Source Code Highlight Filter

Revision History
Revision 8.2.5 18 November 2007

Table of Contents

The *AsciiDoc* distribution includes a source code syntax highlight filter (`source-highlight-filter.conf`) which uses GNU source-highlight [http://www.gnu.org/software/src-highlite/]. GNU source-highlight generates nicely formatted source code for most common programming languages.

Examples

Example 1. Python code snippet

This source-highlight filtered block:

```
[python]
source~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
''' A multi-line
    comment.'''
def sub_word(mo):
    ''' Single line comment.'''
    word = mo.group('word')     # Inline comment
    if word in keywords[language]:
        return quote + word + quote
    else:
        return word
source~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
```

Renders this highlighted source code:

```
''' A multi-line
    comment.'''
def sub_word(mo):
    ''' Single line comment.'''
    word = mo.group('word')     # Inline comment
    if word in keywords[language]:
        return quote + word + quote
    else:
        return word
```

Example 2. Ruby code snippet with line numbering

This source-highlight filtered block:

```
[ruby,numbered]
source~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
#
# Useful Ruby base class extensions.
#

class Array

  # Execute a block passing it corresponding items in
  # +self+ and +other_array+.
  # If self has less items than other_array it is repeated.

  def cycle(other_array)  # :yields: item, other_item
    other_array.each_with_index do |item, index|
      yield(self[index % self.length], item)
    end
  end

end

if $0 == __FILE__
  # Array#cycle test
  # true => 0
  # false => 1
  # true => 2
  # false => 3
  # true => 4
  puts 'Array#cycle test'
  [true, false].cycle([0, 1, 2, 3, 4]) do |a, b|
    puts "#{a.inspect} => #{b.inspect}"
  end
end
source~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
```

Renders this highlighted source code:

```
00001: #
00002: # Useful Ruby base class tension's.
00003: #
00004:
00005: class Array
00006:
00007:   # Execute a block passing it corresponding items in
00008:   # +self+ and +other_array+.
00009:   # If self has less items than other_array it is repeated.
00010:
00011:   def cycle(other_array)  # :yields: item, other_item
00012:     other_array.each_with_index do |item, index|
00013:       yield(self[index % self.length], item)
00014:     end
00015:   end
00016:
00017: end
00018:
00019: if $0 == __FILE__
00020:   # Array#cycle test
00021:   # true => 0
00022:   # false => 1
00023:   # true => 2
00024:   # false => 3
00025:   # true => 4
00026:   puts 'Array#cycle test'
00027:   [true, false].cycle([0, 1, 2, 3, 4]) do |a, b|
00028:     puts "#{a.inspect} => #{b.inspect}"
00029:   end
00030: end
```

Installation

As a prerequisite you will have installed and tested GNU source-highlight [http://www.gnu.org/software/src-highlite/] (most distributions have this package).

Test the filter by converting the test file to HTML with *AsciiDoc*:

```
$ asciidoc -v ./filters/source-highlight-filter-test.txt
$ firefox ./filters/source-highlight-filter-test.html &
```

DocBook Output Definition file

As of Source-highlight 2.3 the *AsciiDoc* DocBook output definition file has been included in the Source-highlight distribution so you can skip this section if you have version 2.3 or greater.

Source-highlight 2.2 and below do not ship with DocBook output definition files. If you want to generate DocBook instead of HTML the following simple DocBook output definition file will do the trick:

```
extension "xml"

bold "<emphasis role=\"strong\">$text</emphasis>"
italics "<emphasis>$text</emphasis>"

translations
"&" "&amp;"
"<" "&lt;"
">" "&gt;"
end
```

To install it:

1. Name it `docbook.outlang` and copy it to the `source-highlight` configuration files directory (typically `/usr/local/share/source-highlight/`).
2. Add the following line to `/usr/local/share/source-highlight/outlang.map`:

   ```
   docbook = docbook.outlang
   ```