Smarter Balanced QTI 3 Custom Operators Reference

Version: 1.1

Date: September 6, 2019

Updated Date: March 26, 2025

Authors: Paul Grudnitski (Pearson), Jeremy Goodell (Pearson), Alex Dean (Smarter Balanced)

Table of Contents

Change Log

Version	Date	Author	Description
0.1	18-Jan-2019	Paul Grudnitski (Pearson) Jeremy Goodell (Pearson)	Initial Draft
0.2	29-Jan-2019	Paul Grudnitski (Pearson)	Fix QTI3 errors in TI example pg. 57
0.3	22-Feb-2019	Paul Grudnitski (Pearson)	Remove “identifier" attribute from <qti-base-value>
0.4	06-Sep-2019	Jeremy Goodell (Pearson)	Added Hot Text (HT) custom operators.

Guidelines

TBD

Introduction

This document describes mappings between Smarter Balanced's (SBAC's) non-QTI-based custom operators library – originally developed by The American Institutes for Research (AIR) and a QTI3-based custom operators library.

The design of these mappings is informed by a desire to:

1. Make the mappings between AIR's operators and the QTI3 operators easy to understand
2. Minimize transformation of the parameters/operands
3. Minimize alteration of the AIR operator semantics
4. Make the operators truly conformant QTI3

Approach

Unlike the AIR custom operators, a conforming QTI (2.x or 3.0) custom operator has two attributes: "class" and "definition".

```
<qti-custom-operator class="name.of.custom.operator" definition="…a string …">
    …custom text…
</qti-custom-operator>
```

Any content inside a qti-custom-operator is custom, and does not even need to be XML. In fact, it is common to encapsulate the content of a custom-operator in CDATA so as to "hide" the custom operator contents from the XML processor that is processing the item; e.g.,

```
<qti-custom-operator class="name.of.custom.operator" definition="…a string …"> <![CDATA[
    …I am invisible to an XML processor…
]]>
</qti-custom-operator>
```

The design approach for converting the AIR custom operators to QTI custom operators is to:

1. Use the same custom operator class names; i.e., an AIR custom operator class name directly maps to a QTI 3 custom operator class name.
2. Encode all other AIR custom operator attributes into the QTI "definition" attribute as key-value pairs, separating each of the key-value pairs with 3 pipe characters ("|||");

   Example: `definition="key1=value1|||key2=value2"`

Package: qti.sbac.customOperators.CTRL

Class Summary

This package contains six custom operators:

qti.sbac.customOperators.CTRL.COUNTBOOL

qti.sbac.customOperators.CTRL.COUNTDOUBLEINRANGE

qti.sbac.customOperators.CTRL.mapExpression

qti.sbac.customOperators.CTRL.MAXINT

qti.sbac.customOperators.CTRL.MININT

qti.sbac.customOperators.CTRL.stringToFloat

COUNTBOOL

A qti customOperator adapted from The American Institutes for Research open source TDS_ItemScoring/tinyctrlscoringengine/CtrlCountBoolean.java.

Class:	**qti.sbac.customOperators.CTRL.COUNTBOOL**	
Description:	Takes a collection or list of Boolean values and counts the number that are either true or false.	
Return Type:	Integer	
Return Description	The number of values that are specified.	
Attributes	**Type**	**Description**
list	String or Identifier	Either: 1) A list of values consisting of parseable Boolean values or Boolean identifiers prefaced with a "$", or 2) An identifier bound to a variable with basetype Boolean
booleanValue	boolean	true or false. The value to be matched.
QTI 3 Encoding		
`<qti-custom-operator class="qti.sbac.customOperators.CTRL.COUNTBOOL" definition="list=$e1,$f2,$g3,$h4|||booleanValue=True" />`		

COUNTDOUBLEINRANGE

A qti customOperator adapted from The American Institutes for Research open source TDS_ItemScoring/tinyctrlscoringengine/CtrlCountDoubleInRange.java.

Class:	**qti.sbac.customOperators.CTRL.COUNTDOUBLEINRANGE**	
Description:	Takes a collection or comma-separated list of numeric values and counts the number of values within the specified range.	
Return Type:	Integer	
Return Description	The number of doubles in the range.	
Attributes	**Type**	**Description**
list	String or Identifier	Either: 1) a list of values consisting of parseable numeric values or numeric variable identifiers prefaced with a "$", or 2) an identifier bound to a variable with basetype float or integer
min	string	Minimum value included in the range.
max	string	Maximum value included in the range.
QTI 3 Encoding		
`<qti-custom-operator class="qti.sbac.customOperators.CTRL.COUNTDOUBLEINRANGE" definition="list=$e1,$f2,$g3,$h4|||min=0.0|||max=2.0" />`		

mapExpression

A qti customOperator adapted from The American Institutes for Research open source TDS_ItemScoring/tinyctrlscoringengine/CtrlMapExpression.java.

Class:	**qti.sbac.customOperators.CTRL.mapExpression**	
Description:	This applies the contained expressions to each element sequentially to each element of the collection identified in “container." All expressions must return a Boolean value. Note that the symbol "@" can be used by expressions to reference each element in the set.	
Return Type:	Container	
Return Description	A container with the same basetype as the input container and an ordered cardinality.	
Attributes	**Type**	**Description**
container	Identifier	An identifier bound to a value with cardinality multiple or ordered.

QTI 3 Encoding

```
<qti-custom-operator class="qti.sbac.customOperators.CTRL.mapExpression"
definition="container=PP_RESPONSE">
  <!-- This baseValue with CDATA prevent a
       QTI XML Processor from evaluating the XML as QTI.  This permits
       the entire string to be handed over to mapExpression for evaluation -->
  <qti-base-value base-type="string"><![CDATA[
    <qti-inside coords="2,301,499,221" shape="rect">
      <qti-custom-operator class="qti.sbac.customOperators.GRAPHIC.GETPOINT"
definition="object=@"/>
    </qti-inside>
  ]]></qti-base-value>
</qti-custom-operator>
```

MAXINT

A qti customOperator adapted from The American Institutes for Research open source TDS_ItemScoring/tinyctrlscoringengine/CtrlMaxInt.java.

Class:	**qti.sbac.customOperators.CTRL.MAXINT**	
Description:	Returns the maximum value of a set of integers.	
Return Type:	Integer	
Return Description	An integer representing the maximum value in the set of integers provided.	
Attributes	**Type**	**Description**
list	String or Identifier	Either: 1) A list of values consisting of parseable integer values or integer variable identifiers prefaced with a "$", or 2) An identifier bound to a variable with basetype integer.
QTI 3 Encoding		
`<qti-custom-operator class="qti.sbac.customOperators.CTRL.MAXINT" definition="list=$e1,$f2,$g3,$h4" />`		

MININT

A qti customOperator adapted from The American Institutes for Research open source TDS_ItemScoring/tinyctrlscoringengine/CtrlMinInt.java.

Class:	**qti.sbac.customOperators.CTRL.MAXINT**	
Description:	Returns the minimum value of a set of integers.	
Return Type:	Integer	
Return Description	An integer representing the minimum value in the set of integers provided.	
Attributes	**Type**	**Description**
list	String or Identifier	Either: 1) A list of values consisting of parseable integer values or integer variable identifiers prefaced with a "$", or 2) An identifier bound to a variable with basetype integer.
QTI 3 Encoding		
`<qti-custom-operator class="qti.sbac.customOperators.CTRL.MININT" definition="list=$e1,$f2,$g3,$h4" />`		

stringToFloat

A qti customOperator adapted from The American Institutes for Research open source TDS_ItemScoring/tinyctrlscoringengine/CtrlStringToFloat.java.

Class:	**qti.sbac.customOperators.CTRL.stringToFloat**	
Description:	Parses a string and converts it to a floating point value	
Return Type:	Float	
Return Description	A numerical value based on parsing of the input string, or NaN if the parse fails.	
Attributes	**Type**	**Description**
inputString	String	A string formatted as a numerical value
QTI 3 Encoding		
`<qti-custom-operator class="qti.sbac.customOperators.CTRL.stringToFloat" definition="inputString=2.0" />`		

Package: qti.sbac.customOperators.EQ

Class Summary

This package contains 13 custom operators:

qti.sbac.customOperators.EQ.EVALUATE

qti.sbac.customOperators.EQ.EXPRESSIONCONTAINS

qti.sbac.customOperators.EQ.GETEQUATIONSCOUNT

qti.sbac.customOperators.EQ.GETINEQUALITIESCOUNT

qti.sbac.customOperators.EQ.ISEMPTY

qti.sbac.customOperators.EQ.ISEQUIVALENT

qti.sbac.customOperators.EQ.ISEQUIVALENTLOG

qti.sbac.customOperators.EQ.ISEQUIVALENTTRIG

qti.sbac.customOperators.EQ.ISMATCH

qti.sbac.customOperators.EQ.LINECONTAINS

qti.sbac.customOperators.EQ.MATCHEXPRESSION

qti.sbac.customOperators.EQ.NUMBERFROMEXPRESSION

qti.sbac.customOperators.EQ.PREPROCESSRESPONSE

EVALUATE

A qti customOperator adapted from The American Institutes for Research open source TDS_ItemScoring/tinyequationscoringengine/ TEEqEvaluate.java.

