TeXDoclet Java Documentation

Created with Javadoc TeXDoclet Doclet

Greg Wonderly Soeren Caspersen Stefan Marx

June 16, 2012

Contents

Class Hierarchy

Classes

- java.lang.Object
  - com.keypoint.PngEncoder (in 2.1, page 18)
    - com.keypoint.PngEncoderB (in 2.2, page 26)
  - com.sun.javadoc.Doclet
    - org.stfm.texdoclet.TeXDoclet (in 1.9, page 14)
  - javax.swing.text.html.HTMLEditorKit.ParserCallback
    - org.stfm.texdoclet.HTMLtoLaTeXBackEnd (in 1.4, page 6)
  - org.stfm.texdoclet.ClassHierachy (in 1.2, page 4)
  - org.stfm.texdoclet.HelpOutput (in 1.3, page 5)
  - org.stfm.texdoclet.InterfaceHierachy (in 1.5, page 8)
  - org.stfm.texdoclet.Package (in 1.6, page 9)
  - org.stfm.texdoclet.TableInfo (in 1.7, page 11)
  - org.stfm.texdoclet.TestFilter (in 1.8, page 13)

Interfaces

- org.stfm.texdoclet.ClassFilter (in 1.1, page 4)

Chapter 1

Package org.stfm.texdoclet

Package Contents *Page*

Interfaces

Classes

This doclet is based on the doclet originally created by Greg Wonderly of C2 technologies Inc. and its revision by XO Software. The project of Greg Wonderly is available here : http://java.net/projects/texdoclet.

1.1 Interface ClassFilter

This interface can be implemented and a class name provided to the Doclet to filter which classes are and are not included in the output document.

1.1.1 Declaration

public interface ClassFilter

1.1.2 All known subinterfaces

TestFilter (in 1.8, page 13)

1.1.3 All classes known to implement interface

TestFilter (in 1.8, page 13)

1.1.4 Method summary

includeClass(ClassDoc) Filters the ClassDoc passed.

1.1.5 Methods

- **includeClass**
  `boolean` **includeClass**`(com.sun.javadoc.ClassDoc` **cd**`)`
  - **Description**
    Filters the ClassDoc passed. If true is returned, the passed class will be included into the output. If false is returned, this document will not be included.

1.2 Class ClassHierachy

Manages and prints a class hierarchy. Use `add` to add another class to the hierarchy. Use `printTree` to print the corresponding LaTeX.

1.2.1 Declaration

public class ClassHierachy
extends java.lang.Object

1.2.2 Field summary

root

1.2.3 Constructor summary

ClassHierachy() Creates new ClassHierachy

1.2.4 Method summary

add(ClassDoc) Adds another class to the hierachy
printBranch(RootDoc, SortedMap, double, double) Prints a branch of the tree.
printTree(RootDoc, double) Prints the LaTeX corresponding to the tree.

1.2.5 Fields

- public java.util.SortedMap **root**

1.2.6 Constructors

- **ClassHierachy**
  `public` **ClassHierachy**`()`
  - **Description**
    Creates new ClassHierachy

1.2.7 Methods

- **add**
  `protected java.util.SortedMap` **add**`(com.sun.javadoc.ClassDoc` **cls**`)`
  - **Description**
    Adds another class to the hierachy
- **printBranch**
  `protected void` **printBranch**`(com.sun.javadoc.RootDoc` **rootDoc**`, java.util.SortedMap` **map**`, double` **indent**`, double` **overviewindent**`)`
  - **Description**
    Prints a branch of the tree. The branch is printed using `TeXDoclet.os`.
- **printTree**
  `public void` **printTree**`(com.sun.javadoc.RootDoc` **rootDoc**`, double` **overviewindent**`)`
  - **Description**
    Prints the LaTeX corresponding to the tree. The tree is printed using `TeXDoclet.os`.

1.3 Class HelpOutput

1.3.1 Declaration

public class HelpOutput
extends java.lang.Object

1.3.2 Constructor summary

HelpOutput()

1.3.3 Method summary

printHelp()

1.3.4 Constructors

- **HelpOutput**
  `public` **HelpOutput**`()`

1.3.5 Methods

- **printHelp**
  `protected static void` **printHelp**`()`

1.4 Class HTMLtoLaTeXBackEnd

This class implements a `ParserCallback` that translates HTML to the corresponding LaTeX. Not all tags a processed but the most common are.

1.4.1 See also

– `javax.swing.text.html.parser.ParserDelegator`

1.4.2 Declaration

public class HTMLtoLaTeXBackEnd
extends javax.swing.text.html.HTMLEditorKit.ParserCallback

1.4.3 Constructor summary

HTMLtoLaTeXBackEnd(StringBuffer) Constructs a new instance.

