# DSL KeyPad — инструмент для многоязычного письма, требующий лишь QWERTY/ЙЦУКЕН.

«DSL KeyPad» — инструмент, дающий пользователю возможность вводить широкий спектр символов юникода посредством ряда функций, описанных ниже. Написан на AutoHotkey V2.

Основной фокус инструмента — расширение доступных ко вводу символов латиницы и кириллицы, при этом требуя от пользователя наличия только стандартных QWERTY/ЙЦУКЕН раскладок.

Идея наличия в системе ряда раскладок для сходных на письме языков кажется мне неудобством — зачем, если можно поверх «единого фундамента» добавить какие захочешь закорючки? И они всегда будут располагаться на одном месте вне зависимости языка. Кто-то может и такое посчитать неудобным, но выбранный мной способ решения проблемы многоязычного ввода для меня же кажется более привлекательным.

## Краткая «история»

Изначально инструмент родился из собственной нужды вводить ряд нестандартных символов. По началу я использовала «Типографскую раскладку» Ильи Бирмана, но в итоге и этого мне оказалось мало. Мне нужен был простой способ вводить шпации, диакритику, расширенную кириллицу и латиницу — для написания лора собственного вымышленного мира и использования сихъ знаковъ в писательстве.

Первая версия исполнения была простой как пробка: ~20 строчек простых биндов вида «<^<!e:: Send("{U+045E}") ; (Ў)». Однако со временем приходили новые идеи и желание сделать что-то более сложное, что могло бы покрыть больше моих же потенциальных потребностей. Как итог — процесс сильно затянулся и получился какой-то монструозный ООП’оид, но результатом я довольна.

Влетала я в это со слабыми навыки в написании кода и постепенно училась в процессе разработки этой штуки её, собственно, и разрабатывать. Конечно, с использованием нейронок, но скорее как указатель на ошибки, справочник и «я тут накакала, сделай по красивее» — «вайбкодить» у меня получалось крайне редко, не умею я с железякой контактировать… и хорошо.

## Краткая сводка фичей инструмента

1. Поддержка широкого спектра символов юникода. На данный момент — более 3300 символов, включая, помимо озвученного: пунктуация, математические символы, символы валют, различные специальные символы, алхимические/астрологические/астрономические символы.
2. GUI для отображения списков доступных символов с горячими клавишами или последовательностями ввода (см. далее); окно настроек; окно для просмотра, создания/изменения/удаления пользовательских рецептов.
3. «Многослоный» набор горячих клавиш, по большей части раздельных для английского и русского языков (например, [англ.] RAlt + A = Ă; RLAlt + A = Â; RAlt + RLShift + A = Ã; [рус.] RAlt + Ф = Ѳ; RAlt + Я = Ѧ).
4. Режим композиции, позволяющий из последовательности одних символов получить какой-то другой (например: AE → Æ, OE → Œ, TH → Þ).
5. «Альтернативные режимы» для ввода форм письменности, отличных от латиницы и кириллицы (Греческая письменность, Германские Руны, Глаголица, Финикийское письмо, Древнесеверноаравийское и т.д.), а также для ввода Международного фонетического алфавита.
6. Режимы для ввода альтернативных форм символов (например: 𝑎, 𝐚, 𝒂, 𝔞, 𝖆, 𝒶, 𝓪, 𝕒, 𝚊, ᴀ).
7. Возможность переключать тип ввода — HTML-энтити/мнемоника и LaTeX (если доступно) вместо символов юникода.
8. Возможность искать символы по их «тегам» в локальной библиотеке (например: «latin capital ligature AE with macron» → Ǣ; запрос может содержать неполный тег: «cap ae mac» → Ǣ; «одал» → ᛟ).
9. Возможность добавлять символы библиотеки в избранное — это позволит всегда отображать их рецепты в подсказке режима композиции, а также отметит избранные символы в GUI звёздочкой.
10. «Внутренние» раскладки клавиатуры (QWERTY, Dvorak, Colemak | ЙЦУКЕН, Диктор ЙІУКЕН 1907).
    1. Включает поддержку «пользовательских раскладок».
11. Поддержка пользовательских биндов, однако ограниченных символами локальной библиотеки.
12. Отдельные вспомогательные режимы для облегчения/ускорения письма на вьетнамском языке/пиньине.
13. Прочие различные возможности…

## Подробнее об основных фичах

### Горячие клавиши (они же — «Быстрые ключи»)

Их много, просто невероятно много, и опытный пианист, безусловно, справится с ними. А если серьёзно — на каждую клавишу может приходиться до 10–11 возможных комбинаций (но в среднем — около 8) (латинская I в этом рекордсмен: Ĭ Ï Ī Ĩ Î Ǐ Į İ Í Ì Ȉ).

