Projekt UnicodeBuild

Cel i główne założenia projektu

Jarosław Kuchta

24.05.2025

# Wprowadzenie

Współcześnie standardem kodowania znaków jest Unicode. Standard ten, prowadzony przez konsorcjum Unicode od 1991 roku, aktualnie definiuje ponad 150 tys. znaków z ponad 160 *systemów pisma* (ang. writing systems), a ma teoretyczną możliwość kodowania ponad miliona znaków w 24-bitowych *punktach kodowych* (ang. code points) w zakresie od 0 do 10FFFF (szesnastkowo). Pod pojęciem systemu pisma rozumie się sposób zapisu określonego języka, przy czym jeden język może być zapisywany w kilku różnych systemach pisma, a jeden system pisma może być stosowany do kilku różnych języków. Z kolei pod pojęciem *języka* (ang. language) rozumie się ustrukturyzowany system komunikacji, który ma swój słownik (ang. vocabulary) oraz gramatykę (ang. grammar). Unicode zajmuje się zarówno systemami pisma używanymi dla języków naturalnych (ang. natural languages), tradycyjnie stosowanych do komunikacji między ludźmi, jak i tymi, które wykorzystuje się w językach sztucznych (ang. artificial languages) lub formalnych (ang. formal languages), jak np. notacje matematyczne lub muzyczne. Oprócz tego Unicode definiuje też zbiory symboli graficznych służących do komunikacji niezależnej od języka (np. symbole pociągu, samolotu), służących do wzbogacania zapisu (jak kółka, kwadraty, gwiazdki), czy też do wyrażania emocji (ang. emoji). Niektóre punkty kodowe reprezentują znaki sterujące (ang. control characters) lub formatujące (ang. format characters), które nie mają prezentacji wizualnej, ale służą do sterowania transmisją tekstu między urządzeniami komputerowymi lub do określania sposobu prezentacji innych znaków. Unicode definiuje też wiele znaków reprezentujących spacje, myślniki i znaki interpunkcji, które są niezależne od języka.

Przy przeglądaniu zapisu łańcuchów Unicode przez człowieka występuje kilka problemów z ich czytelnością i jednoznacznością. I tak:

1. Te znaki Unicode, które nie mają reprezentacji wizualnej, są niezauważalne.
2. Spacje różnej szerokości są trudne do rozróżnienia przez człowieka.
3. Wiele symboli ma podobny wygląd (np. myślniki), co utrudnia ich rozróżnienie.
4. Nie wszystkie znaki mają zdefiniowane glify (obrazy) we wszystkich czcionkach. Jeśli glif jest niezdefiniowany, to w miejscu znaku pojawia się najczęściej pusty prostokąt.

Dla rozwiązania tych problemów stosuje się specjalny sposób wizualizacji niektórych znaków Unicode w łańcuchach znakowych. Polega to zazwyczaj na wstawianiu *sekwencji zastępczych* (ang. escape sequence) w miejsce niektórych znaków. Sekwencja zastępcza rozpoczyna się od znaku odwrotnej kreski ukośnej (‘\’). Dalej stosuje się kilka alternatywnych rozwiązań:

1. pojedyncze litery dla oznaczenia niektórych znaków sterujących, np.; „\t” dla oznaczenia znaku tabulacji, „\r” dla znaku powrotu karetki i „\n” dla znaku nowej linii.
2. kod szesnastkowy (częściej) lub dziesiętny (rzadziej) znaku. Dla odróżnienia kodu szesnastkowego od dziesiętnego poprzedza się go małą literą ‘x’. Przykłady „\0” dla znaku NUL, „\x0D” lub „\13” dla znaku powrotu karetki i „\x0A” (lub „\10”) dla znaku nowej linii.
3. umowną nazwę własną znaku, przy czym nazwy znaków o kodach od 0 do \x1F są zdefiniowane w standardzie ASCII (który jest podzbiorem Unicode), np. „\NUL” dla znaku o kodzie 0, „\HT” dla znaku tabulacji poziomej, „\CR” dla znaku powrotu karetki i „\LF” dla znaku nowej linii. Nazwy własne innych znaków nie są ustandaryzowane. Dla wyróżnienia końca nazwy własnej często przyjmuje się, że spacja po takiej sekwencji nie stanowi dalszego ciągu łańcucha.
4. znak ASCII symbolizujący funkcję przekształcenia diakrytycznego, tzn. takiego, które podany dalej argument funkcji w postaci znaku (lub sekwencji znaków) ASCII przekształca w jego wersję diakrytyczną w pewnym alfabecie narodowym, np. „\’o” oznacza literę ‘ó’, „\"o” oznacza ‘ö’. Jeśli symbol funkcji diakrytycznej jest literą ASCII, to argument funkcji może być oddzielony spacją lub ujęty w nawiasy klamrowe, np. „\H o” oznacza ‘ő’, a „\k{a}” oznacza ‘ą’.

