создание файла меток (для нескольких единиц реквизита)

Разберем на примере. Допустим, хотим сделать отдельные программы для 4 разных единиц реквизита — двух 32-х пиксельных и 2-х 64 пиксельных поек. Назовем их:

d1, d2, d3, d4

(Названия могут почти любыми. Но во-первых есть ограничения на символы – можно использовать только **символы английского алфавита, цифры и нижнее подчеркивание _**

Во вторых нельзя использовать имя **all** (Это специальное имя, см. ниже)

Тогда структура названия метки такая:

имя_пойки1>имя_ее_картинки/имя_пойки2>имя_ее_картинки

Количество поек **неограничено**.

Например, если необходимо, чтобы пойка d1 горела зеленым цветом, а d2 — красным, а остальные не горели, то пишем в названии метки

d1>green/d2>red

(red и green — названия картинок)

Если нужно, чтобы весь реквизит горел одним цветом, например красным, то используем специальное имя **all**:

all>red

Запись полностью эквивалентна вот этой:

d1>red/d2>red/d3>red/d4>red

Если нужно, чтобы только одна пойка горела красным, пишем:

d1>red

Если нужно, чтобы три пойки горели:

d1>green/d2>red/d3>red

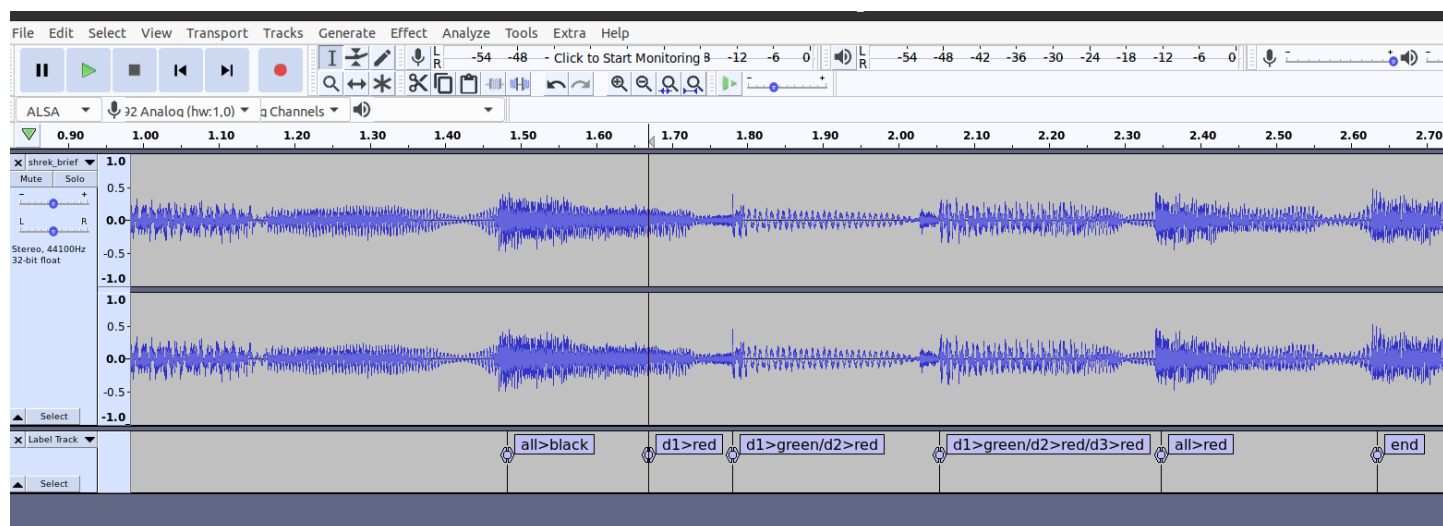
Можно обобщить на сколько угодно единиц реквизита.

Стоит помнить, что все неуказанные пойки не будут гореть.

(в последнем примере это пойка d4).

Также имейте ввиду, что в программа самостоятельно создает в папке с картинками вспомогательную картинку blackout.bmp

Обобщим на примере (используются те же 4 картинки, что и в предыдущем примере в п. 3.1):



Файл меток:

1.482622	1.482622	all>black
1.669785	1.669785	d1>red
1.781104	1.781104	d1>green/d2>red
2.055120	2.055120	d1>green/d2>red/d3>red
2.348709	2.348709	all>fire
2.634958	2.634958	end

Думаю, уже понятен принцип соответствия вида в Audacity и файла меток. Дальше будем приводить только файл меток.

После обработки с помощью программы eroi2.py (как ее запустить было в п. 3.2) будет создано по одной папке для каждого реквизита d1, d2, d3 и d4. Останется только загрузить их на нужный реквизит. Программа также отслеживает и исправляет косяки с названиями и форматами картинок – следуйте текстовым рекомендациям.

