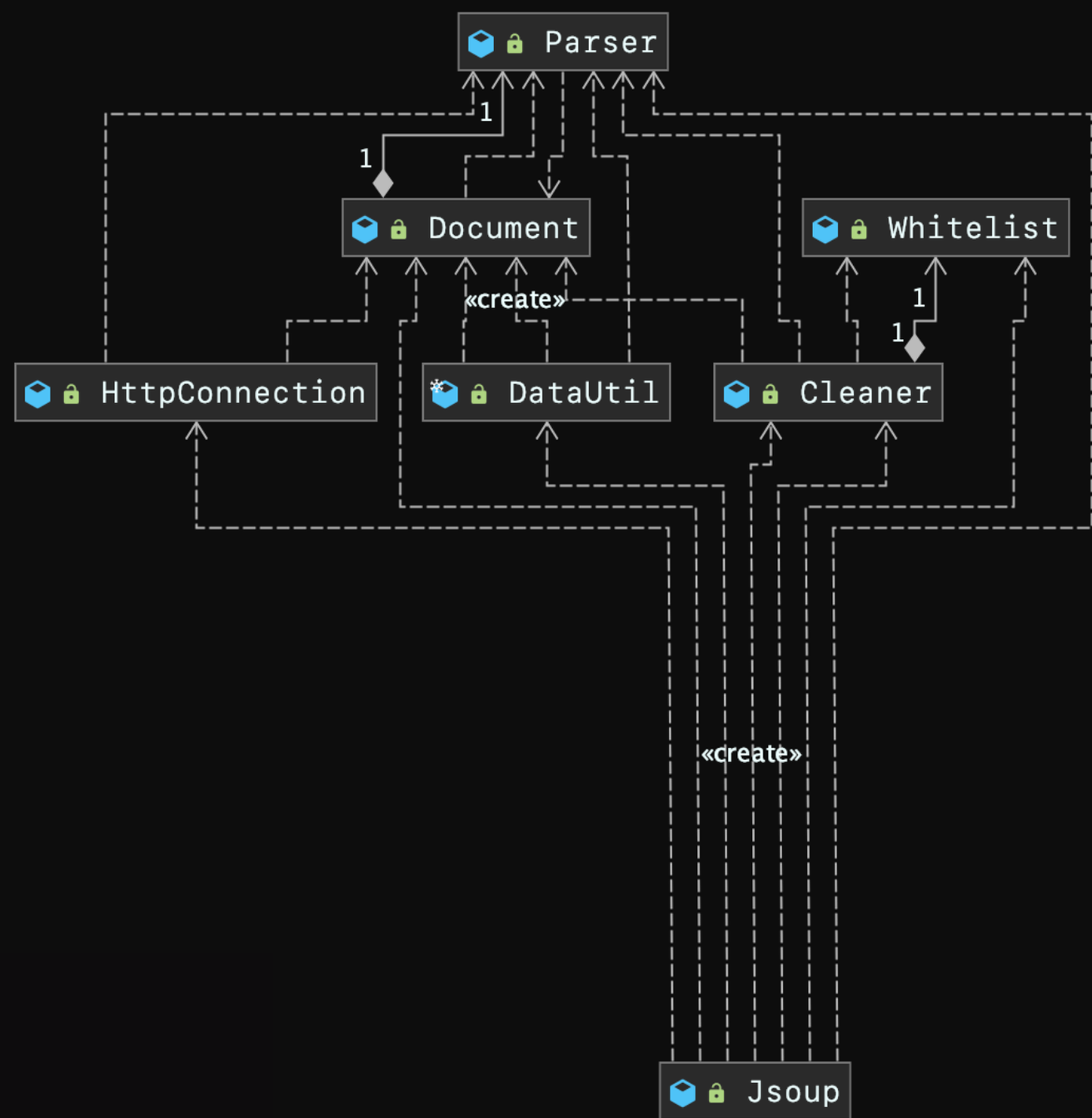
jsoup+

https://github.com/paywteam/jsoup-plus

Design patterns in jsoup

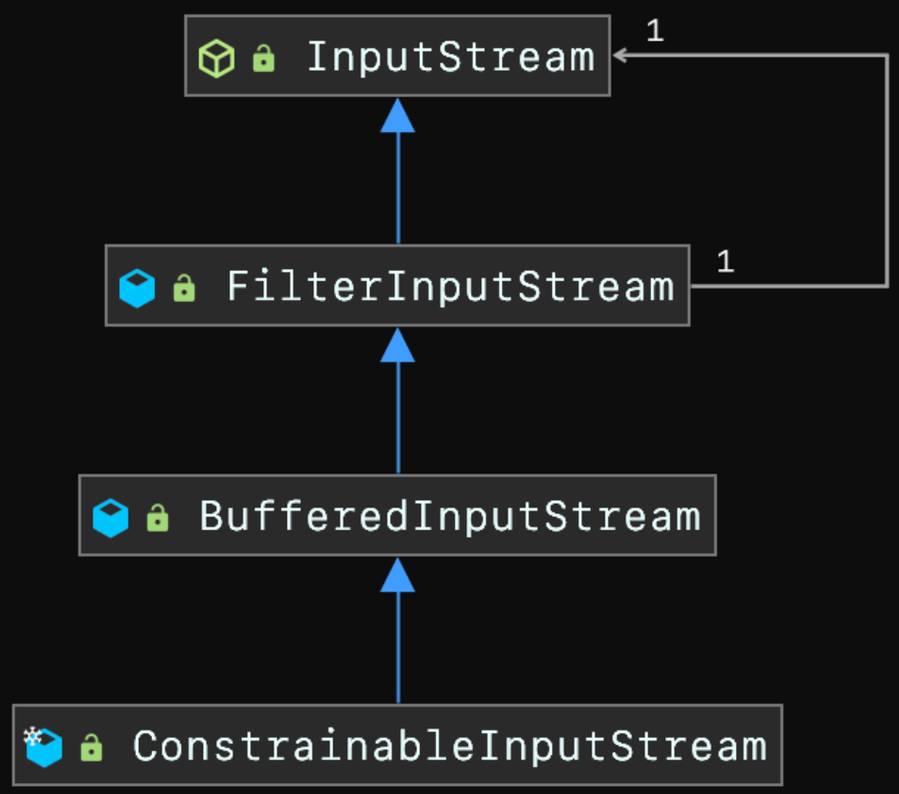
org.jsoup.Jsoup

Facade



org.jsoup.internal.ConstrainableInputStream

Decorator

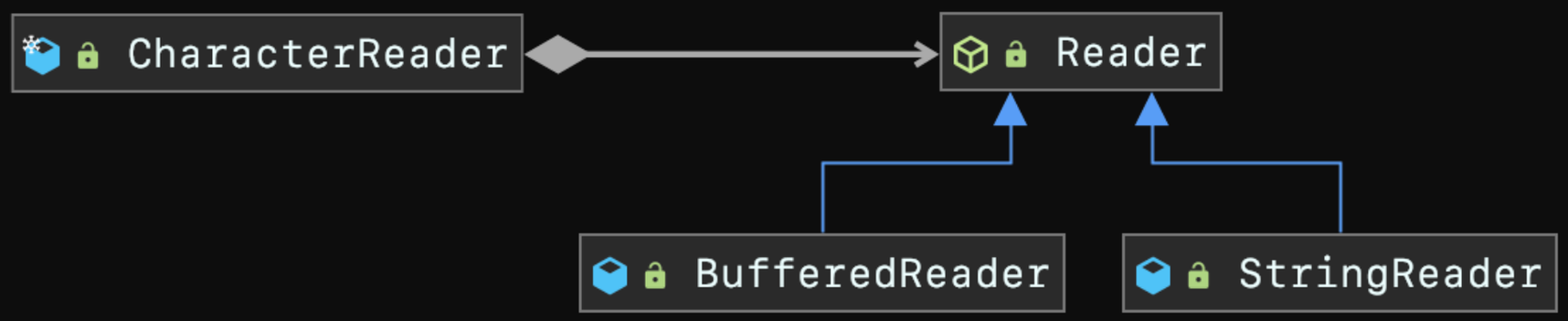


Role	Class
Component	InputStream
ConcreteDecorator	ConstrainableInputStream

```
private ConstrainableInputStream(InputStream in, ...) {
  super(in, bufferSize);
  ...
```

org.jsoup.parser.CharacterReader

Strategy



Role	Class
Context	CharacterReader
Strategy	Reader
ConcreteStrategy	StringReader, BufferedReader

