5.5 Documentation

The ndoc documentation system has been extended to allow developers to provide the glossary entry for the word in the source for the extension. This uses the following XML format.

⟨*glossary*⟩ := <worddef [⟨*name*⟩] [⟨*id*⟩] [⟨*number*⟩] [⟨*wordset*⟩] [⟨*english*⟩] >
⟨*description*⟩
[⟨*rationale*⟩]
[⟨*testing*⟩]
[⟨*implement*⟩]
[⟨*cross ref*⟩]
</worddef>

⟨*name*⟩ := name="⟨*argtext*⟩"

⟨*id*⟩ := id="⟨*argtext*⟩"

⟨*number*⟩ := number="(⟨*digit*⟩ | .)* "

⟨*wordset*⟩ := wordset="⟨*argtext*⟩"

⟨*english*⟩ := english="⟨*argtext*⟩"

⟨*argtext*⟩ := (⟨*upper*⟩ | ⟨*lower*⟩ | ⟨*digit*⟩ | ⟨*punct*⟩ | ⟨*elements*⟩)*

⟨*elements*⟩ := & | " | < | >

A glossary definition is enclosed in <worddef> tags. The tag takes four optional arguments:

name The word name, by default is the name of the method, field or property. Due to the way ndoc works, the attributes are not available when processing comments, thus if the `Primitive` attribute has been used to rename the word, the name attribute must be given.

id The id used to identify the word in the cross referencing system. By default this is the name of the method, field or property.

number Every word in the standard has a number, this is that number. You can provide numbers for your own word lists.

wordset The wordset used for the definition. If not given the word set for the word is take as the current class name, minus the trailing `Words` if given. If the `Wordlist` annotation has been used, this argument should also be given.

english The english pronunciation for the word.

Where ⟨*argtext*⟩ can be any ⟨*upper*⟩ (upper case letter), ⟨*lower*⟩ (lower case letter), ⟨*digit*⟩ (numeric digit), ⟨*punct*⟩ (punctuation character, except &, ", < and >) and ⟨*element*⟩ (XML character elements). Thus word word `S"` should be written `S&quot;`.

The definition must include a <description> and may include <rationale>, <testing>, <implement> and <see> sections.

Description

This is the main description of the operation of the word. The description can be broken down into different ⟨*section*⟩s.

⟨*description*⟩ := <description> ⟨*text*⟩ ⟨*section*⟩* </description>

A <description> should include a general description of the word which may be broken down into further sections: <compile>, <execute>, <init>, <interpret>, <note>, <runtime> and/or <item>. Some sections take an optional type argument. This is normally displayed in the label for that section of the definition, with the exception of the <item> section where the type is the whole of the label.

⟨*section*⟩ := <compile> ⟨*text*⟩ </compile>
| <execute [type="⟨*argtext*⟩"]> ⟨*text*⟩ </execute>
| <init> ⟨*text*⟩ </init>
| <interpret> ⟨*text*⟩ </interpret>
| <note [type="⟨*argtext*⟩"]> ⟨*text*⟩ </note>
| <runtime [type="⟨*argtext*⟩"]> ⟨*text*⟩ </runtime>
| <item type="⟨*argtext*⟩"> ⟨*text*⟩ </item>

Rationale

If the word requires additional discussion that is not part of its definition, such discussion should be included in the rationale. This should include the rationale for including the word in the word set.

⟨*rationale*⟩ := <rationale> ⟨*text*⟩ (⟨*see*⟩ | ⟨*note*⟩ | ⟨*item*⟩)* </rationale>

Testing

The unit testing for a word should be given in the testing section. In this section ⟨*text*⟩ may be mixed with <test> tags to describe an individual test.

⟨*testing*⟩ := <testing> (⟨*text*⟩ | ⟨*test*⟩)* </testing>

⟨*test*⟩ := <test [type="(X | R)*"]>
<pre>⟨*argtext*⟩</pre>
<post>⟨*argtext*⟩</post>
</test>

For example,

```
<test type="XX"><pre>0 DUP</pre><post>0 0</post></test>
```

describes the following test:

```
T{ 0 DUP -> 0 0 XX}T
```

The type argument is used to provide the type of the parameters required for the floating-point test harness.

Implement

If the definition can be implemented in standard Forth, its definition should be given in this section.

⟨*implement*⟩ := <implement> ⟨*text*⟩ </implement>

Cross referencing

Cross references to other words should be included in this section. The section consists of a collection of ⟨*wref*⟩, ⟨*rref*⟩, ⟨*tref*⟩, ⟨*iref*⟩ and ⟨*xref*⟩ tags.