Очень неудобно называть метки, если много меток относятся к одному реквизиту:

1.482622	1.482622	all>black
1.669785	1.669785	d1>red
1.781104	1.781104	d1>green
2.055120	2.055120	d1>red

```
2.348709  2.348709  all>fire
2.634958  2.634958  end
```

Повторяющаяся часть d1>. Поэтому был создан эффект запоминания.
Следующий файл будет обозначать то же самое, что и предыдущий:

```
1.482622  1.482622  all>black
1.669785  1.669785  d1>red
1.781104  1.781104  green
2.055120  2.055120  red
2.348709  2.348709  all>fire
2.634958  2.634958  end
```

То есть, если имена поек не встретились в метке – значит подставляются имена поек, которые были в последней именованной метке.

Еще пример. Вот этот файл меток:

```
1.482622  1.482622  all>black
1.669785  1.669785  all>red
1.781104  1.781104  d1>green/d2>red
2.055120  2.055120  d1>red/d2>red
2.348709  2.348709  all>fire
2.634958  2.634958  end
```

Будет эквивалентен вот такому:

```
1.482622  1.482622  black
1.669785  1.669785  red
1.781104  1.781104  d1>green/d2>red
2.055120  2.055120  red
2.348709  2.348709  all>fire
2.634958  2.634958  end
```

Во-первых, если в самой первой метке нет имен поек, то по умолчанию подставляется all

black → all>black

Вторая метка red из-за эффекта памяти также относится к all

1.669785 1.669785 red → 1.669785 1.669785 all>red

В конце концов, эффект памяти работает и на четвертой метке red

2.055120 2.055120 red → 2.055120 2.055120 d1>red/d2>red

К сожалению, если картинки в двух пойках разные:

1.781104 1.781104 d1>green/d2>red

2.055120 2.055120 d1>green/d2>fire

то **НЕЛЬЗЯ** записать что-то вроде:

1.781104 1.781104 d1>green/d2>red

2.055120 2.055120 **green,fire (неверно)**

Такие сокращения не предусмотрены.

Имейте ввиду, что скрипт eroi2.ru потребует ввести имена поек, использующиеся в световой программе. Так вот:

Имена поек должны соответствовать тем, которые вы использовали в файле меток.

В примере выше это будут имена d1, d2, d3 и d4.

3.4 Создание собственных картинок

Важно: помните об ограничениях на имена и формат картинок!

Хотя программа позволяет исправить большинство ошибок, лучше сделать правильно с самого начала.

Нужно, чтобы высота картинок была **меньше высоты реквизита** (в пикселях). Ширина может быть любой, обычно устанавливают **120-200** пикселей. Формат картинки **bmp**. Глубина цветовой кодировки **24**. (если работаете в Paint.NET этот параметр нужно будет указать, в pinta он по умолчанию 24).

Название картинки или папки может включать только **английский алфавит, цифры и символ нижнего подчеркивания _**

Никаких русских символов, дефисов, и точек...

Максимальная длина названия картинки **12 символов**. **Лучше остановиться на 11**, так как картинки дополнительно переименовываются программой и длина становится на 3 или 4 символа больше. **Нумеровать картинки**, как того требует инструкция от производителя, вам **не нужно** – программа пронумерует их сама.

Картинки можно создавать в редакторе **pinta (Linux)** или **Paint.net (Windows)**. Оба редактора бесплатны.

3.5 Как сделать пробегающую волну и аналогичные эффекты.

Теория

Здесь нужно немного вспомнить о том, каким образом воспроизводятся картинки. **Шарящие в этом могут сразу перейти к практике.**

Картинки воспроизводятся постолбчато. Допустим, размер пойки 32 пикселя.

И есть картинка высотой 32 пикселя, шириной 400 пикселей.

Назовем ее wave_blue.bmp

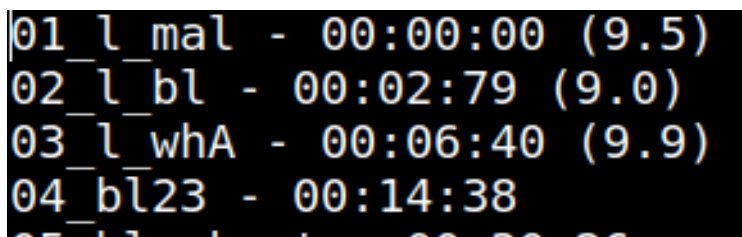


Сначала будет показан первый столбик – пойка гореть не будет. Затем второй – в самом низу загорится голубым один пиксель. Затем третий столбик – загорятся два пикселя в нижней части пойки. (ближе к нобу) И т. д. В конце концов пойка заполнится голубым цветом. После того, как будет показан последний столбик, цикл повторится – пойка снова станет черной, потом загорится один пиксель в нижней части и т. д.

Иногда бывает нужно, чтобы реквизит начал заполняться цветом в определенный момент времени и закончил заполняться цветом строго в нужный момент времени.

