TRPL - Feature #8062

add a SWITCH rule in TRPL

11/20/2023 09:44 AM - Constantin Asofiei

Status:	WIP	**Start date:**	
Priority:	Normal	**Due date:**	
Assignee:	Constantin Asofiei	**% Done:**	0%
Category:		**Estimated time:**	0.00 hour
Target version:			
billable:	No	**vendor_id:**	GCD
Description			

History

#1 - 11/20/2023 09:52 AM - Constantin Asofiei

Greg, I have changes (without having new bytecode being generated) for a structure like this:

```
<switch>...expr

   <case>firstExpr
        <value>secondExpr</value>
        <value>...</value>
        <value></value>
        <value></value>

        /// rules
   </case>

   <default>

   </default>
</switch>
```

The point here is that:

- values for each case can not overlap between them
- there is no 'fall-through' (i.e. the Java break is always assumed)
- on first execution of the switch, the case values are evaluated and assumed to be numbers. This does not assume that the expression is a constant - it can be anything which evaluates to a number. Each value is associated via a map with the case instance to execute.
- once we have this, a map lookup is done to find the switch expression value, and assuming that one is found, the case instance is executed.
- otherwise, the default code is executed (if it exists)

I've created 8062a from trunk rev 14835. The current changes are in rev 14836 - they rely on existing Rule infrastructure, kind of dirty, but more than this will require more in-depth refactoring. Please take a look and let me know what you think.

#2 - 11/21/2023 09:34 AM - Constantin Asofiei

- Project changed from Conversion Tools to TRPL

#3 - 11/24/2023 11:16 AM - Constantin Asofiei

Alexandru, my changes are in 8062a rev 14841 (on top of trunk 14839). Please do a performance test with 7156b. Thanks.

#4 - 11/27/2023 04:43 AM - Alexandru Lungu

Picking them up now and start testing. Expect for a result in the next 2/3 hours.

#5 - 11/27/2023 11:06 AM - Alexandru Lungu

The performance tests on this item are highly volatile. On average of 100 iterations:

- test 1: 8.383s
- test 2: 8.435s
- test 3: 8.835s
- test 4: 8.839s
- test 5: 8.528s
- test 6: 8.858s
- test 7: 8.858s
- test 8: 8.582s

My baseline is ~8.500s.
Weirdly, on some tests (3, 4, 6 and 7), the time was actually increasing in the last iterations. Is there any chance to introduce a leak? Or maybe is just my machine that is going crazy (doing some background work).
If we are to ignore the outsiders (>8.8s), the average is decent representing around 0.3% improvement. However, I can't easily ignore the fact that half of the tests are in fact slower.

#6 - 12/01/2023 05:38 PM - Greg Shah

Code Review Task Branch 8062a Revisions 14853 through 14855

Sorry it took some time to review. Everything looks good.

The only problem I see is in annotations/cleanup.rules:

This original code:

```
      <rule>type == prog.kw_editing and parent.type == prog.editing_block
         <action>copy.remove()</action>
      </rule>
```

does not seem like it can be replaced by this:

```
        <case>
            <value>prog.kw_editing</value>
            <value>prog.editing_block</value>

            <!-- kw_editing under an editing block is useless (easier to remove it
                 here than in the parser) -->
            <action>copy.remove()</action>
         </case>
```