1.4.4 Method summary

fixText(String) Converts a HTML string into LaTeX using an instance of `HTMLtoLaTeXBackEnd`.
handleEndTag(HTML.Tag, int) This method handles HTML tags that mark an ending (e.g.
handleSimpleTag(HTML.Tag, MutableAttributeSet, int) This method handles simple HTML tags (e.g.
handleStartTag(HTML.Tag, MutableAttributeSet, int) This method handles HTML tags that mark a beginning (e.g.
handleText(char[], int) This method handles all other text.

1.4.5 Constructors

- **HTMLtoLaTeXBackEnd**
  `public` **HTMLtoLaTeXBackEnd**`(java.lang.StringBuffer` **ret**`)`
  - **Description**
    Constructs a new instance.
  - **Parameters**
    * `StringBuffer` – The `StringBuffer` where the translated HTML is appended.

1.4.6 Methods

- **fixText**
  `public static java.lang.String` **fixText**`(java.lang.String` **str**`)`
  - **Description**
    Converts a HTML string into LaTeX using an instance of `HTMLtoLaTeXBackEnd`.
- **handleEndTag**
  `public void` **handleEndTag**`(javax.swing.text.html.HTML.Tag` **tag**`, int` **pos**`)`
  - **Description**
    This method handles HTML tags that mark an ending (e.g. </P>-tags). It is called by the parser whenever such a tag is encountered.
- **handleSimpleTag**
  `public void` **handleSimpleTag**`(javax.swing.text.html.HTML.Tag` **tag**`,`
  `javax.swing.text.MutableAttributeSet` **attrSet**`, int` **pos**`)`
  - **Description**
    This method handles simple HTML tags (e.g. <HR>-tags). It is called by the parser whenever such a tag is encountered.
- **handleStartTag**
  `public void` **handleStartTag**`(javax.swing.text.html.HTML.Tag` **tag**`,`
  `javax.swing.text.MutableAttributeSet` **attrSet**`, int` **pos**`)`
  - **Description**
    This method handles HTML tags that mark a beginning (e.g. <P>-tags). It is called by the parser whenever such a tag is encountered.
- **handleText**
  `public void` **handleText**`(char[]` **data**`, int` **pos**`)`
  - **Description**
    This method handles all other text.

1.4.7 Members inherited from class HTMLEditorKit.ParserCallback

`javax.swing.text.html.HTMLEditorKit.ParserCallback`
flush, handleComment, handleEndOfLineString, handleEndTag, handleError, handleSimpleTag, handleStartTag, handleText, IMPLIED

1.5 Class InterfaceHierachy

Manages and prints a interface hierarchy. Use `add` to add another interface to the hierarchy. Use `printTree` to print the corresponding LaTeX.

1.5.1 Declaration

public class InterfaceHierachy
extends java.lang.Object

1.5.2 Field summary

root

1.5.3 Constructor summary

InterfaceHierachy() Creates new InterfaceHierachy

1.5.4 Method summary

add(ClassDoc) Adds another interface to the hierachy
printBranch(RootDoc, SortedMap, double, double) Prints a branch of the tree.
printTree(RootDoc, double) Prints the LaTeX corresponding to the tree.

1.5.5 Fields

- public java.util.SortedMap **root**

1.5.6 Constructors

- **InterfaceHierachy**
  `public` **InterfaceHierachy**`()`
  - **Description**
    Creates new InterfaceHierachy

1.5.7 Methods

- **add**
  `protected java.util.SortedMap` **add**`(com.sun.javadoc.ClassDoc` **cls**`)`

– **Description**
Adds another interface to the hierachy

- **printBranch**
`protected void` **printBranch**`(com.sun.javadoc.RootDoc` **rootDoc**`, java.util.SortedMap` **map**`, double` **indent**`, double` **overviewindent**`)`

  – **Description**
  Prints a branch of the tree. The branch is printed using `TeXDoclet.os`.

- **printTree**
`public void` **printTree**`(com.sun.javadoc.RootDoc` **rootDoc**`, double` **overviewindent**`)`

  – **Description**
  Prints the LaTeX corresponding to the tree. The tree is printed using `TeXDoclet.os`.

1.6 Class Package

This class is used to manage the contents of a Java package. It accepts ClassDoc objects and examines them and groups them according to whether they are classes, interfaces, exceptions or errors. The accumulated Vectors can then be processed to get to all of the elements of the package that fall into each category. If needed the classes, interfaces, exceptions and errors can be sorted using the `sort` method.

