

Assigning a language by character code

If the letter encodings of different languages do not match, then you can assign a language by letter. It will solve the problem of incorrectly specifying the language in the text received for layout, if, for example, it is known in advance that only Russian and English words are in the file, and the markup of the text is such that it is supposedly all Russian.

The Russian letters in the grep search are defined as follows: [а-яА-ЯёЁ], to search for English letters such set is used: [a-zA-Z].

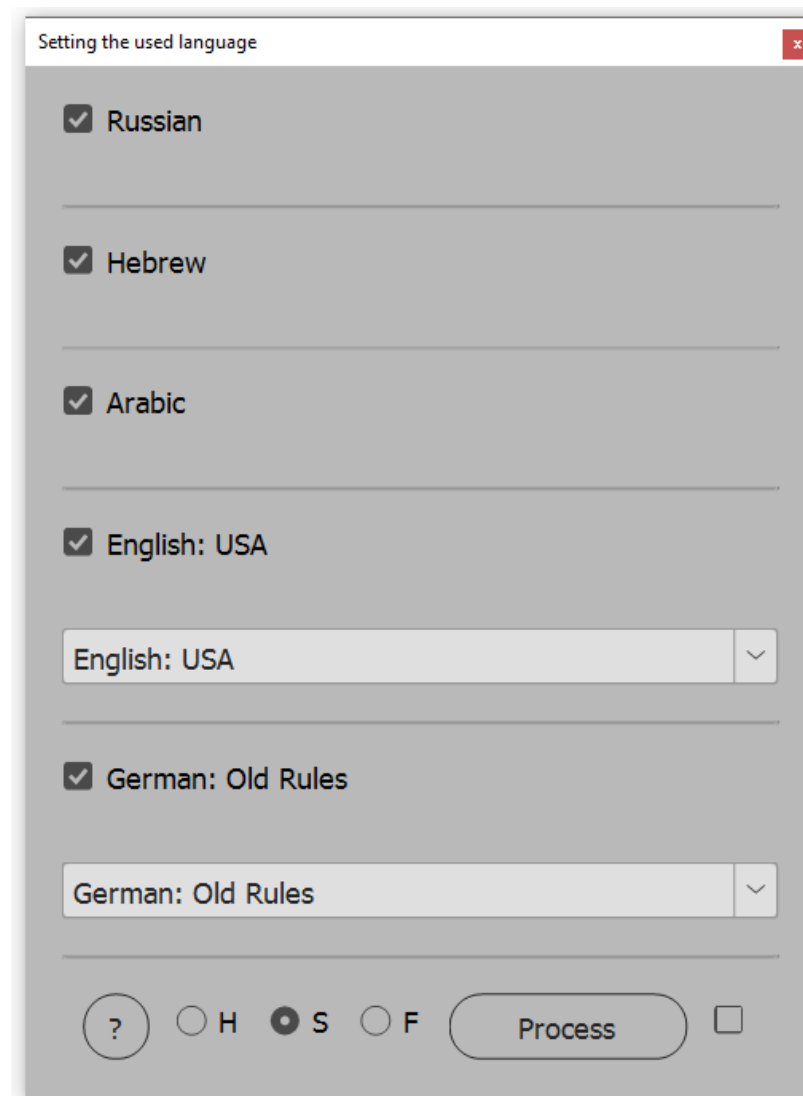
The Hebrew character space is defined by the following line: [\x{0590}-\x{05FF}]. Arabic script is assembled in such a space: [\x{0600}-\x{06FF}].

These spaces do not intersect, which means that you can search for letters from these spaces and assign your own language to them.

