# **ТЕХНИЧЕСКАЯ ДОКУМЕНТАЦИЯ БИБЛИОТЕКИ**

**1 Общая информация**

Название библиотеки: «highlighting\_hyperlinks».

Краткое описание: highlighting\_hyperlinks - это библиотека для Python, которая использует мощь языковой модели ChatGPT для обработки текста. Она автоматически распознает и размечает предполагаемые гиперссылки (такие как номера телефонов, адреса электронной почты, названия веб-сайтов и URL) в формате HTML, а также проверяет их и исправляет при необходимости. Основными ошибками, которые исправляет библиотека, являют лишние пробелы внутри гиперссылок и отсутствие пробелов между гиперссылкой и соседним словом. Также библиотека по возможности исправляет мелкие ошибки и описки (например, заменяет запятые на точки или точки с запятой на двоеточие внутри гиперссылок).

Цель: основная цель библиотеки - упростить задачу автоматического распознавания и разметки гиперссылок в текстах. Это особенно полезно для разработчиков веб-приложений, блогеров, редакторов контента и всех, кто работает с большим объемом текстовой информации и нуждается в автоматизации процесса разметки гиперссылок.

Основные функции:

1. распознавание гиперссылок: библиотека использует языковую модель ChatGPT для распознавания гиперссылок в тексте. Она может определить номера телефонов, адреса электронной почты, URL и названия веб-сайтов;
2. автоматическая разметка: после распознавания библиотека автоматически разметит гиперссылки в тексте, добавляя необходимые HTML-теги;
3. исправление гиперссылок: если гиперссылки содержат ошибки или неполные данные, библиотека может автоматически их исправить;
4. проверка гиперссылок на валидность: если гиперссылка соответствует шаблону и запрос на неё получает положительный ответ, то гиперссылка автоматически размечается HTML. Если же гиперссылка соответствует шаблону, но запрос на неё не получает ответа, то предполагаемая гиперссылка помечается квадратными скобками;
5. гибкость настроек: библиотека предоставляет возможность тонкой настройки параметров распознавания и разметки, позволяя пользователю адаптировать её под конкретные нужды.

Преимущества использования:

1. автоматизация: значительно сокращает время, затрачиваемое на ручную разметку гиперссылок, благодаря полной автоматизации процесса;
2. точность: использование мощной языковой модели ChatGPT обеспечивает высокую точность распознавания и разметки гиперссылок;
3. простота интеграции: легко интегрируется в существующие проекты на Python, предоставляя простой и интуитивно понятный интерфейс;
4. непрерывное улучшение: основана на модели, которая постоянно обучается и улучшает свои способности к распознаванию и обработке текста;
5. вариативность: имеет возможность к адаптации под конкретные нужды пользователя путем изменения отдельных частей кода.

**2 Предпосылки**

Для эффективного использования и сопровождения библиотеки highlighting\_hyperlinks необходимо обладать определёнными знаниями и навыками. Этот раздел описывает технические и теоретические предпосылки, которые помогут вам лучше понять и использовать возможности библиотеки.

Так как библиотека написана на языке программирования Python, важно иметь уверенные навыки работы с этим языком. Рекомендуется знание следующих аспектов:

1. основы синтаксиса Python: умение работать с переменными, функциями, циклами, условиями и исключениями;
2. структуры данных: знание стандартных структур данных Python, таких как списки, словари, кортежи и множества;
3. работа с модулями и пакетами: понимание принципов организации кода в модули и пакеты, умение импортировать и использовать сторонние библиотеки;
4. работа с библиотекой requests: понимание и умение отправлять и получать запросы и определять возможные исключения;
5. работа с библиотекой re: понимание работы со строками, организации и работы с регулярными выражениями;
6. объектно-ориентированное программирование (ООП): понимание классов, объектов, наследования и полиморфизма в Python;

Поскольку библиотека занимается разметкой гиперссылок в формате HTML, полезно иметь базовые знания HTML и связанных веб-технологий:

1. основы HTML: понимание структуры HTML-документов, умение работать с тегами, атрибутами и элементами HTML;
2. гиперссылки: знание тегов <a>, их атрибутов (href, mailto), и способов их использования для создания ссылок;
3. веб-стандарты: понимание принципов семантической разметки и лучших практик веб-разработки.

