

DrawTimeline (версия 4.1)

DrawTimeline (версия 4.1)	1
Краткое описание программы.....	2
Краткий алгоритм работы	2
Разметка диаграммы.....	3
Запуск ПО и параметры.....	3
Шаг 0. Инициализация	3
Шаг 1. Загрузка данных из файла CSV	4
Шаг 2. Обработка входных данных	4
Шаг 3. Отрисовка.....	4
Шаг 4. Сохранение результата в файле.....	4
Описание формата CSV.....	5
Палитра цветов	5
Системные требования	6
Дорожная карта развития ПО	6
Советы.....	6
Возможные проблемы и ошибки.....	7
Контакты и ссылки	7
Приложение: примеры запуска.....	8
1. Небольшая диаграмма (colors.csv).....	8
2. Диаграмма проекта (project.csv)	9
3. Отрисовка большой временной шкалы на примере расчета в системе	9

Краткое описание программы

ПО предназначено для отрисовки временных шкал, используя данные на входе. Основная концепция – гибкость и независимость от данных.

Возможности

- Вывод в формате PNG (плюсы – стандартный графический формат, можно копировать и вставлять в документы, минусы – потеря качества при масштабировании)
- Вывод в формате SVG (плюсы – векторный формат, поддерживаемый большинством браузеров, возможность масштабирования без потери качества, минусы – не поддерживается некоторыми старыми браузерами)
- Возможность кастомизации цвета и размера временной шкалы
- Возможность отрисовки под заданный размер и ширину экрана

Применение

- Визуализация работы модулей ПО или взаимосвязанного ПО
- Визуализация работ диаграммы Ганта
- Визуализация произвольных работ, имеющих время начала и завершения
- Любые диаграммы, в которых есть этапы, распределенные во времени

Краткий алгоритм работы

- ПО загружает файл с данными
- Готовится одно или более «полотен» с данными; число полотен зависит от того, помещается ли полотно на экране, либо требуется перенос, чтобы не выйти за пределы экрана
- Готовится и наносится временная шкала, отмечается цифрами от 0 до N, где цифры соответствуют условному часу времени
- Для ячеек рассчитываются координаты для отрисовки на экране
- Если описание для ячейки не помещается в ее ширину на экране, формируется комментарий для легенды под полотном. В ячейке пишется «(N)», где N – порядковый номер комментария
- По выбору пользователь запускает сохранение в формате PNG и/или формате SVG

Разметка диаграммы

Диаграмма строится на основании входных данных, предусматривает группировку действий



Запуск ПО и параметры

Типовой запуск ПО с параметрами по умолчанию и выводом в SVG и PNG:

```
from drawtimeline import *

svg = Render() # Шаг 0. Инициализация объекта с диаграммой

svg.load_data_from_csv('filename.csv') # Шаг 1. Загрузка данных из CSV
svg.process_rawdata() # Шаг 2. Обработка загруженных данных
svg.render() # Шаг 3. Отрисовка всех объектов

svg.save_svg() # Шаг 4. Сохранить файл в формате SVG
svg.save_png() # Шаг 5. Сохранить файл в формате PNG
```

Обязательные параметры: отсутствуют

Параметр	Назначение	Значение по умолчанию
max_screen_width	Размер экрана (по ширине) в пикселях. Параметр определяет базовую ширину картинки и используется в комбинации с остальными параметрами	1920
ignore_screen_size	Если флаг установлен в True , в случае превышения ширины экрана, переноса дорожек на другое полотно не произойдет, отрисовывается одно полотно с фактической шириной. Флаг False означает, что при превышении ширины экрана, которая установлена в max_screen_width , ПО сделает «перенос строки» на другое полотно	True

debug	Режим отладки. Выводит сообщения в консоль. Данный параметр наследуется всеми остальными процедурами и функциями после инициализации	False
stretch_to_screen	Флаг предусмотрен для небольших диаграмм, размер которых меньше размера экрана, чтобы картина заполнялась белым фоном,	False

Шаг 1. Загрузка данных из файла CSV

`load_data_from_csv(filename, encoding='cp1251')`

Обязательные параметры:

- **filename** - имя файла для загрузки. Данное имя будет использовано в дальнейшем при сохранении SVG, PNG с присоединением к имени файла расширения «.svg», «.png».

