Создание предметного указателя в индизайне

В прошлом веке, во время использования горячего набора, предметный указатель был в большинстве исторических и технических книг. Это было в порядке вещей, несмотря на то что требовало недели если не месяцы кропотливой ручной работы с текстом гранок.

Сейчас вёрстка компьютерная, и вроде как работать с текстом можно эффективнее. Это верно, но книги с предметными указателями не стало особо легче делать.

Программа индизайн, работа которой тут будет обсуждаться, позволяет делать индексы, но интерфейс ввода терминов для разных уровней индексации продуман только для случаев, когда в словах нет окончаний. Это нормально для английского, немецкого, но с нашими падежными формами такой подход не годится. В результате в русских книгах предметные указатели иногда есть, но если термины только в именительном падеже, то такой указатель не особо полезен.

Единственное потенциально работающее решение — использовать grep-запросы, имитирующие изменение слов по падежам. Но и тут не всё просто.

Над указателем должны вместе работать редактор и верстальщик. Редактор определяет, что должно быть в указателе, а верстальщик должен понимать возможные проблемы создания конкретного предметного указателя и уметь их решать.

Проблемы создания указателей

Будем называть терминами отдельные записи предметного указателя, неважно какой он по сути — предметный, именной, географический, и т.д.

Проблемы создания русскоязычных указателей следующие.

1) Совпадение слов в разных терминах. Например, в словнике могут быть термины *Сопротивление* и *Удельное сопротивление*. Если сперва искать в тексте падежные формы термина *Сопротивление*, то в выборку попадут и слова, перед которыми есть слово *удельное*.

2) Другой примечательный пример при работе с именными указателями — фамилия *Александрович*. Она всегда в первых строках словника, и если она уйдёт в обработку первой, то отметится во всех отчествах *Александрович*, что абсолютно неправильно. Это тоже задача пользователя — обработать все термины с отчеством *Александрович* до того как будет обработана фамилия *Александрович* — т.е. подтверждение того, что специалист, собирающий индекс, должен знать текст.

3) Одинаковые названия, имеющие разное значение. Вот пример из словника географических терминов:

Брянск I, станция, Брянская обл., РСФСР
Брянск II, пос., Брянская обл., РСФСР
Брянск, г., РСФСР
или

Биржайская волость, Биржайский уезд, Литовская ССР
Биржайский уезд, Литовская ССР

Надо иметь возможность при поиске отличать Брянск I, Брянск II и Брянск город. Тоже самое с волостью и уездом.

4) Разные написания названия города или фамилии в рамках одного документа.

Вот примеры городов:

Йоэнсуу (Иоэнсуу), г., Финляндия
Святнаволок (Свят–Наволок), с., КФССР

А это варианты фамилий:

Саррай (Саррайль) Морис
Хаккы–паша (Хакки–паша) Ибрагим

Тут не принимаются глубокомысленные сентенции де «документы должны быть оформлены единообразно...», если это оцифрованные сканы бумаг, то там важно оставить всё так, как было при создании данного документа. Но у нас должен быть такой инструмент, для которого не проблема искать термины с разными вариантами написания.

5) Содержащиеся в термине **фамилия имя отчество** обычно бесполезны для поиска. В подавляющем большинстве случаев в тексте фигурируют только фамилии. Как в таких случаях искать именно по фамилиям, не помещая в запрос поиска имя и отчество?

Здесь даже не упомянуто, что для учёта падежей должна быть грамотная генерация grep-запросов, исходя из того, на какую букву заканчивается слово. Это само собой разумеющееся требование.

Как правильно подойти к решению

Можно добавлять термины в указатель так — идти по тексту и выбирать нужные слова. Наверное, это приемлемо, когда, например, в тексте какое-то слово встречается много раз, а для индекса надо только несколько этих слов из конкретных абзацев.

Решение, реализованное тут, берет на себя поиск всех терминов, во всех падежных формах. Но и описанный в предыдущем абзаце подход тоже реализован как частный случай (с. 17).

Нужно, чтобы редактор книги подготовил отдельный текстовый файл со всеми терминами, которые должны быть в указателе, и визуально отметил, на каком уровне иерархии каждый из них.

Например, для книги по астрономии в этом файле может быть такой текст:

Планеты
    Венера
    Земля
    Марс
    ...
Созвездия
    Большая Медведица
    Кассиопея
    Лебедь
    ...
Звёзды с именами
    Альтаир
    Вега
    Денеб
    ...

Тут термины **Планеты, Созвездия, Звёзды с именами** — это первый уровень, а названия астрономических объектов — второй. И для терминов второго уровня будут искаться номера страниц.

Вот пример именного указателя, в котором все записи первого уровня:

Авксентьев Николай Дмитриевич
Аксельрод Александр Ефремович
Александра Федоровна
Алексеев Михаил Васильевич
Анастасов Теодор
Андроников Михаил Михайлович
Антонов-Овсеенко Владимир Александрович
Астафьев Николай Анатольевич
Багратуни Яков Герасимович
Балабанов Маврикий Леонтьевич
Батолин Петр Петрович
....

Такого текстового файла достаточно для создания в индизайне многоуровневого предметного указателя для книги, по которой данный список создан, для этого есть комплект скриптов, с их помощью будут собираться номера страниц, где есть термины, неважно в каком падеже они в тексте.

Дальше дело только в грамотном использовании этих программ. Вот, шаг за шагом, как из таких текстовых файлов получается предметный указатель.

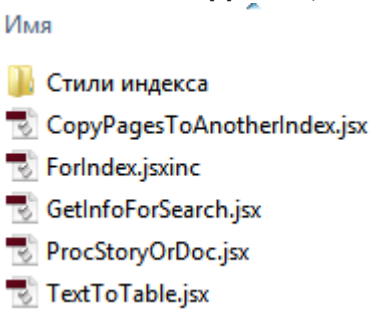
Стилевое оформление списка

Упомянутый в предыдущем пункте текстовый список с визуальным выделением уровней иерархии индекса надо превратить в indd-файл, в котором каждый уровень отмечен своим абзацным стилем.

Для этого надо сделать следующее:

1) поместить в indd-файл этот список;

2) открыть файл **Стили индекса.indd**, в папке со скриптами он в каталоге **Стили индекса**;

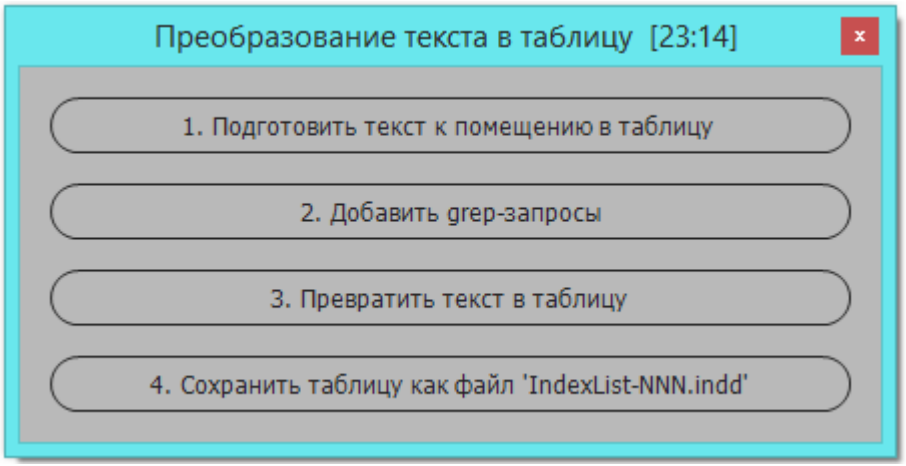


В этом файле есть один текстовый фрейм со всеми уровнями. Скопируйте его и поместите в файл со списком, и сразу удалите. В работе появится папка абзацных стилей **#IndexStyles**, в ней четыре абзацных стиля **#Level1** – **#Level4** и служебный стиль **#PageShow.**

3) отметить все строки текстового списка абзацными стилями, соответствующими их уровням иерархии.