1.6.1 See also

– `Package.sort()` (in 1.6.8, page 10)

1.6.2 Declaration

public class Package
extends java.lang.Object

1.6.3 Field summary

classes The classes this package has in it
errors The errors this package has in it
exceptions The exceptions this package has in it
interfaces The interfaces this package has in it
pkg The name of the package this object is for
pkgDoc

1.6.4 Constructor summary

Package(String, PackageDoc) Construct a new object corresponding to the passed package name.

1.6.5 Method summary

addElement(ClassDoc) Adds a ClassDoc element to this package.
sort() Sorts the vectors of classes, interfaces exceptions and errors.

1.6.6 Fields

- protected com.sun.javadoc.PackageDoc **pkgDoc**
- protected java.lang.String **pkg**
  - The name of the package this object is for
- protected java.util.Vector **classes**
  - The classes this package has in it
- protected java.util.Vector **interfaces**
  - The interfaces this package has in it
- protected java.util.Vector **exceptions**
  - The exceptions this package has in it
- protected java.util.Vector **errors**
  - The errors this package has in it

1.6.7 Constructors

- **Package**
  `public` **Package**`(java.lang.String` **pkg**`, com.sun.javadoc.PackageDoc` **doc**`)`
  - **Description**
    Construct a new object corresponding to the passed package name.
  - **Parameters**
    * `pkg` – the package name to use

1.6.8 Methods

- **addElement**
  `public void` **addElement**`(com.sun.javadoc.ClassDoc` **cd**`)`
  - **Description**
    Adds a ClassDoc element to this package.
  - **Parameters**
    * `cd` – the object to add to this package
- **sort**
  `public void` **sort**`()`
  - **Description**
    Sorts the vectors of classes, interfaces exceptions and errors.

1.7 Class TableInfo

This class provides support for converting HTML tables into LaTeX tables. Some of the things **NOT** implemented include the following:

- valign attributes are not processed, but align= is.
- rowspan attributes are not processed, but colspan= is.
- the argument to border= in the table tag is not used to control line size

Here is an example table.

Column 1 Heading	**Column two heading**	**Column three heading**
data	Span two columns	
more data	right	left

A nested table example		
Column 1 Heading	**Coludliadfuapfd a fia fopia foipapio dupoau foapoifd pdp- fiu apsd oipoioaofid- uaopiu- fopiiiiiiii- iimn two heading**	**Column three heading**
data	Span two columns	
more data	right	left
1 2 3 4	first line second line third line fourth line	

1.7.1 Declaration

public class TableInfo
extends java.lang.Object

1.7.2 Constructor summary

TableInfo()

1.7.3 Method summary

endCol() Ends the current column.
endRow() Ends the current row.
endTable() Ends the table, closing the last row as needed
startCol(MutableAttributeSet) Starts a new column, possibly closing the current column if needed
startHeadCol(MutableAttributeSet) Starts a new Heading column, possibly closing the current column if needed.
startRow(MutableAttributeSet) Starts a new row, possibly closing the current row if needed
startTable(StringBuffer, MutableAttributeSet) Constructs a new table object and starts processing of the table by scanning the <table> passed to count columns.

1.7.4 Constructors

- **TableInfo**
  `public` **TableInfo**`()`

1.7.5 Methods

- **endCol**
  `public void` **endCol**`()`
  - **Description**
    Ends the current column.
  - **Parameters**
    * `ret` – The output buffer to put LaTeX 2ε into.
- **endRow**
  `public void` **endRow**`()`
  - **Description**
    Ends the current row.
  - **Parameters**
    * `ret` – The output buffer to put LaTeX 2ε into.
- **endTable**
  `public java.lang.StringBuffer` **endTable**`()`
  - **Description**
    Ends the table, closing the last row as needed
  - **Parameters**

    * `ret` – The output buffer to put LaTeX 2_ε into.

- **startCol**
  `public void` **startCol**`(javax.swing.text.MutableAttributeSet` **attrSet**`)`
  - **Description**
    Starts a new column, possibly closing the current column if needed
  - **Parameters**
    * `ret` – The output buffer to put LaTeX 2_ε into.
    * `p` – the properties from the <td> tag

- **startHeadCol**
  `public void` **startHeadCol**`(javax.swing.text.MutableAttributeSet` **attrSet**`)`
  - **Description**
    Starts a new Heading column, possibly closing the current column if needed. A Heading column has a Bold Face font directive around it.
  - **Parameters**
    * `ret` – The output buffer to put LaTeX 2_ε into.
    * `p` – The properties from the <th> tag

- **startRow**
  `public void` **startRow**`(javax.swing.text.MutableAttributeSet` **attrSet**`)`
  - **Description**
    Starts a new row, possibly closing the current row if needed
  - **Parameters**
    * `ret` – The output buffer to put LaTeX into.
    * `p` – The properties from the <tr> tag

- **startTable**
  `public java.lang.StringBuffer` **startTable**`(java.lang.StringBuffer` **org**`,`
  `javax.swing.text.MutableAttributeSet` **attrSet**`)`
  - **Description**
    Constructs a new table object and starts processing of the table by scanning the <table> passed to count columns.
  - **Parameters**
    * `p` – properties found on the <table> tag
    * `ret` – the result buffer that will contain the output
    * `table` – the input string that has the entire table definition in it.
    * `off` – the offset into <table> where scanning should start

1.8 Class TestFilter

This class filters out classes beginning with "Test" when applied to the Doclet.