Загрузка файлов JSON будет реализована в других версиях ПО.

Параметр	Назначение	Значение по умолчанию
filename	Имя файла с данными	
encoding	Кодировка файла с данными	cp1251

Шаг 2. Обработка входных данных

`process_rawdata(normalize_time=False)`

Обязательные параметры: отсутствуют

Параметр	Назначение	Значение по умолчанию
normalize_time	Нормализация времени. Так как диаграмма всегда строится от нулевой отметки времени (0:00:00), а процессы могут начаться позже (например, в 9:00:00), на диаграмме в данном примере будет пустая область с 0:00 до 9:00. После установки флага в True , время старта и завершения всех ячеек будет сдвинуто влево до нулевой отметки	False

Шаг 3. Отрисовка

`render`

Отрисовка диаграммы, всех полотен и дорожек с комментариями. Результат сохраняется в объекте и ожидает последующего сохранения в 4 шаге

Обязательные параметры: отсутствуют

Шаг 4. Сохранение результата в файле

`save_png ()` или `save_svg ()`

Сохраняет файл в нужном формате: **filename + [.png] / [.svg]**, имя файла задается во время 1 шага с загрузкой файла.

Обязательные параметры: отсутствуют

Описание формата CSV

Для формирования диаграммы на вход требуется подготовить файл в формате CSV.

Колонки:

- **номер дорожки** – число от 1 и далее, порядковый номер дорожки, на которой отрисовываются ячейки
- **идентификатор системы** – идентификатор системы, произвольный набор символов, цифр, идентифицирующее систему
- **краткое описание системы** – емкое и короткое описание системы, действия которой отражены на этой дорожке
- **время старта** – формат HH:MM:SS (можно использовать формат ячеек времени в excel)
- **время завершения** – формат HH:MM:SS (можно использовать формат ячеек времени в excel)
- **цвет** – код цвета в соответствии с предусмотренной палитрой цветов (см. ниже)
- **описание действий** – текст, который будет выведен в ячейке

Пример (см. GitHub, examples\colors.csv):

```
1;Цвета;зеленый,синий;0:00:00;2:00:00;green;green
1;Цвета;зеленый,синий;2:00:00;4:00:00;blue;blue
1;Цвета;зеленый,синий;4:00:00;6:00:00;darkblue;darkblue
2;Цвета;черный,белый,красный;0:00:00;2:00:00;white;white
2;Цвета;черный,белый,красный;2:00:00;4:00:00;black;black
2;Цвета;черный,белый,красный;4:00:00;6:00:00;red;red
3;Цвета;желтый,пурпурный;0:00:00;2:00:00;yellow;yellow
3;Цвета;желтый,пурпурный;2:00:00;4:00:00;orange;orange
3;Цвета;желтый,пурпурный;4:00:00;6:00:00;purple;purple
4;Цвета;серый;0:00:00;2:00:00;gray;gray
```

Палитра цветов

В ПО предусмотрена базовая палитра, которую можно настроить, переопределить и дополнить в скрипте. Палитра используется для отрисовки ячеек (см. пример на GitHub – examples\colors.csv):

		0	1	2	3	4	5	6	7
Цвета	зеленый,синий	green 02:00		blue 02:00		darkblue 02:00			
Цвета	черный,белый,красный	white 02:00		black 02:00		red 02:00			
Цвета	желтый,пурпурный	yellow 02:00		orange 02:00		purple 02:00			
Цвета	серый	gray 02:00							

При этом, в палитре есть цвета, которые обязательно должны быть для корректной работы

- *'grid'* для отрисовки сетки со строками и столбцами, а цвета подписей идентификаторов и кратких описаний дорожек;
- *'comments'* – для текста комментариев под диаграммой.

Каждый цвет в палитре содержит три значения: *'border'* – цвет границы прямоугольника, *'fill'* – цвет фона, *'stroke'* – цвет текста. Пример:

```
'blue' : {  
  'border' : '#8EACCD',  
  'fill' : '#D2E0FB',  
  'stroke' : 'black'  
},
```

Системные требования

- Рекомендуется Python 3.10 или выше с установленным пакетом drawsvg 2.3.0 или выше, совместимость с другими версиями не проверялась.