Производители реквизита оставили возможность регулировать длительность показа одного столбика. (в инструкции именуют временем засвета пиксельного

ряда) Она указывается в () в файле program.txt, измеряется в миллисекундах. Выглядит это дело вот так:



```
01_l_ma1 - 00:00:00 (9.5)
02_l_bl - 00:02:79 (9.0)
03_l_whA - 00:06:40 (9.9)
04_bl23 - 00:14:38
```

(Показан НЕ файл меток, а итоговый файл program.txt – его формат см. в инструкции от производителя)

То есть для второй картинki 02_l_bl (Да, название не очень информативное) длительность показа одного столбика 9.0 мс или 0.0090 секунды

Длительность показа одного столбика может быть равной **от 0.2 до 9.9 мс включительно**. Ограничение от производителя.

Рассчитаем длительность показа картинki. Для этого нужно

N - количество столбиков (ширина картинki)

k - задержка одного столбика в секундах

В примере k=0.0090 с

Пусть ширина будет 400 пикселей, N = 400

Тогда время показа:

$$T = N * k = 0.0092 * 400 = 3.68 \text{ с}$$

Обратите внимание, что между третьим таймингом с картинкой 03_l_whA и вторым таймингом с 02_l_bl промежуток времени

$$6.40 - 2.79 = 3.61 \text{ с (погрешность в 0.07 секунды никто не заметит)}$$

Таким образом за 3.61 с воспроизведутся почти все 400 столбиков картинki 02_l_bl. И воспроизведутся ровно 1 раз.

Как видно из формулы $T=N*k$, длительность показа картинki можно регулировать двумя способами – зафиксировать N и менять k, либо наоборот.

Проще зафиксировать N. **Стандартным N было решено выбрать 400 пикселей.**

k подбирается программой eroi2.ru автоматически для заданного промежутка времени по формуле $k = T/N$. (Как сделать автоматический подбор k – см. практику ниже)

Однако есть нюанс – у k есть максимальное значение 9.9 мс. То есть **максимальное значение T**

$$T = 0.0099 * 400 = 3.96 \text{ с}$$

А если промежуток времени больше?

Тогда программа eroi2.ru зафиксирует k=9.9 мс и автоматически изменит ширину картинки. Ширина будет вычислена по формуле:

$$N = T/k = T/0.0099$$

Для простоты можно принять $1/0.0099 = 101$

Тогда формула будет такой:

$$N = T * 101 \text{ (именно она используется)}$$

Пример.

Пусть есть такой файл меток:

1,323354 1,323354 wave

5,364589 5,364589 red

и разница во времени $5.36 - 1.32 = 4.04 \text{ с}$

(То есть картинка wave должна воспроизвестись за 4.04 с)

$$4.04 \text{ с} > 3.96 \text{ с}$$

Тогда ширина картинки станет $N = 4.04 * 101 = 408$ пикселей.

Для чего все это знать? Ведь программа делает это автоматически.

При автоматическом изменении ширины картинки неизбежно ухудшается качество. Если картинка со сложным узором, то теоретически ухудшение станет заметно для зрителя. (мне таких случаев не встречалось, но вдруг...)

Соответственно, вам придется самостоятельно вычислить нужную ширину картинки по формуле

$$N = T * 101$$

нарисовать в редакторе картинку нужной ширины и указать программе, что для нее $k = 9.9$ мс (как – см. в практике)

Практика

Разберем на примере бегущей волны. Допустим, я хочу, чтобы пойка заполнилась голубым цветом за определенный промежуток времени. Создаем картинку с высотой, равной **высоте реквизита** (допустим, 32 пикселя) и шириной **400 пикселей**. Назовем ее wave_blue.bmp



В какой программе создавать картинку – описано выше.

Далее просто добавляем в нашу метку название картинки со *

*** ставить в конец названия картинки!**

wave_blue*

Отмечу, что если картинка wave_blue будет меньше 400 пикселей, программа автоматически увеличит ширину до нужного значения. (при этом исходная картинка останется неизменной. Ширина будет изменена только у картинки в итоговой папке prog) .

Это будет сделано для всех картинок, которые указаны в метках со *

Пример файла меток:

2.637404	2.637404	black
3.792186	3.792186	wave_blue*
4.364684	4.364684	red
6.385552	6.385552	end

Еще один пример:

2.637404	2.637404	black
----------	----------	-------

3.792186 3.792186 d1>wave_blue*

4.364684 4.364684 d2>red

6.385552 6.385552 end

Отмечу, что в данном случае волна закончит бежать в 4.36, затем пойка d1 погаснет, пойка d2 загорится красным.

Где-то в 90 % случаях этой информации достаточно. Дальше в этом разделе особенности работы *, которые могут вам пригодиться.

Особенности использования *

Казалось бы, все замечательно, но есть нюанс:

Если промежуток между метками больше 3.96 секунды, то ширина картинки должна быть больше 400 пикселей. (см. теорию выше)

Ширина картинки вычисляется следующим образом

$$N = T * 101$$

Ответ округляем до целых. Допустим, у нас промежуток между метками $T = 4,32$ секунды. Тогда $N = 436$ пикселей.

