HTML Parser API

by David McNicol

Package
cvu.html

cvu.html
Class AttributeList

```
java.lang.Object
   |
   +-cvu.html.AttributeList
```

public class **AttributeList**
extends java.lang.Object

This class represents the attribute list of an tag.
Author:
David McNicol
See Also:
`TagToken`

Constructors

AttributeList

```
public AttributeList()
```

Methods

size

```
public int size()
```

Returns the number of attributes currently defined.

get

```
public java.lang.String get(java.lang.String name)
```

Returns the value of the attribute with the specified name.

Parameters:
`name` - the name of the attribute.

set

```
public void set(java.lang.String name,
         java.lang.String value)
```

Sets the attribute with the specified name to the specified value. If the attribute already has a value it will be overwritten.

Parameters:
`name` - the name of the attribute.
`value` - the new value of the attribute.

(continued on next page)

(continued from last page)

exists

```
public boolean exists(java.lang.String name)
```

Returns true if the specified attribute name exists within the list.

Parameters:
`the` - name of the attribute to check.

---

unset

```
public void unset(java.lang.String name)
```

Removes the specified attribute from the list.

Parameters:
`name` - the name of the attribue to remove.

---

names

```
public java.util.Enumeration names()
```

Returns an enumeration of defined attributes.

---

getQuoted

```
public java.lang.String getQuoted(java.lang.String name)
```

Returns an attribute with all double quote characters escaped with a backslash.

Parameters:
`name` - the name of the attribute.

---

toString

```
public java.lang.String toString(java.lang.String name)
```

Returns a string version of the attribute and its value.

Parameters:
`name` - the name of the attribute.

---

toString

```
public java.lang.String toString()
```

Returns a string version of the attribute list.

cvu.html
Class HTMLNode

```
java.lang.Object
   |
   +-cvu.html.HTMLNode
```

public class **HTMLNode**
extends java.lang.Object

This class represents a single node within an HTML tree. Each node has a name, zero or more attributes and possibly some content. Nodes can appear within the content of other nodes.

End tags do not appear since they only indicate 'end-of-content'. To prevent the system searching for the end of standalone tags, a dynamic list has been implemented. When the HTMLNode class is resolved a setup method is called adding a set of default standalone tags to the list. Standalone tags can then be added and removed dynamically using static method calls.

The list is the only way the internal code can tell whether a tag is standalone. If a problem occurs the tree structure would still be sound, but it would not be accurate, so while the form of the HTML would be conserved, searches would not operate correctly.

Author:
David McNicol

See Also:
`HTMLTree`

Constructors

HTMLNode

```
public HTMLNode(TagToken tag,
                HTMLNode parent,
                java.util.Enumeration src)
```

Constructs a new HTMLNode.

Parameters:
`tag` - the TagToken representing the start of this node.
`standalone` - true if the tag does not have any content.
`src` - enumeration of tag tokens.

HTMLNode

```
public HTMLNode(java.lang.String name)
```

Constructs a new, detached HTMLNode with the specified name.

Parameters:
`name` - the name of the new node.

Methods

getName

```
public java.lang.String getName()
```

Returns the name of this node.

getParent

`public HTMLNode getParent()`

Returns the node's parent node.

getChildren

`public java.util.Enumeration getChildren()`

Returns the node's children.

isHidden

`public boolean isHidden()`

Returns true if the node is currently hidden.

hide

`public void hide()`

Hides the node.

unhide

`public void unhide()`

"Unhides" the node.

getAttribute

`public java.lang.String getAttribute(java.lang.String name)`

Returns the value of the attribute with the given name.

Parameters:

`name` - the name of the attribute.

getAttributes

`public java.util.Enumeration getAttributes()`

Returns an enumeration of attributes defined in this node.

getQuotedAttribute

`public java.lang.String getQuotedAttribute(java.lang.String name)`

Returns an attribute with all double quote characters escaped with a backslash.

Parameters:

`name` - the name of the attribute.

getAttributeToString

`public java.lang.String getAttributeToString(java.lang.String name)`

(continued from last page)

Returns a string version of the attribute and its value.

Parameters:
`name` - the name of the attribute.

toString

```
public java.lang.String toString()
```

Returns a string version of the HTMLNode. If the node is currently hidden then return an empty string.

setParent

```
public void setParent(HTMLNode parent)
```

Sets the node's parent to the specified HTMLNode.

Parameters:
`parent` - the new parent.

isAttribute

```
public boolean isAttribute(java.lang.String name)
```

Returns true if an attribute with the given name exists.

