Оформление сносок в индизайне (FootnotesControl@2022.jsx)

История вопроса

В 2018 году, в год десятилетия DoTextOK появился скрипт оформления сносок DoFootnotesOK.jsx Это было платное расширение скрипта DoTextOK, позволявшее быстро оформлять сноски. Работа с ними выпадала из задач DoTextOK, ну повторяющиеся пробелы после знака сноски скрипт мог обработать. но вариант оформления сноски (это не всегда верхний регистр) и какая шпация должна быть между знаком сноски и текстом — это уже были отдельные задачи. Да и сноски-то не у всех были, поэтому было принято сделать эту опцию платной — кому реально надо, тот купит.

Позднее стало ясно, что вариант встраивания этого дополнения в функционал DoTextOK не очень удобен, гораздо предпочтительнее иметь обработку сносок отдельным скриптом. Так появился вариант этого скрипта под названием FootnotesControl.jsx, и это уже отдельная программа, для запуска в любой момент, когда потребуется привести сноски в порядок, В обоих решениях, и в DoFootnotesOK.jsx, и в FootnotesControl.jsx использовалась идея предварительного сохранения в установках абзацного стиля, во вложенных стилях, настроек оформления сносок. После этих настроек всё работает просто замечательно, но как показала переписка, есть сложности в понимании идеи использования вложенных стилей в установках абзацного стиля.

Поэтому в версии **FootnotesControl@2022.jsx** работа со сносками переделана. Остался неизменным конечный результат — сноски в выделенном тексте будут оформлены безупречно, и при этом больше не надо морочиться, разбираясь с этими вложенными стилями.

Зачем вообще нужен отдельный скрипт

Любая программа решает конкретную задачу. Оформление сносок — это тоже одна из задач вёрстки, и несмотря на то, что разработчики индизайна много сделали для того, чтобы пользователю было удобно оформлять сноски, есть два окна параметров оформления сносок, проблемы всё же остались.

Две главные недоделки: 1) непосредственно после экспорта текста сноски внизу страницы всегда в обычном регистре, несмотря на то, что в символьном стиле оформления сносок указан верхний регистр; 2) после импорта текста в сносках после предустановленной шпации всегда есть пробел. Индизайн, видимо, просто после знака сноски помещает предустановленную шпацию, и всё. Но там всегда есть пробел, установленный в программе Word, и этот пробел остаётся в тексте.

Второстепенные, точнее, реже встречающиеся проблемы — часть текста сноски может быть оформлена каким-то другим шрифтом. Речь не о начертаниях курсив или полужирный, они сохраняются программой DoTextOK, а разные шрифты, разные размеры букв. Если это пропустить, то для таких недооформленных сносок в таблице абзацных стилей будет плюсик в строке с названием стиля. Но это не уберёт неверное оформление сноски.

И совсем редкая ситуация — иногда требуется такое оформление сносок, что для первого абзаца сноски и последующих надо иметь разные абзацные стили. Хорошо бы, чтобы программа принимала решение, что второй и последующие абзацы сноски надо оформлять другим стилем и создавала его.

Очевидно, что всё это по частям можно решить. Но помните присказку «Хорошо уметь самому всё делать, но не дай бог и делать всё это самому». Чем тратить своё ценное время и портить глаза на таких задачах, лучше поручить это хлопотное дело скрипту.

Подготовка к работе

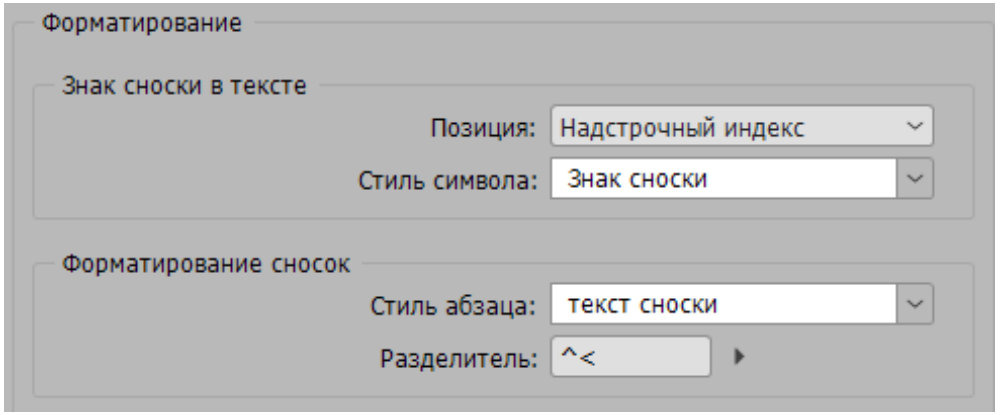
Итак, как это организовано.

Сначала пользователь делает три действия:

1) создает символьный стиль оформления сноски: будет знак просто в верхнем регистре, или пожелает сделать его поизящнее, использовав вариант Надстрочные знаки OpenType — решение за вами.