Получение таблицы «Термин – grep»

Теперь из этого indd-файла с отмеченными уровнями иерархии терминов надо сделать grep-запросы для поиска каждого термина. Это выполняется при помощи скрипта **TextToTable.jsx**, вот его рабочее окно:



Назначение кнопок понятно из названия. Это на картинке они все одинаковой яркости. А при использовании они доступны по очереди. В результате работы скрипта создаётся трёхколонник, в левой колонке термины, во второй буква №, в правой колонке grep-команды для поиска текста первой колонки.

Буква № — указатель, надо ли искать текст данного термина. При запуске скрипта она ставится во всех строках. Потом пользователь может убрать эту букву из строк, для которых номера страниц искать не надо.

В главе «Проблемы создания русского индекса» в п. 5 отмечено, что полная информация о фамилии, имени, отчестве бесполезна для поиска. Но в данный момент задача сделать полный вариант grep-запроса, настойка его под конкретную задачу будет выполняться позже. Используемые правила создания grep-запросов приведены на врезке «GREP-запросы для разных случаев окончания слов».

GREP-запросы для разных случаев окончания слов

Отдельное слово термина обозначим как **word**, граница слова — это **\b**.

Если слово оканчивается на согласную букву, то запрос поиска вариантов слова такой:

\b + word + [а-я]{0,2}\b

Если слово оканчивается на гласную букву, то **word** содержит все буквы слова, кроме последней (назовём этот случай **partOfWord**), и поиск падежных форм слова выполняется так: **\b + partOfWord + [а-я]{1,2}\b**

Если слово оканчивается на 'й', то в **partOfWord** будут все буквы, кроме двух последних, и поиск выполняется так: **\b + partOfWord + [а-я]{2,3}\b**

Слова, оканчивающиеся на мягкий знак, цифру, или любую букву, отсутствующую в русском алфавите, помещаются в строку grep-поиска как есть, только перед ними ставится код **\b**.

Какой оператор использовать для поиска шпаций, зависит от используемой версии индизайна и определяется в служебном текстовом файле **ForIndex.jsxinc**. Это переменная **sepSpace**, по умолчанию её значение **'\\h'**, что подходит для версии CS6 и следующих. Для предшествующих версий надо самим выбрать вариант **[[:blank:]]**.

В процессе работы над процедурой превращения текста в таблицу использовались разные подходы, и тестировалось время выполнения задания. Вывод информации о времени преобразования, числе строк таблицы и темпе обработки (число миллисекунд на обработку одной строки) по умолчанию отключён, это переменная **showTime** в служебном текстовом файле **ForInDex.jsxinc**.

Для справки, на моей небыстрой машине предметный указатель в 379 строк превращается в таблицу за 21 секунду при этом темп 55,7 миллисекунд на строку; 2269 строк потребуют 2 минуты 19 секунд, и темп будет 61,6 миллисекунд на строку. Файл в 3781 строку обработан за 4 мин 13 сек, темп 74,9 миллисекунд на строку. Зависимость времени обработки от объёма похожа на степенную функцию.

Таблица всегда сохраняется с одним и тем же именем **IndexList-nnn.indd**, тут **nnn** — это порядковый номер.

Одинаковое имя нельзя считать недостатком, т.к. эти файлы сохраняются в папке, где размещена работа, для которой делается индекс. Номер это или варианты таблицы, или, если, например, в одной работе два вида указателей, то таблицы для географического и именного указателей должны иметь разные номера.

Итак, словник превращён в список терминов, термины выстроены по алфавиту, для каждого термина есть grep-запрос его поиска. Как этим распорядиться для получения предметного указателя?

Основная программа для индексирования

Все перечисленные ранее проблемы составления разных видов предметных указателей на русском языке можно решить, если иметь возможность выбирать одну или несколько строк из таблицы, определять диапазон поиска — вся статья / документ или только выделенная область, уточнять символьные и/или абзацные стили для искомого термина. Все эти возможности собраны в программе **ProcStoryOrDoc.jsx**, её рабочие окна показаны на следующей странице.

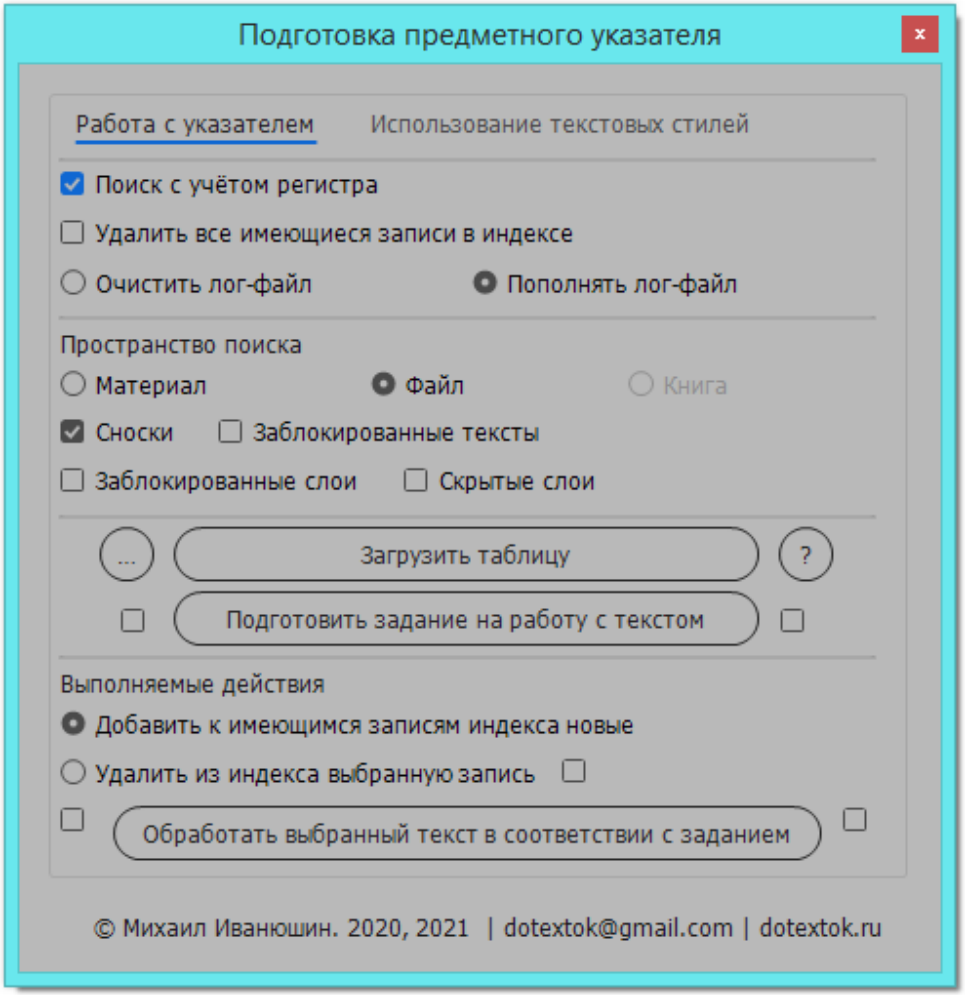
Тут две вкладки **Работа с указателем** и **Использование текстовых стилей**.