Parameters:
`name` - the name of the attribute.

addAttribute

```
public void addAttribute(java.lang.String name,
          java.lang.String value)
```

Adds a new attribute to the node's attribute list with the specified value. If the attribute already exists the old value is overwritten.

Parameters:
`name` - the name of the attribute.
`value` - the value of the attribute.

addChild

```
public void addChild(java.lang.Object child)
```

Adds an object to the end of this node's content

Parameters:
`child` - the node to be added.

removeChild

```
public void removeChild(HTMLNode child)
```

Removes the specified HTMLNode from the current node's list of children.

Parameters:
`child` - the node to be removed.

addChildBefore

```
public void addChildBefore(java.lang.Object child,
          HTMLNode before)
```

Adds an object to this node's content before the specified child node.

Parameters:

`child` - the object to be added.
`before` - the node before which the child will be placed.

removeAttribute

```
public void removeAttribute(java.lang.String name)
```

Removes an attribute with the specified name from the attribute list.

Parameters:

`name` - the name of the attribute to remove.

nextSibling

```
public HTMLNode nextSibling()
```

Returns the node after this one in the parent's list of children.

previousSibling

```
public HTMLNode previousSibling()
```

Returns the node before this one in the parent's list of children.

firstChild

```
public HTMLNode firstChild()
```

Returns the first child of this node.

nextChild

```
public HTMLNode nextChild(HTMLNode child)
```

Returns the HTMLNode after the specified one in this nodes content.

Parameters:

`child` - the HTMLNode before the one we want.

previousChild

```
public HTMLNode previousChild(HTMLNode child)
```

Returns the HTMLNode before the specified one in this nodes content.

Parameters:

`child` - the HTMLNode after the one we want.

(continued from last page)

printDefaultStandaloneList

```
public static void printDefaultStandaloneList()
```

Utility method which people can use to find out exactly which nodes are in the default standalone list. The default list is printed to the standard output.

---

addStandalone

```
public static void addStandalone(java.lang.String name)
```

Adds the specified string to the standalone list.

Parameters:

`name` - the new standalone name.

---

removeStandalone

```
public static void removeStandalone(java.lang.String name)
```

Removes the specified string from the standalone list.

Parameters:

`name` - the standalone name to remove.

---

isStandalone

```
public static boolean isStandalone(java.lang.String name)
```

Checks the standalone list to see if it mentions the specified tag name and returns true if so.

Parameters:

`name` - the tag name to check against the list.

cvu.html
Class HTMLTokenizer

```
java.lang.Object
   |
   +-cvu.html.HTMLTokenizer
```

public class **HTMLTokenizer**
extends java.lang.Object

This class tokenizes a stream of HTML tags and blocks of text. After the stream has been tokenized an Enumeration of tokens can be accessed.

Author:
David McNicol

See Also:
`TagToken`, `TextToken`, `java.util.Enumeration`

Constructors

HTMLTokenizer

```
public HTMLTokenizer(java.lang.String file)
```

Constructs a new HTMLTokenizer using the given filename to create the input stream.

Parameters:
`file` - the name of the file to open.

Methods

getTokens

```
public java.util.Enumeration getTokens()
```

Returns an enumeration of the tokens which have been created by the HTMLTokenizer.

getTokenVector

```
public java.util.Vector getTokenVector()
```

Returns the vector in which the tokens are stored.

cvu.html
Class HTMLTree

```
java.lang.Object
   |
   +-cvu.html.HTMLTree
```

public class **HTMLTree**
extends java.lang.Object

This class stores an HTML file in tree format. It can be constructed from an HTMLTokenizer or a file name, in which case it will create its own tokenizer.

Once the HTML file has been parsed a number of search operations can be performed. The nature of the searches are described below, but some of their uses are highlighted here:

- Subtree - Finding all of the FORM elements within a BODY element.
- Sibling - Finding all the LI elements within the same UL element.
- All - Finding every occurence of the A element.

There is also a context search, which performs a subtree search on the specified element's parent. This can be thought of as a combination between as sibling search and a subtree search.

Author:
David McNicol

See Also:
`HTMLTokenizer`

Constructors

HTMLTree

```
public HTMLTree(java.util.Enumeration e)
```

Constructs a new HTMLTree using the tokens from the specified Enumeration.

HTMLTree

```
public HTMLTree(HTMLTokenizer ht)
```

Constructs a new HTMLTree using the tokens from the specified HTMLTokenizer.

Parameters:
`ht` - the source of the HTML tokens.

HTMLTree

```
public HTMLTree(java.lang.String filename)
```

Constructs a new HTMLTree from the specified HTML file.

Parameters:
`filename` - the name of the HTML file.