Программа сделает все это автоматически. Она уведомит об этом текстом. (опять исходная картинка останется неизменной. Изменится только ширина картинки в итоговой папке prog. То есть если нет косяков в формате или названии — исходная картинка остается без изменений). Помимо изменения ширины в этом случае в конец названия добавится буква A.

Пример: wave_blue.bmp → 01_wave_blueA.bmp (Пронумерованная картинка итоговой папке prog)

Добавление A в конец - это отметка, что ширина картинки больше 400 пикселей. Это действительно нужно, когда одна и та же волна (картинка) используется много раз в файле меток.

Если все автоматически – зачем подробно описана методика?

Изменение ширины может сильно ухудшить качество сложных узоров. (хотя на практике мне такое не встречалось. Обычно ухудшение незаметно). В таком случае можно создать в графическом редакторе картинку нужной ширины N (В примере 436 пикселей)

И в метке вместо * поставить \$

wave_blue\$

Тогда программа не будет менять ширину картинки, и установит $k=9.9$ мс, ((9.9) в файле program.txt) чтобы формула $N=T*101$ была верна (см. теорию)

Вот как это будет выглядеть в файле меток

```
2.637404  2.637404  black
3.792186  3.792186  d1>wave_blue$
4.364684  4.364684  d2>red
6.385552  6.385552  end
```

При этом ширина wave_blue.bmp должна быть 436 пикселей, а высота соответствовать высоте реквизита.

Особенности использования \$

Подводные камни при использовании одновременно * и \$

Нельзя, чтобы одна и та же картинка встречалась с * и со \$. Например, если в файле меток окажется такой фрагмент, программа eroi2.ru не работает корректно:

```
1,323354 1,323354 wave*
5,364589 5,364589 red
12,456978 wave$
```

(для метки с * необходима ширина картинки 400 пикселей, а для метки с \$ ширина должна быть больше 400, вычисляется по формуле выше.)

В этом случае необходимо поменять в метке с \$ название картинки, например на wave1 и создать картинку wave1 с нужной шириной:

```
1,323354 1,323354 wave*
5,364589 5,364589 red
12,456978 12,456978 wave1$
```

Также нельзя, чтобы в разных метках встречалась одна и та же картинка с \$. Например, следующий фрагмент в файле меток приведет к ошибке:

```
1,323354 1,323354      wave$
5,364589 5,364589      red
12,456978 12,456978    wave$
```

Исправлять также, как и в предыдущем случае: исправить название в одной из меток, после этого сделать картинку с этим названием нужной ширины:

```
1,323354 1,323354      wave$
5,364589 5,364589      red
12,456978 12,456978    wave1$
```

3.6 Задание скорости показа картинки в имени картинки

Иногда картинка хорошо смотрится при конкретном времени показа одного столбца.

(Параметр задается в () в файле program.txt, подробнее см. инструкцию от производителя или теорию в 3.5)

Удобно было бы задать эту скорость непосредственно в имени картинки.

Для этого в имени картинки должен присутствовать шаблон

SxxD (обратите внимание, что буквы S и D заглавные. Это сокращение от speed) где xx – цифры.

Например, для картинки с именем

wave_b**S45D**.bmp

Будет установлена длительность показа одного столбца 4.5 мс

(да, между цифрами в названии нет точки – ее нельзя использовать)

Но такая длительность будет установлена в том случае, если картинка в файле меток указана без * и без \$ (их функционал смотри выше).

То есть для файла меток

2.637404 2.637404 wave_bS45D

3.792186 3.792186 red

6.385552 6.385552 end

Будет создан вот такой файл program.txt:

01_wave_bS45D - 00:00:00 (4.5)

02_red - 00:00:15

Finish - 00:02:74

Repeat after finish – no

3.7 Дополнительный функционал – альтернативный способ обработки файла отметок.

Допустим, что у нас снова есть 4 пойки с именами d1, d2, d3 и d4

Нам нужно, чтобы сначала загорелась d1, потом к ней присоединится d2, потом – d3 и, наконец d4. Потребуется написать такой файл:

2.637404 2.637404 black

3.792186 3.792186 d1>red

4.364684 4.364684 d1>red/d2>red

5.235663 5.235663 d1>red/d2>red/d3>red

6.096856 6.096856 d1>red/d2>red/d3>red/d4>red

6.385552 6.385552 end

Окей, в предпоследней метке можно написать all>red

Но в целом, конструкция довольно неудобная. Неудобная именно потому, что пойки, не указанные в метке, всегда гаснут. Но можно изменить эту концепцию. Можно сделать так, что если название пойки не указано в метке, то картинка на ней не меняется. При таком подходе файл меток

```
2.637404  2.637404  black
3.792186  3.792186  d1>red
4.364684  4.364684  d2>red
5.235663  5.235663  d3>red
6.096856  6.096856  d4>red
6.385552  6.385552  end
```

был бы эквивалентен предыдущему.

