Nassi-Shneiderman Diagram Generator

The Nassi-Shneiderman Diagram Generator is a command line tool which converts xml documents describing the structure of a Nassi-Shneiderman diagram into png files displaying said diagrams.

1. Requirements

To use the generator, Java 7 needs to be properly installed and accessible from the command line. A good xml editor is recommended.

2. Creating the XML Files

2.1 The XML Schema

The file nsdg.xml contains the schema for the xml files. If you want to, you can include it in your document like this:

```
<diagram xmlns="http://greenlightning.eu/nsdg"
       xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
       xsi:schemaLocation="http://greenlightning.eu/nsdg path/to/your/nsdg.xml">
```

2.2 Tag Overview

Tag	Attributes	Content
• Root Tag		
diagram	-	(**Element Tag**)*
• Element Tags		
block, procedure, break	-	[String]
branch	[condition=String]	[left], [right]
switch	[expression=String]	(case)+, [default]
loop	[type='*test-first*', '*test-last*', 'infinite'], [condition=String]	(**Element Tag**)*
• Helper Tags		
left, right	[label=String]	(**Element Tag**)*
case, default	[label=String]	(**Element Tag**)*

Attributes and content in square brackets are optional.

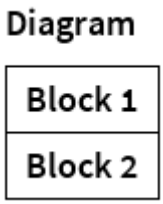
Note: Because there are some special characters (such as '<', '>', '&'...) which you must not use directly in xml, the following escape sequences have been added. To make editing of the xml documents easier, they can be used instead of the xml entities in any string attribute or tag with string content. In addition, escape sequences for control characters such as the line break have been added.

Escape Sequence	Replacement
\t	<tab>
\b	<backspace>
\n	<newline>
\r	<carriage return>
\f	<formfeed>
\a	&
\g	>
\l (alias \k)	<
\p	'
\q	"
\\	\

2.3 Tags

The root of the xml file must be a diagram tag. You can specify the xml schema in the diagram tag as shown above. Furthermore, you can specify a title.

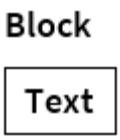
Like many other tags the diagram tag contains a list of element tags. The elements will be drawn one below the other in the same sequence as in the xml file. If the diagram tag contains no element tags, the percentage sign will be displayed in its center to show it is blank.

`<diagram title="Diagram">`
`  <block>Block 1</block>`
`  <block>Block 2</block>`
`</diagram>`
	

`<diagram />`	

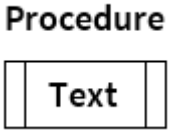
2.3.1 Block

The block element is used to display instructions. The block tag encloses the text which will be drawn inside the block.

`<block>Text</block>`	

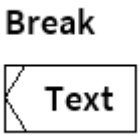
2.3.2 Procedure

The procedure is similar to the block. It denotes the invocation of a subprogram such as a procedure, function or method.

`<procedure>Text</procedure>`	

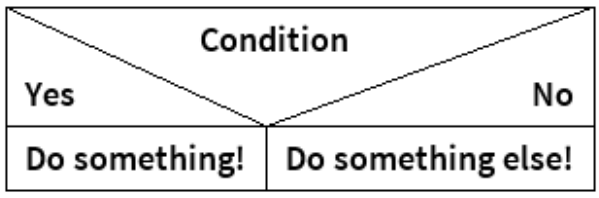
2.3.3 Break

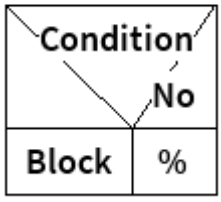
The break or exit element is similar to the block as well, but marks the termination of some part of the program.

`<break>Text</break>`	

2.3.4 Branch

The branch features a condition which is passed to the branch tag as an attribute. It may contain one left tag and / or one right tag. These tags in turn contain the element tags which are displayed in either side of the branch element. If both the left and the right tags are present, the left tag must come first. The left and right tags may have a label attribute which is displayed in the title section of the branch. If these tags do not have a label attribute, nothing is displayed. If they do not contain any elements, the percentage sign is displayed. If one or both of these tags aren't present at all, they are treated as having a blank label and no content.