Class:	**qti.sbac.customOperators.EQ.EVALUATE**	
Description:	Converts expression to floating-point approximation (decimal number).	
Return Type:	Float	
Return Description	A decimal number that represents the expression or Double.NaN if it cannot be parsed.	
Attributes	**Type**	**Description**
object	identifier	identifier of the variable value to evaluate
QTI 3 Encoding		
`<qti-custom-operator class="qti.sbac.customOperators.EQ.EVALUATE" definition="object=EXPRESSIONVARIABLE" />`		

EXPRESSIONCONTAINS

A qti customOperator adapted from The American Institutes for Research open source TDS_ItemScoring/tinyequationscoringengine/TEEqExpressionContains.java.

Class:	**qti.sbac.customOperators.EQ.EXPRESSIONCONTAINS**	
Description:	Determines if a given math expression object contains a given substring	
Return Type:	Boolean	
Return Description	Returns true if the substring is found in the given math expression	
Attributes	**Type**	**Description**
object	identifier	identifier referencing a string representing a mathematical expression
string	string	Substring that we are looking for
QTI 3 Encoding		
`<qti-custom-operator class="qti.sbac.customOperators.EQ.EXPRESSIONCONTAINS" definition="object=EXPRESSIONVARIABLE|||string=needle" />`		

GETEQUATIONSCOUNT

A qti customOperator adapted from The American Institutes for Research open source TDS_ItemScoring/tinyequationscoringengine/TEEqEquationsCount.java.

Class:	**qti.sbac.customOperators.EQ.GETEQUATIONSCOUNT**	
Description:	Determines the number of equations in a math expression object	
Return Type:	Integer	
Return Description	An integer count of the equations in a math expression object	
Attributes	**Type**	**Description**
object	identifier	identifier referencing a string representing a mathematical expression
QTI 3 Encoding		
`<qti-custom-operator class="qti.sbac.customOperators.EQ.GETEQUATIONSCOUNT" definition="object=EXPRESSIONVARIABLE" />`		

GETINEQUALITIESCOUNT

A qti customOperator adapted from The American Institutes for Research open source TDS_ItemScoring/tinyequationscoringengine/TEqGetInequalitiesCount.java.

Class:	**qti.sbac.customOperators.EQ.GETINEQUALITIESCOUNT**	
Description:	Determines the number of inequalities in a math expression object	
Return Type:	integer	
Return Description	An integer count of the number of inequalities in a math expression object	
Attributes	**Type**	**Description**
object	identifier	identifier referencing a string representing a mathematical expression
QTI 3 Encoding		
`<qti-custom-operator class="qti.sbac.customOperators.EQ.GETINEQUALITIESCOUNT" definition="object=EXPRESSIONVARIABLE" />`		

ISEMPTY

A qti customOperator adapted from The American Institutes for Research open source TDS_ItemScoring/tinyequationscoringengine/TEqIsEmpty.java.

Class:	**qti.sbac.customOperators.EQ.ISEMPTY**	
Description:	Determines if a math expression object is empty	
Return Type:	boolean	
Return Description	Returns true if a math expression object is empty	
Attributes	**Type**	**Description**
object	identifier	identifier referencing a string representing a mathematical expression
QTI 3 Encoding		

```
<qti-custom-operator class="qti.sbac.customOperators.EQ.ISEMPTY"
definition="object=EXPRESSIONVARIABLE" />
```

ISEQUIVALENT

A qti customOperator adapted from The American Institutes for Research open source TDS_ItemScoring/tinyequationscoringengine/TEqIsEquivalent.java.

Class:	**qti.sbac.customOperators.EQ.ISEQUIVALENT**	
Description:	Compares the equation expression to an exemplar and determines if they are equivalent.	
Return Type:	boolean	
Return Description	Returns true if the equation referenced by object is equivalent to the exemplar	
Attributes	**Type**	**Description**
object	identifier	identifier referencing a string representing a mathematical expression
exemplar	string	string representing a mathematical expression of an exemplar
simplify	boolean	Flag indicating that simplifications can be performed (e.g. 1+1=2)

QTI 3 Encoding

```
<qti-custom-operator class="qti.sbac.customOperators.EQ.ISEQUIVALENT"
definition="object=EXPRESSIONVARIABLE|||exemplar=Ge(4500,30*m+1500)|||simplify=True" />

<qti-custom-operator class="qti.sbac.customOperators.EQ.ISEQUIVALENT"
definition="object=EXPRESSIONVARIABLE|||exemplar=360/40|||simplify=False" />
```

ISEQUIVALENTLOG

A qti customOperator adapted from The American Institutes for Research open source TDS_ItemScoring/tinyequationscoringengine/TEqIsEquivalentLog.java.

Class:	**qti.sbac.customOperators.EQ.ISEQUIVALENTLOG**	
Description:	Compares the equation expression to an exemplar and determines if they are equivalent using properties of Logs.	
Return Type:	boolean	
Return Description	Returns true if the equation referenced by object is equivalent to the exemplar expression using properties of Logs	
Attributes	**Type**	**Description**
object	identifier	identifier referencing a string representing a mathematical expression
exemplar	string	string representing a mathematical expression of an exemplar
assumptions	boolean	Flag indicating the bases are positive and exponents are real
QTI 3 Encoding		
`<qti-custom-operator class="qti.sbac.customOperators.EQ.ISEQUIVALENTLOG" definition="object=EXPRESSIONVARIABLE|||exemplar=<exemplar to be determined>|||assumptions=True" />`		

ISEQUIVALENTTRIG

A qti customOperator adapted from The American Institutes for Research open source TDS_ItemScoring/tinyequationscoringengine/TEqIsEquivalentTrig.java.

Class:	**qti.sbac.customOperators.EQ.ISEQUIVALENTTRIG**	
Description:	Compares the equation expression to an exemplar and determines if they are equivalent using Trig identities.	
Return Type:	boolean	
Return Description	Returns true if the equation referenced by object is equivalent to the exemplar expression using Trig identities.	
Attributes	**Type**	**Description**
object	identifier	identifier referencing a string representing a mathematical expression
exemplar	string	string representing a mathematical expression of an exemplar
QTI 3 Encoding		
`<qti-custom-operator class="qti.sbac.customOperators.EQ.ISEQUIVALENTTRIG" definition="object=EXPRESSIONVARIABLE|||exemplar=<exemplar to be determined>" />`		

ISMATCH

A qti customOperator adapted from The American Institutes for Research open source TDS_ItemScoring/tinyequationscoringengine/TEqIsMatch.java.

Class:	**qti.sbac.customOperators.EQ.ISMATCH**	
Description:	Determines if an expression can be matched to a parameterized pattern	
Return Type:	boolean	
Return Description	Returns true if expression can be matched by the given parameterized pattern	
Attributes	**Type**	**Description**
object	identifier	identifier referencing a string representing an expression to be matched to a given pattern
pattern	string	string representing a mathematical expression
parameters	String	Comma-separated parameters in the pattern (e.g. a,b,c)
constraints	String	Comma-separated constraints on parameter values (e.g. a>=1 and a<4, b==1 or b==2, c!=0)
variables	string	Comma-separated variables in the pattern (e.g. x,t)
simplify	boolean	Flag indicating that simplifications can be performed (e.g. 1+1=2)
QTI 3 Encoding		
`<customOperator class="qti.sbac.customOperators.EQ.ISMATCH" definition="constraints=a>=0.3 and a<=0.4|||object=EXPRESSIONVARIABLE|||parameters=a|||pattern=a|||simplify=True|||variables="/>`		

LINECONTAINS

A qti customOperator adapted from The American Institutes for Research open source TDS_ItemScoring/tinyequationscoringengine/TEEqLineContains.java.

Class:	**qti.sbac.customOperators.EQ.LINECONTAINS**	
Description:	Returns true if line contains an expression equivalent to exemplar expression. For ex, LineContains(8+8+8=24-2=22, 24-2=22) would return true.	
Return Type:	boolean	
Return Description	True/False	
Attributes	**Type**	**Description**
object	identifier	identifier referencing a string representing an expression to be matched to a given pattern
exemplar	string	string representing a mathematical expression of an exemplar
simplify	boolean	Flag indicating that simplifications can be performed (e.g. 1+1=2)
QTI 3 Encoding		
`<customOperator class="qti.sbac.customOperators.EQ.LINECONTAINS" definition="object=EXPRESSIONVARIABLE|||exemplar=24-2=22|||simplify=True"/>`		

MATCHEXPRESSION

A qti customOperator adapted from The American Institutes for Research open source TDS_ItemScoring/tinyequationscoringengine/TEqMatchExpression.java.

NOTE: this custom operator is not documented in the spec, but is an exact duplicate of EQ.ISMATCH.

Class:	**qti.sbac.customOperators.EQ.MATCHEXPRESSION**	
Description:	Determines if an expression can be matched to a parameterized pattern	
Return Type:	boolean	
Return Description	Returns true if expression can be matched by the given parameterized pattern	
Attributes	**Type**	**Description**
object	identifier	identifier referencing a string representing an expression to be matched to a given pattern
pattern	string	string representing a mathematical expression
parameters	String	Comma-separated parameters in the pattern (e.g. a,b,c)
constraints	String	Comma-separated constraints on parameter values (e.g. a>=1 and a<4, b==1 or b==2, c!=0)
variables	string	Comma-separated variables in the pattern (e.g. x,t)
simplify	boolean	Flag indicating that simplifications can be performed (e.g. 1+1=2)
QTI 3 Encoding		

```
<customOperator class="qti.sbac.customOperators.EQ.MATCHEXPRESSION"
definition="constraints=a>=0.3 and
a<=0.4|||object=EXPRESSIONVARIABLE|||parameters=a|||pattern=a|||simplify=True|||variables="/>
```

NUMBERFROMEXPRESSION

A qti customOperator adapted from The American Institutes for Research open source TDS_ItemScoring/tinyequationscoringengine/TEqNumberFromExpression.java.