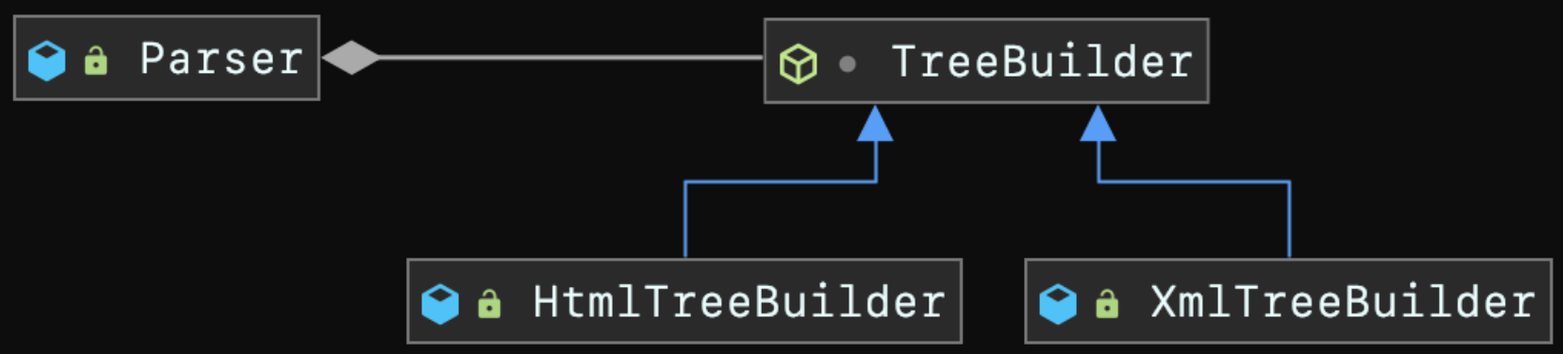
org.jsoup.parser.CharacterReader

Strategy

```
public final class CharacterReader {
  ...
  private final Reader reader;
  ...
  public CharacterReader(Reader input, int sz) {
    Validate.notNull(input);
    Validate.isTrue(input.markSupported());
    reader = input;
...
    final long skipped = reader.skip(pos);
    reader.mark(maxBufferLen);
    final int read = reader.read(charBuf);
    reader.reset();
```