Methods

(continued from last page)

findInSubtree

```
public HTMLNode findInSubtree(java.lang.String name,
        HTMLNode tree)
```

Finds the first element with the specified name in the specified subtree.

Parameters:

`name` - the name of the element to search for.
`tree` - the subtree to search.

---

findNextInSubtree

```
public HTMLNode findNextInSubtree(HTMLNode tree,
        HTMLNode prev)
```

Finds the next element after the specified one in the subtree. If the previous element is not in the subtree then nothing will be found.

Parameters:

`tree` - the subtree to search.
`prev` - a previously found element.

---

findInAll

```
public HTMLNode findInAll(java.lang.String name)
```

Finds the first element with the specified name in the entire tree.

Parameters:

`name` - the name of the element to search for.

---

findNextInAll

```
public HTMLNode findNextInAll(HTMLNode prev)
```

Finds the next element with the same name as the one specified in the entire tree.

Parameters:

`prev` - the previously found element.

---

findInContext

```
public HTMLNode findInContext(java.lang.String name,
        HTMLNode el)
```

Find the first element with the specified name in the specified element's context (that is, the elements parent's subtree).

Parameters:

`name` - the name of the element to search for.
`el` - the element whose context is to be searched.

---

findNextInContext

```
public HTMLNode findNextInContext(HTMLNode el,
        HTMLNode prev)
```

Find the next element with the same name as the specified one in the first element's context (that is, the first elements parent's subtree). If the previous element is not in the subtree then nothing will be found.

(continued from last page)

Parameters:

`el` - the element whose context is to be searched.
`the` - previously found element.

findSibling

`public HTMLNode findSibling(HTMLNode el)`

Finds the next element with the same name as the specified one amongst that elements siblings (that is, the elements parent's children).

Parameters:

`el` - the element whose siblings are to be searched.

toString

`public java.lang.String toString()`

Prints a string representation of the HTMLTree.

cvu.html
Class TagToken

```
java.lang.Object
   |
   +-cvu.html.TagToken
```

public class **TagToken**
extends java.lang.Object

This represents a single HTML tag. Each TagToken has a name and a list of attributes and values.
Author:
David McNicol
See Also:
`HTMLTokenizer`

Fields

ESCAPE

```
public static final char ESCAPE
```

Identifies the escape character.
Constant value: **92**

QUOTE

```
public static final char QUOTE
```

Identifies the quotation character.
Constant value: **34**

Constructors

TagToken

```
public TagToken(java.lang.String line)
```

Constructs a new TagToken converting the specified string into a token name and a list of attributes with values.

Parameters:
`line` - the raw data.

Methods

getName

```
public java.lang.String getName()
```

Returns the name of the TagToken.

getAttributes

```
public AttributeList getAttributes()
```

(continued from last page)

Returns the attribute list of the TagToken.

isEndTag

```
public boolean isEndTag()
```

Indicates whether this token is an end tag.

isAttribute

```
public boolean isAttribute(java.lang.String name)
```

Returns true if the given attribute exists.

Parameters:
`name` - the name of the attribute.

getAttribute

```
public java.lang.String getAttribute(java.lang.String name)
```

Returns the value of the specified attribute or null if the attribute does not exist.

Parameters:
`name` - the name of the attribute.

getQuotedAttribute

```
public java.lang.String getQuotedAttribute(java.lang.String name)
```

Returns an attribute with all double quote characters escaped with a backslash.

Parameters:
`name` - the name of the attribute.

getAttributeToString

```
public java.lang.String getAttributeToString(java.lang.String name)
```

Returns a string version of the attribute and its value.

Parameters:
`name` - the name of the attribute.

toString

```
public java.lang.String toString()
```

Returns a string version of the TagToken.

cvu.html
Class TextToken

```
java.lang.Object
   |
   +-cvu.html.TextToken
```

public class **TextToken**
extends java.lang.Object

This represents a block of text.
Author:
David McNicol
See Also:
`HTMLTokenizer`

Constructors

TextToken

```
public TextToken()
```

Constructs a new token.

Methods

setText

```
public void setText(java.lang.String newText)
```

Sets the content of the Token.

Parameters:
`newText` - the new content of the Token.

setText

```
public void setText(java.lang.StringBuffer newText)
```

Sets the content of the Token.

Parameters:
`newText` - the new content of the Token.

appendText

```
public void appendText(java.lang.String more)
```

Appends some content to the token.

Parameters:
`more` - the new content to add.

(continued from last page)

getText

```
public java.lang.String getText()
```

Returns the contents of the token.

toString

```
public java.lang.String toString()
```

Returns a string version of the TextToken.

Index

U