javasim simtk documentation

Aaron Radke

February 11, 2003
Contents
Chapter 1

Package simtk.variable

Package Contents

<table>
<thead>
<tr>
<th>Classes</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>SimCheckBox</td>
<td>??</td>
</tr>
<tr>
<td>SimComboBox</td>
<td>??</td>
</tr>
<tr>
<td>SimComboBoxNumber</td>
<td>??</td>
</tr>
<tr>
<td>SimComboBoxString</td>
<td>??</td>
</tr>
<tr>
<td>SimDoubleArray</td>
<td>??</td>
</tr>
<tr>
<td>SimInteger</td>
<td>??</td>
</tr>
<tr>
<td>SimNumber</td>
<td>??</td>
</tr>
<tr>
<td>SimStaticNumber</td>
<td>??</td>
</tr>
<tr>
<td>SimString</td>
<td>??</td>
</tr>
<tr>
<td>SimTest</td>
<td>??</td>
</tr>
<tr>
<td>SimTextBox</td>
<td>??</td>
</tr>
<tr>
<td>SimVariable</td>
<td>??</td>
</tr>
<tr>
<td>VariableContainer</td>
<td>??</td>
</tr>
</tbody>
</table>
This is the class that holds all of the variables for a block.
1.1 Classes

1.1.1 Class SimCheckBox

this is a should be the text box variable, but it is not fully implemented yet

**Declaration**

```java
public class SimCheckBox
extends simtk.variable.SimVariable
```

**Constructors**

- `SimCheckBox`
  ```java
  public SimCheckBox()
  ```

- `SimCheckBox`
  ```java
  public SimCheckBox(java.lang.String title, boolean value)
  ```

**Methods**

- `getPanel`
  ```java
  public JPanel getPanel()
  ```

- `getValue`
  ```java
  public boolean getValue()
  ```

- `refreshFields`
  ```java
  public void refreshFields()
  ```

- `setValue`
  ```java
  public void setValue(boolean value)
  ```

- `toString`
  ```java
  public String toString()
  ```
Methods inherited from class `simtk.variable.SimVariable`

- **getPanel**
  
  ```java
  public abstract JPanel getPanel()
  ```

  * Usage
  
  + define the adjustment panel that needs to be displayed

- **refreshFields**

  ```java
  public abstract void refreshFields()
  ```

  * Usage
  
  + requires a function that knows how to refresh the data to the panel

- **variableChanged**

  ```java
  public void variableChanged()
  ```

  * Usage
  
  + this function is called whenever an internal variable changes

Methods inherited from class `simtk.SimBase`

- **getDescription**

  ```java
  public String getDescription()
  ```

- **getReloadBlock**

  ```java
  public Simulatable getReloadBlock()
  ```

  * Usage
  
  + set the block that gets called to resimulate when the a variable is changed.

- **getTitle**

  ```java
  public String getTitle()
  ```

  * Usage
  
  + get the user defined title for the entire block

- **setDescription**

  ```java
  public void setDescription(java.lang.String desc )
  ```

- **setReloadBlock**

  ```java
  public void setReloadBlock( simtk.Simulatable reloadBlock )
  ```

  * Usage
  
  + set the block that gets called to resimulate when the a variable is changed.

- **setTitle**

  ```java
  public void setTitle( java.lang.String title )
  ```

  * Usage
  
  + set the user defined title for the entire block
1.1.2 Class SimComboBox

this is class holds a combo box that can be used for selection of options in a block, mainly and upper level base object

Declaration

```java
public class SimComboBox
  extends simtk.variable.SimVariable
```

Constructors

- `SimComboBox`
  ```java
  public SimComboBox()
  ```
- `SimComboBox`
  ```java
  public SimComboBox(java.lang.String title)
  ```

Methods

- `getPanel`
  ```java
  public JPanel getPanel()
  ```
  - Usage
    * generate a label and the combobox

- `getSelectedIndex`
  ```java
  public int getSelectedIndex()
  ```
  - Usage
    * this makes a selection in the combo box

- `refreshFields`
  ```java
  public void refreshFields()
  ```
  - Usage
    * required field of SimVariables to refresh the fields of the variable

- `setValue`
  ```java
  public void setValue(int index)
  ```
SimComboBox

- **Usage**
  - * this makes a selection in the combo box via an integer

- **setValue**
  public void setValue( java.lang.String anObject )
  - **Usage**
    - * this makes a selection in the combo box via a string object

- **toString**
  public String toString( )

**Methods inherited from class simtk.variable.SimVariable**

- **getPanel**
  public abstract JPanel getPanel( )
  - **Usage**
    - * define the adjustment panel that needs to be displayed

- **refreshFields**
  public abstract void refreshFields( )
  - **Usage**
    - * requires a function that knows how to refresh the data to the panel

- **variableChanged**
  public void variableChanged( )
  - **Usage**
    - * this function is called whenever an internal variable changes

**Methods inherited from class simtk.SimBase**

- **getDescription**
  public String getDescription( )

- **getReloadBlock**
  public Simulatable getReloadBlock( )
  - **Usage**
    - * set the block that gets called to resimulate when the a variable is changed.