Class:	**qti.sbac.customOperators.EQ.NUMBERFROMEXPRESSION**	
Description:	Converts the math expression object to a number	
Return Type:	double	
Return Description	Returns the numerical value represented by the math expression or Double.NaN if the expression cannot be parsed	
Attributes	**Type**	**Description**
object	identifier	identifier referencing a string representing a mathematical expression whose value is of interest
QTI 3 Encoding		
`<qti-custom-operator class="qti.sbac.customOperators.EQ.NUMBERFROMEXPRESSION" definition="object=EXPRESSIONVARIABLE" />`		

PREPROCESSRESPONSE

A qti customOperator adapted from The American Institutes for Research open source TDS_ItemScoring/tinyequationscoringengine/TEqPreProcess.java.

NOTE: This operator is used on virtually every EQ interaction as the first step in evaluating an EQ response by transforming the EQ interaction's XML into the container format expected by other custom operators.

Class:	**qti.sbac.customOperators.EQ.PREPROCESSRESPONSE**	
Description:	Translates equation responses (which are XML) to a collection of strings representing the objects in the set	
Return Type:	basetype	
Return Description	A basetype "string" with cardinality "ordered"	
Attributes	**Type**	**Description**
response	identifier	Names the original XML response to be preprocessed

QTI 3 Encoding

```
<qti-custom-operator class="qti.sbac.customOperators.EQ.PREPROCESSRESPONSE"
definition="response=EQRESPONSEVARIABLE" />


Typical pattern in EQ items:

<qti-outcome-declaration base-type="string" cardinality="ordered" identifier="PP_RESPONSE"/>

<qti-item-body>
  <!-- Equation Editor interaction -->
  <qti-custom-interaction class="tei-sbee" response-identifier="RESPONSE" />
</qti-item-body>

<qti-response-processing>
  <qti-set-outcome-value identifier="PP_RESPONSE">
    <qti-custom-operator class="qti.sbac.customOperators.EQ.PREPROCESSRESPONSE"
definition="response=RESPONSE" />
  </qti-set-outcome-value>

  ... further response evaluation by evaluating PP_RESPONSE ...
</qti-response-processing>
```

qti.sbac.customOperators.EQ – Example Usage and Transformation

Example 1 - Before: QRX

```
<assessmentItem>
  <responseDeclaration baseType="string" cardinality="single" identifier="RESPONSE" />
  <outcomeDeclaration baseType="integer" cardinality="single" identifier="SCORE">
    <defaultValue>
      <value>0</value>
    </defaultValue>
  </outcomeDeclaration>
  <outcomeDeclaration identifier="PP_RESPONSE" baseType="string" cardinality="ordered" />
  <outcomeDeclaration identifier="Line1" baseType="string" cardinality="single" />

  <responseProcessing>
    <setOutcomeValue identifier="PP_RESPONSE">
      <customOperator type="EQ" functionName="PREPROCESSRESPONSE" response="RESPONSE" />
    </setOutcomeValue>
    <setOutcomeValue identifier="Line1">
      <index n="1">
        <variable identifier="PP_RESPONSE" />
      </index>
    </setOutcomeValue>
    <responseCondition>
      <responseIf>
        <or>
          <customOperator type="EQ" functionName="ISEQUIVALENT" object="Line1"
exemplar="Eq(c,((10)/3)*g)" simplify="True" />
          <customOperator type="EQ" functionName="ISEQUIVALENT" object="Line1"
exemplar="Eq(((10)/3)*g,c)" simplify="True" />
          <customOperator type="EQ" functionName="ISMATCH" object="Line1"
pattern="Eq(c,a*g)" parameters="a" constraints="a>=3.3,a<3.4" variables="c,g"
simplify="True" />
          <customOperator type="EQ" functionName="ISMATCH" object="Line1"
pattern="Eq(a*g,c)" parameters="a" constraints="a>=3.3,a<3.4" variables="g,c"
simplify="True" />
        </or>
        <setOutcomeValue identifier="SCORE">
          <baseValue baseType="integer">1</baseValue>
        </setOutcomeValue>
      </responseIf>
    </responseCondition>
  </responseProcessing>
</assessmentItem>
```

Example 1 - After: QTI 3

```
<qti-assessment-item>
  <!-- EQ interactions produce an XML string -->
  <qti-response-declaration base-type="string" cardinality="single" identifier="RESPONSE" />
  <!-- Good Practice to specify normal max/min if not declaring MAXSCORE -->
  <qti-outcome-declaration base-type="float" cardinality="single" identifier="SCORE" normal-
maximum="1" normal-minimum="0">
    <qti-default-value>
      <qti-value>0</qti-value>
    </qti-default-value>
  </qti-outcome-declaration >
  <qti-outcome-declaration identifier="PP_RESPONSE" base-type="string"
cardinality="ordered"/>
  <qti-outcome-declaration identifier="Line1" base-type="string" cardinality="single"/>

  <qti-item-body>
    <!-- item body must contain an interaction for the response declaration -->
    <qti-custom-interaction class="tei-sbee" response-identifier="RESPONSE">
      <qti-custom-option>
        <![CDATA[ EAX ]]>
      </qti-custom-option>
    </qti-custom-interaction>
  </qti-item-body>

  <qti-response-processing>
    <qti-set-outcome-value identifier="PP_RESPONSE">
      <qti-custom-operator class="qti.sbac.customOperators.EQ.PREPROCESSRESPONSE"
definition="response=RESPONSE"/>
    </qti-set-outcome-value>
    <qti-set-outcome-value identifier="Line1">
      <qti-index n="1">
        <qti-variable identifier="PP_RESPONSE"/>
      </qti-index>
    </qti-set-outcome-value>
    <qti-response-condition>
      <qti-response-if>
        <qti-or>
          <qti-custom-operator class="qti.sbac.customOperators.EQ.ISEQUIVALENT"
definition="exemplar=Eq(c,((10)/3)*g)|||object=Line1|||simplify=True"/>
          <qti-custom-operator class="qti.sbac.customOperators.EQ.ISEQUIVALENT"
definition="exemplar=Eq(((10)/3)*g,c)|||object=Line1|||simplify=True"/>
          <qti-custom-operator class="qti.sbac.customOperators.EQ.ISMATCH"
definition="constraints=a>=3.3,a<3.4|||object=Line1|||parameters=a|||pattern=Eq(c,a*g)|||
simplify=True|||variables=c,g"/>
          <qti-custom-operator class="qti.sbac.customOperators.EQ.ISMATCH"
definition="constraints=a>=3.3,a<3.4|||object=Line1|||parameters=a|||pattern=Eq(a*g,c)|||
simplify=True|||variables=g,c"/>
        </qti-or>
        <qti-set-outcome-value identifier="SCORE">
          <qti-base-value base-type="float">1</baseValue>
        </qti-set-outcome-value>
      </qti-response-if>
    </qti-response-condition>
  </qti-response-processing>
</qti-assessment-item>
```

Example 2 - Before: QRX

```
<assessmentItem>
  <responseDeclaration baseType="string" cardinality="single" identifier="RESPONSE" />
  <outcomeDeclaration baseType="integer" cardinality="single" identifier="SCORE">
    <defaultValue>
      <value>0</value>
    </defaultValue>
  </outcomeDeclaration>
  <outcomeDeclaration baseType="string" cardinality="ordered" identifier="PP_RESPONSE" />
  <outcomeDeclaration identifier="PartA" baseType="string" cardinality="single" />
  <outcomeDeclaration identifier="PartB" baseType="string" cardinality="single" />

  <responseProcessing>
    <setOutcomeValue identifier="PP_RESPONSE">
      <customOperator type="EQ" functionName="PREPROCESSRESPONSE" response="RESPONSE" />
    </setOutcomeValue>
    <setOutcomeValue identifier="PartA">
      <index n="1">
        <variable identifier="PP_RESPONSE" />
      </index>
    </setOutcomeValue>
    <setOutcomeValue identifier="PartB">
      <index n="2">
        <variable identifier="PP_RESPONSE" />
      </index>
    </setOutcomeValue>
    <responseCondition>
      <responseIf>
        <and>
          <customOperator type="EQ" functionName="ISEQUIVALENT" object="PartA" exemplar="100"
simplify="True" />
          <customOperator type="EQ" functionName="ISEQUIVALENT" object="PartB" exemplar="96"
simplify="True" />
        </and>
        <setOutcomeValue identifier="SCORE">
          <baseValue baseType="integer">2</baseValue>
        </setOutcomeValue>
      </responseIf>
      <responseElseIf>
        <customOperator type="EQ" functionName="ISEQUIVALENT" object="PartB" exemplar="96"
simplify="True" />
        <setOutcomeValue identifier="SCORE">
          <baseValue baseType="integer">1</baseValue>
        </setOutcomeValue>
      </responseElseIf>
      <responseElseIf>
        <customOperator type="EQ" functionName="ISEQUIVALENT" object="PartA" exemplar="100"
simplify="True" />
        <setOutcomeValue identifier="SCORE">
          <baseValue baseType="integer">1</baseValue>
        </setOutcomeValue>
      </responseElseIf>
    </responseCondition>
  </responseProcessing>
</assessmentItem>
```