1.8.1 Declaration

public class TestFilter
extends java.lang.Object
implements ClassFilter

1.8.2 Constructor summary

TestFilter()

1.8.3 Method summary

includeClass(ClassDoc) Returns false if class name starts with "Test".

1.8.4 Constructors

- **TestFilter**
  `public` **TestFilter**`()`

1.8.5 Methods

- **includeClass**
  `public boolean` **includeClass**`(com.sun.javadoc.ClassDoc` **cd**`)`
  - **Description**
    Returns false if class name starts with "Test".

1.9 Class TeXDoclet

This class provides a Java `javadoc` Doclet which generates a LaTeX 2_ε document out of the java classes that it is used on. This is convenient for creating printable documentation complete with cross reference information.

Supported HTML tags

<a> including an additional attribut "doprinturl". Since the output of the doclet should be printable, the href attribut of tags is printed in parentheses following the link if attribut "doprinturl" is set. Sometimes this is undesirable, and omitting "doprinturl" attribut will prevent this.

<dl> with the associated <dt><dd></dl>tags

<p> but not align=center...yet

 but not clear=xxx

<table> including all the associated <td><th><tr></td></th></tr>

 ordered lists

 unordered lists

<font> font coloring

<pre> preformatted text

<code> fixed point fonts

<i> italized fonts

<b> bold fonts

<sub> subscript

<sup> superscript

 center

<img> image located in java sources (<img src="package path/image name">)

1. example converted from JPG: 

2. example converted from GIF: 

<img> image located in the www: (see image at http://upload.wikimedia.org/wikipedia/commons/9/92/LaTeX_

Extra tags

A new tag is defined: <TEX>. This tag is useful for passing TEX code directly to the TEX compiler. The following code:

```
<TEX txt="\[ F\left( x \right) = \int_{ - \infty }^x {\frac{1}{{\sqrt {2\pi }
            }}e^{ - \frac{{z^2 }}{2}} dz} \]">
<BR><BR><B>This alternative text will appear if the javadoc/HTML is parsed
by any other doclet/browser</B><BR><BR></TEX>
```

will produce the following result:

$$F\left(x\right) = \int_{-\infty}^{x} \frac{1}{\sqrt{2\pi}} e^{-\frac{z^2}{2}} dz$$

The "alternative" text is ignored by the TeXDoclet, but useful if you want to use both the TeXDoclet and a regular HTML based doclet.

1.9.1 See also

- HTMLtoLaTeXBackEnd (in 1.4, page 6)
- TeXDoclet.start(RootDoc) (in 1.9.8, page 17)

1.9.2 Declaration

public class TeXDoclet
extends com.sun.javadoc.Doclet

1.9.3 Field summary

CHAPTER_LEVEL
os Writer for writing to output file
SECTION_LEVEL
SUBSECTION_LEVEL

1.9.4 Constructor summary

TeXDoclet()

1.9.5 Method summary

finish()
init()
initSections()
main(String[])
optionLength(String) Returns how many arguments would be consumed if `option` is a recognized option.
start(RootDoc) Called by the framework to format the entire document
validOptions(String[][], DocErrorReporter) Checks the passed options and their arguments for validity.

1.9.6 Fields

- public static final java.lang.String **SECTION_LEVEL**
- public static final java.lang.String **CHAPTER_LEVEL**
- public static final java.lang.String **SUBSECTION_LEVEL**
- public static java.io.PrintWriter **os**
  - Writer for writing to output file

1.9.7 Constructors

- **TeXDoclet**
  `public` **TeXDoclet**`()`

1.9.8 Methods

- **finish**
  `public static void` **finish**`()`

- **init**
  `public static void` **init**`()`
- **initSections**
  `public static void` **initSections**`()`
- **main**
  `public static void` **main**`(java.lang.String[]` **args**`)`
- **optionLength**
  `public static int` **optionLength**`(java.lang.String` **option**`)`
  - **Description**
    Returns how many arguments would be consumed if `option` is a recognized option.
  - **Parameters**
    * `option` – the option to check
- **start**
  `public static boolean` **start**`(com.sun.javadoc.RootDoc` **root**`)`
  - **Description**
    Called by the framework to format the entire document
  - **Parameters**
    * `root` – the root of the starting document
- **validOptions**
  `public static boolean` **validOptions**`(java.lang.String[][]` **args**`,`
  `com.sun.javadoc.DocErrorReporter` **err**`)`
  - **Description**
    Checks the passed options and their arguments for validity.
  - **Parameters**
    * `args` – the arguments to check
    * `err` – the interface to use for reporting errors