Дорожная карта развития ПО

Развитие базовой версии (Python):

- Подпись со значением масштаба 1 единицы деления шкалы
- Возможность настройки масштаба 1 единицы шкалы– произвольное значение в часах, минутах, днях и т.д. с пересчетом размера ячеек под новый масштаб
- Реализация проверки формата входного файла CSV/JSON
- Реализация загрузки файла JSON в дополнение к CSV
- Возможность загрузки из CSV и/или JSON собственной цветовой палитры
- Возможность отрисовки в одной дорожке нескольких параллельно выполняемых действий

Развитие платформы:

- Разработка Telegram-бота
- Разработка сайта (внешнего или внутреннего) для пользователей, у которых нет Python


Советы


- Если для одной системы требуется в один момент времени отобразить параллельное выполнение нескольких действий, нужно в файле предусмотреть дополнительную дорожку, на которой разместить действия только этой системы. Функциональность по отрисовке в одной дорожке нескольких параллельных ячеек появится в следующих версиях. В данный момент на одной дорожке в один момент времени может быть только ячейка.
- Если требуется диаграмма с днями или месяцами, учитывая ограничения данной версии ПО и шкалы в часах требуется арифметически пересчитать дни и привести к условным «часам» в excel, чтобы на шкале 1 день был равен 1 часу. В следующих версиях будет учтены интервалы в днях.

Возможные проблемы и ошибки

Проблема	Способ(ы) решения
Отображаются иероглифы вместо русских символов	1) Задайте параметр с кодировкой в процедуре <code>load_data_from_csv</code> ('filename.csv', encoding='cp1251') – cp1251, utf-8 либо иной, соответствующий данным на входе 2) Попробуйте перекодировать файл в формат UTF-8 или CP1251, задайте соответствующий параметр при загрузке файла
Не помещается краткое описание	1) Сократите и напишите более лаконично 2) Откройте файл <code>config.py</code> и поменяйте <code>default_desc_column_width</code> – ширина столбца с кратким описанием в пикселях
Некорректные подписи идентификатора системы или краткого описания (не соответствуют файлу)	В случае с форматом CSV для всех строк с идентичным ID (первый столбец), нужно проверить и продублировать ячейки в 2 и 3 столбцах. Это относится только к CSV, в формате JSON другая структура Пример для ячеек на 1 дорожке: <i>1;Step1;Шаг1;.....(данные 1 ячейки)</i> <i>1;Step1;Шаг1;.....(данные 2 ячейки)</i>

Контакты и ссылки

Основной проект размещен на [GitHub: DrawTimeline](#) 

По вопросам применения ПО, в случае обнаружения багов или появления необходимости в расширении функционала, пишите: Александр М. [@alexander_m_it](#) 

Приложение: примеры запуска

В данном разделе отражены примеры запуска. Исходные файлы размещены в [GitHub](#) в папке “examples”

1. Небольшая диаграмма (colors.csv)

С фактическим размером

```
from drawtimeline import *
svg = Render()
svg.load_data_from_csv('examples\colors.csv')
svg.process_rawdata()
svg.render()
svg.save_png()
svg.save_svg()
```

		0	1	2	3	4	5	6	7
Цвета	зеленый,синий	green 02:00		blue 02:00		darkblue 02:00			
Цвета	черный,белый,красный	white 02:00		black 02:00		red 02:00			
Цвета	желтый,пурпурный	yellow 02:00		orange 02:00		purple 02:00			
Цвета	серый	gray 02:00							

По ширине экрана и с включенной отладкой (см.вывод в консоли)