Example 2 - After: QTI 3

```
<qti-assessment-item>
  <!-- EQ interactions produce an XML string -->
  <qti-response-declaration base-type="string" cardinality="single" identifier="RESPONSE" />
  <!-- Good Practice to specify normal max/min if not declaring MAXSCORE -->
  <qti-outcome-declaration base-type="float" cardinality="single" identifier="SCORE" normal-
maximum="2" normal-minimum="0">
    <qti-default-value>
      <qti-value>0</qti-value>
    </qti-default-value>
  </qti-outcome-declaration >
  <qti-outcome-declaration base-type="string" cardinality="ordered"
identifier="PP_RESPONSE"/>
  <qti-outcome-declaration base-type="string" cardinality="single" identifier="PartA"/>
  <qti-outcome-declaration base-type="string" cardinality="single" identifier="PartB"/>

  <qti-item-body>
    <!-- item body must contain an interaction for the response declaration -->
    <qti-custom-interaction class="tei-sbee" response-identifier="RESPONSE">
      <qti-custom-option>
        <![CDATA[ EAX ]]>
      </qti-custom-option>
    </qti-custom-interaction>
  </qti-item-body>

  <qti-response-processing>
    <qti-set-outcome-value identifier="PP_RESPONSE">
      <qti-custom-operator class="qti.sbac.customOperators.EQ.PREPROCESSRESPONSE"
definition="response=RESPONSE"/>
    </qti-set-outcome-value>
    <qti-set-outcome-value identifier="PartA">
      <qti-index n="1">
        <qti-variable identifier="PP_RESPONSE"/>
      </qti-index>
    </qti-set-outcome-value>
    <qti-set-outcome-value identifier="PartB">
      <qti-index n="2">
        <qti-variable identifier="PP_RESPONSE"/>
      </qti-index>
    </qti-set-outcome-value>
    <qti-response-condition>
      <qti-response-if>
        <qti-and>
          <qti-custom-operator class="qti.sbac.customOperators.EQ.ISEQUIVALENT"
definition="exemplar=100|||object=PartA|||simplify=True"/>
          <qti-custom-operator class="qti.sbac.customOperators.EQ.ISEQUIVALENT"
definition="exemplar=96|||object=PartB|||simplify=True"/>
        </ qti-and>
        <qti-set-outcome-value identifier="SCORE">
          <qti-base-value base-type="float">2</qti-base-value>
        </qti-set-outcome-value>
      </qti-response-if>
      <qti-response-else-if>
        <qti-custom-operator class="qti.sbac.customOperators.EQ.ISEQUIVALENT"
definition="exemplar=96|||object=PartB|||simplify=True"/>
        <qti-set-outcome-value identifier="SCORE">
```

```
          <qti-base-value base-type="float">1</qti-base-value>
        </qti-set-outcome-value>
      </qti-response-else-if>
      <qti-response-else-if>
        <customOperator class="qti.sbac.customOperators.EQ.ISEQUIVALENT"
definition="exemplar=100|||object=PartA|||simplify=True"/>
        <qti-set-outcome-value identifier="SCORE">
          <qti-base-value base-type="float">1</qti-base-value>
        </qti-set-outcome-value>
      </qti-response-else-if>
    </qti-response-condition>
  </qti-response-processing>
</qti-assessment-item>
```

Package: qti.sbac.customOperators.GRAPHIC

Class Summary

This package contains 14 custom operators:

qti.sbac.customOperators.GRAPHIC.COUNTSIDES

qti.sbac.customOperators.GRAPHIC.GETLENGTH

qti.sbac.customOperators.GRAPHIC.GETNAME

qti.sbac.customOperators.GRAPHIC.GETPOINT

qti.sbac.customOperators.GRAPHIC.GETSELECTEDREGIONSCOUNT

qti.sbac.customOperators.GRAPHIC.GETSLOPE

qti.sbac.customOperators.GRAPHIC.GETVECTOR

qti.sbac.customOperators.GRAPHIC.HASVERTEX

qti.sbac.customOperators.GRAPHIC.HASVERTEXT

qti.sbac.customOperators.GRAPHIC.INTERSECTSPOINT

qti.sbac.customOperators.GRAPHIC.INTERSECTSREGION

qti.sbac.customOperators.GRAPHIC.ISGRAPHICTYPE

qti.sbac.customOperators.GRAPHIC.ISREGIONSELECTED

qti.sbac.customOperators.GRAPHIC.PREPROCESSRESPONSE

COUNTSIDES

A qti customOperator adapted from The American Institutes for Research open source TDS_ItemScoring/tinygrscoringengine/TGRECountSides.java.

Class:	**qti.sbac.customOperators.GRAPHIC.COUNTSIDES**	
Description:	Counts the number of line segments that comprise the object	
Return Type:	Integer	
Return Description	An integer representing the number of sides of the object. If the object passed in was not a graphic object or it was not an object comprised of line segments, zero is returned.	
Attributes	**Type**	**Description**
object	identifier	Identifier bound to a string representing a graphic object
QTI 3 Encoding		
`<qti-custom-operator class="qti.sbac.customOperators.GRAPHIC.COUNTSIDES" definition="object=EXPRESSIONVARIABLE" />`		

GETLENGTH

A qti customOperator adapted from The American Institutes for Research open source TDS_ItemScoring/tinygrscoringengine/TGREGetLength.java.

Class:	**qti.sbac.customOperators.GRAPHIC.GETLENGTH**	
Description:	Returns the length of the referenced vector	
Return Type:	float	
Return Description	A value indicating the length of the vector or NaN if anything but a vector	
Attributes	**Type**	**Description**
vector	identifier	Identifier bound to a string representing a vector
QTI 3 Encoding		
`<qti-custom-operator class="qti.sbac.customOperators.GRAPHIC.GETLENGTH" definition="vector=EXPRESSIONVARIABLE" />`		

GETNAME

A qti customOperator adapted from The American Institutes for Research open source TDS_ItemScoring/tinygrscoringengine/TGREGetLength.java.

Class:	**qti.sbac.customOperators.GRAPHIC.GETNAME**	
Description:	Returns the name of the named graphic object	
Return Type:	string	
Return Description	The name of the graphic object or an empty string if the graphic object is unnamed	
Attributes	**Type**	**Description**
object	identifier	Identifier bound to a string representing a graphic object.
QTI 3 Encoding		
`<qti-custom-operator class="qti.sbac.customOperators.GRAPHIC.GETNAME" definition="object=EXPRESSIONVARIABLE" />`		

GETPOINT

A qti customOperator adapted from The American Institutes for Research open source TDS_ItemScoring/tinygrscoringengine/TGREGetSinglePoint.java.

Class:	**qti.sbac.customOperators.GRAPHIC.GETNAME**	
Description:	Gets a point object corresponding to the location of a point or atomic object in the response space	
Return Type:	point	
Return Description	A point in the response space, or null if an inappropriate object was passed in	
Attributes	**Type**	**Description**
object	identifier	Identifier bound to a string representing a point or an atomic object
pointIndex	integer	The index of the specified point [Note: this attribute is not used in the original code]
QTI 3 Encoding		
`<qti-custom-operator class="qti.sbac.customOperators.GRAPHIC.GETPOINT" definition="object=EXPRESSIONVARIABLE" />`		

GETSELECTEDREGIONSCOUNT

A qti customOperator adapted from The American Institutes for Research open source TDS_ItemScoring/tinygrscoringengine/TGREGetSelectedCount.java.

Class:	**qti.sbac.customOperators.GRAPHIC.GETSELECTEDREGIONSCOUNT**	
Description:	Returns the quantity of the regions within a region group that are selected	
Return Type:	Integer	
Return Description	A point in the response space, or null if an inappropriate object was passed in	
Attributes	**Type**	**Description**
object	identifier	Identifier bound to a string representing a region group
QTI 3 Encoding		

```
<qti-custom-operator class="qti.sbac.customOperators.GRAPHIC.GETSELECTEDREGIONSCOUNT"
definition="object=EXPRESSIONVARIABLE" />
```

GETSLOPE

A qti customOperator adapted from The American Institutes for Research open source TDS_ItemScoring/tinygrscoringengine/TGREGetSlope.java.

Class:	**qti.sbac.customOperators.GRAPHIC.GETSLOPE**	
Description:	Calculates the slope of a line	
Return Type:	float	
Return Description	A value representing the slope of the line. Returns NaN if the object is not a line.	
Attributes	**Type**	**Description**
vector	identifier	Identifier bound to a string representing a line
QTI 3 Encoding		
`<qti-custom-operator class="qti.sbac.customOperators.GRAPHIC.GETSLOPE" definition="vector=EXPRESSIONVARIABLE" />`		

GETVECTOR

A qti customOperator adapted from The American Institutes for Research open source TDS_ItemScoring/tinygrscoringengine/TGREGetSlope.java.

Class:	**qti.sbac.customOperators.GRAPHIC.GETVECTOR**	
Description:	Returns the index-th vector of a multi-vector object. Vectors are sorted starting with the vector containing the top-most, left-most point, and moving clockwise. Where more than two line segments meet at the same point, the top-most, left-most comes first.	
Return Type:	String	
Return Description	A string representation of a vector.	
Attributes	**Type**	**Description**
object	identifier	Identifier bound to a string representing a graphic object
order	integer	Index of desired vector
QTI 3 Encoding		

```
<qti-custom-operator class="qti.sbac.customOperators.GRAPHIC.GETVECTOR"
definition="object=EXPRESSIONVARIABLE|||order=0" />
```

HASVERTEX

A qti customOperator adapted from The American Institutes for Research open source TDS_ItemScoring/tinygrscoringengine/TGREHasVertex.java.