1.9.9 Members inherited from class Doclet

`com.sun.javadoc.Doclet`
languageVersion, optionLength, start, validOptions

Chapter 2

Package com.keypoint

Package Contents *Page*

Classes

2.1 Class PngEncoder

2.1.1 Declaration

public class PngEncoder
extends java.lang.Object

2.1.2 All known subclasses

PngEncoderB (in 2.2, page 26)

2.1.3 Field summary

bytePos
bytesPerPixel
compressionLevel
crc
crcValue
dataPos
ENCODE_ALPHA Constant specifying that alpha channel should be encoded.
encodeAlpha
endPos
filter
FILTER_LAST
FILTER_NONE Constants for filters

FILTER_SUB
FILTER_UP
hdrPos
height
image
leftBytes
maxPos
NO_ALPHA Constant specifying that alpha channel should not be encoded.
pngBytes
priorRow
width

2.1.4 Constructor summary

PngEncoder() Class constructor
PngEncoder(Image) Class constructor specifying Image to encode, with no alpha channel encoding.
PngEncoder(Image, boolean) Class constructor specifying Image to encode, and whether to encode alpha.
PngEncoder(Image, boolean, int) Class constructor specifying Image to encode, whether to encode alpha, and filter to use.
PngEncoder(Image, boolean, int, int) Class constructor specifying Image source to encode, whether to encode alpha, filter to use, and compression level.

2.1.5 Method summary

filterSub(byte[], int, int) Perform "sub" filtering on the given row.
filterUp(byte[], int, int) Perform "up" filtering on the given row.
getCompressionLevel() Retrieve compression level
getEncodeAlpha() Retrieve alpha encoding status.
getFilter() Retrieve filtering scheme
pngEncode() Creates an array of bytes that is the PNG equivalent of the current image.
pngEncode(boolean) Creates an array of bytes that is the PNG equivalent of the current image, specifying whether to encode alpha or not.
resizeByteArray(byte[], int) Increase or decrease the length of a byte array.
setCompressionLevel(int) Set the compression level to use
setEncodeAlpha(boolean) Set the alpha encoding on or off.
setFilter(int) Set the filter to use
setImage(Image) Set the image to be encoded
writeByte(int, int) Write a single byte into the pngBytes array at a given position.
writeBytes(byte[], int) Write an array of bytes into the pngBytes array.
writeBytes(byte[], int, int) Write an array of bytes into the pngBytes array, specifying number of bytes to write.
writeEnd() Write a PNG "IEND" chunk into the pngBytes array.
writeHeader() Write a PNG "IHDR" chunk into the pngBytes array.
writeImageData() Write the image data into the pngBytes array.

writeInt2(int, int) Write a two-byte integer into the pngBytes array at a given position.
writeInt4(int, int) Write a four-byte integer into the pngBytes array at a given position.
writeString(String, int) Write a string into the pngBytes array at a given position.

2.1.6 Fields

- public static final boolean **ENCODE_ALPHA**
  - Constant specifying that alpha channel should be encoded.
- public static final boolean **NO_ALPHA**
  - Constant specifying that alpha channel should not be encoded.
- public static final int **FILTER_NONE**
  - Constants for filters
- public static final int **FILTER_SUB**
- public static final int **FILTER_UP**
- public static final int **FILTER_LAST**
- protected byte **pngBytes**
- protected byte **priorRow**
- protected byte **leftBytes**
- protected java.awt.Image **image**
- protected int **width**
- protected int **height**
- protected int **bytePos**
- protected int **maxPos**
- protected int **hdrPos**
- protected int **dataPos**
- protected int **endPos**
- protected java.util.zip.CRC32 **crc**
- protected long **crcValue**
- protected boolean **encodeAlpha**
- protected int **filter**
- protected int **bytesPerPixel**
- protected int **compressionLevel**