Из-за своего же числа, по умолчанию — выключены, но переключаются на RAlt + F1. Состояние вкл/выкл сохраняется в конфигурационном файле. Привязки могут конфликтовать с хоткеями приложений (точнее — блокировать их).

Делятся комбинации на ряд условных подгрупп, по «иницирующим» модификаторам: LCtrl LAlt-группа; LAlt-группа; RAlt-…; LShift-…; RShift-….

Больше всего привязок включает в себя RAlt-группа, и она же является основной. На втором месте LCtrl LAlt-группа, чей фокус сосредоточен на комбинируемой диакритике, но ею не ограничивается. Остальные — «по остаточному принципу».

Для привязок латинских букв я попыталась сделать какую-то логику для большей интуитивности, например: все комбинации RAlt + LAlt + <БУКВА> = <БУКВА> с циркумфлексом (ŴÊŶÛÎÔÂŜĜĤĴẐĈ). Однако это не заимело полноценной реализации — части символов с соответствующими диакритическими знаками в юникоде по просту не существует, и в то же время есть буквы с более высоким приоритетом над другими. Так, например, буквы Å и Ů заменили буквы Ǎ и Ǔ на позиции RAlt + LAlt + LShift <БУКВА>, изначально задуманной для букв с гачеком (ĚŘŤǏǑŠĎǦȞǨĽŽČŇ), так как, по моей оценке, надобность в их вводе будет возникать значительно чаще, чем в случае A и U с гачеком. В случае же кириллицы — там тот ещё разброд и шатание, лишь в единичных случаях пытающийся соответствовать описанной выше логике: RAlt + RShift + <БУКВА> = ӢӮ ([лат.] ĀĒȲŪĪŌḠ).

Карту привязок можно представить следующим образом:

Пришлось несколько «нестандартно» отобразить альты и шифты, чтобы уместить на одном изображении сразу кучу комбинаций, а потому небольшое пояснение: если какой-то цвет встречается на позиции альтов и/или шифтов — значит их и нужно нажимать для получения соответствующего символа.

< ВСТАВИТЬ КАРТИНКИ С БИНДАМИ >

В дополнение есть некоторые обособленные сочетания клавиш, не использующие клавиши-модификаторы (Alt, Shift…):

* «Минус» на нампаде — сам по себе вместо дефиса теперь вводит настоящий минус (−), а в сочетании с «плюсом» на нампаде вводит плюс-минус (±), в обратной последовательности: знак минус-плюс (∓).
* «Умножение» на нампаде — вместо астериска теперь вводит символ умножения (×), а в сочетании со «слэшем» на нампаде — символ ⋇.
* «Слэш» (деление) на нампаде теперь вводит обелюс (÷).

Отдельно так же упомяну комбинации на стрелочки — они относятся к RAlt-группе и позволят вводить, собственно, стрелочки: ←↑↓→ (с LShift: ⮌⮍⮏⮎, RShift: ↺↻). При этом, как и клавиши на нампаде, их можно комбинировать: ↖↗↙↘↕↔ (RAlt + влево-вверх [или наоборот], вправо-вверх, влево-вниз, вправо-вниз, вверх-вниз и влево-вправо).

### «Плавильня»

Упомянутый выше «режим композиции» является основным способом использовать то, что я назвала «Плавильней» — фичу для «переплавки» последовательности одних символов в другие. Вдохновляясь, конечно, Compose key из Линукса и проектом WinCompose.