NOTE: this method is incorrectly referenced as HASVERTEXT in response processing in sample items. Hence the class name has been changed to make those items work.

Class:	**qti.sbac.customOperators.GRAPHIC.HASVERTEX**	
Description:	Determines if a point is among the vertices of an object	
Return Type:	Boolean	
Return Description	Returns true if the point (x,y) is among the vertices of the object, false otherwise	
Attributes	**Type**	**Description**
object	identifier	Identifier bound to a string representing a graphic object
tolerance	float	allowable distance from the point still considered intersecting in the same units used in the coordinate plane of the item
Parameters	Type	Description
x	Integer	x-value of point to check
y	integer	y-value of point to check

QTI 3 Encoding

```
<qti-custom-operator class="qti.sbac.customOperators.GRAPHIC.HASVERTEX"
definition="object=EXPRESSIONVARIABLE|||tolerance=3.0">
  <qti-base-value base-type="integer">5</qti-base-value>
  <qti-base-value base-type="integer">6</qti-base-value>
</qti-custom-operator>
```

HASVERTEXT

A qti customOperator adapted from The American Institutes for Research open source TDS_ItemScoring/tinygrscoringengine/TGREHasVertex.java.

NOTE: HASVERTEX is incorrectly referenced as HASVERTEXT in response processing in sample items. Hence the class name has been changed to make those items work.

Class:	**qti.sbac.customOperators.GRAPHIC.HASVERTEXT**	
Description:	Determines if a point is among the vertices of an object	
Return Type:	Boolean	
Return Description	Returns true if the point (x,y) is among the vertices of the object, false otherwise	
Attributes	**Type**	**Description**
object	identifier	Identifier bound to a string representing a graphic object
tolerance	float	allowable distance from the point still considered intersecting in the same units used in the coordinate plane of the item
Parameters	Type	Description
x	Integer	x-value of point to check
y	integer	y-value of point to check

QTI 3 Encoding

```
<qti-custom-operator class="qti.sbac.customOperators.GRAPHIC.HASVERTEXT"
definition="object=EXPRESSIONVARIABLE|||tolerance=3.0">
  <qti-base-value base-type="integer">5</qti-base-value>
  <qti-base-value base-type="integer">6</qti-base-value>
</qti-custom-operator>
```

INTERSECTSPOINT

A qti customOperator adapted from The American Institutes for Research open source TDS_ItemScoring/tinygrscoringengine/TGREIntersectsPoint.java.

Class:	**qti.sbac.customOperators.GRAPHIC.INTERSECTSPOINT**	
Description:	Determines if the object intersects the designated point	
Return Type:	Boolean	
Return Description	Returns true if the object intersects the designated point, false otherwise	
Attributes	**Type**	**Description**
object	identifier	Identifier bound to a string representing a graphic object
tolerance	float	allowable distance from the point still considered intersecting in the same units used in the coordinate plane of the item
x	Integer	x coordinate of the point to check the intersection against
y	integer	Y coordinate of the point to check the intersection against
QTI 3 Encoding		
`<qti-custom-operator class="qti.sbac.customOperators.GRAPHIC.INTERSECTSPOINT" definition="object=EXPRESSIONVARIABLE|||tolerance=3.0|||x=5|||y=6" />`		

INTERSECTSREGION

A qti customOperator adapted from The American Institutes for Research open source TDS_ItemScoring/tinygrscoringengine/TGREIntersectsRegion.java.

Class:	**qti.sbac.customOperators.GRAPHIC.INTERSECTSREGION**	
Description:	Checks whether an object intersects a rectangular region. For usability reasons intersection is defined as within 5 pixels	
Return Type:	Boolean	
Return Description	Boolean indicating whether the object intersected the region	
Attributes	**Type**	**Description**
object	identifier	Identifier bound to a string representing a graphic object
topY	integer	Top of the region
leftX	Integer	Leftmost part of the region
bottomY	integer	Bottom of the region
rightX	integer	Rightmost part of the region
QTI 3 Encoding		
`<qti-custom-operator class="qti.sbac.customOperators.GRAPHIC.INTERSECTSREGION" definition="object=EXPRESSIONVARIABLE|||topY=5|||leftX=2|||bottomY=0|||rightX=7" />`		

ISGRAPHICTYPE

A qti customOperator adapted from The American Institutes for Research open source TDS_ItemScoring/tinygrscoringengine/TGREIsGraphicType.java.

Class:	**qti.sbac.customOperators.GRAPHIC.ISGRAPHICTYPE**	
Description:	Determines if the object is a graphic object of the identified type	
Return Type:	Boolean	
Return Description	Returns true if the graphic object is an object of the identified type	
Attributes	**Type**	**Description**
object	identifier	Identifier bound to a string representing a graphic object
graphicType	string	must be one of the following: "PALETTEIMAGE", "VECTOR", "ARROW", "POINT"
QTI 3 Encoding		
`<qti-custom-operator class="qti.sbac.customOperators.GRAPHIC.ISGRAPHICTYPE" definition="object=EXPRESSIONVARIABLE|||graphicType=ARROW" />`		

ISREGIONSELECTED

A qti customOperator adapted from The American Institutes for Research open source TDS_ItemScoring/tinygrscoringengine/TGREIsRegionSelected.java.

Class:	**qti.sbac.customOperators.GRAPHIC.ISREGIONSELECTED**	
Description:	Determines if a region is selected	
Return Type:	Boolean	
Return Description	Returns true if the region is selected, false otherwise	
Attributes	**Type**	**Description**
region	identifier	Identifier bound to a string representing a region
QTI 3 Encoding		

```
<qti-custom-operator class="qti.sbac.customOperators.GRAPHIC.ISREGIONSELECTED"
definition="region=1FirstObject" />
```

PREPROCESSRESPONSE

A qti customOperator adapted from The American Institutes for Research open source TDS_ItemScoring/tinygrscoringengine/TGREPreProcess.java.

NOTE: This operator is used on virtually every Grid interaction as the first step in evaluating an Grid response by transforming the Grid interaction's XML into the container format expected by other custom operators.

Class:	**qti.sbac.customOperators.GRAPHIC.PREPROCESSRESPONSE**	
Description:	Translates equation responses (which are XML) to a collection of strings representing the objects in the set	
Return Type:	basetype	
Return Description	A basetype "string" with cardinality "ordered"	
Attributes	**Type**	**Description**
response	identifier	Names the original XML response to be preprocessed

QTI 3 Encoding

```
<qti-custom-operator class="qti.sbac.customOperators.GRAPHIC.PREPROCESSRESPONSE"
definition="response=GIRESPONSEVARIABLE" />


Typical pattern in Grid items:

<qti-outcome-declaration base-type="string" cardinality="ordered" identifier="PP_RESPONSE"/>

<qti-item-body>
  <!-- Grid interaction -->
  <qti-custom-interaction class="tei-sbgrid" response-identifier="RESPONSE" />
</qti-item-body>

<qti-response-processing>
  <qti-set-outcome-value identifier="PP_RESPONSE">
    <qti-custom-operator class="qti.sbac.customOperators.GRAPHIC.PREPROCESSRESPONSE"
definition="response=RESPONSE" />
  </qti-set-outcome-value>

  ... further response evaluation by evaluating PP_RESPONSE ...
</qti-response-processing>
```

