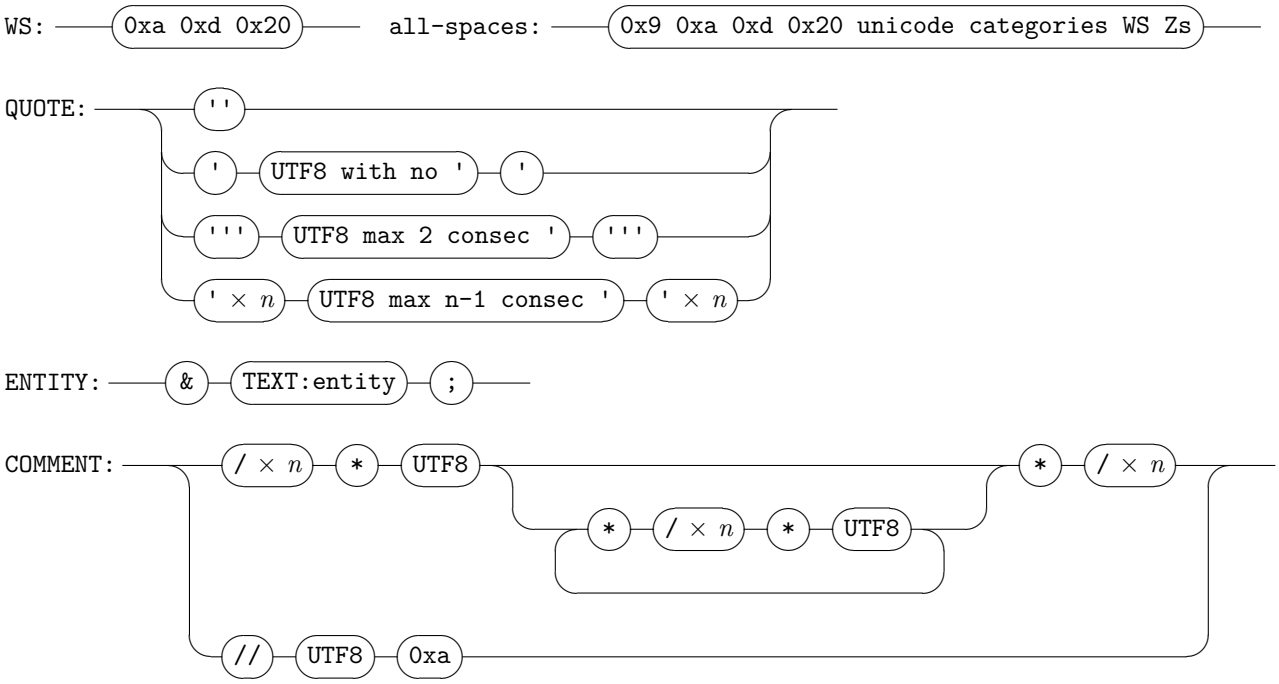
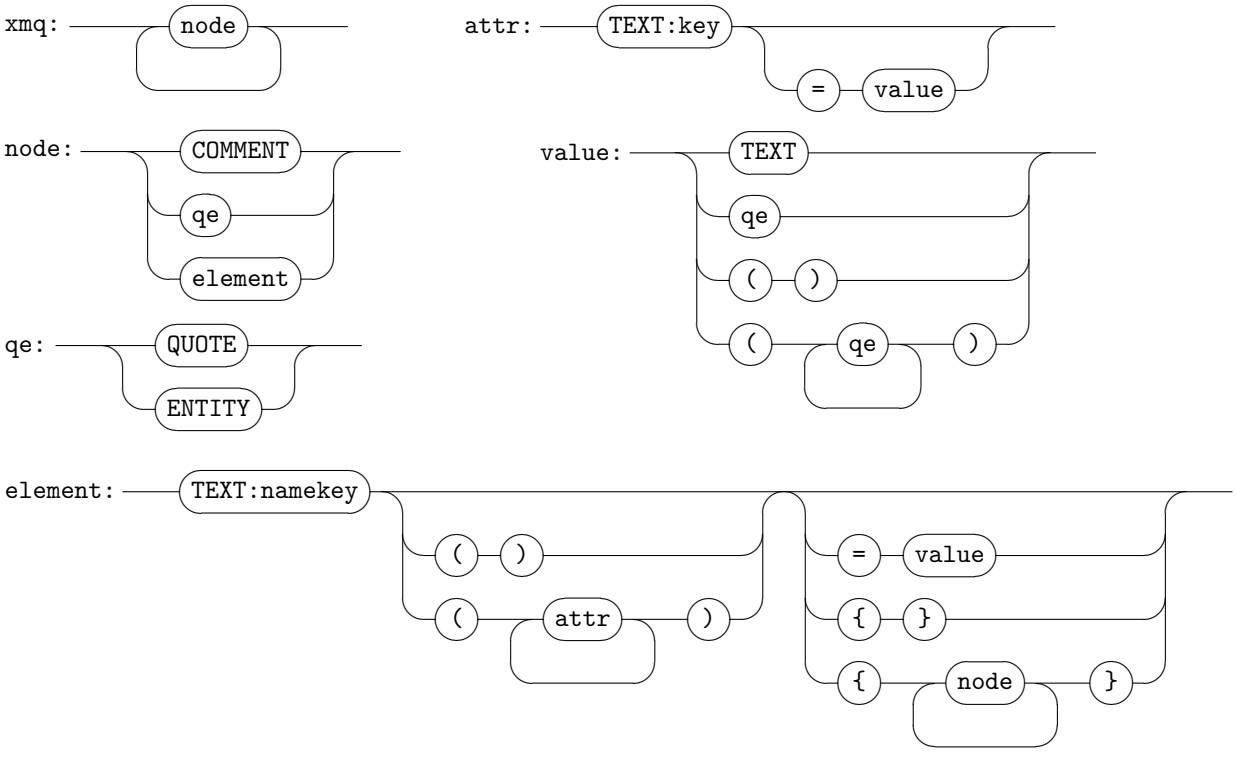
Specification for XMQ by Fredrik Öhrström 2023-12-29 oehrstroem@gmail.com