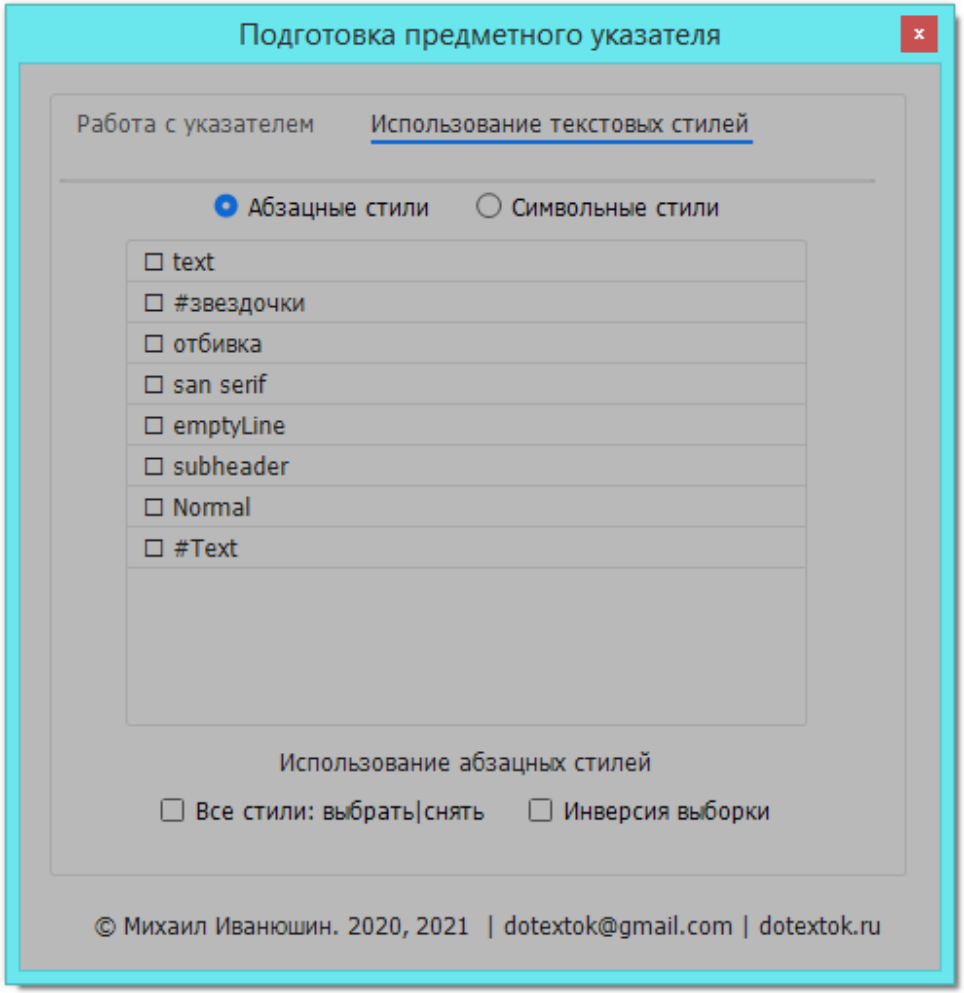
Назначение почти всех опций понятно из их названия. Как работает флажок **Поиск с учётом регистра**, объяснено во врезке на следующей странице.

Самое важное на вкладке **Работа с указателем** — возможность загружать таблицу, выбирать из неё нужные строки для обработки, это выполняет кнопка **Подготовить задание на работу с текстом**.

Если требуется уточнение, каким символьным/абзацным стилем оформлен искомый термин, то для этого предусмотрена вкладка **Использование текстовых стилей**.

Да, вы можете достаточно быстро обработать сложный словник, если сначала пройдете все особые случаи, например, упоминавшаяся ранее необходимость обработать сперва термин *Удельное сопротивление*, а потом уже *Сопротивление*. Или случай с Александровичем, сначала надо найти все отчества и обработать эти записи. Каждая запись окрашивается красным





Поиск с учётом регистра

В приводившихся примерах словников есть слова с прописными буквами, но будут они учитываться или нет, определяет флажок **Поиск с учётом регистра** на вкладке **Работа с указателем**. Вот как он работает.

Допустим, есть текст «Белый снег и белый мел, белый заяц тоже бел».

Тогда, если флажок установлен, в строке grep-поиска это будет обозначаться оператором **(?-i)**, поиск **(?-i)Белый** найдёт только одно слово «Белый». Запрос **(?-i)белый** в той же строке найдёт два слова «белый».

Если флажок сброшен, т.е. ищем без учёта регистра, это grep-оператор **(?i)**, то любой из запросов **(?i)Белый** или **(?i)белый**, и даже **(?i)БЕЛЫЙ** найдёт и «Белый», и «белый».

цветом. Этот цвет, во-первых, запрещает индексировать слово, именно поэтому в окрашенном *удельном сопротивлении* слово *сопротивление* отмечено уже не будет. Во-вторых, легко просматривать текст в поиске пропущенных имён, названий и пр.

И попавшие в задание записи, они могут как быть добавлены в индекс, так и исключены из него, всё определяется радиокнопкой блока **Выполняемые действия**.

Осталось понять, как настраивать таблицу с grep-запросами для разных случаев подготовки предметного указателя. Это решается скриптом **GetInfoForSearch.jsx**. Его можно запустить в любой момент, когда открыт файл с готовой таблицей. Справа его рабочее окно.

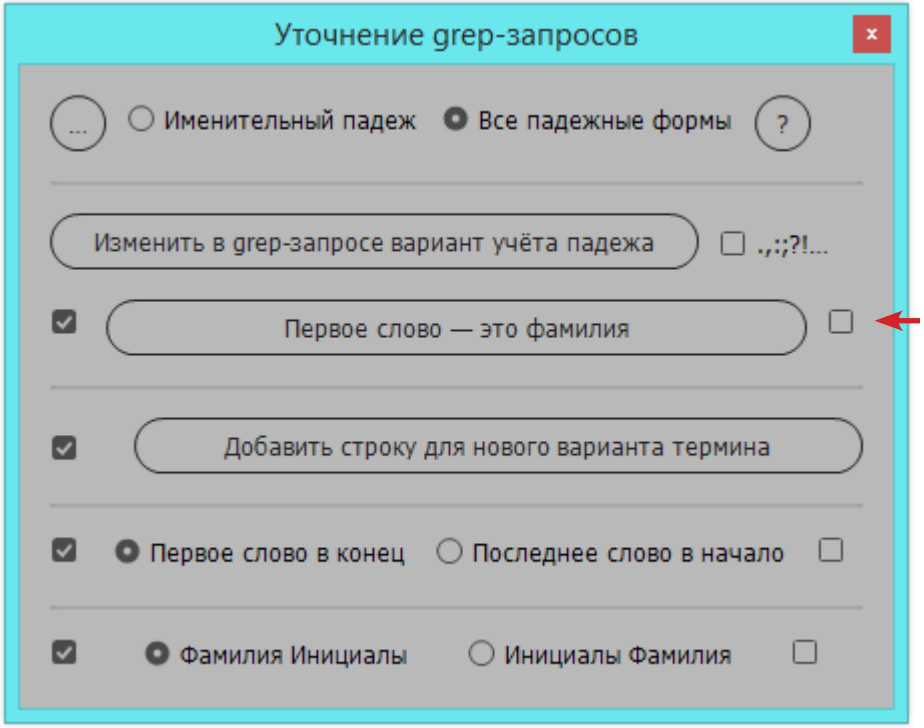
При создании таблицы все термины оформляются так, что можно искать их во всех падежных формах. Например, термин *Нижний Новгород* в третьем столбце будет указан так:

\bНижн[а-я]{2,3}\b\h\bНовгород[а-я]{0,2}\b

Но можно изменить запрос, чтобы он был только в именительном падеже. Выбор варианта учёта падежа определяется радиокнопками, а по нажатию кнопки **Изменить в grep-запросе вариант учета падежа** в третьей колонке будет термин в соответствии с сделанным выбором. Вот как изменится grep-запрос для поиска термина *Нижний Новгород*, если был выбран вариант **Только именительный падеж:**

\bНижний\b\h\bНовгород\b

Флажок справа от этой кнопки определяет, останутся ли знаки пунктуации в запросе поиска. Для поиска по фамилии или названию города они не нужны.



Поиск вариантов одного термина

В окне уточнения grep-запросов, выводимого скриптом **GetInfoForSearch.jsx**, есть решение для случая, когда в словнике отмечено, что термин может иметь два варианта написания, см. пункт 4 в главе «Проблемы создания русского индекса». И это ещё лёгкий случай. В книге о полководцах Барклай-де-Толли искался как:

- Барклай-де-Толли
- Барклай
- Михаил Богданович

естественно, во всех падежах, три варианта поиска для одного термина.

Эта задача решается кнопкой **Добавить строку для нового варианта термина**. В добавленной строке в первой ячейке останется тот же термин. Если флажок слева от этой кнопки установлен, то текст в левой ячейке не будет виден. В правой ячейке вы поместите новые варианты поиска этого термина.

Тут можно быстро подготовить разные варианты для поиска, этот скрипт делался в первую очередь для работы с фамилиями и именами. В словнике принято указывать первым фамилию, потом имя и отчество, а в тексте обычно только фамилия. Для таких случаев предусмотрен флажок-кнопка, оставляющий в тексте третьей ячейки только первое слово. На картинке с. 7 он справа от кнопки **Первое слово — это фамилия** и отмечен красной стрелкой.