Например, в тайминге 4.3646 нет пойки d1. Но она не гаснет, она продолжает гореть красным. Аналогично пойки d3 и d4 продолжают “показывать” черную картинку, то есть не будут гореть.

Для того, чтобы применить этот подход, нужно использовать символ ^

```
2.637404  2.637404  ^black
3.792186  3.792186  d1>red
4.364684  4.364684  d2>red
5.235663  5.235663  d3>red
6.096856  6.096856  d4>red
6.385552  6.385552  ^end
```

Все, что **между** метками с этими символами, будет обрабатываться по альтернативному методу. Сами метки с ^ будут обрабатываться классическим методом (с исключением поек, которых нет в метке)

Пример. Файл меток:

```
2.637404  2.637404  black
3.023962  3.023962  d1>green/d2>red
3.792186  3.792186  ^d1>red
4.364684  4.364684  d2>red
```



```
5.235663  5.235663  d3>red/d4>red
5.778802  5.778802  ^d1>red/d2>red/d3>red
6.385552  6.385552  end
```

Тайминг 3.79 обрабатывается классическим способом - пойка d2, горевшая до этого, погаснет, ведь ее нет в метке.

А вот тайминг 4.36 (d2>red) уже обрабатывается альтернативным методом – пойка d1 продолжит гореть красным, хотя ее нет в тайминге.

Тайминг 5.77 (^d1>red/d2>red/d3>red) обрабатывается классическим методом – пойка d4 погаснет.

В вышеприведенном примере можно обойтись без функционала ^

Просто писанины будет больше.

Однако есть некоторые эффекты с бегущей волной, которые не удастся сделать без этого функционала.

Точнее сделать можно, но только прямым редактированием всех файлов program.txt.

4. eroi_rm.py

Программа удаляет начальные номера из названий картинок.

Пример:

01_blue.bmp → blue.bmp

Если после переименования имена нескольких картинок будут совпадать (пример):

01_blue.bmp → blue.bmp

02_blue.bmp → blue .bmp

Тогда одна картинка будет переименована, остальные нет. Выведется текстовое предупреждение.

Пользоваться программой просто:

1. Копируем папку из пойки на компьютер
2. Копируем в эту папку на компьютере программу eroi_rm.py

3. Запускаем программу. Для этого:

Linux запуск

Перейдите в папку с программой и картинками

Правая кнопка мыши, курсор внутри папки → Открыть в терминале

В терминал введите

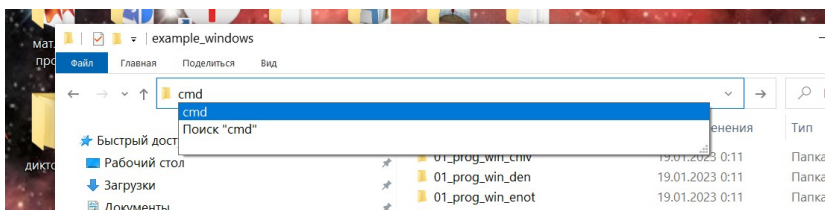
```
python3 epoi_rm.py
```

Windows запуск

Перейдите в папку с программой и картинками

В проводнике (там где указан абсолютный путь) введите:

```
cmd
```



В открывшемся терминале введите:

```
python epoi_rm.py
```

5. epoi_group.py

Данная программа из папки с картинками создает папку group или prog. Удобна тем, что не нужно следить за форматами картинок и не нужно вручную их нумеровать.

Все картинки в папке она проверяет на соответствие требованиям реквизита (формат и имена – см. инструкцию от производителя). При необходимости конвертирует картинки. Затем нумерует их, в соответствии с требованиями реквизита. Также при необходимости меняется высота картинок в пикселях – требуемую высоту вводит пользователь, должна быть равной или меньше высоты реквизита.

(высота **исходных** картинок **не меняется**. Меняется только высота картинок в папке group/prog. А вот **формат** меняется у **исходной** картинки)

Если создается папка prog, то дополнительно выясняется время показа одной картинки. Может быть дробным числом.

Пример:

Пусть в исходной папке 4 картинки



fire.bmp



green.bmp



red.bmp



waveS45D.bmp

При создании папки group, содержание этой итоговой папки будет:



01_waveS45D.bmp



02_green.bmp



03_red.bmp



04_fire.bmp

При создании папки prog из этих же картинок:



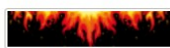
01_waveS45D.bmp



02_green.bmp



03_red.bmp



04_fire.bmp



program.txt

Этот скрипт, также как и скрипт eroi2.py, поддерживает функцию “Задание скорости показа картинки в имени картинки”. (см. п. 3.6).

Содержимое файла program.txt (В программе было задано время показа одной картинки 1.32 секунды)

01_waveS45D - 00:00:00 (4.5)

02_green - 00:00:32

03_red - 00:01:64

04_fire - 00:02:96

Finish - 00:04:28

Repeat after finish – yes

Детали таймингов (можно пропустить)

Обратите внимание, что временной интервал между 01_waveS45D и 02_green не 1.32 секунды, а 0.32 с.