Dla odróżnienia „zwykłego” znaku odwrotnej kreski ukośnej stosuje się sekwencję dwóch znaków odwrotnej kreski ukośnej („\\”).

Ze względu na czytelność sekwencji zastępczych najlepsze jest trzecie rozwiązanie – z nazwami własnymi. Dla przykładu „a \rightarrow b” jest łatwiejsze do zrozumienia niż „a \x2192 b”. Przy tym rozwiązaniu pojawia się jednak problem ze standaryzacją nazw własnych znaków. W standardzie Unicode wprawdzie każdy znak ma nazwę, ale są to nazwy długie, opisowe. Dla przykładu strzałka w prawo ma nazwę „RIGHTWARDS ARROW”, a strzałka ‘↲’ nazwę „DOWNWARDS ARROW WITH TIP LEFTWARDS”. Takie długie nazwy nie są wygodne do praktycznego zastosowania. Zachodzi potrzeba zdefiniowania krótszych nazw własnych.

Ze względu na ogromną liczbę zdefiniowanych znaków Unicode wskazane jest stworzenie algorytmu, który będzie przekształcał długie, opisowe nazwy Unicode znaków w skrócone nazwy znaków, które będą jednoznaczne, zrozumiałe i łatwe do zapamiętania. Projekt UnicodeBuild ma na celu stworzenie aplikacji służącej do wizualizacji efektów działania takiego algorytmu i umożliwiającej człowiekowi korygowanie nazw znaków. Algorytm ma działać na bazie danych UCD (Unicode Character Database) opublikowanej przez konsorcjum Unicode.

# Baza danych UCD

Konsorcjum Unicode publikuje najświeższą wersję swojego standardu na stronie WWW <https://www.unicode.org/Public/UCD/latest/>. Strona ta zawiera trzy katalogi:

* ucd – zawierający bazę danych UCD w postaci tekstowej,
* ucdxml – zawierający tę samą bazę danych w postaci plików XML spakowanych do ZIP,
* charts – zawierający plik CodeCharts.pdf wizualizujący definicje znaków.

Tekstowa postać bazy danych zawiera kilkadziesiąt plików tekstowych (w kodowaniu ASCII), z których najważniejszy to plik UnicodeData.txt. Przedstawia on definicje punktów kodowych w postaci sekwencji pól oddzielonych średnikami. Każda definicja jest zapisana w osobnej linii tekstu. Przykład:

0020;SPACE;Zs;0;WS;;;;;N;;;;;

Struktura tego pliku i pozostałych plików jest opisana w dokumencie UAX #44, "Unicode Character Database". Znaczenie pól w pliku UnicodeData.txt jest następujące:

1. Code – kod szesnastkowy znaku,
2. Name – nazwa opisowa znaku,
3. General Category – kategoria ogólna znaku,
4. Canonical Combining Class – kanoniczna klasa łączenia,
5. Bidi Class – klasa dwukierunkowości,
6. Decomposition – sposób dekompozycji i odwzorowanie,
7. Decimal Digit Value – wartość znaku jako cyfry dziesiętnej,
8. Digit Value – wartość znaku jako cyfry,
9. Numeric Value – wartość znaku jako liczby,
10. Bidi Mirrored – czy odbicie lustrzane w tekście dwukierunkowym,
11. Unicode 1 Name – nazwa znaku w standardzie Unicode 1.0 (nieaktualne),
12. ISO Comment – komentarz w standardzie ISO 10646 (nieaktualne),
13. Simple CAPITAL Mapping – kod odpowiadającego znaku wielkiej litery,
14. Simple Lowercase Mapping – kod odpowiadającego znaku małej litery,
15. Simple Titlecase Mapping – kod odpowiadającego znaku wielkiej/małej litery.

