Named Character Escape Sequences

P1097 – TARGETING C++20

R. Martinho Fernandes | cpp@rmf.io

Presented By: JeanHeyd Meneide & Tom Honermann

phdofthehouse@gmail.com & tom@honermann.net

https://wg21.link/p1097

WHAT… ARE THESE?

- u8"´"
- u8"‟"
- u8"´"
- u8"A"
- u8"𝐀"
- u8"A"
- u8"□"

WHAT ARE THESE?

- `u8"\u00B4"`
- `u8"\u0301"`
- `u8"\u1FFD"`
- `u8"\uD090"`
- `u8"\u13AA"`
- `u8"\uA4EE"`
- `u8"\U0016F40"`

WHAT ARE THESE?

- u8"\N{ACUTE ACCENT}"
- u8"\N{COMBINING ACUTE ACCENT}"
- u8"\N{GREEK OXIA}"
- u8"\N{CRYLLIC CAPITAL LETTER A}"
- u8"\N{CHEROKEE LETTER GO}"
- u8"\N{LISU LETTER A}"
- u8"\N{MIAO LETTER ZZYA}"

Raw	\x	\u	\U	\N
u8""	u8"\xE2\x80\x8B"	u8"\u200B"	u8"\U0000200B"	u8"\N{ZERO WIDTH SPACE}"
u8""	u8"\xE2\x80\x8C"	u8"\u200C"	u8"\U0000200C"	u8"\N{ZERO WIDTH NON-JOINER}"
u8""	u8"\xE2\x80\x8D"	u8"\u200D"	u8"\U0000200D"	u8"\N{ZERO WIDTH JOINER}"
u8""	u8"\xE2\x81\xA2"	u8"\u2062"	u8"\U00002062"	u8"\N{INVISIBLE TIMES}"
u8"´"	u8"\xC2\xB4"	u8"\u00B4"	u8"\U000000B4"	u8"\N{ACUTE ACCENT}"
u8"́"	u8"\xCC\x81"	u8"\u0301"	u8"\U00000301"	u8"\N{COMBINING ACUTE ACCENT}"
u8"᾽"	u8"\xE1\xBF\xBD"	u8"\u1FFD"	u8"\U00001FFD"	u8"\N{GREEK OXIA}"
u8";"	u8"\x3B"	u8"\u003B"	u8"\U0000003B"	u8"\N{SEMICOLON}"
u8";"	u8"\xCD\xBE"	u8"\u037E"	u8"\U0000037E"	u8"\N{GREEK QUESTION MARK}"
u8"Ω"	u8"\xCE\xA9"	u8"\u03A9"	u8"\U000003A9"	u8"\N{GREEK CAPITAL LETTER OMEGA}"
u8"Ω"	u8"\xE2\x84\xA6"	u8"\u2126"	u8"\U00002126"	u8"\N{OHM SIGN}"
u8"A"	u8"\x41"	u8"\u0391"	u8"\U00000391"	u8"\N{LATIN CAPITAL LETTER A}"
u8"A"	u8"\xCE\x91"	u8"\u0391"	u8"\U00000391"	u8"\N{GREEK CAPITAL LETTER ALPHA}"
u8"A"	u8"\xD0\x90"	u8"\uD090"	u8"\U0000D090"	u8"\N{CYRILLIC CAPITAL LETTER A}"
u8"A"	u8"\xE1\x8E\xAA"	u8"\u13AA"	u8"\U000013AA"	u8"\N{CHEROKEE LETTER GO}"
u8"A"	u8"\xEA\x93\xAE"	u8"\uA4EE"	u8"\U0000A4EE"	u8"\N{LISU LETTER A}"
u8"A"	u8"\xF0\x90\x8A\xA0"	n/a	u8"\U000102A0"	u8"\N{CARIAN LETTER A}"
u8"𖽀"	u8"\xF0\x96\xBD\x80"	n/a	u8"\U00016F40"	u8"\N{MIAO LETTER ZZYA}"

WHAT'S IN A NAME?

- Readability
  - Know what is actually being put in your string
- Consistency
  - Identifiers are created out of ASCII alpha-numeric characters with dashes and spaces
  - All name -> character mappings are ***absolutely stable*** (even the mistakes…)

NAME ALIASES

- Shorthand for longer names that sometimes make it even shorter than their U-codes
  - `"\N{NO-BREAK SPACE}"` `// matches Name for U+00A0`
  - `"\N{KANNADA LETTER LLLA}"` `// matches correction alias for U+0CDE`
  - `"\N{NBSP}"` `// matches abbreviation alias for U+00A0`
  - `"\N{LINE FEED}"` `// matches control character alias for U+000A`
  - `"\N{LF}"` `// matches abbreviation alias for U+000A`

MINUTAE

- In UAX #44 – Unicode Character Database (https://www.unicode.org/reports/tr44/#UAX44-LM2):
  - Extremely loose forward-compatible matching is demonstrated (ignore underscore, medial hyphen, spaces, etc..)
- Most languages do not do complete loose matching
  - Python: Case insensitivity
  - Perl: Exact match

MATCHING FOR C++

- Proposed: Case Insensitivity
  - Simplest, reasonably easy to implement
  - Some impedance mismatch with {ZERO-WIDTH-SPACE} versus {ZERO-WIDTH SPACE}
    - Compiler can error on inability to find a proper escape sequence to prevent silent spelling errors

- E.g.:
  - `"\N{nbsp}"` `// matches abbreviation alias for U+00A0`
  - `"\N{LiNe fEED}"` `// matches control character alias for U+000A`
  - `"\N{lF}"` `// matches abbreviation alias for U+00OA`

WORDING

- The paper has wording and is ready to send to Core