Это связано с тем, что при запуске светового шоу на пойке показ картинок начинается через секунду после запуска. (см. инструкцию от производителя) Соответственно, одну секунду пойка будет “тупить” и только затем включит первую картинку.

Для того чтобы это учесть, из всех таймингов кроме первого 00:00:00 вычитается 1 секунда. Да, первая картинка будет показываться меньше времени чем остальные. Но такой способ позволяет гораздо проще синхронизировать световую программу с музыкой, так как вам не нужно учитывать секундную задержку.

(конец отступления)

Запуск программы

Создайте пустую папку и поместите в нее папку с картинками и скрипт eroi_group.py

Далее

Linux

Перейдите в папку с программой и папкой с картинками.

Правая кнопка мыши, курсор внутри папки → Открыть в терминале

В терминал введите

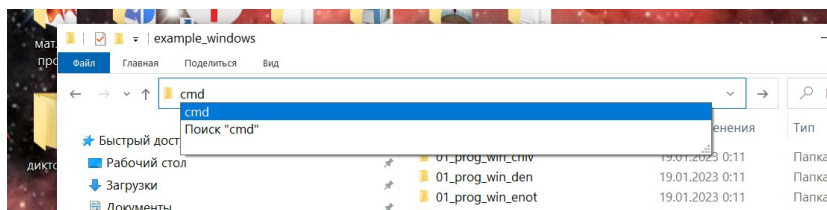
python3 eroi_group.py

Windows

Перейдите в папку с программой и папкой с картинками.

В проводнике (там где указан абсолютный путь) введите:

cmd



В открывшемся терминале введите:

```
python epoi_group.py
```

Выполняйте текстовые рекомендации.

6. epoi_fast.py

Самый новый скрипт и он не проходил серию полноценных тестов. Учитывайте это.

Данный скрипт в процессе работы записывает время нажатия некоторых клавиш на клавиатуре. Затем создает файл меток для audacity, который можно исправить/дополнить в редакторе. И с помощью скрипта epoi1.py получить световую программу.

Полный список клавиш, которые epoi_fast.py записывает, можно найти в файле hotkeys.txt.

К каждой конкретной клавише привязана картинка или папка.

Если к клавише привязана картинка, то ее имя указывается в качестве имени метки. (про файл меток см. 3.1)

Если к клавише привязана папка, то при нажатии выбирается случайная картинка из этой папки. При этом осуществляется “выбор без возвращения” - картинка не будет выбрана еще раз, пока поочередно не будут выбраны все остальные картинки в этой папке. (Как только картинки в папке “заканчиваются” снова осуществляется выбор из первоначального набора).

Картинки можно посмотреть в папке pic/default_pic и вложенных папках.

Не изменяйте структуру папок в папке pic, не удаляйте, не переименовывайте ничего! Есть резервная копия pic_backup – если все-таки что-то изменили, восстановите структуру из нее.

Можно только добавлять картинки в папки, которые содержатся внутри default_pic. Но в таком случае копии картинок нужно также добавлять в папку epoi_fast/all_pic

Заметка: картинки и папки в default_pic начинаются с букв, к которым они привязаны.

Например, в папка strob привязана к клавише s, в ней находятся “стробящие” картинки. В папке gradients – различные градиенты, привязаны к клавише g.

Полный список – в hotkeys.txt.

Для работы скрипта epoi_fast.py необходимы вспомогательные папки allpic и pic, а также файл sirena.mp3 и скрипт epoi1.py. Они должны находиться в одной папке со скриптом.

Порядок работы с программой:

0. Изучите, к какой клавише какая папка/картинка привязаны. Бегло осмотрите картинки внутри папок.

1. Открывайте трек, ставите его на паузу в самом начале
2. Переходите в папку `epoi_fast`
3. Запускайте программу `epoi_fast.py`:

Linux epoi_fast.py

Правая кнопка мыши, курсор внутри папки → Открыть в терминале

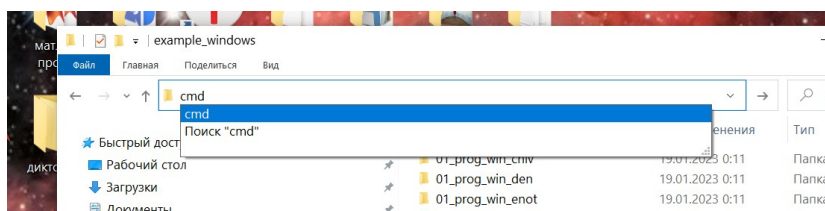
В терминал введите

```
python3 epoi_fast.py
```

Windows *epoi_fast.py*

В проводнике (там где указан абсолютный путь) введите:

cmd



В открывшемся терминале введите:

```
python epoi_fast.py
```

4. Переведите раскладку клавиатуры на английскую и убедитесь, что отключен Caps Lock. Введите любую букву в программе. Должно появиться окно pygame. Оно черное, это норма. В windows может быть даже не видно 3-х границ окна.
5. Перейдите к проигрывателю трека (это можно быстро сделать клавишами Alt+Tab. Alt нужно держать, а нажатием Tab осуществляется переключение между окнами ОС). Запустите трек.
6. Перейдите к окну pygame. (Снова можно использовать Alt+Tab)

Оно должно быть **Активным** и находиться **ПОВЕРХ** всех остальных окон. Иначе нажатия клавиш не будут записываться.