⟨*cross ref*⟩ := <see> ⟨*reference*⟩* </see>

⟨*reference*⟩ := ⟨*wref*⟩ | ⟨*rref*⟩ | ⟨*tref*⟩ | ⟨*iref*⟩ | ⟨*xref*⟩

⟨*wref*⟩ := <wref word="⟨*argtext*⟩" [⟨*wordset*⟩] />

⟨*rref*⟩ := <rref word="⟨*argtext*⟩" [⟨*wordset*⟩] />

⟨*tref*⟩ := <tref word="⟨*argtext*⟩" [⟨*wordset*⟩] />

⟨*iref*⟩ := <iref word="⟨*argtext*⟩" [⟨*wordset*⟩] />

⟨*xref*⟩ := <xref label="⟨*argtext*⟩"> ⟨*argtext*⟩ </xref>

The <wref>, <rref>, <tref> and <iref> tags provide a full cross references to the glossary, rationale, testing and implementation sections of a definition respectively. The word argument must give the id of the word it is referencing. The wordset argument must be given when referencing a word in a different word set. For example, the following would be used to provide a cross reference to the `S"` word in the Core word set:

```
<wref word="string" wordset="core" />
```

which would produce the following text, complete with a link to the definition.

6.1.2165 S"

Where the number is derived from the position of the word definition in the document and its number argument, if given.

Finally, the <xref> tag allows for the cross referencing between sections of the document. If the section identified by the label argument can be found, a full reference to the section is produced, complete with a link to the section. However, if the section cannot be found the ⟨*argtext*⟩ is used.

Normal text

The ⟨*text*⟩ consists of a collection of paragraphs, which may contain a combination of a limited number of XHTML tags and a few additional tags for the processing or Forth documentation.

⟨*text*⟩ := (⟨*xhtml*⟩ | ⟨*special*⟩)*

The XHTML tags are well known so no further discussion is required, except to say that any arguments are passed through to the glossary file without an further processing.

⟨*xhtml*⟩ := ⟨*para text*⟩
| ⟨*item*⟩*
| ⟨*item*⟩*
| <table> [⟨*caption*⟩] [⟨*header*⟩] ⟨*body*⟩ [⟨*footer*⟩] </table>

⟨*item*⟩ := ⟨*text*⟩

⟨*caption*⟩ := <caption> ⟨*argtext*⟩ </caption>

⟨*header*⟩ := <thead> ⟨*row*⟩ </thead>

⟨*body*⟩ := <tbody> ⟨*row*⟩* </tbody>

⟨*footer*⟩ := <tfoot> ⟨*row*⟩ </tfoot>

⟨*row*⟩ := <tr>(<th> ⟨*text*⟩ </th> | <td> ⟨*text*⟩ </td>)*</tr>

There are only three additional tags at the paragraph level.

⟨*special*⟩ := <para> ⟨*para text*⟩ </para>
| <stack [type="⟨*upper*⟩"]>
[<pre> ⟨*stack item*⟩* </pre>]
[<post> ⟨*stack item*⟩* </post>]
</stack>
| <code> (⟨*text*⟩ | ⟨*white space*⟩)* </code>

⟨*stack item*⟩ := ⟨*argtext*⟩ | " (⟨*argtext*⟩ | <⟨*lower*⟩/>)* "

<para> This is the ndoc equivalent of the XHTML <p> tag.

<stack> A stack description giving both the before (<pre>) and after (<post>) items. A stack description may be applied to any of the stacks, thus the type argument should be used when describing a stack other than the data stack (I.e., type="R" for the return stack).

A stack item may include a syntactic element, which is always enclosed in quotes. The standard defines a number of descriptors for use in the syntactic descriptions, such as <char>. Unfortunately, this is the same syntax as a tag, thus it has been necessary to make such syntactic descriptors self closing tags (<char/>). Any valid self-closed tag will be treated as a syntactic descriptor.

The stack items may include a few special formatting characters:

`x_{i}`	x subscript i	x_i
`x*i`	x multiplied by i	$x * i$
`n|u`	n or u	$n \mid u$

<code> Represents a multi-line forth code example. The code is placed in a indented paragraph of its own. The system pays attention to the white space within the definition. Particularly the line breaks and indentation.

If a word in the code has a definition within the current word set, the word will be linked back to its definition, unless formatted with another tag.

Formatted text

The <p> and <para> tags allow for both formatted and unformatted text.

⟨*para text*⟩ := ⟨*argtext*⟩ | "
| ⟨*text*⟩
| ⟨*text*⟩
| ⟨*reference*⟩
| <word [⟨*wordset*⟩]> ⟨*argtext*⟩ </word>
| <param> ⟨*stack item*⟩* </param>
| <arg> ⟨*argtext*⟩ </arg>
| <value> ⟨*argtext*⟩ </value>
| <c> ⟨*text*⟩ </c>

 and are taken from XHTML and need no further discussion. The following additional formatting tags are provided:

<word> Represents a forth word. If the word is defined within the current word set, it will link the word to its definition. The wordset argument must be used when referring to a word in a different word set.

<param> Represents a stack item within the the text. (*c-addr len*)

<arg> Represents an argument (⟨*text*⟩).

<value> Represents a constant value (`text`).

<c> Represents program code within the body of the text. If a word in the code has a definition within the current word set, the word will be linked back to its definition. In effect this is the same as applying <word> to every word in ⟨*text*⟩, unless formatted with another tag.