Поменять в новом варианте термина имя и фамилию местами, поставить фамилию после имени отчества, превратить имя отчество в инициалы — все эти операции подготовки данных для grep-поиска выполняются кнопками окна уточнения grep-запросов.

ⓘ Когда открыто окно подготовки предметного указателя, скрипт **GetInfoForSearch.jsx** можно вызвать флажком-кнопкой слева от кнопки **Подготовить задание на работу с текстом** см. с. 6.

Когда будете выбирать строки для подготовки очередного задания, выделяйте их целиком.

Работа с текстовыми стилями

На вкладке **Использование текстовых стилей** есть списки стилей, перед каждых названием флажок, и два флажка внизу окна — для выбора/сброса и инверсии выбора.

Выборка названия стиля всегда выполняется *двойным* щелчком, а нижние флажки переключаются *одиночным* щелчком. Обратите на это внимание.

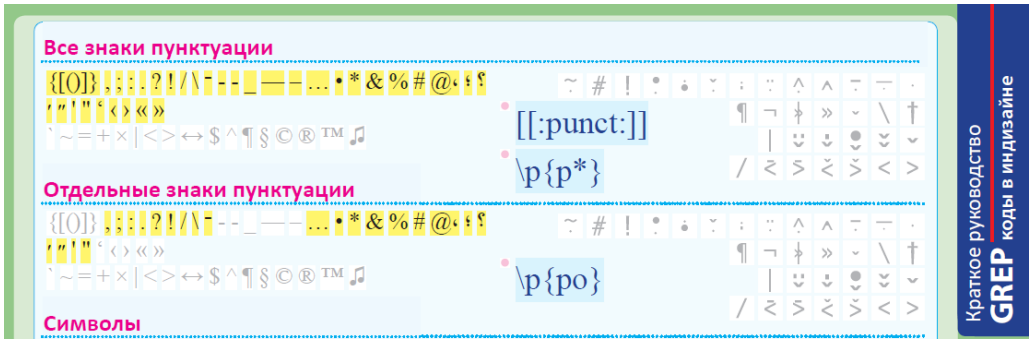
Особые случаи

Это в большей мере касается именных указателей, если там есть фамилии, оканчивающиеся на -е, -и, -о, -у, -э, -ю (Гаридзе, Иоселиани, Лещенко, Рытхэу, Друцэ, Цзю), то в большинстве случаев они не имеют падежных форм, и это надо учесть — поправить grep-запрос. Но бывают исключения, тут опять повторюсь, что редактор должен быть в теме, например, в документах первых лет после революции фамилия Родзянко почему-то нередко склонялась: Родзянке, Родзянкой.

Результаты индексирования желательно просматривать, благо что они окрашены красным цветом. Могут быть неожиданности. Например, в словнике была *Франц Мария*. Учёл, что фамилия не склоняется. В grep-поиске **\bФранц\b**, т.е. ищется граница слова: после буквы пробел, перевод строки или знак пунктуации. И после обработки текста неожиданно вижу:

Франц[узский] посол [Ж.] Нуланс не уезжает — искомое слово и открывающая квадратная скобка, хотя в запросе

определено, что ищется целое слово. Теоретически это не ошибка, а особенность инструмента grep, он считает скобки знаками пунктуации.



(Картинка выше — это экранная ссылка на русскоязычный документ по grep-кодам)

Но в указатель такие случаи попадать не должны.

Если в индексе номера страниц не нужны

На второй странице есть пример двухуровневого указателя. Там **Планеты**, **Созвездия**, **Звёзды с именами** — это первый уровень, просто название, для них номера страниц искать не надо. А вот названия астрономических объектов должны иметь номера страниц. Указателем, что номера надо искать, служит буква **№** во второй ячейке строки. При создании таблицы она есть везде, и уже пользователь должен убрать её из тех строк, для которых номера страниц искать не надо.

Начинаем собирать указатель

Думаю, уже сложилось понимание, что сперва надо обработать сложные случаи, когда надо в таблице выбрать одну-две строки и в окне **Сбор индексного указателя** нажать на кнопку **Подготовить задание на работу с текстом.**

Задание включает в себя следующее:

- из информации, какой абзацный стиль текста в первой колонке, определяется уровень индекса;
- по содержимому второй ячейки выясняется, надо ли grep-запросами искать номера страниц;
- образец grep-запроса из третьей колонки дополняется информацией, учитывать или нет регистр при поиске.

Это определяет пользователь флажком **Поиск с учётом регистра** на вкладке **Работа с указателем**, и он должен быть установлен до нажатия кнопки **Подготовить задание на работу с текстом.**

Если для поиска требуется определить, какие текстовые стили надо использовать, то это тоже в руках пользователя, вкладка **Использование текстовых стилей.** На этой вкладке сплывающие подсказки радиокнопок **Абзацные стили** и **Символьные стили** показывают, стили какого документа используются. Если уже открыт другой файл, то повторный выбор радиокнопок переключит скрипт на работу с текстовыми стилями нового документа. И когда всё настроено, нажать кнопку **Обработать выбранный текст в соответствии с заданием.**

Маловероятно, что у вас *все* строки будут особенными. Достаточно быстро наступит момент, когда захочется

выделить всю таблицу и отдать её скрипту на обработку. Но как обеспечить, чтобы пропускались уже обработанные строки?

Обрабатывать только один раз

Чтобы исключить выбор уже обработанной строки, надо изменить цвет текста в третьей ячейке. Если он не чёрный, то строка пропускается. Сделайте привычкой сразу менять этот цвет, чтобы не терять время на попытки понять, что не так в указателе, повторная обработка уже обработанного текста может дать неожиданный результат.

ⓘ В программе подготовки предметного указателя для изменения текста в ячейке с grep-запросом предусмотрен флажок-кнопка справа от кнопки **Подготовить задание на работу с текстом**. Работает, когда на экране в файл с таблицей.

Уверенность в результате

Итак, текст обработан, предметный указатель сделан. Но все ли grep-запросы были безупречны? Совсем не в кайф ломать глаза, просматривая много раз весь текст в поисках возможных ошибок. От работы надо получать удовольствие, и проверка правильной работы grep-запросов не исключение.

Чтобы видеть, как выполнился каждый запрос, в процессе работы с документом ведётся регистрационный файл выполнения grep-запросов. Это текстовый файл,

Персонажи	№	\bПерсонаж[а-я]{1,2}\b
Старик	№	\bСтарик[а-я]{0,2}\b
Старуха	№	\bСтарух[а-я]{1,2}\b
Золотая рыбка	№	\bЗолота[а-я]{1,2}\b\h\bрыбк[а-я]{1,2}\b
Предметы быта	№	\bПредмет[а-я]{1,2}\b\h\bбыт[а-я]{1,2}\b
землянка	№	\bземлянк[а-я]{1,2}\b
невод	№	\bневод[а-я]{0,2}\b
пряжа	№	\bпряж[а-я]{1,2}\b
Старухины хотелки	№	\bСтарухин[а-я]{1,2}\b\h\bхотелк[а-я]{1,2}\b
корыто	№	\bкорыт[а-я]{1,2}\b
дворянка	№	\bдворянк[а-я]{1,2}\b
царица	№	\bцариц[а-я]{1,2}\b
владычица морская	№	\bвладычиц[а-я]{1,2}\b\h\bморска[а-я]{1,2}\b

его имя совпадает с названием работы, а последние буквы в имени **=log.txt**. Дальше он будет называться лог-файл.

В папке с примерами есть файл «Сказка о рыбаке и рыбке», там имеется текст предметного указателя, в нём два уровня.

Персонажи
старик
старуха
золотая рыбка
Предметы быта
землянка
невод
пряжа

Старухины хотелки
корыто
дворянка
царица
владычица морская

В видео показано, как из этого текста делается таблица, как программа собирает предметный указатель.