org.jsoup.parser.Parser

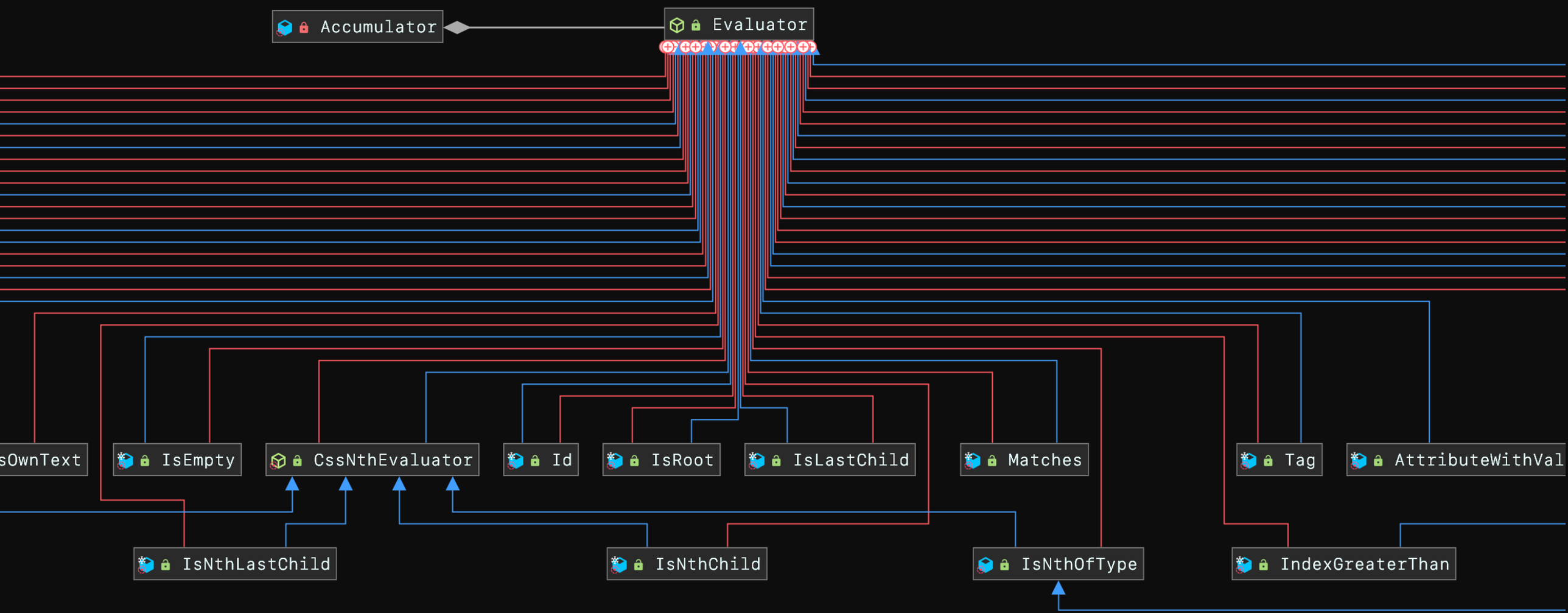
Strategy



Role	Class
Context	Parser
Strategy	TreeBuilder
ConcreteStrategy	HtmlTreeBuilder, XmlTreeBuilder

org.jsoup.select.Collector.Accumulator

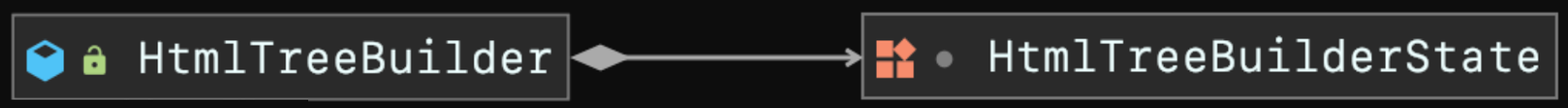
Strategy



Role	Class
Context	Accumulator
Strategy	Evaluator
ConcreteStrategy	Refer to the diagram

org.jsoup.parser.HtmlTreeBuilder

State



Role	Class
Context	HtmlTreeBuilder
State	HtmlTreeBuilderState
ConcreteState	Many nested states