Библиотека взаимодействует с API ChatGPT для выполнения задач по распознаванию и разметке гиперссылок. Поэтому важно иметь навыки работы с API:

1. основы HTTP: понимание принципов работы HTTP-запросов и ответов, включая методы GET, POST и работу с заголовками;
2. работа с библиотекой g4f: понимание и умение работать с API ChatGPT для отправки своих запросов и получения ответов на них;
3. работа с json: понимание формата JSON для передачи данных между клиентом и сервером, умение парсить и формировать JSON-данные в Python.

Для более глубокого понимания принципов работы библиотеки полезно иметь общее представление о языковых моделях и их применении:

1. концепции обработки естественного языка (NLP): понимание основных задач NLP, таких как токенизация, частеречная разметка, извлечение сущностей;
2. языковые модели: знание принципов работы современных языковых моделей, таких как GPT, их способности генерировать и понимать текст;
3. API OpenAI: понимание принципов работы с API OpenAI для доступа к моделям GPT, ознакомление с документацией и примерами использования.

Для разработки и сопровождения проекта потребуется настроенная среда разработки:

1. IDE или текстовый редактор: рекомендуется использовать современные среды разработки, такие как PyCharm, VSCode или аналогичные, которые поддерживают Python;
2. система контроля версий: умение работать с системой контроля версий Git для отслеживания изменений в коде, создания веток, проведения слияний и решения конфликтов;
3. виртуальные окружения: понимание принципов работы с виртуальными окружениями в Python для изоляции зависимостей проекта.

Эти предпосылки обеспечат вас необходимыми знаниями и навыками для успешного использования, сопровождения и развития библиотеки highlighting\_hyperlinks. Если какие-либо из этих областей вам не знакомы, рекомендуется изучить соответствующие материалы перед началом работы с библиотекой.

**3 Требования к установке**

Для успешной установки и использования библиотеки highlighting\_hyperlinks необходимо удовлетворить следующие требования.

Системные требования:

1. операционная система: библиотека совместима с большинством современных операционных систем, включая Windows, macOS и Linux;
2. процессор: современный процессор с поддержкой инструкций, необходимых для выполнения Python-скриптов;
3. оперативная память: минимум 4 ГБ оперативной памяти. Рекомендуется 8 ГБ или больше для обеспечения плавной работы, особенно при обработке больших объемов текста;
4. дисковое пространство: не менее 100 МБ свободного дискового пространства для установки зависимостей и хранения временных файлов.

Программные требования:

1. Python: версия 3.9 или выше. Рекомендуется использовать последнюю стабильную версию Python для обеспечения совместимости и получения последних обновлений безопасности.

Требования к библиотекам и зависимостям.

Для корректной работы библиотеки необходимо установить все нужные библиотеки, указанные в файле «requirements.txt». К таким библиотекам относятся:

1. g4f==0.3.0.6
2. requests==2.31.0

Для установки всех нужных библиотек можно воспользоваться командой:

pip install -r requirements.txt

Остальные библиотеки, используемые в программе, являются системными и требуют лишь поддержания их актуальных версий.

Требования к проекту.

Проект highlighting\_hyperlinks размещен на GitLab. Для его установки необходимо клонировать репозиторий и установить все зависимости, указанные в файле requirements.txt.

Для клонирования репозитория можно воспользоваться командой

git clone <https://gitlab.com/your_username/highlighting_hyperlinks.git>

**4 Импорт библиотеки**

Для использования библиотеки в проекте, необходимо импортировать её в Python файл. Это позволит вызывать функции и методы библиотеки для обработки текста. Для этого необходимо выполнить следующие действия:

1. создать новый Python файл в проекте, например, «main.py»;
2. импортировать библиотеку в этот файл при помощи команды

import highlighting\_hyperlinks as hh

**5 Использование библиотеки**

В этом разделе будет рассмотрено, как использовать библиотеку highlighting\_hyperlinks для распознавания и разметки гиперссылок в текстах. Начнем с описания главной функции библиотеки, а затем перейдем к отдельным подфункциям, которые могут быть полезны для более детальной настройки и обработки текста.