2) создает абзацный стиль оформления сносок. Привязка к базовой сетке в абзацном стиле должна быть снята, поскольку индизайн сам выравнивает нижнюю строку сноски по низу полосы. Настраивать вложенные стили — символьный стиль для знака сноски, определять шпацию — больше не нужно.

3) в окне **Текст > Параметры сносок документа...** указывает эти стили для знака и текста и определяет,



Часть окна **Параметры сносок**. Тут надо определить стиль сносок, стиль абзаца и разделитель. Информация поля **Позиция** скриптом не используется.

какой будет разделитель между знаком сноски и текстом. В списке указаны разные одиночные шпации, и можно эту строку дополнить своими знаками.

Справа показаны шпации, предлагаемые в выпадающем списке **Разделитель**, и те, что вы сами можете поставить: слева от названия приводится код знака.

В частности, заказчик иногда хочет, чтобы первые строки сносок были оформлены втяжкой. Это можно сделать установкой нужных параметров, но иногда были сетования, что абзацы со сноской в одну цифру смотрятся не так, как абзацы со сносками в две цифры. И чтобы таких нареканий не было, поступить

Шпации в окне Параметры сносок

^t	табуляция
^m	круглая шпация
^>	полукруглая шпация
^|	волосяная шпация
^<	тонкая шпация

Другие знаки, которые могут быть в строке Разделитель

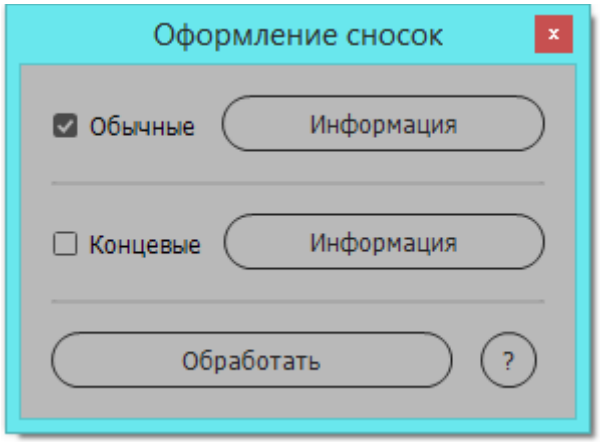
^S	неразрывный эластичный пробел
^s	неразрывный фиксированный пробел
^%	шпация на 1/6 круглой
^4	шпация на 1/4 круглой
^3	шпация на 1/3 круглой
^.	шпация на точку
^/	шпация на цифру
^i	произвольный отступ

надо иначе: параметры левого отступа и абзацного отступа оставить нулевыми, а в поле **Разделитель** после нужной шпации поставить ^i — это буквенный код знака «Произвольный отступ» (Ctrl+\).

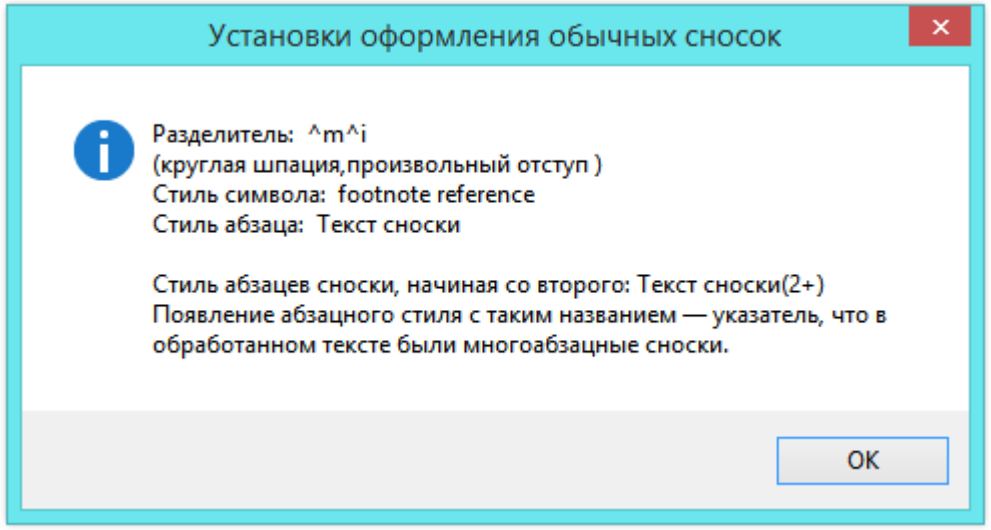
Как вариант, можно посмотреть, как смотрится сноска с этим знаком, добавив его на странице, а потом взять в буфер и вставить после шпации в поле «Разделитель». Сразу после вставки этот знак будет выглядеть чёрной точкой, а после повторного открытия окна он превратится в ^i.