И попутно пишется лог-файл обработки grep-запросов. В него информация добавляется при каждом нажатии на кнопку **Обработать выбранный текст в соответствии с заданием.**

Сперва идут дата и время. В начале строки указатель действия: >> для случая, когда найденный текст помещается в индекс, и << для случая удаления найденного текста из индекса (добавление или удаление определяется радиокнопками блока **Выполняемые действия**), после двоеточия иерархические термины разных уровней, разделённые двумя вертикальными чертами, эту последовательность завершает обрабатываемый термин, указанный в первой ячейке строки, затем знак равенства как разделитель и использовавшийся grep-запрос, где отражено состояние флажка **Поиск с учётом регистра.** А дальше с новой строки результаты поиска, найденные термины в разных падежных формах, в конце каждой записи в квадратных скобках номер страницы.

Рабочий файл называется **Golden Fish.indd**, а лог-файл — **Golden Fish=log.txt.**

На странице 4 приведены правила создания grep-запросов для разных случаев окончания слов, на предыдущей странице таблица для поиска этих терминов. Искаться будут только те, где во второй колонке есть буква №.

И вот начало лог-файла обработки данной таблицы

```
-----|  Tue Oct 27 2020 22:02:11 GMT+0300  |-----
>> : Старухины хотелки||владычица морская = (?i)\bвладычиц[а-я]{1,2}\b\h\bморска[а-я]{1,2}\b
>> : Персонажи||Золотая рыбка = (?i)\bЗолота[а-я]{1,2}\b\h\bрыбк[а-я]{1,2}\b
        золотая рыбка [8]
        золотая рыбка [6]
        золотая рыбка [4]
        золотая рыбка [3]
        золотая рыбка [2]
        золотая рыбка [1]
>> : Предметы быта||землянка = (?i)\bземлянк[а-я]{1,2}\b
```

Как видите, тут не всё в порядке: термин *владычица морская* вообще не найден, при обработке термина *золотая рыбка* пропущены случаи золотую рыбку. Это по причине того, что не был учтён случай, что если в слове две последние буквы гласные, то шаблон поиска должен быть таким же как для слов, оканчивающихся на букву **й.** Это исправлено, теперь результат работы выглядит так:

```
-----|  Wed Oct 28 2020 00:35:56 GMT+0300  |-----
>> : Старухины хотелки||владычица морская = (?i)\bвладычиц[а-я]{1,2}\b\h\bморск[а-я]{2,3}\b
        владычицей морскою [10]
        владычицей морскою [9]
>> : Персонажи||Золотая рыбка = (?i)\bЗолот[а-я]{2,3}\b\h\bрыбк[а-я]{1,2}\b
        золотую рыбку [10]
        золотая рыбка [8]
        золотую рыбку [7]
        золотая рыбка [6]
        золотую рыбку [5]
        золотая рыбка [4]
        золотую рыбку [4]
```

золотая рыбка [3]
золотую рыбку [3]
Золотую рыбку [2]
золотая рыбка [2]
золотая рыбка [1]

И лог-файл можно распечатать, чтобы редактор проверил правильность помещения терминов в указатель.

Работа с лог-файлом

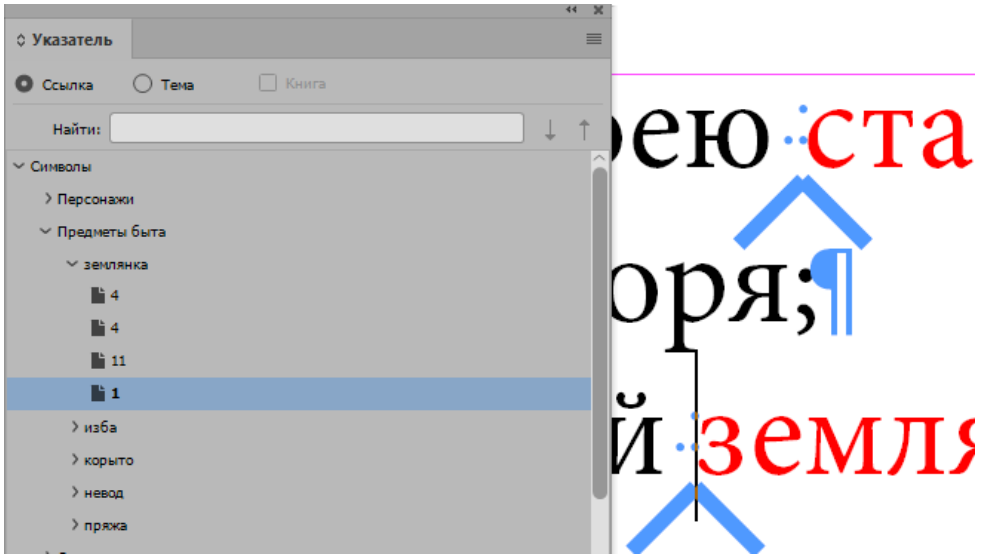
Наличие лог-файла, конечно, хорошее подспорье для внесения исправлений в grep-запросы по результатам текущей работы, чтобы встреченная сейчас ошибка больше не появлялась. Но в текстовом виде, как он есть, совсем неудобно переходить в текст, чтобы поработать с проблемным запросом. Можно, конечно, зная номер страницы, перейти на неё с помощью сочетания клавиш **Ctrl+J**.

Но есть способ лучше — скрипт **LogFileUsage.jsx**. При запуске он начнёт работать, если для открытого indd-файла есть его лог-файл. Будет создан indd-файл с именем рабочего файла, но оканчивающимся на **=log.indd**. В тестовом примере для рабочего файла **GoldenFish.indd**, при наличии созданного ранее текстового файла **GoldenFish=log.txt**, будет создан файл **GoldenFish=log.indd**. А если такой indd-файл уже есть, он будет открыт. В любом случае будут открыты два indd-файла — один с работой, второй с текстом лог-файла.

Для удобства работы в текстовый лог-файл теперь

Полная информация о термине в тексте

Слева от слова, попавшего в индекс, всегда есть маркер. Если его выделить, то в панели **Указатель** (**Index**) откроется ветка всей иерархии указателей для этого термина, и будет отмечена страница, где этот маркер был выделен. Это очень полезная опция, особенно для случаев, когда видишь ошибку применения grep-запроса. С её помощью можно быстро разобраться, что надо исправить.

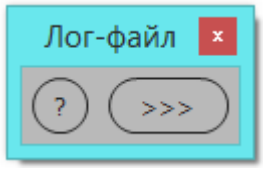


Чтобы после выделения в указателе страницы термина перейти в текст к маркеру используйте команду **Перейти к выделенному маркеру**, она в выпадающем меню панели **Указатель**.

помещается не только номер страницы термина, но и идентификатор индексного маркера, стоящего слева от термина. Например, так:

```
>> Старухины хотелки||царица = (?i)\bцариц[аейямихаыуо]*\b
        царицей [10]  id=11270
        царицей [9]  id=11271
```

Как только на экране появятся вот эти две кнопки:



значит, **LogFileUsage.jsx** готов к работе.

Когда активно окно с лог-файлом, перед нажатием кнопки [>>>] надо поставить курсор в строку, где есть термин и после него номер страницы в квадратных скобках и идентификатор позиции термина.

После нажатия кнопки термин этой строки будет в центре экрана с увеличением 300%, курсор будет мигать на индексном маркере данного термина.

Когда на экране страница вёрстки, нажатие на эту кнопку выведет на экран лог-файл. Строка, из которой только что перешли в рабочий файл, будет окрашена цветом LogLineColor.

Если термин в фрейме на рабочем столе

Ну в общем-то это ситуация, которой в свёрстанном тексте быть не должно — текстовые фреймы на полях. Но если такие фреймы есть и в них находится термин, то при его обнаружении скрипт помещает в лог-файл обрабатываемый термин, после него вместо номера страницы [**???**], и переходит к обработке следующей записи.

```
-----|  Thu Oct 29 2020 01:37:48 GMT+0300  |-----
>> : Волконская = (?-i)\bВолконск[а-я]{2,3}\b
        Волконская [ ??? ]
        Волконскую [213]
        Волконская [213]
```

Опыт настройки grep-запросов

В первой версии диапазон возможных букв в конце слова определялся так: [а-я], это вот и в примере несколько строк выше. И с таким диапазоном была головная боль, что ищешь **Петров[а-я]{0,2}** и в лог-файле находишь в примерах обработки этого grep-запроса Петрович, Петровна. С **Павлов[а-я]{0,2}** была та же проблема: находились Павловна, Павлович. В результатах обработки запроса **Корнилов[а-я]{0,2}** нашёлся Корниловец.