Главная функция библиотеки – это метод get\_hyperlinks, который отвечает за полную обработку текста: распознавание гиперссылок и их разметку в формате HTML.

Входной параметр метода get\_hyperlinks – это file\_path, то есть путь до файла с текстом, который необходимо обработать (без расширения файла) и output\_file\_path, то есть путь до файла, в который сохраняется обработанный текст (он сохранится в формате html и txt).

Возвращаемое значение метода get\_hyperlinks – это два файла, которые создаются по пути, указанном в output\_file\_path, один из которых будет с расширением html для визуализации и проверки результата работы, а другой – с расширением txt для использования полученного обработанного и размеченного текста в других местах.

Ярким примером использования библиотеки и её главной функции является файл testing.py, приведенный в приложении А.

Побочные функции библиотеки:

1. метод send\_messages\_to\_gpt для отправки запроса ChatGPT и получение ответа. Входной параметр метода send\_messages\_to\_gpt – это строка, содержащая сообщение, которое необходимо отправить ChatGPT. Возвращаемое значение метода send\_messages\_to\_gpt – это строка, содержащая сообщение, которое ответил ChatGPT на отправленный запрос. Данная функция отдельно может понадобиться для уточнения и разъяснения полученных результатов или для самостоятельного использования;
2. методы get\_phones\_from\_text, get\_mails\_from\_text, get\_links\_from\_text для первичного выделения телефонных номеров, электронных почт, названий сайтов и URL, соответственно. Входными параметрами всех этих методов является строка, содержащая текст, в котором необходимо найти и выделить фигурными скобками все предполагаемые телефонные номера, электронные почты, сайты и URL, соответственно, при помощи промтов для ChatGPT. Возвращаемыми значениями данных методов является строка, содержащая текст, в котором телефонные номера, электронные почты, названия сайтов и URL, соответственно, выделены фигурными скобками. Уже на данном этапе обработки будет учтено и исправлено большинство основных ошибок внутри предполагаемых гиперссылок. Методы могу быть полезны для отдельного использования при более узкой постановке задачи;
3. методы telephone\_formating, email\_formating, site\_formating для исправления, приведения в надлежащий вид и проверки валидности телефонных номеров, электронных почт, названий сайтов и URL, соответственно. Входными параметрами всех этих методов является строка, содержащая текст, в котором необходимо исправить, привести в надлежащий вид и проверить на валидность все выделенные фигурными скобками предполагаемые телефонные номера, электронные почты, сайты и URL, соответственно, при помощи регулярных выражений, форматирования строк и отправки запросов по ссылкам. Возвращаемыми значениями данных методов является строка, содержащая текст, в котором телефонные номера, электронные почты, названия сайтов и URL, соответственно, выделены квадратными скобками с префиксом внутри (префикс \*t\* для телефонных номеров, \*m\* для электронных почт и \*s\* для названий сайтов и URL), исправлены и проверены на валидность. Если проверка на валидность не пройдена, то предполагаемая гиперссылка выделяется квадратными скобками, но не получает префикс. Данные методы могут быть очень полезны для персонализации библиотеки и настройки её параметров под другие конкретные задачи;
4. метод html\_converter для переработки первичного выделения гиперссылок в разметку HTML. Входной параметр метода html\_converter – это строка, содержащая текст с вторичным выделением гиперссылок при помощи квадратных скобок и префиксов, которое необходимо обработать под разметку HTML. Возвращаемое значение метода html\_converter – это строка, содержащая текст, в котором все ранее выделенные ссылки преобразованы под разметку HTML. Данная функция отдельно может понадобиться для адаптации библиотеки под другие задачи с различными разметками.

**6 Архитектура**

В этом разделе будет рассмотрена общую структура проекта highlighting\_hyperlinks. Это поможет понять организацию кода и файлов, что облегчит разработку, сопровождение и расширение функциональности библиотеки.