Композиция активируется двойным нажатием на RAlt — ввод пользователя будет отображаться во всплывающей подсказке, крепящейся к полю ввода (или к курсору, если по какой-то причине поле ввода не обнаружено). Там же, по мере ввода, будут отображаться предложения совпадающих рецептов (\*последовательностей), при этом рецепты избранных символов будут отображены всегда.

< ВСТАВИТЬ АНИМАЦИЮ КОМПОЗИЦИИ >

«Плавильня» позволила значительно расширить возможности ввода. Клавиатура не резиновая и бесконечное количество привязок, увы, сделать не получится. Композиция же принимает любой ввод (в том числе с «Быстрых ключей») и позволяет печатать прямо с клавиатуры ощутимо больше символов.

Режим композиции триггерится на первое точное совпадение рецепта, что означает невозможность ввести, например, Ǣ. Однако композицию можно перевести в состояние удержания нажатием клавиши паузы, что позволит вводить любые сходные рецепты, не опасаясь срыва ввода первым же совпадением.

Вместо паузы при вводе можно использовать «оператор разделения» — символ «`». Это не предотвратит триггер на первое точное совпадение, но позволит продолжить ввод последовательности, если существуют рецепты, начинающиеся с результата первого совпадения, например, AE → Æ, AE<МАКРОН> → Ǣ; тогда — RAlt×2: A`E → (Æ) → Æ<МАКРОН> → Ǣ.

Перечень доступных клавиш:

* Enter — подтвердить ввод
* Escape — отменить режим композиции
* Backspace — стереть последний символ
* Insert — вставка содержимого буфера обмена
* Pause — переключение удержания

< ВСТАВИТЬ АНИМАЦИЮ С ` >

С активной паузой можно использовать ещё несколько «операторов»: (ЧИСЛО), (~) и (ЧИСЛО~) — пишутся в начале строки и отбиваются пробелом от ввода; «ЧИСЛО» — количество повторений результата (например, «(4) alc sulfur» → «🜍🜍🜍🜍»); тильда же позволяет обрабатывать комбинации внутри какой-либо фразы, например, попытка ввести «apa<КРАТКА>» приведёт к ошибке «рецепт не найден», но «(~) apa<КРАТКА>» успешно вставит румынское «apă» (вода).

И ещё есть оператор «##», который нужен для предотвращения обработки нежелательных последовательностей: ввод «(~) tho<АКУТ>rr» приведёт к вставке «þórr», но если мы хотим оставить «th», тогда вводить следует «(~) t##ho<АКУТ>rr» — «thórr».

Дополнительно, режим композиции можно использовать для вставки символов через Альт-коды или кодовые позиции Юникода напрямую, начав ввод с A+/А+ или U+/Ю+ соответственно. В тултипе будет отображён символ, соответствующий введённому коду, а также его кодовая страница (для альт-кода) или блок (для юникода). Можно вводить множество кодов сразу, а для подтверждения их вставки необходимо нажать Enter.

Для альт-кодов играет роль язык раскладки: числа 128–255 и 0128–0255 используют страницы 850/Windows-1522, если активна английская, и 866/Windows-1251, если русская.

< ВСТАВИТЬ СКРИНЫ ТУЛТИПА >

### Пользовательские рецепты для «Плавильни»

Как вручную, так и через окно интерфейса («Мои рецепты» в меню трея или символ кубика в панели настроек (RCtrl F9 / «Настройки» в трее)) пользователь может создать собственные рецепты, чьи последовательности будут доступны к использованию сразу (если создавать его через окно интерфейса).

В качестве результата пользователь может вписывать сколь угодно объёмный многострочный текст, позволяя использовать такие рецепты и не совсем по прямому назначению инструмента — сниппеты кода, фрагменты статей и т.д. Но всё же не рекомендую вставлять туда через чур большой текст.

При первом запуске создаётся несколько демонстрационных пользовательских рецептов — шаблон HTML, иероглиф «義», тег <kbd> и эмодзи «🧊».

Примечание: в случае ручного добавления рецепта в .ini файл важно следить за тем, чтобы этот файл оставался в кодировке UTF-16 LE.