2.1.7 Constructors

- **PngEncoder**
  `public` **PngEncoder**`()`
  - **Description**
    Class constructor
- **PngEncoder**
  `public` **PngEncoder**`(java.awt.Image` **image**`)`
  - **Description**
    Class constructor specifying Image to encode, with no alpha channel encoding.
  - **Parameters**
    * `image` – A Java Image object which uses the DirectColorModel
  - **See also**
    * `java.awt.Image`
- **PngEncoder**
  `public` **PngEncoder**`(java.awt.Image` **image**`,` `boolean` **encodeAlpha**`)`
  - **Description**
    Class constructor specifying Image to encode, and whether to encode alpha.
  - **Parameters**
    * `image` – A Java Image object which uses the DirectColorModel
    * `encodeAlpha` – Encode the alpha channel? false=no; true=yes
  - **See also**
    * `java.awt.Image`
- **PngEncoder**
  `public` **PngEncoder**`(java.awt.Image` **image**`,` `boolean` **encodeAlpha**`,` `int` **whichFilter**`)`
  - **Description**
    Class constructor specifying Image to encode, whether to encode alpha, and filter to use.
  - **Parameters**
    * `image` – A Java Image object which uses the DirectColorModel
    * `encodeAlpha` – Encode the alpha channel? false=no; true=yes
    * `whichFilter` – 0=none, 1=sub, 2=up
  - **See also**
    * `java.awt.Image`
- **PngEncoder**
  `public` **PngEncoder**`(java.awt.Image` **image**`,` `boolean` **encodeAlpha**`,` `int` **whichFilter**`,` `int` **compLevel**`)`

- **Description**
  Class constructor specifying Image source to encode, whether to encode alpha, filter to use, and compression level.
- **Parameters**
  - `image` – A Java Image object
  - `encodeAlpha` – Encode the alpha channel? false=no; true=yes
  - `whichFilter` – 0=none, 1=sub, 2=up
  - `compLevel` – 0..9
- **See also**
  - `java.awt.Image`

2.1.8 Methods

- **filterSub**
  `protected void` **filterSub**`(byte[]` **pixels**`, int` **startPos**`, int` **width**`)`
  - **Description**
    Perform "sub" filtering on the given row. Uses temporary array leftBytes to store the original values of the previous pixels. The array is 16 bytes long, which will easily hold two-byte samples plus two-byte alpha.
  - **Parameters**
    - `pixels` – The array holding the scan lines being built
    - `startPos` – Starting position within pixels of bytes to be filtered.
    - `width` – Width of a scanline in pixels.
- **filterUp**
  `protected void` **filterUp**`(byte[]` **pixels**`, int` **startPos**`, int` **width**`)`
  - **Description**
    Perform "up" filtering on the given row. Side effect: refills the prior row with current row
  - **Parameters**
    - `pixels` – The array holding the scan lines being built
    - `startPos` – Starting position within pixels of bytes to be filtered.
    - `width` – Width of a scanline in pixels.
- **getCompressionLevel**
  `public int` **getCompressionLevel**`()`
  - **Description**
    Retrieve compression level
  - **Returns** – int in range 0-9
- **getEncodeAlpha**
  `public boolean` **getEncodeAlpha**`()`

  - **Description**
    Retrieve alpha encoding status.
  - **Returns** – boolean false=no, true=yes

- **getFilter**
  `public int` **getFilter**`()`
  - **Description**
    Retrieve filtering scheme
  - **Returns** – int (see constant list)

- **pngEncode**
  `public byte[]` **pngEncode**`()`
  - **Description**
    Creates an array of bytes that is the PNG equivalent of the current image. Alpha encoding is determined by its setting in the constructor.
  - **Returns** – an array of bytes, or null if there was a problem

- **pngEncode**
  `public byte[]` **pngEncode**`(boolean` **encodeAlpha**`)`
  - **Description**
    Creates an array of bytes that is the PNG equivalent of the current image, specifying whether to encode alpha or not.
  - **Parameters**
    * `encodeAlpha` – boolean false=no alpha, true=encode alpha
  - **Returns** – an array of bytes, or null if there was a problem

- **resizeByteArray**
  `protected byte[]` **resizeByteArray**`(byte[]` **array**`, int` **newLength**`)`
  - **Description**
    Increase or decrease the length of a byte array.
  - **Parameters**
    * `array` – The original array.
    * `newLength` – The length you wish the new array to have.
  - **Returns** – Array of newly desired length. If shorter than the original, the trailing elements are truncated.