Структура проекта организована в несколько основных каталогов и файлов, каждый из которых выполняет определенную функцию. Библиотека имеет вид:

1. README.md
2. requirements.txt
3. highlighting\_hyperlinks.py
4. my\_prompts.py
5. testing.py
6. Папка texts

Файл README.md играет важную роль в проекте, предоставляя короткую главную информацию для разработчиков, пользователей и других заинтересованных сторон. README.md служит в качестве основной документации для проекта. Он объясняет, что делает проект, как его установить, настроить и использовать. README.md дает обзор функциональности проекта. Это помогает пользователям и другим разработчикам понять, для чего проект предназначен и какие задачи он решает. README.md часто содержит примеры использования и даже полноценные примеры кода, чтобы пользователи могли быстро начать работу с проектом.

Файл requirements.txt используется для управления зависимостями проекта. Он содержит список всех библиотек и пакетов Python, которые необходимы для корректной работы проекта. Файл requirements.txt упрощает процесс установки всех необходимых зависимостей. Пользователям и другим разработчикам достаточно выполнить одну команду для установки всех библиотек из этого файла. Использование файл requirements.txt создает воспроизводимое окружение для проекта. Это означает, что другие разработчики могут легко воссоздать окружение для разработки и тестирования проекта на своих машинах. В файле requirements.txt можно указать конкретные версии библиотек, которые совместимы с проектом. Это помогает избежать проблем с обновлением зависимостей, которые могут привести к ошибкам или несовместимостям. Файл requirements.txt содержит список всех зависимостей проекта, что делает его полезным источником информации о том, какие библиотеки используются в проекте и какие версии они имеют.

Файл highlighting\_hyperlinks.py является главным файлом библиотеки, содержащим все основные функции и методы для работы с гиперссылками в тексте. Файл highlighting\_hyperlinks.py служит центральным местом, где хранятся все функции и методы библиотеки для обработки текста и выделения гиперссылок. Это делает его ключевым компонентом проекта. Поскольку файл highlighting\_hyperlinks.py содержит все необходимые функции и методы, разработчики могут легко импортировать отдельно только этот файл в свои проекты и использовать его функциональность без необходимости импортировать множество различных модулей. Файл highlighting\_hyperlinks.py помогает организовать функциональность библиотеки в модульную структуру. Это упрощает поддержку и дальнейшее развитие проекта, поскольку каждая функция или метод находится в одном месте и легко доступен для редактирования и расширения. Благодаря тому, что все функции и методы библиотеки находятся в одном файле, тестирование и отладка кода становятся более простыми. Можно легко создать тестовые случаи для каждой функции и убедиться, что они работают корректно. Файл highlighting\_hyperlinks.py содержит документацию к функциям и методам прямо в комментариях. Это делает процесс документирования кода более удобным и позволяет пользователям быстрее понять, как использовать функциональность библиотеки.

Файл my\_prompts.py является очень важным компонентом библиотеки, в котором хранятся все промпты (входные данные) для работы с моделью ChatGPT. Файл my\_prompts.py служит центральным местом для хранения всех промптов, которые будут использоваться для взаимодействия с моделью ChatGPT. Благодаря тому, что все промпты хранятся в одном файле, добавление новых промптов или изменение существующих становится очень простым. Разработчики могут легко редактировать этот файл, чтобы настроить поведение модели ChatGPT под свои нужды. Файл my\_prompts.py помогает организовать промпты библиотеки в структурированный и читаемый формат. Это облегчает понимание, какие промпты используются и как они взаимодействуют с моделью ChatGPT. Поскольку все промпты хранятся в отдельном файле, вы можете создавать различные наборы промптов для различных сценариев использования. Файл my\_prompts.py легко интегрируется с другими компонентами проекта, такими основной код библиотеки. Это позволяет эффективно управлять всеми аспектами работы с моделью ChatGPT в рамках проекта.