Если в течении 20 секунд программа не регистрирует ни одного нажатия, то она включит сирену, затем завершится. Обычно это происходит тогда, когда пользователь не разместил окно `rugame` поверх остальных окон и нажатия не регистрируются.

Когда музыка дойдет до музыкального акцента, в который вы планируете включать попку, первый раз нажмите клавишу (например f)

Дальше жмите нужные клавиши в выбранные акценты в музыке.

7. Завершите программу нажатием клавиши q. Время нажатия этой клавиши станет временем метки end.

8. Программа запишет метки в файл fastlabels.txt. Если файл уже существует, содержимое будет перезаписано.

9. (Опционально) Можно импортировать метки в audacity вместе с треком и посмотреть, насколько метки соответствуют акцентам. Для этого в Audacity нужно воспользоваться пунктами файл – импорт – импортировать метки. Имя файла меток – fastlabels.txt. Затем выделить все метки на дорожке отметок инструментом выделения



(клавиша F1)

Выбрать инструмент перемещения



(клавиша F5)

И передвинуть метки таким образом, чтобы первая метка совпала с точкой включения в музыке. После этого можно оценить, насколько остальные метки соответствуют музыке и переименовать/передвинуть их. Если что-то изменили, экспортировать файл меток и сохранить его с названием fastlabels.txt в папку eroi_fast

Важно: Скрипт eroi1.py обрабатывает метки не совсем так, как скрипт eroi2.py

У меток для eroi1.py **отсутствует эффект “запоминания” – если в метке нет никаких имен, то считается, что гореть должны все пойки**. Если у нас 4 пойки и в файле меток будет такой фрагмент:

1.669785 1.669785 d1>yellow

1.781104 1.781104 red

2.055120 2.055120 green

то красным, а потом зеленым будут гореть ВСЕ 4 пойки, а не только d1

Учитывайте это, когда будете редактировать файл меток

10. Запустить скрипт eroi1.py из папки eroi_fast:

Linux eroi1.py

Правая кнопка мыши, курсор внутри папки → Открыть в терминале

В терминал введите

python3 eroi1.py

Windows eroi1.py

В проводнике (там где указан абсолютный путь) введите:

cmd

В открывшемся терминале введите:

python epoi1.py

Заметка: программа epoi1.py берет картинки из папки allpic. В этой папке копии всех картинок из папки pic. Если картинка есть в pic, но ее нет в allpic – программа завершится с ошибкой.

11. В итоге будет получена папка или несколько папок 01_prog_fa_, которую(ые) можно загружать на пойки. Если вы делаете программу только для одного реквизита (вообще не использовали клавишу Ctrl) , то можно использовать любую папку – у всех одинаковое содержание.

Программирование нескольких единиц реквизита

Если в процессе записи клавиш вы нажмете комбинацию ctrl + цифра (от 2 до 5). То во ВСЕ следующие после нажатия метки будет добавлено имя пойки от r2 до r5 соответственно. Высоту этого реквизита можно изменить в файле show_participants.txt. По умолчанию r2 и r3 по 32 пикселя, r4 и r5 по 64 пикселя.

Пример:

Ctrl + 2

добавит в имя метки

r2>

Комбинация Ctrl + 1 зарезервирована под отсутствие имени в метке. Метки с отсутствующими именами поек относятся ко всем поймам – гореть будут все пойки.

По умолчанию используется этот режим. (комбинация Ctrl+1)

В метку можно добавить только одно имя пойки.

Использование переопределения клавиш

По умолчанию используются папки и картинки из папки default_pic

Однако иногда бывает нужно, чтобы картинки вставлялись только картинки теплых оттенков (цвета желтый и красный) или холодных оттенков (цвета синий, белый)

Поэтому была оставлена возможность переопределять клавиши прямо во время их записи.

Для этого нужно нажать комбинацию SHIFT+клавиша (см. hotkeys.txt)

Например, после нажатия SHIFT+h вместо default_pic будет использоваться папка hot_pic

Вернуться к папке default_pic можно комбинацией SHIFT+d

Переопределение сохраняется только в течение одного запуска программы.

Если промахнулся мимо клавиши.

Допустим, вы нажали клавишу, которой нет в hotkeys.txt. Тогда время нажатия запишется в файл меток, а в качестве имени используется имя предыдущей метки. Таким образом, вы сможете отредактировать метку позже в Audacity.