- **setCompressionLevel**
  `public void` **setCompressionLevel**`(int` **level**`)`
  - **Description**
    Set the compression level to use
  - **Parameters**

* `level` – 0 through 9

- **setEncodeAlpha**
  `public void` **setEncodeAlpha**`(boolean` **encodeAlpha**`)`
  - **Description**
    Set the alpha encoding on or off.
  - **Parameters**
    * `encodeAlpha` – false=no, true=yes

- **setFilter**
  `public void` **setFilter**`(int` **whichFilter**`)`
  - **Description**
    Set the filter to use
  - **Parameters**
    * `whichFilter` – from constant list

- **setImage**
  `public void` **setImage**`(java.awt.Image` **image**`)`
  - **Description**
    Set the image to be encoded
  - **Parameters**
    * `image` – A Java Image object which uses the DirectColorModel
  - **See also**
    * `java.awt.Image`
    * `java.awt.image.DirectColorModel`

- **writeByte**
  `protected int` **writeByte**`(int` **b**`, int` **offset**`)`
  - **Description**
    Write a single byte into the pngBytes array at a given position.
  - **Parameters**
    * `n` – The integer to be written into pngBytes.
    * `offset` – The starting point to write to.
  - **Returns** – The next place to be written to in the pngBytes array.

- **writeBytes**
  `protected int` **writeBytes**`(byte[]` **data**`, int` **offset**`)`
  - **Description**
    Write an array of bytes into the pngBytes array. Note: This routine has the side effect of updating maxPos, the largest element written in the array. The array is resized by 1000 bytes or the length of the data to be written, whichever is larger.

- **Parameters**
  - `data` – The data to be written into pngBytes.
  - `offset` – The starting point to write to.
- **Returns** – The next place to be written to in the pngBytes array.

- **writeBytes**
  `protected int` **writeBytes**`(byte[] data, int nBytes, int offset)`
  - **Description**
    Write an array of bytes into the pngBytes array, specifying number of bytes to write. Note: This routine has the side effect of updating maxPos, the largest element written in the array. The array is resized by 1000 bytes or the length of the data to be written, whichever is larger.
  - **Parameters**
    - `data` – The data to be written into pngBytes.
    - `nBytes` – The number of bytes to be written.
    - `offset` – The starting point to write to.
  - **Returns** – The next place to be written to in the pngBytes array.

- **writeEnd**
  `protected void` **writeEnd**`()`
  - **Description**
    Write a PNG "IEND" chunk into the pngBytes array.

- **writeHeader**
  `protected void` **writeHeader**`()`
  - **Description**
    Write a PNG "IHDR" chunk into the pngBytes array.

- **writeImageData**
  `protected boolean` **writeImageData**`()`
  - **Description**
    Write the image data into the pngBytes array. This will write one or more PNG "IDAT" chunks. In order to conserve memory, this method grabs as many rows as will fit into 32K bytes, or the whole image; whichever is less.
  - **Returns** – true if no errors; false if error grabbing pixels

- **writeInt2**
  `protected int` **writeInt2**`(int n, int offset)`
  - **Description**
    Write a two-byte integer into the pngBytes array at a given position.
  - **Parameters**
    - `n` – The integer to be written into pngBytes.

* `offset` – The starting point to write to.

– **Returns** – The next place to be written to in the pngBytes array.

- **writeInt4**
  `protected int` **writeInt4**`(int n, int offset)`

  – **Description**
  Write a four-byte integer into the pngBytes array at a given position.

  – **Parameters**

  * `n` – The integer to be written into pngBytes.
  * `offset` – The starting point to write to.

  – **Returns** – The next place to be written to in the pngBytes array.

- **writeString**
  `protected int` **writeString**`(java.lang.String s, int offset)`

  – **Description**
  Write a string into the pngBytes array at a given position. This uses the getBytes method, so the encoding used will be its default.

  – **Parameters**

  * `n` – The integer to be written into pngBytes.
  * `offset` – The starting point to write to.

  – **Returns** – The next place to be written to in the pngBytes array.

  – **See also**

  * `java.lang.String.getBytes()`

2.2 Class PngEncoderB

2.2.1 Declaration

public class PngEncoderB
extends com.keypoint.PngEncoder (in 2.1, page 18)

2.2.2 Field summary

image
tType
wRaster

2.2.3 Constructor summary

PngEncoderB() Class constructor
PngEncoderB(BufferedImage) Class constructor specifying BufferedImage to encode, with no alpha channel encoding.

PngEncoderB(BufferedImage, boolean) Class constructor specifying BufferedImage to encode, and whether to encode alpha.
PngEncoderB(BufferedImage, boolean, int) Class constructor specifying BufferedImage to encode, whether to encode alpha, and filter to use.
PngEncoderB(BufferedImage, boolean, int, int) Class constructor specifying BufferedImage source to encode, whether to encode alpha, filter to use, and compression level

2.2.4 Method summary

establishStorageInfo() Get and set variables that determine how picture is stored.
pngEncode() Creates an array of bytes that is the PNG equivalent of the current image.
pngEncode(boolean) Creates an array of bytes that is the PNG equivalent of the current image, specifying whether to encode alpha or not.
setImage(BufferedImage) Set the BufferedImage to be encoded
writeHeader() Write a PNG "IHDR" chunk into the pngBytes array.
writeImageData() Write the image data into the pngBytes array.
writePalette(IndexColorModel)