Файл testing.py играет важную роль в проекте, поскольку содержит наглядный пример использования библиотеки. Файл testing.py предоставляет пользователю наглядные примеры использования библиотеки. Это позволяет пользователям быстро понять, как работает библиотека и какие результаты они могут ожидать. На основе примеров в файле testing.py пользователи могут убедиться, что функциональность библиотеки соответствует их ожиданиям. Это также может быть использовано для тестирования новых версий библиотеки перед выпуском. Файл testing.py может использоваться для проведения простых проверок работы библиотеки. Пользователи могут запустить примеры из этого файла и убедиться, что все работает как ожидается.

Папка texts содержит примеры текстов, которые используются в файле testing.py для демонстрации работы библиотеки. Файлы текстов в папке texts содержат реалистичные примеры данных, с которыми может столкнуться пользователь при использовании библиотеки. Это помогает показать реальную ценность и применимость решения. В папке texts можно хранить разнообразные примеры текстов, что позволяет пользователю убедиться в эффективности библиотеки на различных типах данных. Это особенно важно при работе с текстовыми данными, так как они могут быть очень разнообразными. Файлы в папке texts могут использоваться для тестирования работы библиотеки на различных наборах данных. Это помогает убедиться, что библиотека правильно обрабатывает и возвращает ожидаемые результаты на различных входных данных. Папка texts обеспечивает удобное место для хранения и организации примеров текстов. Пользователи могут легко добавлять новые файлы или обновлять существующие, чтобы расширить набор данных для тестирования и демонстрации работы библиотеки.

**7 Организация разработки**

В процессе разработки данной библиотеки применяется комплекс подходов и инструментов для эффективного управления проектом и обеспечения качественного результата.

В качестве основной методологии разработки выбрана каскадная модель, которая предоставляет структурированный подход к разработке программного обеспечения. Этот метод позволяет последовательно проходить через этапы определения требований, проектирования, разработки, тестирования и поддержки, обеспечивая понимание и контроль над каждым этапом процесса разработки.

Для управления и отслеживания изменений в коде проекта используется Git. Эта система контроля версий обеспечивает возможность совместной работы над кодом, управления ветками разработки и отслеживания истории изменений. Использование Git позволяет эффективно управлять разработкой проекта и поддерживать целостность кодовой базы.

В рамках проекта активно поддерживается техническая документация, которая предоставляет подробную информацию о структуре, функциях и API библиотеки. Эта документация помогает разработчикам лучше понять функциональность и особенности реализации проекта, обеспечивая информационную прозрачность и доступность для всех участников процесса разработки.

README файл представляет собой основной источник информации о проекте для пользователей. В нем содержится описание проекта, инструкции по установке, примеры использования и контактная информация. Это обеспечивает удобство работы с библиотекой и помогает пользователям быстрее начать использовать ее в своих проектах.

Весь код проекта снабжен комментариями, которые помогают разработчикам понять структуру и логику работы каждой части программы. Комментарии облегчают поддержку и дальнейшее развитие проекта, так как предоставляют контекст и объяснения к каждому участку кода.

**8 Оформление багов и фич**

Для эффективного управления процессом исправления багов и реализации новых функций в проекте используется подход с созданием новых веток в системе контроля версий Git. Этот метод позволяет изолировать изменения, обеспечивая прозрачность и контроль над процессом разработки.

Создание веток для багов.

При обнаружении бага необходимо создать новую ветку в Git, которая будет посвящена исправлению данной проблемы. Это помогает изолировать изменения, связанные с багом, от основной ветки разработки, что минимизирует риск внесения непредвиденных ошибок в кодовую базу. Рекомендуется следовать следующей последовательности:

1. создание ветки: ветка создается из основной ветки (обычно main или master). Название ветки должно отражать суть бага и может включать уникальный идентификатор задачи из системы управления задачами;
2. исправление бага: внесение изменений в код для исправления обнаруженной проблемы. Важно снабдить изменения комментариями и при необходимости добавить новые тесты, подтверждающие исправление бага;
3. создание pull request: после завершения работы над исправлением создается pull request (PR) для объединения изменений с основной веткой. В PR необходимо описать суть исправления и ссылаться на соответствующую задачу в системе управления задачами;
4. код ревью и тестирование: команда проводит код ревью и тестирование изменений. Это позволяет убедиться в том, что исправление не приводит к новым ошибкам и не нарушает существующую функциональность;
5. объединение изменений: после успешного прохождения код ревью и тестирования изменения объединяются с основной веткой.