На примере .ini файла структура рецепта выглядит следующим образом:

[alchemical\_sulfuric\_acid]

name=ru:Алх. Серная кислота|en:Alc. Sulfuric Acid

recipe=alc sulf acid

result=🜊🜖

[chemical\_sulfuric\_acid]

name=ru:Серная кислота|en:Sulfuric Acid

recipe=chem sulf acid|H2SO4

result=H₂SO₄

[emoji\_castle]

name=ru:Сиро|en:Shiro

recipe=cls|змк

result=🏯

; [custom\_entry\_name] — Имя записи, может содержать только «a–Z» и «\_» символы

; name — Отображаемое имя в окне пользовательских рецептов. Может быть простой

; строкой, так и быть с указанием локализации для поддерживаемых языков

; recipe — Сам рецепт, может быть простой строкой или содежать несколько рецептов,

; разделяемых «|». Можно использовать ссылки на другие записи как ${имя\_записи}

; result — Результат рецепта. Также может содержать ${ссылки}

; то же самое с примером ссылок:

[alchemical\_sulfuric\_acid]

name=ru:Алх. Серная кислота|en:Alc. Sulfuric Acid

recipe=alc sulf acid

result=${alchemical\_acid\_vinegar}${alchemical\_vitriol\_1}

[chemical\_sulfuric\_acid]

name=ru:Серная кислота|en:Sulfuric Acid

recipe=chem sulf acid|H2SO4

result=H${digit\_2::subscript}SO${digit\_4::subscript}

${имя\_записи::альтерация×количество} — ссылка, которая позволяет обратиться к любому элементу библиотеки, если тот уже инициализирован. Через «::» можно задать альтернативную форму, если она существует (${digit\_1} → «1» | ${digit\_1::modifier → «¹»). В конце ссылки через символ «×» можно указать количество повторений вывода записи, на которую мы ссылаемся. Это можно использовать и между пользовательскими рецептами, чтобы создать зависящие друг от друга последовательности, например:

[white\_heart]

name=Белое сердце

recipe=бср|whr

result=🤍

[white\_heart\_triple]

name=Три белых сердца

recipe=3бср|3whr

result=${white\_heart×3} ; вставляет результат первого рецепта трижды

Помимо основного файла «DSLKeyPad\User\profile-<ПРОФИЛЬ>\CustomRecipes.ini» можно использовать и дополнительные .ini-файлы рецептов, прикрепляя их «вручную» соответствующей кнопкой в интерфейсе или помещая в директорию для автоматической подгрузки (при запуске программы): «DSLKeyPad\User\profile-<ПРОФИЛЬ>\Autoimport.ini\».

Однако взаимодействовать с такими рецептами через окно интерфейса не выйдет — редактирование, добавление и удаление доступны только для файла «CustomRecipes.ini».

Для дополнительных файлов рецептов есть возможность устанавливать несколько параметров для всех содержащихся в них рецептов (указываются вручную в начале файла):

[options]

prefix=<prefix>

; последовательность, устанавливаемая в начало

; всех рецептов в файле

no\_whitespace=1

; отключает добавление пробела после префикса

Если в файле есть двойные (и более) рецепты (например «лёд|ice»), для них можно задать и подобного вида префиксы (одинарные рецепты всегда берут первый из них):

[options]

prefix=эмодзи|emoji

[recipe\_name]

name=Лёд

recipe=лёд|ice

; потребует ввода «эмодзи лёд» или «emoji ice»

result=🧊

Присутствует и частичная поддержка последовательностей из файлов «\*.XCompose» линукса. Их необходимо поместить в директорию автоподгрузки «DSLKeyPad\User\profile-<ПРОФИЛЬ>\Autoimport.linux\» и обновить рецепты соответствующей кнопкой в «Моих рецептах».

Импортируются только простые последовательности XCompose, как, например, «<Multi\_key> <g> <r> <i> <n> : "😁"». При первом запуске в демонстрационных целях по указанному пути создаётся файл «demo.XCompose» с последовательностью «<Multi\_key> <0> <0> : "∞"».

< ВСТАВИТЬ КАРТИНКИ С ОКНОМ РЕЦЕПТОВ >

## Поиск символов по тегам

….

Ссылаться можно и напрямую на названия записей: «cyr\_c\_lig\_yus\_big\_iotified» → «Ѭ»; «reference\_mark» → «※».