qti.sbac.customOperators.GRAPHIC – Example Usage and Transformation

Before: QRX

```
<assessmentItem>
  <responseDeclaration baseType="string" cardinality="single" identifier="RESPONSE" />
  <outcomeDeclaration baseType="integer" cardinality="single" identifier="SCORE">
    <defaultValue>
      <value>0</value>
    </defaultValue>
  </outcomeDeclaration>
  <outcomeDeclaration baseType="string" cardinality="ordered" identifier="PP_RESPONSE" />
  <outcomeDeclaration identifier="1" baseType="string" cardinality="ordered" />
  <outcomeDeclaration identifier="1FirstObject" baseType="string" cardinality="single" />
  <outcomeDeclaration identifier="RC1" baseType="boolean" cardinality="single" />
  <outcomeDeclaration identifier="5" baseType="string" cardinality="ordered" />
  <outcomeDeclaration identifier="5FirstObject" baseType="string" cardinality="single" />
  <outcomeDeclaration identifier="RC5" baseType="boolean" cardinality="single" />
  <outcomeDeclaration identifier="7" baseType="string" cardinality="ordered" />
  <outcomeDeclaration identifier="7FirstObject" baseType="string" cardinality="single" />
  <outcomeDeclaration identifier="RC7" baseType="boolean" cardinality="single" />
  <outcomeDeclaration identifier="12" baseType="string" cardinality="ordered" />
  <outcomeDeclaration identifier="12FirstObject" baseType="string" cardinality="single" />
  <outcomeDeclaration identifier="RC12" baseType="boolean" cardinality="single" />
  <outcomeDeclaration identifier="15" baseType="string" cardinality="ordered" />
  <outcomeDeclaration identifier="15FirstObject" baseType="string" cardinality="single" />
  <outcomeDeclaration identifier="RC15" baseType="boolean" cardinality="single" />
  <outcomeDeclaration identifier="CC" baseType="integer" cardinality="single" />

  <responseProcessing>
    <setOutcomeValue identifier="PP_RESPONSE">
      <customOperator type="GRAPHIC" functionName="PREPROCESSRESPONSE" response="RESPONSE" />
    </setOutcomeValue>
    <setOutcomeValue identifier="1">
      <customOperator type="CTRL" functionName="mapExpression" container="PP_RESPONSE">
        <stringMatch caseSensitive="True">
          <baseValue baseType="string">1</baseValue>
          <customOperator type="GRAPHIC" functionName="GETNAME" object="@" />
        </stringMatch>
      </customOperator>
    </setOutcomeValue>
    <setOutcomeValue identifier="1FirstObject">
      <index n="1">
        <variable identifier="1" />
      </index>
    </setOutcomeValue>
    <setOutcomeValue identifier="RC1">
      <customOperator type="GRAPHIC" functionName="ISREGIONSELECTED" region="1FirstObject" />
    </setOutcomeValue>
    <setOutcomeValue identifier="5">
      <customOperator type="CTRL" functionName="mapExpression" container="PP_RESPONSE">
        <stringMatch caseSensitive="True">
          <baseValue baseType="string">5</baseValue>
          <customOperator type="GRAPHIC" functionName="GETNAME" object="@" />
        </stringMatch>
      </customOperator>
    </setOutcomeValue>
```

```
    <setOutcomeValue identifier="5FirstObject">
      <index n="1">
        <variable identifier="5" />
      </index>
    </setOutcomeValue>
    <setOutcomeValue identifier="RC5">
      <customOperator type="GRAPHIC" functionName="ISREGIONSELECTED" region="5FirstObject" />
    </setOutcomeValue>
    <setOutcomeValue identifier="7">
      <customOperator type="CTRL" functionName="mapExpression" container="PP_RESPONSE">
        <stringMatch caseSensitive="True">
          <baseValue baseType="string">7</baseValue>
          <customOperator type="GRAPHIC" functionName="GETNAME" object="@" />
        </stringMatch>
      </customOperator>
    </setOutcomeValue>
    <setOutcomeValue identifier="7FirstObject">
      <index n="1"><variable identifier="7" /></index>
    </setOutcomeValue>
    <setOutcomeValue identifier="RC7">
      <customOperator type="GRAPHIC" functionName="ISREGIONSELECTED" region="7FirstObject" />
    </setOutcomeValue>
    <setOutcomeValue identifier="12">
      <customOperator type="CTRL" functionName="mapExpression" container="PP_RESPONSE">
        <stringMatch caseSensitive="True">
          <baseValue baseType="string">12</baseValue>
          <customOperator type="GRAPHIC" functionName="GETNAME" object="@" />
        </stringMatch>
      </customOperator>
    </setOutcomeValue>
    <setOutcomeValue identifier="12FirstObject">
      <index n="1">
        <variable identifier="12" />
      </index>
    </setOutcomeValue>
    <setOutcomeValue identifier="RC12">
      <customOperator type="GRAPHIC" functionName="ISREGIONSELECTED" region="12FirstObject" />
    </setOutcomeValue>
    <setOutcomeValue identifier="15">
      <customOperator type="CTRL" functionName="mapExpression" container="PP_RESPONSE">
        <stringMatch caseSensitive="True">
          <baseValue baseType="string">15</baseValue>
          <customOperator type="GRAPHIC" functionName="GETNAME" object="@" />
        </stringMatch>
      </customOperator>
    </setOutcomeValue>
    <setOutcomeValue identifier="15FirstObject">
      <index n="1">
        <variable identifier="15" />
      </index>
    </setOutcomeValue>
    <setOutcomeValue identifier="RC15">
      <customOperator type="GRAPHIC" functionName="ISREGIONSELECTED" region="15FirstObject" />
    </setOutcomeValue>
    <setOutcomeValue identifier="CC">
      <customOperator type="CTRL" functionName="COUNTBOOL" list="$RC1,$RC5,$RC7,$RC12,$RC15"
booleanValue="True" />
    </setOutcomeValue>
    <responseCondition>
      <responseIf>
```

```
        <equal>
          <variable identifier="CC" />
          <baseValue baseType="float">5</baseValue>
        </equal>
        <setOutcomeValue identifier="SCORE">
          <baseValue baseType="integer">1</baseValue>
        </setOutcomeValue>
      </responseIf>
    </responseCondition>
  </responseProcessing>
</assessmentItem>
```

After: QTI 3

```
<qti-assessment-item>
  <!-- GRID interactions produce an XML string -->
  <qti-response-declaration base-type="string" cardinality="single" identifier="RESPONSE" />
  <!-- Good Practice to specify normal max/min if not declaring MAXSCORE -->
  <qti-outcome-declaration base-type="float" cardinality="single" identifier="SCORE" normal-
maximum="1" normal-minimum="0">
    <qti-default-value><qti-value>0</qti-value></qti-default-value>
  </qti-outcome-declaration >

  <qti-outcome-declaration base-type="string" cardinality="ordered"
identifier="PP_RESPONSE"/>
  <qti-outcome-declaration base-type="string" cardinality="ordered" identifier="1"/>
  <qti-outcome-declaration base-type="string" cardinality="single"
identifier="1FirstObject"/>
  <qti-outcome-declaration base-type="boolean" cardinality="single" identifier="RC1"/>
  <qti-outcome-declaration base-type="string" cardinality="ordered" identifier="5"/>
  <qti-outcome-declaration base-type="string" cardinality="single"
identifier="5FirstObject"/>
  <qti-outcome-declaration base-type="boolean" cardinality="single" identifier="RC5"/>
  <qti-outcome-declaration base-type="string" cardinality="ordered" identifier="7"/>
  <qti-outcome-declaration base-type="string" cardinality="single"
identifier="7FirstObject"/>
  <qti-outcome-declaration base-type="boolean" cardinality="single" identifier="RC7"/>
  <qti-outcome-declaration base-type="string" cardinality="ordered" identifier="12"/>
  <qti-outcome-declaration base-type="string" cardinality="single"
identifier="12FirstObject"/>
  <qti-outcome-declaration base-type="boolean" cardinality="single" identifier="RC12"/>
  <qti-outcome-declaration base-type="string" cardinality="ordered" identifier="15"/>
  <qti-outcome-declaration base-type="string" cardinality="single"
identifier="15FirstObject"/>
  <qti-outcome-declaration base-type="boolean" cardinality="single" identifier="RC15"/>
  <qti-outcome-declaration base-type="integer" cardinality="single" identifier="CC"/>

  <qti-item-body>
    <!-- item body must contain an interaction for the response declaration -->
    <qti-custom-interaction class="tei-sbgrid" response-identifier="RESPONSE">
      <qti-custom-option><![CDATA[
        GAX
      ]]></qti-custom-option>
    </qti-custom-interaction>
  </qti-item-body>

  <qti-response-processing>
    <qti-set-outcome-value identifier="PP_RESPONSE">
      <qti-custom-operator class="qti.sbac.customOperators.GRAPHIC.PREPROCESSRESPONSE"
definition="response=RESPONSE"/>
    </qti-set-outcome-value>
    <qti-set-outcome-value identifier="1">
      <qti-custom-operator class="qti.sbac.customOperators.CTRL.mapExpression"
definition="container=PP_RESPONSE">
        <!-- This baseValue with CDATA prevent a
             QTI XML Processor from evaluating the XML as QTI. This permits
             the entire string to be handed over to mapExpression for evaluation -->
        <qti-base-value base-type="string"><![CDATA[
          <qti-string-match case-sensitive="true">
            <qti-base-value base-type="string">1</qti-base-value>
```

```
            <qti-custom-operator class="qti.sbac.customOperators.GRAPHIC.GETNAME"
definition="object=@"/>
          </qti-string-match>
        ]]></qti-base-value>
      </qti-custom-operator>
    </qti-set-outcome-value>
    <qti-set-outcome-value identifier="1FirstObject">
      <qti-index n="1">
        <qti-variable identifier="1"/>
      </qti-index>
    </qti-set-outcome-value>
    <qti-set-outcome-value identifier="RC1">
      <qti-custom-operator class="qti.sbac.customOperators.GRAPHIC.ISREGIONSELECTED"
definition="region=1FirstObject"/>
    </qti-set-outcome-value>
    <qti-set-outcome-value identifier="5">
      <qti-custom-operator class="qti.sbac.customOperators.CTRL.mapExpression"
definition="container=PP_RESPONSE">
        <!-- This baseValue with CDATA prevent a
             QTI XML Processor from evaluating the XML as QTI.  This permits
             the entire string to be handed over to mapExpression for evaluation -->
        <qti-base-value base-type="string"><![CDATA[
          <qti-string-match case-sensitive="true">
            <qti-base-value base-type="string">5</qti-base-value>
            <qti-custom-operator class="qti.sbac.customOperators.GRAPHIC.GETNAME"
definition="object=@"/>
          </qti-string-match>
        ]]></qti-base-value>
      </qti-custom-operator>
    </qti-set-outcome-value>
    <qti-set-outcome-value identifier="5FirstObject">
      <qti-index n="1">
        <qti-variable identifier="5"/>
      </qti-index>
    </qti-set-outcome-value>
    <qti-set-outcome-value identifier="RC5">
      <qti-custom-operator class="qti.sbac.customOperators.GRAPHIC.ISREGIONSELECTED"
definition="region=5FirstObject"/>
    </qti-set-outcome-value>
    <qti-set-outcome-value identifier="7">
      <qti-custom-operator class="qti.sbac.customOperators.CTRL.mapExpression"
definition="container=PP_RESPONSE">
        <!-- This baseValue with CDATA prevent a
             QTI XML Processor from evaluating the XML as QTI.  This permits
             the entire string to be handed over to mapExpression for evaluation -->
        <qti-base-value base-type="string"><![CDATA[
          <qti-string-match case-sensitive="true">
            <qti-base-value base-type="string">7</qti-base-value>
            <qti-custom-operator class="qti.sbac.customOperators.GRAPHIC.GETNAME"
definition="object=@"/>
          </qti-string-match>
        ]]></qti-base-value>
      </qti-custom-operator>
    </qti-set-outcome-value>
    <qti-set-outcome-value identifier="7FirstObject">
      <qti-index n="1">
        <qti-variable identifier="7"/>
      </qti-index>
    </qti-set-outcome-value>
    <qti-set-outcome-value identifier="RC7">
```

```
      <qti-custom-operator class="qti.sbac.customOperators.GRAPHIC.ISREGIONSELECTED"
definition="region=7FirstObject"/>
    </qti-set-outcome-value>
    <qti-set-outcome-value identifier="12">
      <qti-custom-operator class="qti.sbac.customOperators.CTRL.mapExpression"
definition="container=PP_RESPONSE">
        <!-- This baseValue with CDATA prevent a
             QTI XML Processor from evaluating the XML as QTI.  This permits
             the entire string to be handed over to mapExpression for evaluation -->
        <qti-base-value base-type="string"><![CDATA[
          <qti-string-match case-sensitive="true">
            <qti-base-value base-type="string">12</qti-base-value>
            <qti-custom-operator class="qti.sbac.customOperators.GRAPHIC.GETNAME"
definition="object=@"/>
          </qti-string-match>
        ]]></qti-base-value>
      </qti-custom-operator>
    </qti-set-outcome-value>
    <qti-set-outcome-value identifier="12FirstObject">
      <qti-index n="1">
        <qti-variable identifier="12"/>
      </qti-index>
    </qti-set-outcome-value>
    <qti-set-outcome-value identifier="RC12">
      <qti-custom-operator class="qti.sbac.customOperators.GRAPHIC.ISREGIONSELECTED"
definition="region=12FirstObject"/>
    </qti-set-outcome-value>
    <qti-set-outcome-value identifier="15">
      <qti-custom-operator class="qti.sbac.customOperators.CTRL.mapExpression"
definition="container=PP_RESPONSE">
        <!-- This baseValue with prevent a
             QTI XML Processor from evaluating the XML as QTI.  This permits
             the entire string to be handed over to mapExpression for evaluation -->
        <qti-base-value base-type="string"><![CDATA[
          <qti-string-match case-sensitive="true">
            <qti-base-value base-type="string">15</qti-base-value>
            <qti-custom-operator class="qti.sbac.customOperators.GRAPHIC.GETNAME"
definition="object=@"/>
          </qti-string-match>
        ]]></qti-base-value>
      </qti-custom-operator>
    </qti-set-outcome-value>
    <qti-set-outcome-value identifier="15FirstObject">
      <qti-index n="1">
        <qti-variable identifier="15"/>
      </qti-index>
    </qti-set-outcome-value>
    <qti-set-outcome-value identifier="RC15">
      <qti-custom-operator class="qti.sbac.customOperators.GRAPHIC.ISREGIONSELECTED"
definition="region=15FirstObject"/>
    </qti-set-outcome-value>
    <qti-set-outcome-value identifier="CC">
      <qti-custom-operator class="qti.sbac.customOperators.CTRL.COUNTBOOL"
definition="booleanValue=True|||list=$RC1,$RC5,$RC7,$RC12,$RC15"/>
    </qti-set-outcome-value>

    <responseCondition>
      <responseIf>
        <equal>
          <qti-variable identifier="CC"/>
```

```
          <qti-base-value base-type="float">5</qti-base-value>
        </equal>
        <qti-set-outcome-value identifier="SCORE">
          <qti-base-value base-type="integer">1</qti-base-value>
        </qti-set-outcome-value>
      </responseIf>
    </responseCondition>
  </qti-response-processing>
</qti-assessment-item>
```