Создание веток для фич.

Для добавления новой функциональности также создается отдельная ветка в Git. Это обеспечивает независимость разработки новых функций от основной ветки, позволяя разрабатывать и тестировать новые возможности без риска нарушения текущей стабильной версии проекта. Процесс создания веток для фич включает следующие шаги:

1. создание ветки: ветка создается из основной ветки. Название ветки должно быть информативным и отражать суть добавляемой функциональности;
2. разработка фичи: внесение изменений в код для реализации новой функциональности. Важно следить за чистотой кода, снабжать его комментариями и добавлять необходимые тесты;
3. создание pull request: по завершении работы над функциональностью создается pull request для объединения изменений с основной веткой. В PR должно быть подробно описано, какая функциональность была добавлена и как она была реализована;
4. код ревью и тестирование: команда проводит код ревью и тестирование новой функциональности. Это позволяет убедиться в том, что новая функция работает корректно и не нарушает существующую функциональность;
5. объединение изменений: после успешного прохождения код ревью и тестирования изменения объединяются с основной веткой.

Общие рекомендации.

Рекомендуется делать частые и небольшие коммиты, описывающие конкретные изменения. Это облегчает отслеживание истории изменений и упрощает процесс код ревью. Сообщения к коммитам должны быть информативными и краткими, отражая суть внесенных изменений. Важно соблюдать общий стиль кода проекта, что делает код более читабельным и поддерживаемым. Все изменения должны сопровождаться соответствующими тестами, что позволяет поддерживать высокое качество кода и предотвращать регрессии.

**9 Часто задаваемые вопросы (FAQ)**

Вопрос 1: Как установить библиотеку?

Ответ: Для установки библиотеки необходимо клонировать проект с GitLab и установить все зависимости из файла requirements.txt. Подробные инструкции приведены в разделе "Установка" документации.

Вопрос 2: Как использовать основную функцию библиотеки?

Ответ: Основная функция библиотеки highlight\_hyperlinks принимает на вход необработанный текст и возвращает его с разметкой гиперссылок в формате HTML. Подробные примеры использования приведены в разделе "Использование" документации.

Вопрос 3: Какие форматы гиперссылок поддерживаются?

Ответ: Библиотека поддерживает распознавание и разметку телефонных номеров, адресов электронной почты, названий сайтов и URL. Все поддерживаемые форматы описаны в разделе "Введение" документации.

Вопрос 4: Как добавить новую функциональность в библиотеку?

Ответ: Для добавления новой функциональности рекомендуется создать отдельную ветку в Git, внести необходимые изменения и затем создать pull request для их объединения с основной веткой. Подробный процесс описан в разделе "Оформление багов и фич" документации.

Вопрос 5: Как сообщить о баге?

Ответ: Для сообщения о баге следует связаться с разработчиками библиотеки. После этого создается новая ветка в Git для исправления бага. Подробные инструкции приведены в разделе "Оформление багов и фич" документации.

Вопрос 6: Где можно найти примеры использования библиотеки?

Ответ: Примеры использования библиотеки можно найти в файле testing.py, который содержит наглядные примеры работы с библиотекой. Также можно обратиться к папке texts, где хранятся примеры текстов для тестирования.

Вопрос 7: Как обновить библиотеку до новой версии?

Ответ: Для обновления библиотеки до новой версии следует клонировать актуальную версию проекта с GitLab и обновить зависимости, установив их из обновленного файла requirements.txt.

Вопрос 8: Какой формат используется для промптов?

Ответ: Все промпты для взаимодействия с моделью ChatGPT хранятся в файле my\_prompts.py. Этот файл содержит текстовые шаблоны, которые используются для генерации запросов к модели.

Вопрос 9: Как связаться с разработчиками библиотеки?

Ответ: Контактная информация для связи с разработчиками указана в файле README. Вы можете отправить письмо на указанный адрес электронной почты.