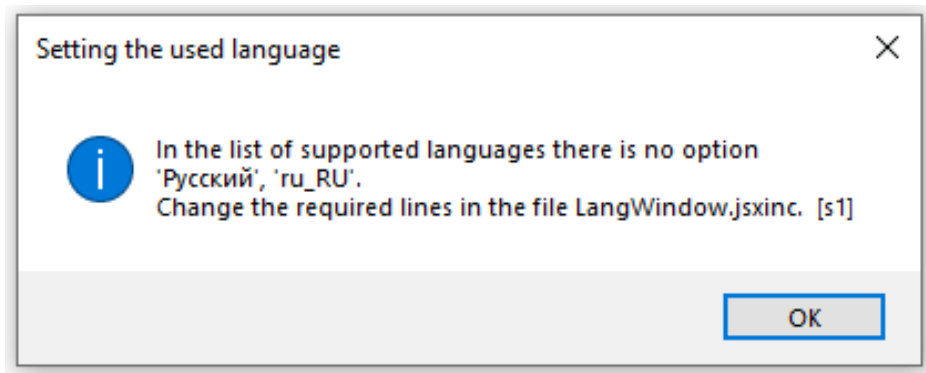
This idea is perfectly suitable for the letters of Russian, Hebrew, Arabic, and the Devanagari alphabet, and it is obvious that this does not automatically work for European languages. But if it is known in advance that the text typed in Latin is, for example, Italian, then it would be good to be able to assign an attribute of this language to Latin letters..

There is another problem with European languages: the set of letters mentioned above is not enough for everyone. So for the German language it is necessary to have such a space of signs: [a-zA-ZäöüÄÖÜß]. Norwegian, Czech and other languages have their own special sets of letters. But if you are able to determine the necessary space of signs, then the letters of these languages can be identified.

The task is not easy, but it has been solved. The working window of this program is on the right **SetLanguageByCharCode.jsx**.



At the first launch, instead of the expected window, the following message may appear on the screen:



This script with settings for working with the Russian language was released in the English version of InDesign.

It is necessary to change the settings in the **LangWindow.jsxinc** file. Here's what the parameter block for the Russian language looks like:

```
18 // русский язык
19 var lng1_name = "Русский";
20 //var lng1_name = "Russian";
21 var lng1_icuLN = "ru_RU";
22 var lng1_untranslatedName = "Russian";
23 var lng1_help = "[а-яА-ЯёЁ]";
```

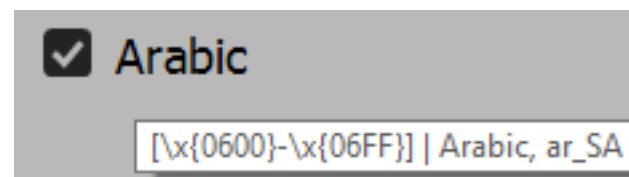
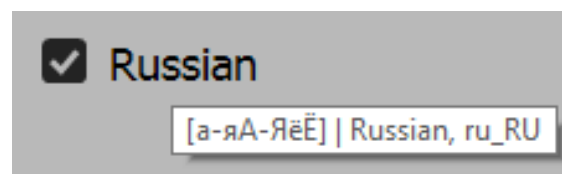
lng1_name — this is the name of the language in the InDesign localization used; lng1_icuLN — short name of the language; lng1_untranslatedName — a single version of the language name for all localizations; lng1_help — the search space for the letters of this language.

There is such a block for each language. It is necessary to specify the correct values in all blocks for _name and _icuLN.

There is a script to get these parameters for the used version of the InDesign **All_InDesign_Languages.jsx**.

After that, the program window will appear on the screen.

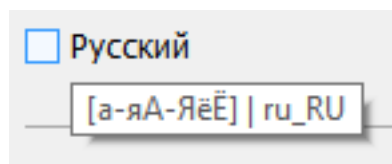
Information about the search space is displayed in the helptips for selecting the language to be processed, followed by a vertical line and a variant of the name of this language, which is the same in all InDesign localizations, and a short language designation.





All the examples above, which show tooltips, are made in one of the latest versions. Everywhere there is a single version of the language name for all localizations. And before the 2015 version, there was no such parameter in the language settings.

And below is an example of a pop-up window in CS6, there is only a letter designation of the language.



For the two lower options, the language is selected from the drop-down lists. InDesign has several variants of English, several variants of German; the same situation with French, with the languages used in India. It is reasonable to assume that you may need to assign a specific language attribute to the text, and the script provides this option.

Obviously, the chosen language must match the encoding used in this version. So in this example, for the penultimate option, you can choose either English or Italian. But in Spanish, Czech, there are letters that are missing from the encoding used. In the bottom line, you can select any of the German languages.