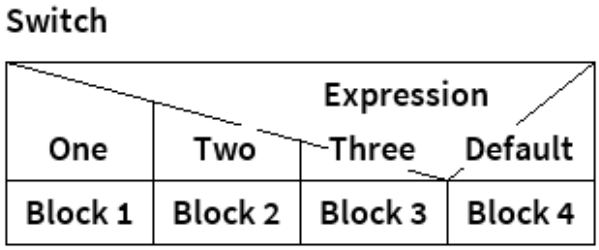
`<branch condition="Condition">`
`  <left label="Yes">`
`  <block>Do something!</block>`
`  </left>`
`  <right label="No">`
`  <block>`
`  Do something else!`
`  </block>`
`  </right>`
`</branch>`
	

`<branch condition="Condition">`
`  <left>`
`  <block>Block</block>`
`  </left>`
`  <right label="No" />`
`</branch>`
	

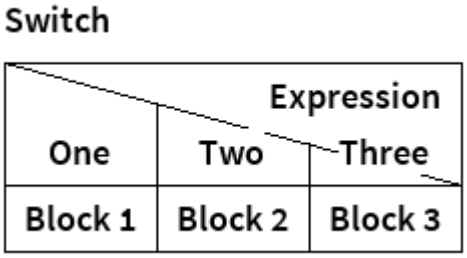
2.3.5 Switch

The switch is similar to the branch. It must contain one or more case tags and might contain one default tag at the end. The case and default tags behave like the left and right tags of the branch. If the default tag is not present, though, it is not displayed at all.

```
<switch expression="Expression">
	<case label="One">
		<block>Block 1</block>
	</case>
	<case label="Two">
		<block>Block 2</block>
	</case>
	<case label="Three">
		<block>Block 3</block>
	</case>
	<default label="Default">
		<block>Block 4</block>
	</default>
</switch>
```



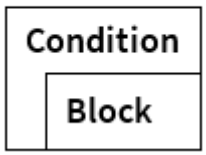

```
<switch expression="Expression">
	<case label="One">
		<block>Block 1</block>
	</case>
	<case label="Two">
		<block>Block 2</block>
	</case>
	<case label="Three">
		<block>Block 3</block>
	</case>
</switch>
```



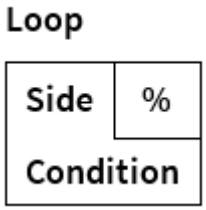
2.3.6 Loop

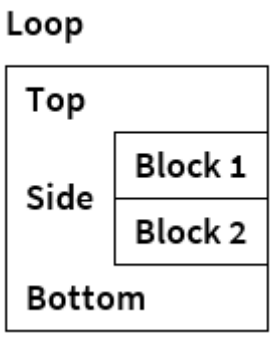
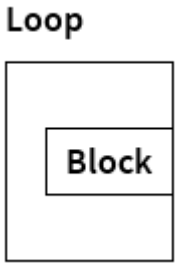
The loop tag has a type attribute which may be either 'test-first', 'test-last' or 'infinite', where 'test-first' is the default value. This attribute determines how the loop is rendered. Also, test loops have the condition and side attributes, whereas the infinite loop uses a top, a side and a bottom attribute.

```
<loop condition="Condition">
	<block>Block</block>
</loop>
```




```
<loop type="test-last" side="Side" condition="Condition" />
```



<loop type="infinite" top="Top" side="Side" bottom="Bottom"> <block>Block 1</block> <block>Block 2</block> </loop>	
<loop type="infinite"> <block>Block</block> </loop>	

3. Using the Generator

Start the generator as follows:

```
java -jar nsdg.jar <path-to-xml-file>...
```

You can specify multiple xml files separated by spaces and they will be handled individually, one after another.

The generator will put the png files in the same directory and give them the same names as the original xml files, but with the png file extension instead.

Files will be overwritten without warning!