Ограничено можно использовать регулярные выражения, например:

* «fut.\*?\s+\S+al» (ищем тег с любыми символами и пробелами между fut и al) → «ᚺ» (руна Хагалаз), полный тег: «elder **fut**hark hag**al**az»
* «az$» (ищем тег с az в конце) → «ᛞ» (руна Манназ), полный тег: «elder futhark mann**az**»
* «^фра» (ищем тег с фра в начале) → «₣» (Франк)

Для каждого из запросов можно указать и альтерацию символа, если она сщуествует, добавлением «::<АЛЬТЕРАЦИЯ>» в конец, например: «лигатура ae, лигатура ae::капитель» → «Æᴁ»; «!латин h::фрактур» → «𝔥».

В поиске не обязательно, как показано выше, указывать «внутренние названия» альтераций. Для каждой из них допустны следующие записи:

* Bold — полужирный, п, b
* Italic — курсив, ку, it
* ItalicBold — курсив полужирный, куп, itb
* Modifier — верхний индекс, ви, sup
* Subscript — нижний индекс, ни, sub
* Fraktur — фрактур, ф, f
* FrakturBold — полужирный фрактур, пф, fb
* Script — рукописный, р, s
* ScriptBold — полужирный рукописный, пр, sb
* DoubleStruck — ds
* SansSerif — без засечек, бз, ss
* SansSerifItalic — курсив без засечек, кубз, ssit
* SansSerifItalicBold — курсив полужирный без засечек, купбз, ssitb
* SansSerifBold — полужирный без засечек, пбз, ssb
* Monospace — моноширинный, м, m
* SmallCapital — капитель, к, sc
* Small — маленькая, мал, sm
* Combining — комбинируемый, ко, c
* Uncombined — некомбинируемый, неко, uc

Помимо непосредственно поиска символов, окно ввода запроса поддерживает и другое поведение:

* «\» перезапустит программу.
* /<ИМЯ ФУНКЦИИ / ПУТЬ ДО МЕТОДА> позволяет вызывать некоторые внутренние функции:
  + /Reload — тоже перезапустит программу…
  + /ChrLib.Print — создаёт по пути «DSLKeyPad\User\» файл «printed\_pairs.html» с перечислением записей локальной библиотеки формата «результат + доступные альтернативные формы — имя записи».
  + /MyRecipes.Update — ре-инициализирует пользовательские рецепты.
  + /Update.Get или Update.Repair — получить обновление или восстановить текущую версию.
  + /ChrCrafter.ComposeMode — аналогично RAlt×2 активирует режим композиции…
  + /KeyboardBinder.Init — ре-инициализирует раскладки и привязки.

## Иконка в трее

Просто небольшое перечисление взаимодействий:

* ЛКМ — открывает панель с перечислением доступных символов
* Ctrl+ЛКМ — открывает настройки
* Alt+ЛКМ — открывает список пользовательских рецептов
* Shift+ЛКМ — открывает окно выбора режима «Альтернативного ввода»
* Ctrl+Shift+ЛКМ — открывает окно выбора режима «Вариаций глифов»
* ПКМ — открывает контекстное меню

## Нюансы и потанцевальные проблемы

Самая, пожалуй, большая проблема — блокировка клавиш. AutoHotkey перехватывает комбинации, и особенно это заметно при использовании «внутренних» клавиатурных раскладок и «альтернативных режимов». Напимер, в играх с активной «внутренней» раскладкой, отличной от пары QWERTY-ЙЦУКЕН, попросту нельзя будет перемещаться на WASD.

Усесть на все стулья разом у меня не получилось, а потому для компромисса в программе есть как и RAlt + F1 для переключения «Быстрых ключей», так и RCtrl + F10 для отключения/включения вообще всех привязок (за исключением самой этой комбинации).

Может встретиться проблема со стороны определения софтом вводимого языка. Так, например, MS Word может повести себя неожиданным образом, видя, что у вас активна раскладка English (USA), но в окно попадают какие-то «странные закорючки». Когда я тестировала ввод на вьетнамском — ворд так и норовил заменить все введённые символы на… греческие. Это можно подправить отключением одной настройки в ворде, но я, к сожалению, забыла какой.