Package: qti.sbac.customOperators.HT

Class Summary

This package contains 4 custom operators:

qti.sbac.customOperators.HT.GETHTELEMENTBYID

qti.sbac.customOperators.HT.GETHTELEMENTSCOUNT

qti.sbac.customOperators.HT.GETSELECTEDHTELEMENTSFROMLIST

qti.sbac.customOperators.HT.ISHTELEMENTSELECTED

GETHTELEMENTBYID

A qti customOperator reverse engineered after it was discovered in SBAC items.

Class:	**qti.sbac.customOperators.HT.GETHTELEMENTBYID**	
Description:	Returns the specified elementId if it's in the specified interactionSet.	
Return Type:	string	
Return Description	A string representing the specified elementId (in square brackets) if the specified elementId is in the specified interactionSet. If the elementId is not in the interactionSet, then a string representing an empty set [] is returned.	
Attributes	**Type**	**Description**
interactionSet	identifier	Interaction qti-response-declaration identifier.
elementId	string	The element-id to get from the selected interactionSet (if present).
QTI 3 Encoding		
`<qti-custom-operator class="qti.sbac.customOperators.HT.GETHTELEMENTBYID" definition="elementId=4|||interactionSet=G1"/>`		

GETHTELEMENTSCOUNT

A qti customOperator reverse engineered after it was discovered in SBAC items.

Class:	**qti.sbac.customOperators.HT.GETHTELEMENTSCOUNT**	
Description:	Returns the count of elements in the specified ordered container.	
Return Type:	integer	
Return Description	The count of elements in the specified ordered container.	
Attributes	**Type**	**Description**
elementSet	ordered container	An ordered container to count the elements in.
QTI 3 Encoding		

```
<qti-custom-operator class="qti.sbac.customOperators.HT.GETHTELEMENTSCOUNT"
definition="elementSet=All"/>
```

GETSELECTEDHTELEMENTSFROMLIST

A qti customOperator reverse engineered after it was discovered in SBAC items.

Class:	**qti.sbac.customOperators.HT.GETSELECTEDHTELEMENTSFROMLIST**	
Description:	Returns an ordered container of elementIds which is just a copy of the elements in the specified interactionSet.	
Return Type:	ordered container	
Return Description	An ordered container of the elementIds in the specified interactionSet.	
Attributes	**Type**	**Description**
elementsSet	ordered container	An ordered container of possible element ids (?). The actual purpose of this parameter is unknown and it is subsequently currently not evaluated or utilized in the code. Everything needed to return the selected HT elements is already included in the interactionSet.
interactionSet	identifier	Interaction qti-response-declaration identifier.
QTI 3 Encoding		
`<qti-custom-operator class="qti.sbac.customOperators.HT.GETSELECTEDHTELEMENTSFROMLIST" definition="elementsSet=1,2,3,4,5,6,7|||interactionSet=G1"/>`		

ISHTELEMENTSELECTED

A qti customOperator reverse engineered after it was discovered in SBAC items.

Class:	**qti.sbac.customOperators.HT.ISHTELEMENTSELECTED**	
Description:	Returns a boolean indicating if the specified element parameter is non-blank.	
Return Type:	boolean	
Return Description	true if the specified element is non-blank [<elementId>] or false for empty set [].	
Attributes	**Type**	**Description**
elements	string	An elementId surrounded by square brackets (if the specified elementId is selected) or an empty set [] if not.
QTI 3 Encoding		
`<qti-custom-operator class="qti.sbac.customOperators.HT.ISHTELEMENTSELECTED" definition="element=A4"/>`		

qti.sbac.customOperators.HT – Example Usage and Transformation

Before: QRX

```
<assessmentItem>
  <responseDeclaration baseType="string" cardinality="multiple" identifier="G1"/>
  <outcomeDeclaration baseType="integer" cardinality="single" identifier="SCORE">
    <defaultValue>
      <value>0</value>
    </defaultValue>
  </outcomeDeclaration>
  <outcomeDeclaration identifier="A3" baseType="string" cardinality="single"/>
  <outcomeDeclaration identifier="All" baseType="string" cardinality="ordered"/>
  <outcomeDeclaration identifier="Allc" baseType="integer" cardinality="single"/>
  <responseProcessing>
    <setOutcomeValue identifier="A3">
      <customOperator type="HT" functionName="GETHTELEMENTBYID" interactionSet="G1"
elementId="3"/>
    </setOutcomeValue>
    <setOutcomeValue identifier="All">
      <customOperator type="HT" functionName="GETSELECTEDHTELEMENTSFROMLIST"
interactionSet="G1" elementsSet="1,2,3,4,5"/>
    </setOutcomeValue>
    <setOutcomeValue identifier="Allc">
      <customOperator type="HT" functionName="GETHTELEMENTSCOUNT" elementSet="All"/>
    </setOutcomeValue>
    <responseCondition>
      <responseIf>
        <and>
          <customOperator type="HT" functionName="ISHTELEMENTSELECTED" element="A3"/>
          <equal>
            <variable identifier="Allc"/>
            <baseValue baseType="float">1</baseValue>
          </equal>
        </and>
        <setOutcomeValue identifier="SCORE">
          <baseValue baseType="integer">1</baseValue>
        </setOutcomeValue>
      </responseIf>
    </responseCondition>
  </responseProcessing>
</assessmentItem>
```