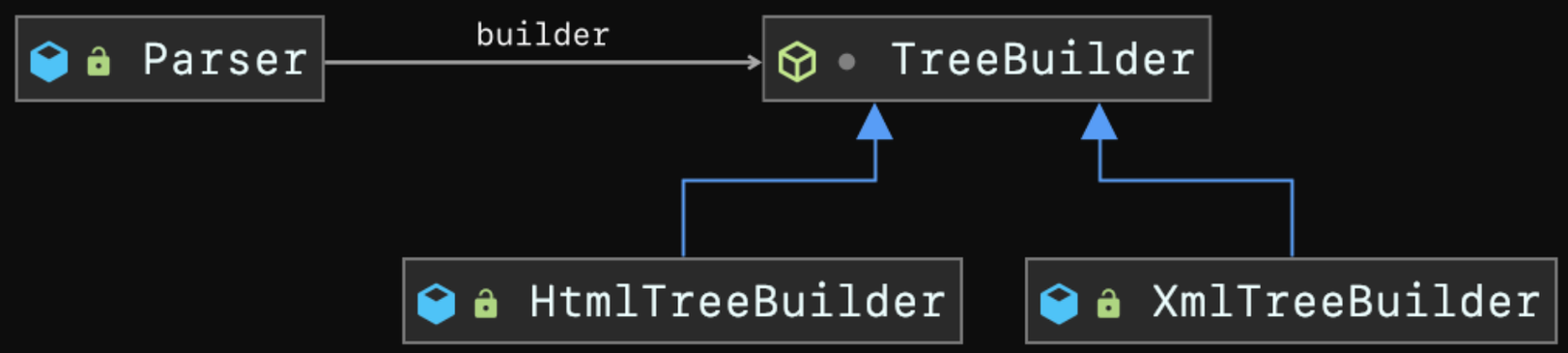
org.jsoup.parser.HtmlTreeBuilder

State

```
private HtmlTreeBuilderState state; // the current state
...
protected boolean process(Token token) {
  currentToken = token;
  return this.state.process(token, this);
}
...
void transition(HtmlTreeBuilderState state) {
  this.state = state;
}
```

org.jsoup.parser.HtmlTreeBuilder

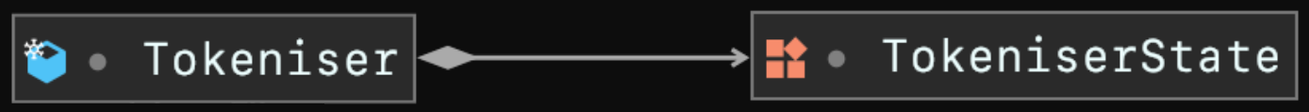
State



Role	Class
Director	Parser
Builder	TreeBuilder
Concrete Builder	HtmlTreeBuilder, XmlTreeBuilder
Product	Document

org.jsoup.parser.Tokeniser

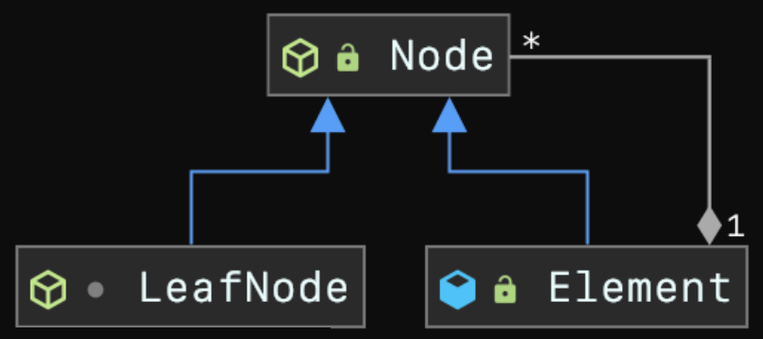
State



Role	Class
Context	Tokeniser
State	TokeniserState
ConcreteState	Many nested states

org.jsoup.nodes.Node

Composite



Role	Class
Component	Node
Composite	Element
Leaf	LeafNode

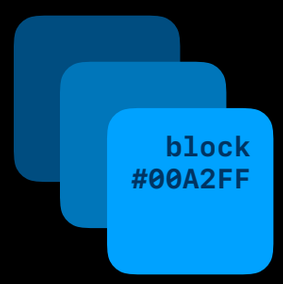
org.jsoup.nodes.**Node**

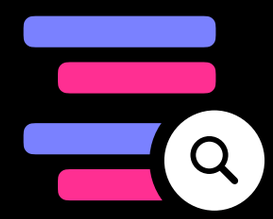
Composite

```java
// Element.java
public class Element extends Node {
    ...
    List<Node> childNodes;
```

```java
// LeafNode.java
abstract class LeafNode extends Node {
```

New Features for jsoup





SQLish

Load nested iframe documents

```
<iframe src="somewhere"></iframe>
```

```html
<body>
  <div>
    <h1>foo</h1>
    <p>bar</p>
    <iframe src="somewhere" />
  </div>
</body>
```

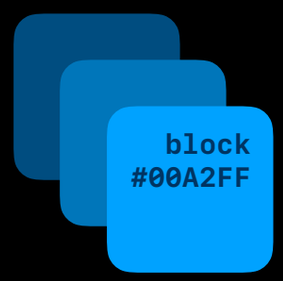
somewhere

```html
<body>
  <section>
    <aside>welcome</aside>
  </section>
</body>
```

```
<body>
  <div>
    <h1>foo</h1>
    <p>bar</p>
    <iframe src="somewhere">
      <body>
        <section>
          <aside>welcome</aside>
        </section>
      </body>
    </iframe>
  </div>
</body>
```

somewhere

```
<body>
  <section>
    <aside>welcome</aside>
  </section>
</body>
```