Обработка сносок

Установка и запуск. Архив надо распаковать в папке Пользователь или User, выбрать текст для обработки, и после запуска появится окно



Кнопка **Информация** показывает, что установлено в окне **Параметры сносок**:



но в несколько ином виде: коды шпаций в строке **Разделитель** сопровождаются их текстовым объяснением.

В процессе обработки название кнопки **Обработать** сменится на **Оформление**, на экране будет прогрессбар, как он исчезнет и вернётся прежнее название кнопки, значит, обработка завершена.

Многоабзацные сноски. Если они встретятся, то скрипт дублирует стиль абзаца и добавит в имени суффикс (2+). При необходимости установки этого стиля можно изменить.

Концевые сноски. Обрабатываются все имеющиеся в материале концевые сноски, независимо от объёма выделенного текста. Отдельного абзацного стиля для многоабзацных сносок не предусмотрено.

Итог, в чём преимущества этого скрипта в сравнении с предыдущими версиями:

1) Оформление сносок не запустится, если в выделенной области есть потерянные сноски. Эта проблема импорта текстов, похоже, не будет решена разработчиками. Но скрипт скажет о том, что есть потерянные сноски и поможет их найти.

2) не нужно настраивать вложенные стили.

3) не нужно создавать отдельный абзацный стиль для оформления второго и последующих абзацев. Он будет создан на базе пользовательского абзацного стиля с суффиксом в названии (2+). В момент создания по установкам это тот же абзацный стиль, что и для оформления сноски. Но поскольку бывают хотелки заказчиков, что абзац знака сноски должен быть без абзацного отступа, а второй и следующие абзацы должны иметь ненулевой абзацный отступ, то наличие отдельных абзацных стилей этот вопрос снимает.

Да, возможно, такое оформление начала сноски противоречит классическим правилам вёрстки, но тут верстальщику не следует придерживаться максимы «не могу поступиться принципами». Тут конструктивнее подход «любой каприз за ваши деньги».

4) табличные сноски тоже приводятся в порядок.

5) все, кто купил предыдущую версию, могут зайти со своей учётной записью на shop.dotextok.ru и в архиве своего заказа скачать обновление.

И ещё такое отличие от предыдущей версии: убраны флажки выбора — удалять или нет повторяющиеся шпации и снимать стилевое форматирование сноски, эти действия выполняются безусловно. Работа стала проще: запустить скрипт, нажать на кнопку **Информация**, дабы убедиться, что активны именно те установки, что нужны (служебные коды шпаций дублируются их понятными названиями), и запустить обработку.

В результате:

- все знаки сносок будут оформлены нужным символьным стилем, и между знаком сноски и текстом не будет лишнего пробела. Для большинства верстающих — в только что импортированном тексте знак сноски в обычном регистре и лишний пробел после шпации — это как некая данность, которую надо пройти руками. Теперь это всё исправляет программа.
- случайное форматирование текста будет снято (чтобы не потерять курсив и полужирный, индексы и пр. важно предварительно обработать весь текст скриптом DoTextOK). Но если в сносках импортированного текста будет символьное оформление с цветом или изменением величины интерлиньяжа или кегля, то оно будет снято.
- если есть сноски в несколько абзацев, то второй и последующие абзацы будут оформлены абзацным

стилем, отличающимся от стиля первого абзаца. Скрипт сам создаст этот абзацный стиль на базе стиля, указанного в окне **Параметры сносок**.

▶ обновляется и стиль оформления знака сноски в тексте, уже не будет ситуации, что случайно изменился регистр и сноска стала цифрой после слова.

▶ программа приведёт в порядок и табличные сноски.

▶ если стало ясно, что не всё учтено, то клавиши (Ctrl+Z) откатят вёрстку к состоянию до запуска этого скрипта.

* * *

Я уже упоминал присказку, что хорошо уметь всё самому делать, но не дай бог всё и делать самому. Что касается вёрстки, то одно дело — уметь выбирать нужные шпации, обрабатывать текст grep-запросами, искать глазами висячие строки, и что-то с ними делать, и прочие мелкие задачи, которые должен уметь решать верстальщик. Но совсем иное — иметь в достаточном количестве разнообразные инструменты, которые могут быстро и качественно решать за вас большинство таких оформительских задач. И тут верстальщик — уже не простой исполнитель, а руководитель процесса.

Вот оформление сносок — это тоже одна из многих задач процесса доведения сырого материала до конечного результата, И этот скрипт — тот инструмент, который в тысячу раз быстрее, чем вы руками, подставит нужные шпации, приложит нужные стили, выполнит нужные grep-запросы. Речь — об экономии времени и сбережении остроты зрения в этом кропотливом процессе, а вёрстка состоит из множества мелких задач.

А если верстальщик устанет от книги, еще не сделав её, то результат будет далёк от совершенства.

Я за то, чтобы вёрстка была в удовольствие, чтобы получать радость от управления этим процессом, а не отвращение от однообразной работы. И в этом деле именно скрипты дарят такую радость.

Михаил Иванюшин
dotextok@gmail.com | https://dotextok.ru