Dla generowania skróconych nazw znaków istotne są tylko trzy pola: kod, nazwa opisowa i kategoria ogólna. Pole General Category zawiera dwuliterowe kody ogólnych kategorii znaków Unicode, które są przedstawione w tab. 1.

Tab. . Ogólne kategorie znaków Unicode

|  |  |  |
| --- | --- | --- |
| **Skrót** | **Nazwa** | **Opis** |
| Lu | Uppercase Letter | wielka litera |
| Ll | Lowercase Letter | mała litera |
| Lt | Titlecase Letter | dwuznak zakodowany jako pojedynczy znak, którego pierwsza część jest wielka |
| Lm | Modifier Letter | litera modyfikująca |
| Lo | Other Letter | inne litery, w tym sylaby i ideogramy |
| Mn | Nonspacing Mark | znak łączący bez odstępu (o zerowej szerokości) |
| Mc | Spacing Mark | znak łączący z odstępami (o niezerowej szerokości) |
| Me | Enclosing Mark | znak łączący obejmujący |
| Nd | Decimal Number | cyfra dziesiętna |
| Nl | Letter Number | znak numeryczny przypominający literę |
| No | Other Number | znak numeryczny innego typu |
| Pc | Connector Punctuation | łączący znak interpunkcyjny, np. krawat (ang. tie) |
| Pd | Dash Punctuation | znak interpunkcyjny będący kreską lub myślnikiem |
| Ps | Open Punctuation | znak interpunkcyjny otwierający (parę) |
| Pe | Close Punctuation | znak interpunkcyjny zamykający (parę) |
| Pi | Initial Punctuation | cudzysłów początkowy |
| Pf | Final Punctuation | cudzysłów końcowy |
| Po | Other Punctuation | znak interpunkcyjny innego typu |
| Sm | Math Symbol | symbol matematyczny |
| Sc | Currency Symbol | znak waluty |
| Sk | Modifier Symbol | symbol modyfikujący niebędący literą |
| So | Other Symbol | symbol innego typu |
| Zs | Space Separator | znak spacji (o różnych szerokościach niezerowych) |
| Zl | Line Separator | separator wiersza (tylko znak o kodzie U+2028) |
| Zp | Paragraph Separator | separator akapitów (tylko znak o kodzie U+2029) |
| Cc | Control | kod sterujący C0 lub C1 |
| Cf | Format | znak sterujący formatem |
| Cs | Surrogate | zastępczy punkt kodowy |
| Co | Private Use | znak do użytku prywatnego |
| Cn | Unassigned | zarezerwowany, nieprzypisany punkt kodowy |

Unicode wyróżnia też zbiorcze kategorie znaków, które są zbiorami łączącymi kategorie ogólne (Tab. 2).

Tab. . Zbiorcze kategorie znaków

|  |  |  |
| --- | --- | --- |
| **Skrót** | **Nazwa** | **Opis** |
| LC | Cased Letter | Lu | Ll | Lt |
| L | Letter | Lu | Ll | Lt | Lm | Lo |
| M | Mark | Mn | Mc | Me |
| N | Number | Nd | Nl | No |
| P | Punctuation | Pc | Pd | Ps | Pe | Pi | Pf | Po |
| S | Symbol | Sm | Sc | Sk | So |
| Z | Separator | Zs | Zl | Zp |
| C | Other | Cc | Cf | Cs | Co | Cn |

Poza danymi zapisanymi w pliku UnicodeData.txt do skracania nazw opisowych znaków potrzebna jest też znajomość systemów pisma, do których należą poszczególne znaki.