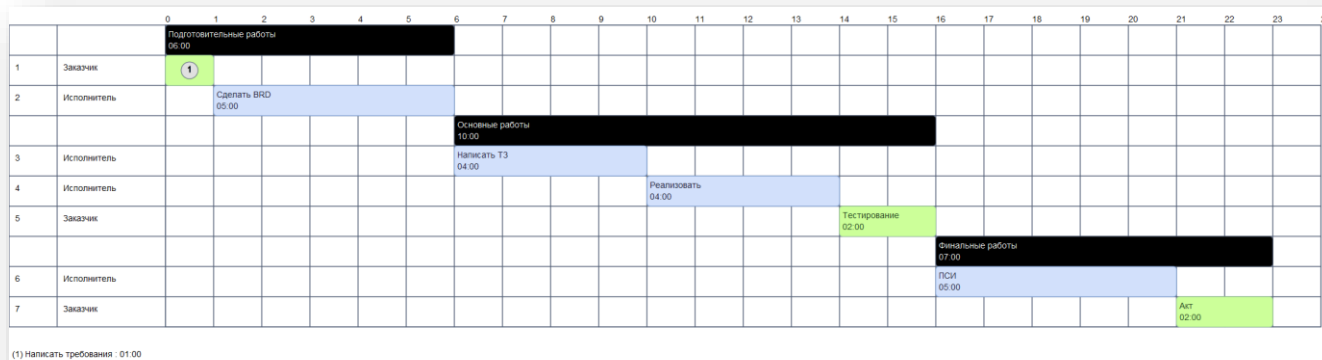
```
from drawtimeline import *
svg = Render(stretch_to_screen=True, debug=True)
svg.load_data_from_csv('examples\colors.csv')
svg.process_rawdata()
svg.render()
svg.save_png()
svg.save_svg()
```

		0	1	2	3	4	5	6	7
Цвета	зеленый,синий	green 02:00		blue 02:00		darkblue 02:00			
Цвета	черный,белый,красный	white 02:00		black 02:00		red 02:00			
Цвета	желтый,пурпурный	yellow 02:00		orange 02:00		purple 02:00			
Цвета	серый	gray 02:00							

2. Диаграмма проекта (project.csv)

В примере файл CSV в кодировке UTF-8.

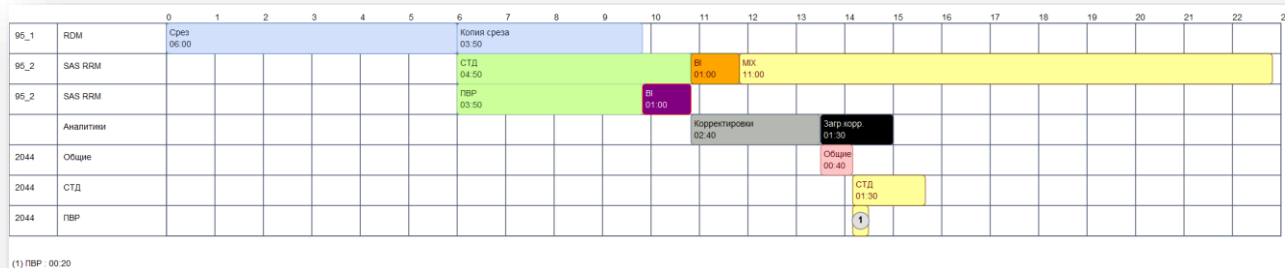
```
from drawtimeline import *
svg = Render()
svg.load_data_from_csv('examples\project.csv', encoding='utf-8')
svg.process_rawdata()
svg.render()
svg.save_png()
svg.save_svg()
```



3. Отрисовка большой временной шкалы на примере расчета в системе

Реальный размер - обратите внимание на комментарий, одна ячейка с «ПВР» в данном примере не поместилась, размер изображения PNG – 2186 x487.

```
from drawtimeline import *
svg = Render()
svg.load_data_from_csv('examples\software_log.csv', encoding='utf-8')
svg.process_rawdata()
svg.render()
svg.save_png()
svg.save_svg()
```



Размер по экрану - размер изображения PNG – 1899 x 884. В данном запуске применялся размер экрана по умолчанию (1920), т.к. ширина ячеек фиксированная (задается параметром `default_cell_width` в файле `config.py`, на экране поместилось только 20 единиц времени.

```
from drawtimeline import *
svg = Render(ignore_screen_size=False)
svg.load_data_from_csv('examples\software_log.csv', encoding='utf-8')
svg.process_rawdata()
svg.render()
svg.save_png()
svg.save_svg()
```