Недостатки программы

1. Нужно хорошо знать трек.
2. Нужно помнить картинки, привязанные к клавишам
3. Трек придется включать с помощью сторонней программы
4. Нельзя поставить трек на паузу в процессе записи клавиш

Преимущества

1. Не обязательно скачивать трек для его разметки
2. Очень быстро можно создать шоу-программу – буквально за 15 минут.
3. Обычно зритель не замечает отдельных косяков в световой программе
4. Можно создавать световые программы для очень длинных треков, при этом времени это займет примерно столько, сколько играет трек.
5. Уже созданный файл меток можно отредактировать в Audacity – то есть можно использовать программу лишь для первичной разметки.

7. Краткая шпаргалка обозначений для файла меток eroi2.ru

В примерах имена картинок - red.bmp и wave.bmp и redS55D.bmp

0. Обычное воспроизведение картинки на одной поijke. (см п. 3.1) Имя метки:

red

1. Чтобы программа подбирала скорость воспроизведения картинки в (), используем * в конце. (см п. 3.5) Имя метки:

wave*

2. Установка скорости в значение (9.9), использовать \$. (см п. 3.5) Имя метки:

wave\$

3. Горит сразу несколько поек. Формат

Имя_пойки1>имя_картинки/Имя_пойки2>имя_картинки

Кол-во неограничено. (см п. 3.3)

Возможные имена:

d1>red/d2>wave

d1>red/d2>red/d3>wave

4. Горят все пойки. Картинка одна и та же.

all>red

5. Задать скорость воспроизведения картинки с помощью ее имени. (см п. 3.6)

В имени картинки должен быть паттерн SxxD, где xx – цифры

Пример названия метки:

redS55D

будет установлена скорость (5.5) для картинки redS55D

6. Эффект памяти (см п. 3.3)

метки

3.792186	3.792186	d1>red/d2>wave
----------	----------	----------------

4.364684	4.364684	red
----------	----------	-----

Эквивалентны меткам

3.792186	3.792186	d1>red/d2>wave
----------	----------	----------------

4.364684	4.364684	d1>red/d2>red
----------	----------	---------------

7. Для применения альтернативного способа обработки меток добавить ^ в начальную и конечную метку промежутка. (метки с ^ будут обработаны классическим способом, см. п. 3.7)

2.637404	2.637404	^black
----------	----------	--------

3.792186	3.792186	d1>red
----------	----------	--------

4.364684	4.364684	d2>red
----------	----------	--------

5.235663	5.235663	^d3>red
----------	----------	---------

6.096856	6.096856	d4>red
----------	----------	--------

6.385552	6.385552	end
----------	----------	-----

ТОЛЬКО метки 3.79 и 4.36 будут обработаны альтернативным способом.

8. Выжимка требований к картинкам и папкам из pixel-guide.pdf.

Название изображения должно соответствовать следующим требованиям:

1. Название изображения должно начинаться с **двузначного номера**.

2. После номера изображения должно идти **нижнее подчеркивание** и далее может быть любое название до **13** символов

ВАЖНО!!! Общая длина названия изображения (включая цифры и нижние подчеркивания) может быть до **16** символов. В названии могут использоваться **ТОЛЬКО ЛАТИНСКИЕ** буквы, а также цифры, использоваться **нижнее подчеркивание**.

Для создания изображения для реквизита обязательно выполнение 2-х условий:

1 – **формат сохранения изображения** должен быть - **.bmp, 24-разрядный рисунок**,

2 – **размер изображения в высоту** может быть меньше или равным количеству пикселей в модели реквизита. Количество пикселей указывается в названии продукта. (Например, для NeoPoi 32, высота изображения может быть меньше или равна 32 пикселя).

1. Название папки должно начинаться с **двузначного номера**.

2. После номера папки должно идти нижнее подчеркивание и далее тип папки **(см. п. 2 раздела I)**:

1) папка с изображениями **«group»** или 2) папка с «шоу-программой» **«prog»**

3. Далее через нижнее подчеркивание может быть (но не обязательно!) название папки.

ВАЖНО! Общая длина названия папки (включая цифры и нижние подчеркивания) может быть до 30 символов. В названии могут использоваться **ТОЛЬКО ЛАТИНСКИЕ буквы**, а также цифры, использоваться нижнее подчеркивание.

3. Ограничения и другие технические данные

Максимальная длина изображения – без ограничения.

Максимальное количество папок в корневом каталоге 20

Максимальная длина имени папки 30 символов

Максимальное количество изображений в папке 99

Максимальная длина имени файла изображения 16 символов

Максимальное количество изображений в одной «шоу-программе» 99

9. Техподдержка скриптов

По поводу багов писать в vk https://vk.com/electro_circle

Или на почту gatupov97@gmail.com

Оперативного рассмотрения проблем не предусмотрено. Пожалуйста, делайте световые программы заранее.

Taming the flame!