Pod pojęciem *systemu pisma* (ang. writing system) rozumie się zbiór symboli (zwanych *grafemami*) oraz reguł, według których zapisuje się konkretny język. Nie musi być to język mówiony przez ludzi. System pisma, który służy do zapisu języka mówionego jest nazywany po prostu pismem (ang. script). Część systemów pisma nie służy do zapisu języka naturalnego, ale są stosowane w wybranych dziedzinach wiedzy, jak matematyka, chemia czy muzyka. Takie systemy pisma nazywane są *notacją* (ang. notation).

Pisma nie należy mylić z alfabetem. Są różne rodzaje pisma: alfabetyczne, sylabowe (sylabariusze, ang. abugida), spółgłoskowe z domyślnymi samogłoskami (ang. abjad), ideograficzne i inne. Jeden system pisma może być stosowany w wielu językach. Najszerzej stosowanym jest pismo zwane Latin, bazujące na alfabecie łacińskim, które wraz z różnymi symbolami *diakrytycznymi* (typu kreski, daszki, ogonki, haczyki) tworzą alfabety większości języków europejskich, północno- i południowoamerykańskich, a także jest stosowane w wielu państwach afrykańskich, w Australii i Nowej Zelandii oraz w niektórych krajach azjatyckich.

Unicode nie określa, w jakich językach świata są używane poszczególne znaki, ale określa, do jakiego systemu pisma należą poszczególne znaki. Przypisanie systemów pisma do konkretnych języków może odczytać z innych źródeł. Jeden system pisma może być stosowany do kilku języków, ale również kilka systemów pisma można stosować do tego samego języka.

Systemy pisma są widoczne w nazwach opisowych znaków, które często różnią się tylko nazwą tego systemu. Dla przykładu wielka litera ‘A’ występuje w wielu alfabetach, a odpowiednie znaki mają nazwy:

* 0041 LATIN CAPITAL LETTER A
* 0410 CYRILLIC CAPITAL LETTER A
* 104B0 OSAGE CAPITAL LETTER A
* 10570 VITHKUQI CAPITAL LETTER A
* 10C80 OLD HUNGARIAN CAPITAL LETTER A
* 10D50 GARAY CAPITAL LETTER A
* 118A1 WARANG CITI CAPITAL LETTER A
* 16E4D MEDEFAIDRIN CAPITAL LETTER A

Przy zapisywaniu ciągów znakowych bardzo często kolejne znaki należą do tego samego systemu pisma. Dla zwięzłości zapisu można zastosować grupowanie znaków należących do tego samego systemu. Tak więc zamiast pisać:

* \latinA\latinB\latinC
* \cyrillicA\cyrillicB\cyrillicC

można zapisać w sposób bardziej zwięzły:

* \latin{ABC}
* \cyrillic{ABC}

Przypisanie znaków do poszczególnych systemów pisma można rozpoznać nie tylko po nazwach opisowych, ale też można odczytać z pliku Scripts.txt. W pliku tym pierwsza kolumna (przed średnikiem) zawiera kod znaku lub zakres znaków, a druga – nazwę pisma. Po znaku ‘#’ w linii znajduje się komentarz, który podaje kategorię ogólną, liczbę znaków (dla zakresu) i nazwę opisową znaku (lub zakresu znaków). Przykład:

0041..005A ; Latin # L& [26] LATIN CAPITAL LETTER A..LATIN CAPITAL LETTER Z

0061..007A ; Latin # L& [26] LATIN SMALL LETTER A..LATIN SMALL LETTER Z

00AA ; Latin # Lo FEMININE ORDINAL INDICATOR

00BA ; Latin # Lo MASCULINE ORDINAL INDICATOR

00C0..00D6 ; Latin # L& [23] LATIN CAPITAL LETTER A WITH GRAVE..LATIN CAPITAL LETTER O WITH DIAERESIS

Z tego przykładu widać, że znaki o kodach 00AA (‘ª’) i 00BA (‘º’), chociaż przypominają odpowiednio małe litery łacińskie ‘a’ i ‘o’ w indeksie górnym, nie mają w nazwie „LATIN SMALL LETTER”, ale są przypisane do pisma łacińskiego.