2.2.5 Fields

- protected java.awt.image.BufferedImage **image**
- protected java.awt.image.WritableRaster **wRaster**
- protected int **tType**

2.2.6 Constructors

- **PngEncoderB**
  `public` **PngEncoderB**`()`
  - **Description**
    Class constructor
- **PngEncoderB**
  `public` **PngEncoderB**`(java.awt.image.BufferedImage` **image**`)`
  - **Description**
    Class constructor specifying BufferedImage to encode, with no alpha channel encoding.
  - **Parameters**
    * `image` – A Java BufferedImage object
- **PngEncoderB**
  `public` **PngEncoderB**`(java.awt.image.BufferedImage` **image**`, boolean` **encodeAlpha**`)`

- **Description**
  Class constructor specifying BufferedImage to encode, and whether to encode alpha.
- **Parameters**
  * `image` – A Java BufferedImage object
  * `encodeAlpha` – Encode the alpha channel? false=no; true=yes

- **PngEncoderB**
  `public` **PngEncoderB**`(java.awt.image.BufferedImage` **image**`, boolean` **encodeAlpha**`, int` **whichFilter**`)`
  - **Description**
    Class constructor specifying BufferedImage to encode, whether to encode alpha, and filter to use.
  - **Parameters**
    * `image` – A Java BufferedImage object
    * `encodeAlpha` – Encode the alpha channel? false=no; true=yes
    * `whichFilter` – 0=none, 1=sub, 2=up

- **PngEncoderB**
  `public` **PngEncoderB**`(java.awt.image.BufferedImage` **image**`, boolean` **encodeAlpha**`, int` **whichFilter**`, int` **compLevel**`)`
  - **Description**
    Class constructor specifying BufferedImage source to encode, whether to encode alpha, filter to use, and compression level
  - **Parameters**
    * `image` – A Java BufferedImage object
    * `encodeAlpha` – Encode the alpha channel? false=no; true=yes
    * `whichFilter` – 0=none, 1=sub, 2=up
    * `compLevel` – 0..9

2.2.7 Methods

- **establishStorageInfo**
  `protected boolean` **establishStorageInfo**`()`
  - **Description**
    Get and set variables that determine how picture is stored. Retrieves the writable raster of the buffered image, as well its transfer type. Sets number of output bytes per pixel, and, if only eight-bit bytes, turns off alpha encoding.
  - **Returns** – true if 1-byte or 4-byte data, false otherwise

- **pngEncode**
  `public byte[]` **pngEncode**`()`

- **Description**
  Creates an array of bytes that is the PNG equivalent of the current image. Alpha encoding is determined by its setting in the constructor.
- **Returns** – an array of bytes, or null if there was a problem

- **pngEncode**
  `public byte[] pngEncode(boolean encodeAlpha)`
  - **Description**
    Creates an array of bytes that is the PNG equivalent of the current image, specifying whether to encode alpha or not.
  - **Parameters**
    - `encodeAlpha` – boolean false=no alpha, true=encode alpha
  - **Returns** – an array of bytes, or null if there was a problem

- **setImage**
  `public void setImage(java.awt.image.BufferedImage image)`
  - **Description**
    Set the BufferedImage to be encoded
  - **Parameters**
    - `BufferedImage` – A Java BufferedImage object

- **writeHeader**
  `protected void writeHeader()`
  - **Description**
    Write a PNG "IHDR" chunk into the pngBytes array.

- **writeImageData**
  `protected boolean writeImageData()`
  - **Description**
    Write the image data into the pngBytes array. This will write one or more PNG "IDAT" chunks. In order to conserve memory, this method grabs as many rows as will fit into 32K bytes, or the whole image; whichever is less.
  - **Returns** – true if no errors; false if error grabbing pixels

- **writePalette**
  `protected void writePalette(java.awt.image.IndexColorModel icm)`

2.2.8 Members inherited from class PngEncoder

`com.keypoint.PngEncoder` (in 2.1, page 18)

bytePos, bytesPerPixel, compressionLevel, crc, crcValue, dataPos, ENCODE_ALPHA, encodeAlpha, endPos, filter, FILTER_LAST, FILTER_NONE, FILTER_SUB, FILTER_UP, filterSub, filterUp, getCompressionLevel, getEncodeAlpha, getFilter, hdrPos, height, image, leftBytes, maxPos, NO_ALPHA, pngBytes, pngEncode, pngEncode, priorRow, resizeByteArray, setCompressionLevel, setEncodeAlpha, setFilter, setImage, width, writeByte, writeBytes, writeBytes, writeEnd, writeHeader, writeImageData, writeInt2, writeInt4, writeString