6.3. Regular expression syntax for `xsd:`

The regular expression syntax is fairly similar to that of Perl. Refer to the Appendix F of the XML Schema Datatypes specification for a complete definition of the regular expressions allowed in the `pattern` parameter of any of the `xsd:` datatypes.

Note

If you are working with Unicode, you should read the full specification, as there are a number of advanced features, not discussed here, that are most useful in Unicode work.

Here is a summary of most of the commonly used features.

p|*q*	Either pattern *p* or pattern *q*.
pq	Pattern *p* followed by pattern *q*.
p?	Matches pattern *p* or nothing at all. You could think of it as saying "*p* occurs optionally here."
*p**	Matches zero or more occurrences of *p*.
p+	Matches one or more occurrences of *p*.
p{*n*}	Matches exactly *n* occurrences of pattern *p*.
p{*n*,*m*}	Matches at least *n* occurrences, but no more than *m* occurrences, of pattern *p*.
p{*n*,}	Matches *n* or more occurrences of pattern *p*.
[c_1c_2...]	Matches any single character from inside the square brackets. For example, the pattern "`xsd:string { pattern='[abc]' }`" matches any of the characters `a`, `b`, or `c`. You can specify ranges of characters as "[c_1-c_2]". For example, the pattern "`[a-zA-Z]`" matches any letter, lowercase or uppercase.
[^c_1c_2...]	Matches any single character *except* those enumerated inside the square brackets. For example, the regular expression "`xsd:string { pattern='[^abc]' }`" matches any single character *except* `a`, `b`, or `c`.
(*p*)	Parentheses may be used for grouping. For example, pattern "`(ab)+`" matches "`ab`", "`abab`", "`ababab`", and so on.
`\r`	Matches the carriage return (ASCII CR) character.
`\n`	Matches the newline (ASCII LF) character.
`.`	Matches any character except newline or carriage return.
`\t`	Matches the tab (ASCII HT) character.
`\`*C*	Any of the following characters must be escaped by preceding them with a backslash: "`\ | . - ^ ? * + { } ( ) [ ]`". For example, "`pattern='\[\*\]'`" matches the string "`[*]`".
`\s`	Matches a *whitespace character*: space, tab, newline, or carriage return.

`\S`	Matches any character except a whitespace character.
`\i`	Matches a *name start character*: a letter, “`_`”, or “`:`”.
`\I`	Matches any character except a name start character.
`\c`	Matches a *name character*, that is, a name start character or digit.
`\C`	Matches any character except a name character.
`\d`	Matches a decimal digit (same as “`[0-9]`”).
`\D`	Matches any character except a decimal digit.

Here's an example of a pattern for a U. S. Postal Service zip code:

```
    xsd:string { pattern='[0-9]{5}(-[0-9]{4})?' }
```

That is, five digits, optionally followed by a hyphen and four more digits.

Contents: Relax NG Compact Syntax (RNC)
Previous: 5. The definition patterns
Help: Tech Computer Center: Help System
Home: About New Mexico Tech

John W. Shipman
Comments welcome: tcc-doc@nmt.edu
Last updated: 2008-02-14 18:43
URL: `http://www.nmt.edu/tcc/help/pubs/rnc/xsd.html`