И хоть лог-файл помогает находить такие огрехи, хотелось бы, чтобы таких случаев вообще не было. Проблема появления Петровны, Павловича, Корниловца и пр. обусловлена большой информационной избыточностью набора [а-я], ну действительно, тут все буквы алфавита. В поиске решения я разделил слова на группы по вариантам окончания, чтобы у каждой группы были в наборе только те буквы, которые могут появиться. Вот какие группы получились:

Обозначение трёх последних букв слова: **s** — согласная, **g** — гласная, **m** — мягкий знак, **t** — твёрдый знак.

ssg = «йоувамыихея»; // мечта кровля рикша вопли мосты ‖ **мечТА: \bмеч[тайоувамыихея]+\b**
ssm = «июяем»; // честь ‖ **честЬ: \bчест[ьиюяем]+\b**
gsm = «июяеём»; // тень ‖ **тЕНЬ: \bт[еньиюяеём]+\b**
msg = «аыеуовмийх»; // кольцо ходьба деньги Ванька ‖ **колЬЦО: \bкол[ьцоаыеуовмийх]+\b**
smg = «иеюёйявмх»; // ладья листья друзья ‖ **ладЬЯ: \bлад[ьяиеюёйявмх]+\b**
sgg = «йяогюемуивхё»; // милая светлый каравай караваи края ‖ **милАЯ: \bмил[аяйяогюемуивхё]+\b**
gsg = «ейямихаыуо»; // вода воевода олово поля вдовы ‖ **воДА: \bво[даейямихаыуо]+\b**
smg = «ейиюё»; // семья ‖ **семьЯ: \bсемь[яейиюё]+\b**
sgs = «мыауое»; // Петров ‖ **\bПетров[мыаоуе]*\b**
sms = «ауоме»; // Гарольд ‖ **\bГарольд[ауоме]*\b**
mgs = «ауоме»; // батальон ‖ **\bбатальон[ауоме]*\b**
gss = «ауоме»; // торт пациент ‖ **\bторт[аоуме]*\b**
sss = «аоуме» ; // текст контекст ‖ **\bтекст[аоуме]*\b**
ggs = «мыауое»; // Малеев ‖ **\bМалеев[мыауое]*\b**
tgs = «ауоме»; // объём подъём ‖ **\bобъём[ауоме]*\b**

Группы имеют свои наборы, и в них букв намного меньше, чем в стандартном [а-я]. У групп слов, оканчивающиеся на согласную, всегда один и тот же набор.

Вверху таблица со страницы 10, когда grep-запросы сделаны с новыми наборами.

Приведённая выше информация, какие буквы входят в эти группы, не самая последняя. Эти данные периодически обновляются, и последняя версия содержится в файле настроек **ForIndex.jsxinc**.

Персонажи	№	\bПерсонаж[иейямихаыуо]*\b
Старик	№	\bСтарик[мауое]*\b
Старуха	№	\bСтарух[аейямихаыуо]*\b
Золотая рыбка	№	\bЗолот[аяайяогюемуивхё]+\b\h\bрыбк[айоувамыихея]*\b
Предметы быта	№	\bПредмет[ыейямихаыуо]*\b\h\bбыт[аейямихаыуо]*\b
землянка	№	\bземлянк[айоувамыихея]*\b
невод	№	\bневод[мауое]*\b
пряжа	№	\bпряж[аейямихаыуо]*\b
Старухины хотелки	№	\bСтарухин[ыейямихаыуо]*\b\h\bхотелк[ийоувамыихея]*\b
корыто	№	\bкорыт[оейямихаыуо]*\b
дворянка	№	\bдворянк[айоувамыихея]*\b
царица	№	\bцариц[аейямихаыуо]*\b
владычица морская	№	\bвладычиц[аейямихаыуо]*\b\h\bморск[аяайяогюемуивхё]+\b

Выбор, какие наборы использовать — стандартный [а-я] или отдельные для каждой группы слов — определяется переменной **simpleSearch** в файле **ForIndex.jsxinc**. Если она равна **true**, то используется [а-я], если **false** (это значение по умолчанию), то наборы отдельных групп.

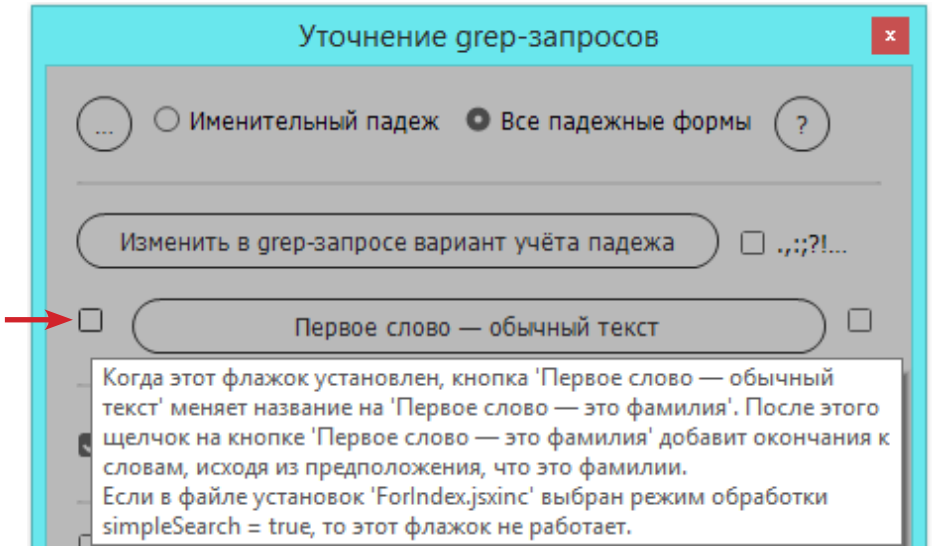
Поиск фамилий

В главе **«Опыт настройки grep-запросов»** рассказано о подготовке grep-запросов, учитывающих особенности русского языка, и это решение заметно помогло в поиске терминов в разных падежах.

Но оно потребовало доработки, если искать этими запросами фамилии. Надо было учесть два момента: во-первых, есть фамилии, которые не имеют падежных форм, во-вторых, есть двойные фамилии. И эти ограничения были легко сняты.

В файле **ForIndex.jsxinc** есть и grep-запросы, и код для обеих ситуаций, когда в тексте для указателя обычные слова, и когда фамилии. Но как сообщить программе

уточнения grep-запросов, какой из вариантов использовать? Для решения этого вопроса в интерфейс программы **GetInfoForSearch.jsx** добавлен флажок, определяющий, какой это текст:



Скрипт проверяет этот флажок и использует один из вариантов в зависимости от его установки.

В этом подходе заключена та важная гибкость, требующаяся для подготовки предметного указателя: когда флажок сброшен, мы можем искать в тексте одно-два-три слова в разных падежах, ну в общем, все появления многословного термина. А если в словнике фамилия имя отчество, то в большинстве случаев результативен только поиск по фамилии, и запрос этого поиска должен быть правильным. И задача создания запроса, учитывающего все особенности фамилий на русском языке, тут решена, пожалуй, для всех случаев. Из известных мне не охвачен вариант, когда фамилия кончается на -ких, -пых, -тых (Мягких, Тупых, Толстых), мне кажется, что они тоже не склоняются независимо от того кто это — мужчина или женщина. Но если это не так, то grep-запрос для этих случаев несложно добавить в код файла **ForIndex.jsxinc**.

Ещё вопрос — фамилия Павел (или Орел). Мужчина по имени Павел, и фамилия такая же, как *фамилия имя* должны быть написаны в родительном падеже? **Певела Павла** или **Павла Павла**? Тему падежей фамилий, оканчивающихся на -ел, оставим филологам, а скрипт находит оба варианта.

Гораздо большая проблема — однофамильцы.

