Q and A on Extended Identifiers and Extended Literals
Yuriko Sawatani, Norihiro Kumagai, Ichiro Koshida

Q: How does it work when a character, which is not defined in the execution character set is used as _universal-character-name_?
A: For identifiers, it is OK if they are distinguished each other even if there are no correspondence characters in the execution character set. For literals of strings or characters, a character is translated by an implementation-defined encoding. For comments, they are removed during translation phase and need not to think about this.

Q: Will size of an external identifier be too long if a non basic character in an identifier is converted to universal-character-name?
A: In C++, there is no limitation of the size for identifiers.

Q: In C, for example, uppercase and lowercase can be treated as the same in a linking time. In this situation, will this proposal have a problem?
A: In C++, uppercase and lowercase are treated as the different characters. So it is OK.

Q: I suppose that identifiers have no problem if they can be recognized each other, but how about in a debug environment?
A: It is OK to show a character using ??uNNNN, ??UNNNNNNNN if needed.

Q: Does it need to prepare all of the mapping of characters in ISO10646?
A: No. You only need to prepare tables from supported source file character to universal-character-name in translation phase 1, and tables from universal-character-name to the execution character set in translation phase 5.

Q: Is a run-time code conversion from universal-character-set to the execution character set required in this proposal? If it is true, then, will it has an impact for run-time environment, ex. code conversion tables and a performance impact for a run-time code conversion?
A: This proposal does not assume a run-time code conversion. In phase 5 defined in 2.1 Phases of translation, universal-character-name is converted to a member of the execution character set.

Q: If a compiler supports universal-character-name, i.e. ??uNNNN, ??UNNNNNNNN, can it not be used on an OS which does not support ISO/IEC 10646?
A: Yes, it can. universal-character-name is introduced to have a portability from one platform to another. When the OS does not support ISO/IEC 10646, then universal-character-name is replaced to a character defined in translation phase 5.

Q: Is it possible to map ??u2323 to a Japanese character x2323 in JIS code since I want to develop a compiler for JIS environment?
A: It will not ISO Strictly Conforming Implementation. The character designated by the universal-character-name ??uNNNN is that character whose encoding in ISO/IEC 10646.