Plik ScriptExtensions.txt zawiera dodatkowe informacje o systemach pisma dla tych znaków, które są wykorzystywane w wielu systemach. Pierwsza kolumna zawiera kod znaku lub zakres znaków, druga – skrócone, 4-literowe nazwy systemów pisma (lista nazw oddzielonych spacjami). Komentarz zawiera kategorię ogólną, liczbę znaków (dla zakresu) i nazwę opisową znaku (lub zakresu znaków). Przykład:

00B7 ; Avst Cari Copt Dupl Elba Geor Glag Gong Goth Grek Hani Latn Lydi Mahj Perm Shaw #Po MIDDLE DOT

02BC ; Beng Cyrl Deva Latn Lisu Thai Toto #Lm MODIFIER LETTER APOSTROPHE

02C7 ; Bopo Latn # Lm CARON

02C9..02CB ; Bopo Latn # Lm [3] MODIFIER LETTER MACRON..MODIFIER LETTER GRAVE ACCENT

02CD ; Latn Lisu # Lm MODIFIER LETTER LOW MACRON

02D7 ; Latn Thai # Sk MODIFIER LETTER MINUS SIGN

02D9 ; Bopo Latn # Sk DOT ABOVE

Skrócone nazwy systemów pisma są zgodne z normą ISO 15924.

Inny sposób rozpoznania przypisania znaku do systemu pisma stanowią *bloki Unicode* (ang. Unicode blocks). Bloki są zdefiniowane w pliku Block.txt w katalogu ucd. Unicode wyróżnia 338 bloków. Każdy blok ma określony zakres (początek i koniec) oraz nazwę. Początek listy bloków jest pokazany poniżej:

0000..007F; Basic Latin

0080..00FF; Latin-1 Supplement

0100..017F; Latin Extended-A

0180..024F; Latin Extended-B

0250..02AF; IPA Extensions

02B0..02FF; Spacing Modifier Letters

0300..036F; Combining Diacritical Marks

0370..03FF; Greek and Coptic

0400..04FF; Cyrillic

0500..052F; Cyrillic Supplement

0530..058F; Armenian

0590..05FF; Hebrew

0600..06FF; Arabic

0700..074F; Syriac

0750..077F; Arabic Supplement

Użycie bloków do rozpoznania przypisania znaków Unicode do systemów pismamoże być niejednoznaczne. Wiele bloków jest przypisanych do pojedynczego systemu pisma, ale kilkadziesiąt bloków zawiera znaki przypisane do dwóch, trzech, a nawet czterech systemów pisma. Są też takie bloki, które nie są przypisane do żadnego konkretnego systemu pisma.

# Oczekiwane efekty

Efektem zastosowania projektu UnicodeBuild ma być opracowanie takiego algorytmu, który będzie odwzorowywał kody znaków Unicode na ciągi znaków ASCII. Algorytm ma być zastosowany do serializacji i deserializacji łańcuchów znakowych. Algorytm musi:

* być wzajemnie jednoznaczny,
* zapewniać czytelność wyników dla człowieka,
* dawać zwięzłe wyniki,
* umożliwiać łatwą modyfikację treści,
* zapewniać odpowiednią szybkość konwersji,
* podążać za zmianami w standardzie Unicode.

Zwięzłość i wzajemną jednoznaczność zapewnia najprostsze odwzorowanie znaków Unicode w kod szesnastkowy, ale nie zapewnia on czytelności wyników. Czytelność oznacza, że człowiek przeglądający skonwertowany na ASCII tekst Unicode może zrozumieć znaczenie tekstu bez odwoływania się do jakichkolwiek materiałów pomocniczych.

## Zwięzłość

Używanie pełnych, opisowych nazw znaków nie daje zwięzłości. Można to uzyskać przechodząc szereg kroków konwersji:

1. Pierwszym krokiem w uzyskaniu zwięzłości może być łączenie znaków napisanych tym samym pismem. Dla przykładu zamiast pisać

\CYRILLIC\_CAPITAL\_LETTER\_A\CYRILLIC\_SMALL\_LETTER\_BE

można by napisać:

\CYRILLIC{\CAPITAL\_LETTER\_A\ SMALL \_LETTER\_BE}

Nawiasy klamrowe obejmują tutaj znaki pisane tym samym pismem.