Работа с однофамильцами

На левой картинке кнопка [**?**] — это помощь в поиске однофамильцев в упорядоченном по алфавиту тексте: ставишь курсор в текст, и скрипт начинает искать однофамильцев, начиная с той строки, где стоит курсор. В расчёт берутся только строки чёрного цвета, т.е. не обработанные ранее. Найденные строки выделяются и выводится сообщение о завершении поиска.

После этого надо пройти по всем однофамильцам, каждый из них может быть в тексте не один раз, отметить их всех несовпадающими символьными стилями, а потом выбрать каждую строку, указать для неё, какой символьный стиль использовать (см. страница 6, правый рисунок), и обработать.

Да, это кропотливая работа, но она с предсказуемым результатом и выполняется один раз.

Частичное совпадение терминов

О том, что возможно частичное совпадение терминов, было сказано на первой странице этого руководства. Было сказано как о проблеме, и теперь смотрите — как она решена. Для начала — текст

Певец Александрович
Царь Павел Александрович

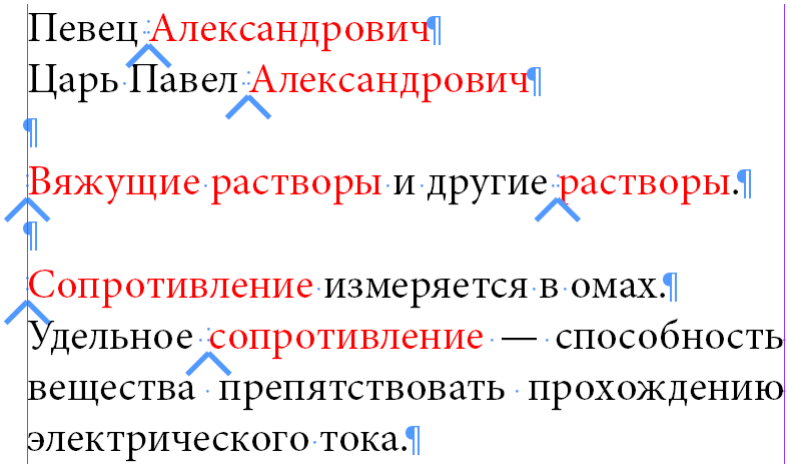
Вяжущие растворы и другие растворы.

Сопротивление измеряется в омах.
Удельное сопротивление — способность вещества препятствовать прохождению электрического тока.

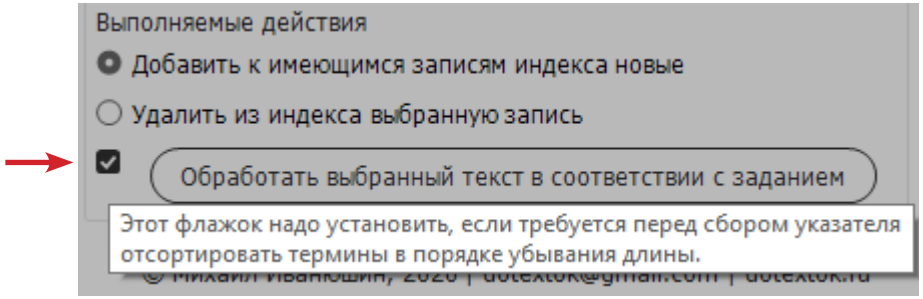
и таблица

Дополнительно
Александрович
Павел Александрович
Вяжущие растворы
Растворы
Сопротивление
Удельное сопротивление

Очевидно, что если обрабатывать текст в очерёдности терминов этой таблицы, то получится вот что:



правильно обработаны только Вяжушие растворы и растворы. Термины Павел Александрович и Удельное сопротивление в указатель не попали, т. к. в их составе оказались другие, более короткие термины, которые обработаны раньше. Решением этой частной задачи будет предварительная сортировка терминов в порядке убывания длины: флажок слева от кнопки **Обработать выбранный текст в соответствии с заданием**, если он установлен перед нажатием этой кнопки, то данные сперва будут отсортированы в порядке убывания длины терминов, а потом обработаны.



Это исключит данную ошибку, и результат будет таким:

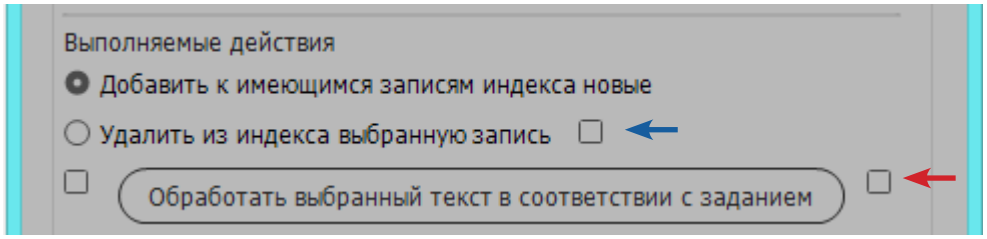
Певец Александрович¶
Царь Павел Александрович¶
¶
Вяжущие растворы и другие растворы.¶
¶
Сопротивление измеряется в омах.¶
Удельное сопротивление — способность вещества препятствовать прохождению электрического тока.¶

По умолчанию этот флажок в момент нажатия кнопки **Подготовить задание на работу с текстом** сбрасывается. Фактически он получает состояние, определённое в переменной **sortChBxValue**, размещённой в служебном файле **ForIndex.jsxinc**, стандартно это **false**.

Можно сделать стандартной установку этого флажка, достаточно изменить значение упомянутой переменной. Плюс такого решения — гарантированно не будет потерь терминов при частичном совпадении. Минус — в лог-файле данные будут не в той очерёдности, что в таблице.

Работа с отдельными терминами

В главе **«Как правильно подойти к решению»** (с. 2) упомянута ситуация, что верстальщик просматривает текст, выделяет нужные термины и отмечает их, чтобы они попали в предметный указатель.



Стандартно кнопка **Обработать выбранный текст в соответствии с заданием** доступна только после обработки загруженной таблицы, и доступ к ней снимается по завершении обработки текста. Предполагается, что следующая обработка файла будет после новой загрузки таблицы. Очевидно, что это расходится с подходом «идти по тексту и отмечать термины». Поэтому был добавлен флажок-кнопка (красная стрелка), делающий доступной кнопку обработки, без необходимости снова загружать таблицу.

Если сейчас такой случай, что нужно использовать текущие результаты работы кнопки **Подготовить задание на работу с текстом**, то щелчок на этом флажке-кнопке сделает активной кнопку **Обработать выбранный текст в соответствии с заданием**, и можно продолжить работу, используя прежнее задание. Этот флажок-кнока становится доступным после первого нажатия на кнопку **Подготовить задание на работу с текстом** — как только появится задание на обработку.

И есть ещё один вариант: сперва отметить все термины, а потом идти по тексту, выделять ненужные и удалять их. Это реализовано флажком-кнопкой, синяя стрелка на верхнем рисунке. Выбирайте вариант работы, который в данный момент лучше походит.

Сбой в размещении индексного маркера

Редко, но к сожалению, бывает иногда так, что скрипт не может разместить маркер. Вот как это выглядит в лог-файле

```
>> Рычков Н. М., нарком юстиции СССР = (?i)\bРычко[ваyеоым]+\b
        Рычковым [449]
        Рычковым [421]
Запись 'Рычков Н. М., нарком юстиции СССР' не удалось поместить в индексный указатель.  ←
        Рычков [421]
Запись 'Рычков Н. М., нарком юстиции СССР' не удалось поместить в индексный указатель.  ←
        Рычкова [420]
        Рычков [419]
        Рычкова [417]
```

Видите, дважды Запись 'Рычков Н. М., нарком юстиции СССР' не удалось поместить в индексный указатель. Эта проблема не новая, она зарегистрирована ещё в версии CS3, но похоже, эту часть кода так и не поправят.

Я обсуждал с Питером Карелом этот вопрос, он считает, что это имеет место только в файлах, где есть таблицы. Он предложил решение, я его включил в программу, но оно у меня не всегда срабатывает. Если интересно, то можете тут почитать об этой проблеме:

https://community.adobe.com/t5/indesign/index-marker-is-not-placed-in-the-right-location-when-added-via-a-script/m-p/10967488?page=1#M178124

С другой стороны, я об этой ситуации знаю, и фиксирую в файле результатов. И если она возникла, вы это узнаете из лог-файла. Я видел другие программы сбора индексов, там эта ситуация пролетает мимо.

