PB138 — Characters in XML

Outline

- Character sets (charsets) and their encodings
- Unicode and its encodings
- Characters in XML
- Character entities

Characters in XML documents

The specification allows at some places in XML documents (eg. element name, attribute content...) not all characters.

It is good to know the meaning of:

- **character sets** (set of characters with respective codes/numbers), tj. attaching the ordinal value to character (eg. Unicode) a
- **character encoding** (from a given set), eg. UTF-8, eg. the ordinal value is encoded into a sequence of bytes
- (slightly) different in XML 1.0 and XML 1.1

Unicode and ISO 10646 Standards

Both standards try to resolve the problem: charsets with more than 256 chars were impossible to encode as *one byte = one character*.

Originally 16-bit Unicode

upto 64 K chars, enough for European languages/alphabets, not sufficient for world languages (eg. Chinese).

32-bit Unicode

covers "everything", every alphabet in the worlds.

Unicode and ISO 10646 Standards

- Nowadays, out of the 32-bit set just the *Basic Multilingual Plane (BMP)* is used, covering most of the typical languages.
- For names in XML (non-terminal Qualified Name - QName) only BMP chars may be used.
- Otherwise any Unicode char may be used.

Unicode encodings

All XML applications (particularly parsers) must be able to process some Unicode encodings. The most common in CZ/SK/EU are:

8-bit, traditional

US-ASCII, ISO-8859-2 (ISO Latin 2), Windows-1250 (= Cpl250) - just a subset of Unicode.

UTF-8

encoding of all chars in Unicode, each char to 1-6 bytes (different), US-ASCII to 1 byte, Czech/Slovak chars to 2 bytes.

UTF-16

same principle as UTF-8, but 16 bit (2 bytes) word is the basic unit

Unicode encodings

UCS-2

direct encoding of Unicode, chars from BMP are directly represented as their ordinal numbers

UCS-4

dtto, but for whole Unicode at 4 bytes - not efficient, 4 bytes even for US-ASCII, EU-langs…

UTF

encodings are the most important for XML, particularly UTF-8 (but parsers must know both).

Allowed chars

- Any chars from UNICODE upto `x10FFFF` (except of `xFFFE`, `xFFFF` and the range `xD800` — `xDFFF`).
- names must be composed of non-whitespace chars: numerals, letters, . (dot) - (comma, minus) _ (underscore) : etc., must start with a letter or
- *Encoding* of the UNICODE chars is not important.
- Now in more detail…

Allowed chars in XML 1.0

Unicode code points in the following ranges are valid in XML 1.0 documents (according to Wikipedia, https://en.wikipedia.org/wiki/Valid_characters_in_XML):

- `U+0009`, `U+000A`, `U+000D`: these are the only C0 controls accepted in XML 1.0;
- `U+0020` … `U+D7FF`, `U+E000` … `U+FFFD`: this excludes some (not all) non-characters in the BMP (all surrogates, `U+FFFE` and `U+FFFF` are forbidden);
- `U+10000` … `U+10FFFF`: this includes all code points in supplementary planes, including non-

characters.

- The preceding code points ranges contain the following controls which are only valid in certain contexts in XML 1.0 documents, and whose usage is restricted and highly discouraged:
- `U+007F` … `U+0084`, `U+0086` … `U+009F`: this includes a C0 control character and all but one C1 control.

Allowed chars in XML 1.1

Unicode code points in the following code point ranges are always valid in XML 1.1 documents (https://en.wikipedia.org/wiki/Valid_characters_in_XML)

- `U+0001` … `U+D7FF`, `U+E000` … `U+FFFD`: this includes most C0 and C1 control characters, but excludes some (not all) non-characters in the BMP (surrogates, `U+FFFE` and `U+FFFF` are forbidden);
- `U+10000` … `U+10FFFF`: this includes all code points in supplementary planes, including non-characters.
- The preceding code points ranges contain the following controls which are only valid in certain contexts in XML 1.1 documents, and whose usage is restricted and highly discouraged:
- `U+0001` … `U+0008`, `U+000B` … `U+000C`, `U+000E` … `U+001F`: this includes most (not all) C0 control characters
- `U+007F` … `U+0084`, `U+0086` … `U+009F` : this includes a C0 control character, and all but one C1 control.

Allowed encodings

- Implicitly unless indicated otherwise in prolog (eg. `<?xml version=" 1. 0" encoding="Windows-1250"?>`) then UTF-8 or UTF-16 is used.
- The distinction between UTF-8 and UTF-16 is done according to the first two bytes of the document entity (ie. file), by so-called byte-order-mark `xFFFE`.
- If not present, UTF-8 is assumed, thus UTF-8 is the implicit encoding of UNICODE in XML.

Character entities

- Enable writing characters not present in current available *font* on screen/print
- Mean to write chars having otherwise a special meaning (i.e. markup in XML)

Syntax of entities

A numeric character reference refers to a character by its Universal Character Set/Unicode code point, and uses the format:

```
&#nnnn;
```

or

```
&#xhhhh;
```

where `nnnn` is the code point in decimal form, and `hhhh` is the code point in hexadecimal form. The x must be lowercase in XML documents. The `nnnn` or `hhhh` may be any number of digits and may include leading zeros. The `hhhh` may mix uppercase and lowercase, though *uppercase* is the usual style.

Named entities

In contrast, a *character entity reference* refers to a character by the name of an entity which has the desired character as its replacement text. The entity must either be predefined (built into the markup language) or explicitly declared in a Document Type Definition (DTD). The format is the same as for any entity reference:

```
&name;
```

where `name` is the case-sensitive name of the entity. The semicolon is required.