1. Drugim krokiem może być zastąpienie długich nazw pisma ich krótkimi (4-literowymi) odpowiednikami ze standardu ISO 15924. Dla przykładu cyrylica ma skróconą nazwę „Cyrl”, przez co można uzyskać:

\Cyrl{\CAPITAL\_LETTER\_A\ SMALL\_LETTER\_BE}

W niektórych przypadkach można uzyskać dość duże skrócenie wyniku, np. pismo „ANATOLIAN HIEROGLYPHS” ma skróconą nazwę „Hluw”, ale jak widać po tym przykładzie takie skrócenie może się odbywać kosztem czytelności.

Innym problemem może być brak obecności danego systemu pisma w standardzie ISO 15924, co ma miejsce wówczas, gdy trzeba skracać nazwy notacji lub innych zbiorów symboli, np. „GREEK INSTRUMENTAL NOTATION”

1. Trzecim krokiem może być wyeliminowanie takich rzeczowników ogólnych z nazw opisowych, jak „LETTER”, „DIGIT”, „NUMBER”, „VOWEL”, „CONSONTANT”, „SIGN”, „MARK”. W powyższym przykładzie wynikiem byłoby:

\Cyrl{\CAPITAL\_A\ SMALL\_BE}

Trzeba to jednak robić ostrożnie, gdyż może to w niektórych przypadkach naruszyć wymaganie wzajemnej jednoznaczności. Dla przykładu w piśmie BHAIKSUKI występują zarówno cyfry „1”, „2” itd. o nazwach odpowiednio „BHAIKSUKI DIGIT ONE”, „BHAIKSUKI DIGIT TWO”, …, jak i liczby 1, 2 itd. o nazwach odpowiednio „BHAIKSUKI NUMBER ONE”, „BHAIKSUKI NUMBER TWO”. Znaki reprezentujące cyfry i liczby mają inny wygląd i (oczywiście) inne kody.

1. Kolejnym krokiem może być wyeliminowanie słów „CAPITAL” i „SMALL” i odróżnianie wielkości liter w sekwencjach zastępczych. W naszym przykładzie można napisać po prostu:

\Cyrl{\A\be}

Tu może pojawić się problem z różnymi znaczeniami słowa „SMALL”, które niekoniecznie oznacza „małą” literę, ale „pomniejszoną” literę lub inny znak.

1. Można wyeliminować znaki kreski ukośnej przed jednoliterowymi nazwami liter i napisać:

\Cyrl{Abe}

Nazwy wieloliterowe (dwu-, trzyliterowe) muszą być poprzedzone znakiem kreski ukośnej. Po takiej nazwie musi wystąpić kolejna kreska ukośna oznaczająca początek kolejnej sekwencji zastępczej, znak nawiasu klamrowego zamykającego (oznaczający koniec łańcucha) lub spacja. Ta spacja jest traktowana jako separator, a nie jako część konwertowanego łańcucha. Dla przykładu napis „Babcia i dziadek” po rosyjsku w cyrylicy:

Бабушка и дедушка

byłby zapisany jako

\Cyrl{\BE a\be\sha\ka a i \de\ie\de u \sha\ka a}

Pierwsza sekwencja „\BE” oznacza wielką literę ‘B’ w cyrylicy, po niej następuje spacja separująca, potem samogłoska ‘a’ bez spacji, gdyż po niej od razu występuje sekwencja „\be”.

## Czytelność i łatwość modyfikacji

Zwięzłość zapisu może wchodzić w konflikt z wymaganiem czytelności. Jak widać z ostatniego przykładu w poprzednim podrozdziale, napis:

\Cyrl{\BE a\be\sha\ka a i \de\ie\de u \sha\ka a}

Nie jest łatwy do odczytania przez człowieka. Problem ten można rozwiązać np. stosując transliterację romańską, czyli zamiast nazw (już teraz skróconych) spółgłosek pisać po prostu zamienniki z alfabetu łacińskiego (tam, gdzie to możliwe). Wówczas przykładowy napis

Бабушка и дедушка

byłby zapisany jako:

\Cyrl{Babushka i dedushka}

Oczywiście takie rozwiązanie wymaga znajomości zasad transliteracji, które nie są zapisane w standardzie Unicode, lecz w innych dokumentach.

