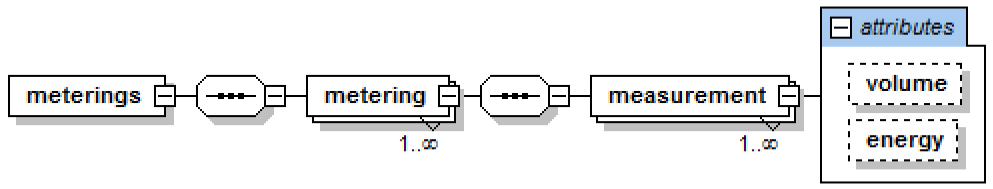
- **Date:** 2014-02-12
- **JVM:** 1.7.0_45

I wanted to know how fast the evaluator was so I set up a test case where I pitted it against the default xpath implementation in java. I started off with a reasonably sized XML of 13.3 MB. The schema of the XML is +- as follows:



For a proper comparison I picked a random value near the end of the XML.

```
public static void main(String...args) throws Exception {
        Document largeDoc = load(new File("largedoc.xml"));
        XPath xPath =  XPathFactory.newInstance().newXPath();
        XPathExpression expression = xPath.compile("metering/measurement[@volume = '171.69040']/@energy");
        evaluate(expression, largeDoc.getDocumentElement(), 10);

        XMLContent content = new XMLContent(largeDoc.getDocumentElement());
        Operation operation = PathAnalyzer.analyze(
                QueryParser.getInstance().parse("metering/measurement[@volume = '171.69040']/@energy"));
        evaluate(operation, content, 10);
}

public static void evaluate(XPathExpression expression, Element largeDoc, int times) throws
XPathExpressionException {
        for (int i = 0; i < times; i++) {
                Date date = new Date();
                String result = expression.evaluate(largeDoc);
                // expecting 1957.50902
                System.out.println("XPATH  [" + (new Date().getTime() - date.getTime()) + "ms]: "
                        + result);
        }
}

public static void evaluate(Operation operation, ComplexContent content, int times) throws
EvaluationException {
        for (int i = 0; i < times; i++) {
                Date date = new Date();
                List<String> result = (List<String>) operation.evaluate(content);
                // expecting 1957.50902
                System.out.println("CUSTOM [" + (new Date().getTime() - date.getTime()) + "ms]: "
                        + result);
        }
}
```

The code is pretty straight forward in that it simply constructs an xpath expression and runs it a number of times, then constructs an operation and runs that a few times. Note that due to a lot of lazy loading and internal optimization I would expect the first run of the operation to be slow but for it to pick up speed fast. It was not optimized for one-off evaluations. The result surprised me:

```
XPATH  [669ms]: 1957.50902
XPATH  [379ms]: 1957.50902
XPATH  [139ms]: 1957.50902
XPATH  [146ms]: 1957.50902
XPATH  [162ms]: 1957.50902
XPATH  [142ms]: 1957.50902
XPATH  [155ms]: 1957.50902
XPATH  [140ms]: 1957.50902
XPATH  [137ms]: 1957.50902
XPATH  [149ms]: 1957.50902
CUSTOM [1344ms]: [1957.50902]
CUSTOM [142ms]: [1957.50902]
CUSTOM [62ms]: [1957.50902]
CUSTOM [45ms]: [1957.50902]
CUSTOM [42ms]: [1957.50902]
CUSTOM [43ms]: [1957.50902]
CUSTOM [67ms]: [1957.50902]
CUSTOM [40ms]: [1957.50902]
CUSTOM [42ms]: [1957.50902]
CUSTOM [42ms]: [1957.50902]
```

As you can see the custom engine starts of quite a bit slower but quickly gains speed and in the end performs the query 3 times faster than the xpath engine.

Also note that the engine always returns a list because of the way the query is constructed (multiple hits could happen) whereas you have to tell the xpath engine what exactly it should return.

I increased the size of the XML to 106.5 MB and ran the test again (the record to find is once again at the end of the XML):

```
XPATH  [3678ms]: 1957.50902
XPATH  [1960ms]: 1957.50902
XPATH  [1041ms]: 1957.50902
XPATH  [1469ms]: 1957.50902
XPATH  [972ms]: 1957.50902
XPATH  [902ms]: 1957.50902
XPATH  [897ms]: 1957.50902
XPATH  [916ms]: 1957.50902
XPATH  [916ms]: 1957.50902
XPATH  [905ms]: 1957.50902
CUSTOM [1716ms]: [1957.50902]
CUSTOM [429ms]: [1957.50902]
CUSTOM [377ms]: [1957.50902]
CUSTOM [345ms]: [1957.50902]
CUSTOM [403ms]: [1957.50902]
CUSTOM [317ms]: [1957.50902]
CUSTOM [446ms]: [1957.50902]
CUSTOM [311ms]: [1957.50902]
CUSTOM [433ms]: [1957.50902]
CUSTOM [311ms]: [1957.50902]
```

This result tells us that the engine remains consistently 3 times faster and also that the initial overhead does not increase as a function of the size whereas for XPath it would seem that it does.