So, the key to proper language markup is the correct character encoding. And it can be changed in this program. For example, instead of searching for German letters, you can include the Devanagari alphabet, which is used in Hindi. It's not difficult if you understand how the script code for InDesign is prepared.

This is how the German language processing connection looks in the **LangWindow.jsxinc** file:

```
46 // немецкий
47 //var lng5_name = "Немецкий: Старые правила"; // rus
48 var lng5_name = "German: Old Rules"; // eng
49 var lng5_icuLN = "de_DE";
50 var lng5_untranslatedName = "German: Traditional";
51 var lng5_help = "[a-zA-ZäöüÄÖÜß]";
```

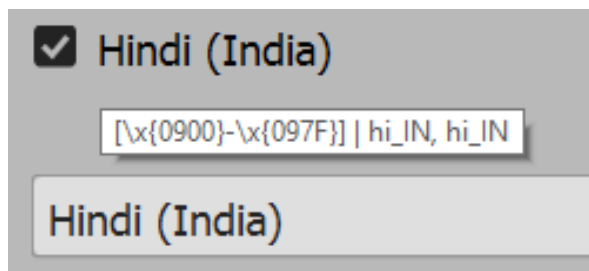
The commented block for devanagari is in the same file:

```
53 /*
54 // Подключение обработки деванагари вместо немецкого языка
55 var lng5_name = "Хинди (Индия)"; // rus
56 // var lng5_name = Hindi (India); // eng
57 var lng5_icuLN = "hi_IN";
58 var lng5_untranslatedName = "hi_IN";
59 var lng5_help = "[\x{0900}-\x{097F}]";
60 */
```

It is enough to comment out the block with the German language and make the devanagari processing active, and you can expect that the view of the window will change. But most likely, this will not happen. The old window with German language processing will be on the screen. This is not an error, but a feature of the program: when you start, the settings of these five checkboxes are taken from the save file of the previous launch settings.

The **#sets** folder is created in the folder of the processed file, and it

has a **#sets.ini** file, from which all the window design data is taken. To enable the default settings, i.e. those included in the program, click on the check box to the right of the **Process** button. And if there was a replacement of the space of German letters with the Devanagari alphabet, then the bottom button will be like this:



What will be processed is determined by one of three radio buttons: **H** — the highlighting text, **S** — the story, **F** — the file, or the entire document. In the **Highlighting text** option, the processing area expands to the borders of the paragraphs that fall into the selection.

When processing, the name of the trigger button changes, it will be **Processing**.

In addition, there will be a progress bar on the screen. It contains information about what is currently being processed. In one story, letter processing is performed as many times as there are checkboxes. The progress bar will contain information about which letters of the language are being processed, for example, Letters — Russian. And by the time all the letters are processed, we have the following sit-

uation: the letters in the article, footnotes, and tables have a language attribute corresponding to its encoding, but the language attribute has not changed for numbers, punctuation marks, and punctuation marks. Therefore, at the end of the article processing, another passage through the text is performed, searching for numbers, punctuation and punctuation marks, and assigning them the desired language attribute. This problem is solved as follows: in each paragraph, the first letter is searched for, and all these characters of this paragraph will have its language attribute. During this pass, there will be a message in the progress bar Punctuation, numbers, spaces — all languages. There is also a message for working with tables, but they are usually processed so quickly that you don't notice it.

When the processing process is completed, the previous button name will return: **Process**.

If you decide to supplement the program with your own options, it is better to have different versions of this program for different languages. It is advisable that they have their own setting files (see Fig. below).

In the CS6 version, it is not always immediately possible to match the name of the lower buttons being changed and the selected language. It may turn out that a new language has been selected in the list, but the name of the checkbox remains the same. In this case, you need to clear the checkbox and set it again, and then confirm the selection again. It usually helps.

```
129 var rezFileName = "#sets.ini"; // you can have variations of this script
130 // for different sets of processed languages.
131 // And these options should differ by the name of the installation file.
132 // #sets1.ini, #sets2. ini, etc.
```

Mikhail Ivanyushin
<https://shop.dotextok.ru/en/dotextok@gmail.com>