Trzeba też zauważyć, że pewne zasady romanizacji wychodzą poza dozwolony zakres znaków wynikowych (ASCII). Dla przykładu tabela transliteracji zapisana w Bibliotece Kongresu USA dla języka rosyjskiego jest przedstawiona poniżej:

Tab. . Tablica romanizacji ALA-LC dla języka rosyjskiego

|  |  |  |  |
| --- | --- | --- | --- |
| Znak rosyjski | Romanizacja | Znak rosyjski | Romanizacja |
| А | A | а | a |
| Б | B | б | b |
| В | V | в | v |
| Г | G | г | g |
| Д | D | д | d |
| Е | E | е | e |
| Ё | Ë | ё | ë |
| Ж | Zh | ж | zh |
| З | Z | з | z |
| И | I | и | i |
| І | Ī | і | ī |
| Й | Ĭ | й | ĭ |
| К | K | к | k |
| Л | L | л | l |
| М | M | м | m |
| Н | N | н | n |
| О | O | о | o |
| П | P | п | p |
| Р | R | р | r |
| С | S | с | s |
| Т | T | т | t |
| У | U | у | u |
| Ф | F | ф | f |
| Х | Kh | х | kh |
| Ц |  | ц |  |
| Ч | Ch | ч | ch |
| Ш | Sh | ш | sh |
| Щ | Shch | щ | shch |
| Ъ | ʺ (hard sign) | ъ | ʺ (hard sign) |
| Ы | Y | ы | y |
| Ь | ʹ (soft sign) | ь | ʹ (soft sign) |
| Ѣ |  | ѣ |  |
| Э | Ė | э | ė |
| Ю |  | ю |  |
| Я |  | я |  |
| Ѳ | Ḟ | ѳ | ḟ |
| Ѵ | Ẏ | ѵ | ẏ |

Jak widać, w wielu przypadkach wymagane jest użycie liter łacińskich ze znakami diakrytycznymi (kropką, dwiema kropkami, kreską itp.), a w pewnych szczególnych – użycie łuków łączących po dwie litery. Zwłaszcza to ostatnie rozwiązanie wydaje się być odbiegające od wymagania łatwości modyfikacji tekstu.

## Znaki diakrytyczne

Popularnie znakami diakrytycznymi nazywa się litery alfabetu łacińskiego z dodatkowymi ozdobnikami ponad lub pod bazowymi znakami łacińskimi, np. ‘ó’, ‘ś’ ‘ą’, ‘ę’. To nie jest prawidłowe rozumienie, gdyż znakami diakrytycznymi są właśnie te ozdobniki, a zmodyfikowane w ten sposób litery nazywa się literami diakrytyzowanymi lub literami diakrytycznymi. Znaki diakrytyczne mogą być umieszczane nie tylko ponad i pod literami, ale też obok litery, czasem w sposób obejmujący literę. Dotyczy to nie tylko liter, ale również symboli oznaczających sylaby. Są również znaki diakrytyczne łączące dwie litery, a nawet grupy liter. Niektóre znaki diakrytyczne mogą przedłużać pewne segmenty liter (np. nogę litery).

W Unicode litery diakrytyczne mają z reguły nazwy opisowe zawierające frazę „WITH”, np. mała litera łacińska ‘ą’ ma nazwę „LATIN SMALL LETTER A WITH OGONEK”.

Wiele liter diakrytycznych ma zdefiniowane glify w różnych czcionkach, ale z niektórymi może być problem. Wówczas warto zastosować funkcję diakrytyczną, która będzie dodawać ozdobniki do podstawowych wersji litery. Dla przykładu napis:

Zażółć gęślą jaźń

można wyrazić jako

Za\overdot{z}\acute{o}\stroke{l}\acute{c} g\ogonek{e}\acute{s}l\ogonek{a} ja\acute{z}\acute{n}

W pewnych systemach składu pisma funkcje diakrytyczne wyraża się prostymi nazwami lub symbolami. Np. w systemie LaTeX poprzedni przykład można zapisać jako:

Za\.z\’o\l\’c g\k{e}\’sl\k{a} ja\’z\’n