Get elements by inline style

```
<div style="display: block;">No-named DOM</div>
```

```html
<div style="display: block;">No-named DOM</div>
```

```
doc.select("div[style*=\"display: block\"]")
```

✗ display : block

✓ display: block

```
doc.select("div[style*=\"display: block\"]")
```

✗ display:block

```
public class Element {
  ...
  private ArrayList<Style> styles;
  ...
```

Key	Value
display	flex
align-items	center
justify-content	center
color	crimson
text-align	right

```
doc.getElementsByInlineStyle("display", "block")
```

Preserve text content line breaks using HTML block style

```
<div>
  <h1>My First Program</h1>
  <p><span>Hello</span> World</p>
</div>
```

```html
<div>
  <h1>My First Program</h1>
  <p><span>Hello</span> World</p>
</div>
```

```
My First Program Hello World // Element.text()
My First ProgramHello World // Element.wholeText()
```

```
Element.formattedText()
```

My First Program

Hello World

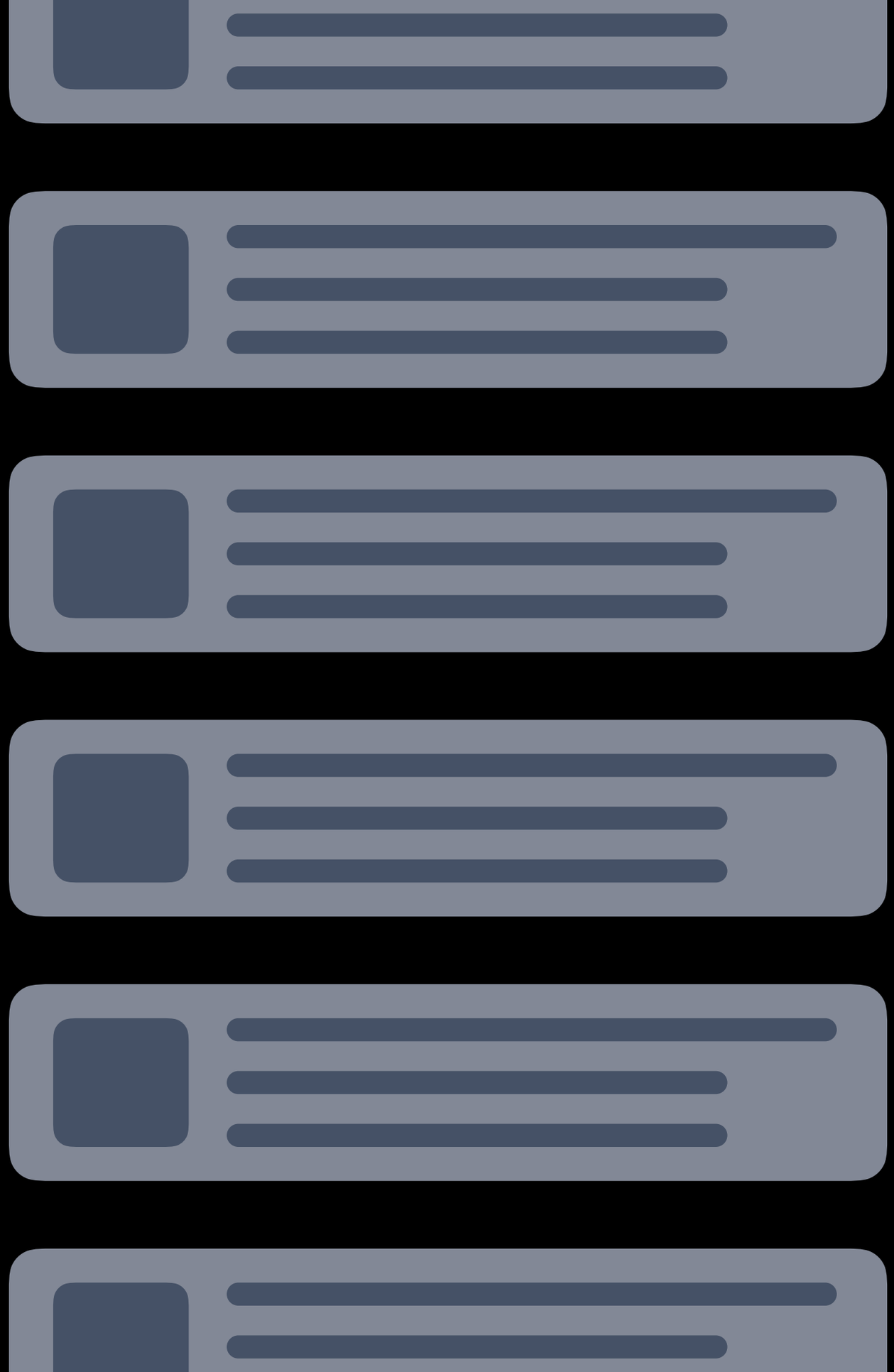
A + Visitor

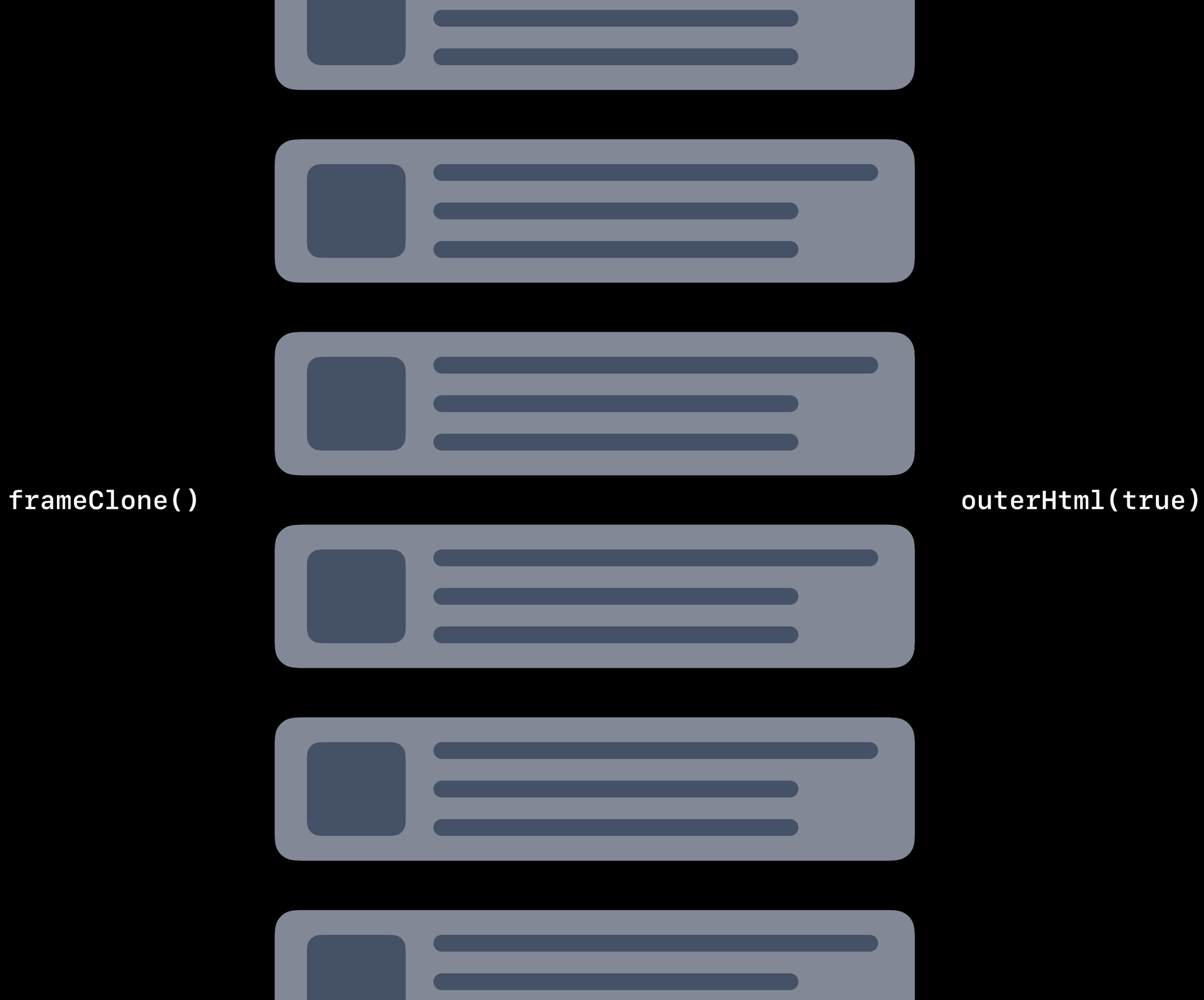
```java
public void visit(Element element) {
  ...
}

public void visit(TextNode textNode) {
  ...
}
```

```java
public void accept(FormattedTextVisitor visitor) {
  visitor.visit(this);
}
```

```java
FormattedTextVisitor visitor = new FormattedTextVisitor();
node.accept(visitor);
```