Input must be valid UTF8 (0x9 | 0xa | 0xd | [20-d7ff] | [e000-fffd] | [10000-10ffff])
CRLF pairs (0xd 0xa) and standalone CR (0xd) are treated as LF (0xa) when parsing.

LEXER




PARSER



## RULES

```
    TEXT:namekey TEXT:key and TEXT:entity
r1. Must start with a letter or underscore.
r2. Cannot start with the letters xml (or XML, or Xml, etc).
r3. Can contain letters, digits, hyphens, underscores and periods.
r4. Can contain a single colon separating the TEXT into two parts, each following r1,r2,r3.

    Two permitted exceptions to TEXT:namekey rule r1 and r3.
r5. A single !DOCTYPE before the first element and ?pi elements.

    If quoted content contains at least one newline then:
r6. Leading or ending WS with at least one newline will be trimmed.
r7. All spaces before a newline are removed.
r8. Incidental indentation (some spaces after a newline) is removed.
r9. The indent to be removed is the minimum source code indentation
    for text within the block where empty lines are ignored.
r10. The first line is prefixed with spaces if the following lines
     have lesser source code indentation.

     TEXT:namekey is (only for syntax highlighting) either a name or a key.
r11. it is a key if '=' follows immediately (ie no attributes), otherwise it is a name.

A quote with only spaces and at least one newline is equivalent to the empty string.
The element age=123 is shorthand for age{'123'}
Use (...) for attributes with newlines and explicit leading/ending spaces.
Inside a comment */* means a newline, used for compact xmq without newlines.
The tab 0x9 is not in WS since its variable size would confuse incidental indentation.
```

## EXAMPLES

```
car {
    // An example structure.
    regnr = 'ABC 123'
    color = red
    img   = /www/y.png
    tag   = <car>
}

div(id = 32) {
    h1 = Welcome!
    'Rest here weary
     traveller:'
    a(href = https://a.b.c) {
        img(url = /img/i.png)
        'Click here!'
    }
}
```

```
<car>
  <!-- An example structure. -->
  <regnr>ABC 123</regnr>
  <color>red</color>
  <img>/www/y.png</img>
  <tag>&lt;car&gt;</tag>
</car>


<div id="32">
  <h1>Welcome!</h1>
    Rest here weary
traveller:<a href="https://a.b.c">↩
↪<img url="/img/i.png">Click here!</a>
</div>
```

Html cannot be pretty printed with newlines here,
whereas xmq can be pretty printed without introducing whitespace.

## CORNERS

```
Explicit spaces: abc = '    '    abc {'    '}
A single newline: abc = &#10;       abc { &#10; }
Spaces surrounding newline: abc = (' ' &#10; ' ')   abc {' ' &#10; ' '}
Value with leading/ending quotes: x = ( &#39; 'quoted quote' &#39; ) or:
'''
'quoted quote'
'''
or: x { &#39; 'quoted quote' &#39; }
When highlighting: p = 123 is a key value and p(x) = 123 is a name value.
```