After: QTI 3

```
<qti-assessment-item>
  <qti-response-declaration base-type="string" cardinality="multiple" identifier="G1"/>
  <!-- Good Practice to specify normal max/min if not declaring MAXSCORE -->
  <qti-outcome-declaration base-type="float" cardinality="single" identifier="SCORE" normal-
maximum="1" normal-minimum="0">
    <qti-default-value><qti-value>0</qti-value></qti-default-value>
  </qti-outcome-declaration >
  <qti-outcome-declaration base-type="string" cardinality="single" identifier="A3"/>
  <qti-outcome-declaration base-type="string" cardinality="ordered" identifier="All"/>
  <qti-outcome-declaration base-type="integer" cardinality="single" identifier="Allc"/>

  <qti-item-body>
    <!-- item body must contain an interaction for the response declaration -->
    <qti-hottext-interaction class="sbac" max-choices="0" response-identifier="G1">
      <qti-hottext identifier="1">Hottext Sentence 1.</qti-hottext>
      <qti-hottext identifier="2">Hottext Sentence 2.</qti-hottext>
      ...
      <qti-hottext identifier="5">Hottext Sentence 5.</qti-hottext>
    </qti-hottext-interaction>
  </qti-item-body>

  <qti-response-processing>
    <qti-set-outcome-value identifier="A3">
      <qti-custom-operator class="qti.sbac.customOperators.HT.GETHTELEMENTBYID"
definition="elementId=3|||interactionSet=G1"/>
    </qti-set-outcome-value>
    <qti-set-outcome-value identifier="All">
      <qti-custom-operator class="qti.sbac.customOperators.HT.GETSELECTEDHTELEMENTSFROMLIST"
definition="elementsSet=1,2,3,4,5|||interactionSet=G1"/>
    </qti-set-outcome-value>
    <qti-set-outcome-value identifier="Allc">
      <qti-custom-operator class="qti.sbac.customOperators.HT.GETHTELEMENTSCOUNT"
definition="elementSet=All"/>
    </qti-set-outcome-value>
    <qti-response-condition>
      <qti-response-if>
        <qti-and>
          <qti-custom-operator class="qti.sbac.customOperators.HT.ISHTELEMENTSELECTED"
definition="element=A3"/>
          <qti-equal>
            <qti-variable identifier="Allc"/>
            <qti-base-value base-type="float">1</qti-base-value>
          </qti-equal>
        </qti-and>
        <qti-set-outcome-value identifier="SCORE">
          <qti-base-value base-type="integer">1</qti-base-value>
        </qti-set-outcome-value>
      </qti-response-if>
    </qti-response-condition>
  </qti-response-processing>
</qti-assessment-item>
```

Package: qti.sbac.customOperators.TABLE

Class Summary

This package contains 3 custom operators:

qti.sbac.customOperators.TABLE.GETCOLUMN

qti.sbac.customOperators.TABLE.GETHEADERROW

qti.sbac.customOperators.TABLE.GETVALUENUMERIC

GETCOLUMN

A qti customOperator adapted from The American Institutes for Research open source TDS_ItemScoring/tinytablescoringengine/TTGetColumn.java.

Class:	**qti.sbac.customOperators.TABLE.GETCOLUMN**	
Description:	Obtains the requested column	
Return Type:	string	
Return Description	A string representing the requested column. The string may be empty	
Attributes	**Type**	**Description**
table	identifier	Identifier bound to a string representing a table
columnName	string	The name of the desired column
QTI 3 Encoding		

```
<qti-custom-operator class="qti.sbac.customOperators.TABLE.GETCOLUMN"
definition="table=TABLEVARIABLE
|||columnName=column2" />
```

GETHEADERROW

A qti customOperator adapted from The American Institutes for Research open source TDS_ItemScoring/tinytablescoringengine/TTGetHeaderRow.java.

Class:	**qti.sbac.customOperators.TABLE.GETHEADERROW**	
Description:	Obtains the header row of the table	
Return Type:	string	
Return Description	A string representing the requested header row. The string may be empty	
Attributes	**Type**	**Description**
table	identifier	Identifier bound to a string representing a table
QTI 3 Encoding		
`<qti-custom-operator class="qti.sbac.customOperators.TABLE.GETHEADERROW" definition="table=TABLEVARIABLE" />`		

GETVALUENUMERIC

A qti customOperator adapted from The American Institutes for Research open source TDS_ItemScoring/tinytablescoringengine/TTGetValueNumeric.java.

Class:	**qti.sbac.customOperators.TABLE.GETVALUENUMERIC**	
Description:	Obtains the numeric value of the requested cell	
Return Type:	float	
Return Description	The numeric value of the indicated cell or NaN if the value cannot be successfully parsed	
Attributes	**Type**	**Description**
tableVector	identifier	Identifier bound to a string representing a table
index	Integer	The index of the desired cell (0-based)
QTI 3 Encoding		
`<qti-custom-operator class="qti.sbac.customOperators.TABLE.GETVALUENUMERIC" definition="tableVector=TABLEVARIABLE|||index=1" />`		

qti.sbac.customOperators.TABLE – Example Usage and Transformation

Before: QRX

```
<assessmentItem>
  <responseDeclaration baseType="string" cardinality="single" identifier="RESPONSE" />
  <outcomeDeclaration baseType="integer" cardinality="single" identifier="SCORE">
    <defaultValue><value>0</value></defaultValue>
  </outcomeDeclaration>
  <outcomeDeclaration identifier="line1" baseType="string" cardinality="single" />
  <outcomeDeclaration identifier="newline1" baseType="float" cardinality="single" />
  <outcomeDeclaration identifier="newline2" baseType="float" cardinality="single" />
  <responseProcessing>
    <setOutcomeValue identifier="line1">
      <customOperator type="TABLE" functionName="GETCOLUMN" table="#" columnName="col1" />
    </setOutcomeValue>
    <setOutcomeValue identifier="newline1">
      <customOperator type="TABLE" functionName="GETVALUENUMERIC" tableVector="line1"
index="0" />
    </setOutcomeValue>
    <setOutcomeValue identifier="newline2">
      <customOperator type="TABLE" functionName="GETVALUENUMERIC" tableVector="line1"
index="1" />
    </setOutcomeValue>
    <responseCondition>
      <responseIf>
        <and>
          <equal>
            <variable identifier="newline1" />
            <baseValue baseType="float">20</baseValue>
          </equal>
          <equal><variable identifier="newline2" />
            <baseValue baseType="float">25</baseValue>
          </equal>
        </and>
        <setOutcomeValue identifier="SCORE">
          <baseValue baseType="integer">1</baseValue>
      </setOutcomeValue>
    </responseIf>
  </responseCondition>
</responseProcessing>
</assessmentItem>
```

After: QTI 3

```
<qti-assessment-item>
  <!-- TABLE interactions produce an XML string -->
  <qti-response-declaration base-type="string" cardinality="single" identifier="RESPONSE" />
  <!-- Good Practice to specify normal max/min if not declaring MAXSCORE -->
  <qti-outcome-declaration base-type="float" cardinality="single" identifier="SCORE" normal-
maximum="1" normal-minimum="0">
    <qti-default-value><qti-value>0</qti-value></qti-default-value>
  </qti-outcome-declaration >
  <qti-outcome-declaration identifier="line1" base-type="string" cardinality="single" />
  <qti-outcome-declaration identifier="newline1" base-type="float" cardinality="single" />
  <qti-outcome-declaration identifier="newline2" base-type="float" cardinality="single" />

  <qti-item-body>
    <!-- item body must contain an interaction for the response declaration -->
    <qti-custom-interaction class="tei-sbtable" response-identifier="RESPONSE">
      <qti-custom-option>
        <![CDATA[ <table class="tableItem"> … </table> ]]>
      </qti-custom-option>
    </qti-custom-interaction>
  </qti-item-body>

  <qti-response-processing>
    <qti-set-outcome-value identifier="line1">
      <qti-custom-operator class="qti.sbac.customOperators.TABLE.GETCOLUMN"
definition="table=#|||columnName=col1" />
    </qti-set-outcome-value>
    <qti-set-outcome-value identifier="newline1">
      <qti-custom-operator class="qti.sbac.customOperators.TABLE.GETVALUENUMERIC"
definition="tableVector=line1|||index=0" />
    </qti-set-outcome-value>
    <qti-set-outcome-value identifier="newline2">
      <qti-custom-operator class="qti.sbac.customOperators.TABLE.GETVALUENUMERIC"
definition="tableVector=line1|||index=1" />
    </qti-set-outcome-value>
    <qti-response-condition>
      <qti-response-if>
        <qti-and>
          <qti-equal>
            <qti-variable identifier="newline1" />
            <qti-base-value base-type="float">20</qti-base-value>
          </qti-equal>
          <qti-equal><qti-variable identifier="newline2" />
            <qti-base-value base-type="float">25</qti-base-value>
          </qti-equal>
        </qti-and>
        <qti-setOutcomeValue identifier="SCORE">
          <qti-base-value base-type="float">1</qti-base-value>
      </qti-set-outcome-value>
    </qti-response-if>
  </qti-response-condition>
</qti-response-processing>
</qti-assessment-item>
```