Inspection




frameClone()
outerHtml(true)

```html
<!-- clone() -->
<div id="wrapper">
  <h1 class="title typography-big" data-title-id="123">
    This is title
  </h1>
</div>


<!-- frameClone() -->
<div id="">
  <h1 class="" data-title-id=""></h1>
</div>


<!-- frameClone(new String[]{"id", "class"}) -->
<div id="wrapper">
  <h1 class="title typography-big" data-title-id=""></h1>
</div>
```

frameClone()

```
<div>
  <h1>foo</h1>
  <p>bar</p>
</div>
```

↓

```
<div><h1>foo</h1><p>bar</p></div>
```

outerHtml(true)

frameClone() + minifying
+ Levenshtein Distance

```
Element.inspect();
```

```
## This kind of element has been repeated 4 times ##

Query Recommendation: div.car

<div class="car">
 Tesla
</div>

## This kind of element has been repeated 3 times ##

Query Recommendation: li

<li>item 1</li>
```

SQLish

SQL-like utility for handling data from elements

ownText() →

```
<span class="price">
  <span class="low">지마켓</span>
  1112640
</span>

<span class="price">
  <span class="low">11번가</span>
  890910
</span>

<span class="price">
  <span class="low">옥션</span>
  993424
</span>

<span class="price">
  <span class="low">쿠팡</span>
  873420
</span>

<span class="price">
  <span class="low">위메프</span>
  943202
</span>
```

```
SQLish sql = new SQLish(
  doc.select(".price"),
  new TextExtractor.OwnText()
);

sql.lteByText(1000000).orderByTextAsc().limit(3).exec();
```

Chaining methods

```java
SQLish sql = new SQLish(
  doc.select(".price"),
  new TextExtractor.OwnText()
);

sql.lteByText(1000000).orderByTextAsc().limit(3).exec();
```

Facade

```java
private ArrayList<SQLCommand> commands; // sql command list
...
public Elements exec() {
    Elements copyElements = this.elements.clone();

    for (SQLCommand command: this.commands) {
        command.execute(copyElements);
    }

    return copyElements;
}
```

Command

```java
private TextExtractor extractor; // text extractor from element
...
public void setExtractor(TextExtractor extractor) {
    this.extractor = extractor;
}
...
public SQLish orderByTextAsc() {
    this.commands.add(new SQLCommand.OrderByTextAscCommand(this.extractor));
    return this;
}
```

Strategy

```html
<span class="price">
  <span class="low">지마켓</span>
  1112640
</span>

<span class="price">
  <span class="low">11번가</span>
  890910
</span>

<span class="price">
  <span class="low">옥션</span>
  993424
</span>

<span class="price">
  <span class="low">쿠팡</span>
  873420
</span>

<span class="price">
  <span class="low">위메프</span>
  943202
</span>
```

→

```html
<span class="price">
  <span class="low">쿠팡</span>
  873420
</span>

<span class="price">
  <span class="low">11번가</span>
  890910
</span>

<span class="price">
  <span class="low">위메프</span>
  943202
</span>
```

jsoup+

https://github.com/paywteam/jsoup-plus