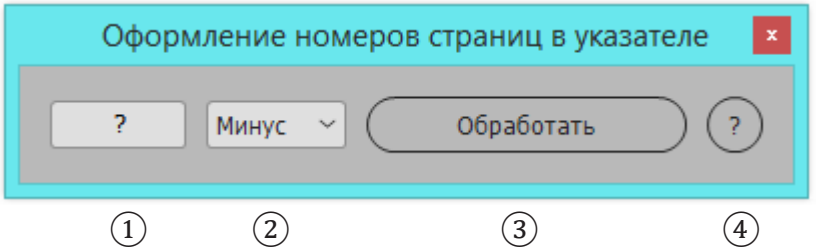
Что тут можно сделать? Пока только одно — добавить вручную потерянный термин, если это нужно. Когда на одной странице фамилия несколько раз, то добавлять её ещё смысла, конечно, нет. Но если термин вообще потерян, то это надо исправить.

Упорядочение номеров в указателе

Особенность подготовленного индизайном указателя — это повторяющиеся одинаковые номера, и последовательности номеров идущих одна за другой страниц.

17, 17, 17, 27, 28, 29, 35, 39, 40, 41 надо преобразовать в 17, 27–29, 35, 39–41.

Это выполняет скрипт **ProcNumberLine.jsx:**



① Во время приведения строк указателя в порядок можно определить, на какое число изменить все номера страниц. Для этого надо ввести тут положительное или отрицательное целое число. При введении отрицательного числа сперва вводится число, а потом знак. Эта опция полезна и для случая, когда указатель уже собран, оформлен, но что-то изменилось, и вся вёрстка сдвинулась на несколько стра-

ниц. Чем переоформлять указатель, проще ввести поправку на этот сдвиг.

② Вариант разделителя номеров. Дефис минус тире. По умолчанию используется минус.

③ Кнопка запуска. ④ Справка о программе.

Добавление прописных букв

Вот как можно добавить прописные буквы в упорядоченный список терминов, если не хотите использовать список абзацев с прописными буквами и запускать сортировку.

Есть такой перечень	*Его нужно оформить вот так:*
Абажур	А
Абазин	Абажур
Аббат	Абазин
Агава	Аббат
Бахрома	Агава
Бахча	
Бацилла	Б
Век	Бахрома
Вексель	Бахча
Вектор	Бацилла
	В
	Век
	Вексель
	Вектор

Искать: **^(\u).+\r(\1.+\r)+**
Заменить: **\r$1\r$0**

^(\u) поиск прописной буквы в начале абзаца.

.+\r поиск до конца абзаца с включением в результат перевода строки.

\1.+\r Метасимвол **\1** обозначает, что это адресация к шаблону поиска прописной буквы в начале строки, следовательно, это поиск такой же прописной буквы и затем всего абзаца, включая перевод строки.

(\1.+\r)+ знак плюс обозначает, что этот поиск выполнить по крайней мере один раз.

В замене **\r$1\r$0** **\r** — перевод строки, **$1** — первый заключенный в скобки шаблон поиска, т.е. прописная буква, **$0** — весь текст, который был найден в процессе поиска.

Если теперь эти прописные буквы надо оформить своим абзацным стилем, то для их поиска используйте grep-запрос *Искать*: **(\r)(^\u\r)** *Заменить*: **$2** с указанием абзацного стиля замены.

Дополнения

Инструкция, страница 6, правая картинка: чекбоксы для выбора стилей устанавливаются/сбрасываются двойным щелчком.

Остальные флажки всех окон скриптов управляются одиночным нажатием кнопки.

***

В инициалах должна быть шпация: А. С. Пушкин (Пушкин А. С.) — между 'А.' и 'С.' должен быть любой пробельный элемент, хоть волосяная шпация. Если это условие не выполнено, поиск будет безрезультатным.

***

Файл **Группировка слов по их окончаниям.pdf** , размещённый в папке **Info** — это один из промежуточных этапов группировки слов по вариантам окончаний.

Для использования программы не обязателен. Помещён тут для тех пользователей, кому интересен вопрос grep-обработки слов русского языка в разных падежах.

***

Для grep-запросов трёхбуквенных имён — Юра, Оля — уточнение, что это это поиск фамилии, не действует. Только варианты **Именительный падеж** и **Все падежные формы.**

В первом случае рузультат будет \bЮра\b

во втором \bЮр[аыеёуойюи]{1,2}\b

***

В словнике составное слово или двойная фамилия разделены обычным дефисом, а в строке grep-поиска на его месте будет группа из обычного дефиса (0x2D) и неразрывного (0x2011): [--]

Эта замена обычного дефиса на вариант из двух нужна для того, чтобы обойти случай, что в тексте в какой-то момент обычный дефис был заменён на неразрывный, а в тексте словника стоит обычный дефис.

В тексте может быть дискреционный перенос (0xAD) после обычного дефиса, но grep-поиск этот случай отслеживает, поэтому данный знак в строку поиска не включается.

***

О переменной simpleSearch в файле ForIndex.jsxinc.

Если simpleSearch равно true, то используется простой вариант создания grep-запроса, но он сильно избыточный из-за того что используются все буквы [а-я]. Избыточность проявляется, например, в том, что можно искать по тексту фамилии Петров, это делается так \bПетров[а-я]{0,2}\b , а будут найдены и отмечены Петрович, Петровна. Эта проблема и с Петров, Корнилов, Никитов, Титов, Александров, Андреев и пр.

Есть другой вариант получения набора букв, которые могут быть в окончаниях падежных форм слова. Он запускается, когда simpleSearch равна false. Буквенный разбор словника не требует заметно больше времени, всё обрабатывается быстро, и при этом неожиданных случаев — Семенович при поиске Семенов, Андреевна при поиске Андреев, Толпа при поиске Толь, и др. — уже не будет. По умолчанию simpleSearch = false.

***

Во всех вариантах окончания слов gss, sgs, ggs, tgs, sms, mgs и пр., если в них есть буква е, добавлена буква ё, чтобы не пропустить в тексте термины, где стоит е, тогда как в словнике ё.

***

В индексируемом тексте не должно быть фреймов и ячеек таблиц с переполнением. Не должно быть фреймов на рабочем столе, в тексте которых есть искомые термины.

***

О сбое в размещении индексного маркера, с. 18.

Есть тестовые файлы, в которых:

- этого сбоя нет, когда нет таблицы;
- устойчиво возникает сбой, когда таблица помещается в текстовый поток;
- не возникает сбоя, когда эта таблица в отдельном фрейме, не важно, стоит он на полосе сам по себе, или это прикреплённый фрейм.

Надо будет общими усилиями всех, кто будет пользоваться этой программой, обращать внимание на эти моменты: если появился этот сбой, то какие таблицы этому способствовали.

Пишите об этой проблеме, её проявлениях — при каких таблицах она возникла у вас, и о найденных решениях, если они есть, на dotextok@gmail.com

Тогда есть надежда, что будет найдено решение этой проблемы.

***

С Книгой (Book) эта версия программы не работает, пока есть только кнопка для переключения в этот режим. Очень много задач и проблем пришлось решить, пока фомировалось то решение, о котором повествует данный документ. Как только все, пока не решённые, но уже мелкие вопросы будут сняты, я займусь созданием предметного указателя в рамках Книги.

Заключение

На основе словника с терминами скрипты помогут собрать предметный указатель. Порукой тому, что в нем всё правильно, служит лог-файл, уделите ему внимание. Может, какие слова не были учтёны или встретится ошибка grep-запроса И конечно, важно смотреть цветные выделения терминов в вёрстке.

Имеющийся сейчас штатный инструмент позволяет собирать бесполезный для дела дежурный вариант, где имеются ссылки на некоторые термины, которые есть в тексте только в именительном падеже. А этим комплектом программ можно делать в научных изданиях и учебниках полноценные и удобные в работе предметные указатели, с иерархией логической соподчинённости терминов.

Михаил Иванюшин
dotextok@gmail.com dotextok.ru