- **getTitle**
  public String getTitle( )
  - **Usage**
    - * get the user defined title for the entire block
### setDescription

```java
public void setDescription(java.lang.String desc)
```

- **setReloadBlock**

```java
public void setReloadBlock(simtk.Simulatable reloadBlock)
```

  - **Usage**
    * set the block that gets called to resimulate when the a variable is changed.

- **setTitle**

```java
public void setTitle(java.lang.String title)
```

  - **Usage**
    * set the user defined title for the entire block

### 1.1.3 Class SimComboBoxNumber

This class holds a combo box that can be used for selection of SimNumbers. This is useful for loading up Sim Numbers and then selecting one for an output.

#### Declaration

```java
public class SimComboBoxNumber
extends simtk.variable.SimComboBox
```

#### Fields

- **private Vector vec**
  * vector to hold the SimNumbers

#### Constructors

- **SimComboBoxNumber**

```java
public SimComboBoxNumber(java.lang.String title)
```

  - **Usage**
    * constructor to set the title
Methods

• add
  public void add( simtk.variable.SimNumber sn )
  
  – Usage
  * add a simnumber to the combobox

• getValue
  public double getValue( )
  
  – Usage
  * get the value of the selected SimNumber

Methods inherited from class simtk.variable.SimComboBox

• getPanel
  public JPanel getPanel( )
  
  – Usage
  * generate a label and the combobox

• getSelectedIndex
  public int getSelectedIndex( )
  
  – Usage
  * this makes a selection in the combo box

• refreshFields
  public void refreshFields( )
  
  – Usage
  * required field of SimVariables to refresh the fields of the variable

• setValue
  public void setValue( int index )
  
  – Usage
  * this makes a selection in the combo box via an integer

• setValue
  public void setValue( java.lang.String anObject )
  
  – Usage
  * this makes a selection in the combo box via a string object

• toString
  public String toString( )
Methods inherited from class `simtk.variable.SimVariable`

- `getPanel`
  public abstract JPanel getPanel( )
  - Usage
  * define the adjustment panel that needs to be displayed

- `refreshFields`
  public abstract void refreshFields( )
  - Usage
  * requires a function that knows how to refresh the data to the panel

- `variableChanged`
  public void variableChanged( )
  - Usage
  * this function is called whenever an internal variable changes

Methods inherited from class `simtk.SimBase`

- `getDescription`
  public String getDescription( )

- `getReloadBlock`
  public Simulatable getReloadBlock( )
  - Usage
  * set the block that gets called to resimulate when the a variable is changed.

- `getTitle`
  public String getTitle( )
  - Usage
  * get the user defined title for the entire block

- `setDescription`
  public void setDescription( java.lang.String desc )

- `setReloadBlock`
  public void setReloadBlock( simtk.Simulatable reloadBlock )
  - Usage
  * set the block that gets called to resimulate when the a variable is changed.

- `setTitle`
  public void setTitle( java.lang.String title )
  - Usage
  * set the user defined title for the entire block
1.1.4 Class SimComboBoxString

This class holds a combo box that can be used for selection of strings. This is useful for loading up a selection of strings which a value is selected from and a simulation is done when a new value is selected.

Declaration

```java
public class SimComboBoxString
    extends simtk.variable.SimComboBox
```

Fields

- private Vector vec
  - vector to hold the SimNumbers

Constructors

- `SimComboBoxString`  
  ```java
  public SimComboBoxString( java.lang.String title )
  ```
  - Usage
    * constructor to set the title

Methods

- `add`  
  ```java
  public void add( java.lang.String s )
  ```
  - Usage
    * add a string to the combobox

- `getValue`  
  ```java
  public String getValue( )
  ```
  - Usage
    * get the value of the selected SimNumber
Methods inherited from class `simtk.variable.SimComboBox`

- **getPanel**
  ```java
  public JPanel getPanel()
  ```
  - Usage
    * generate a label and the combobox

- **getSelectedIndex**
  ```java
  public int getSelectedIndex()
  ```
  - Usage
    * this makes a selection in the combobox

- **refreshFields**
  ```java
  public void refreshFields()
  ```
  - Usage
    * required field of SimVariables to refresh the fields of the variable

- **setValue**
  ```java
  public void setValue(int index)
  ```
  - Usage
    * this makes a selection in the combobox via an integer

- **setValue**
  ```java
  public void setValue(java.lang.String anObject)
  ```
  - Usage
    * this makes a selection in the combobox via a string object

- **toString**
  ```java
  public String toString()
  ```

Methods inherited from class `simtk.variable.SimVariable`

- **getPanel**
  ```java
  public abstract JPanel getPanel()
  ```
  - Usage
    * define the adjustment panel that needs to be displayed

- **refreshFields**
  ```java
  public abstract void refreshFields()
  ```
  - Usage
    * requires a function that knows how to refresh the data to the panel

- **variableChanged**
  ```java
  public void variableChanged()
  ```
  - Usage
    * this functions is called whenever an internal variable changes
1.1.5 Class SimDoubleArray

this is an array that for now holds a double type

Declaration

```java
public class SimDoubleArray
    extends simtk.variable.SimVariable
```
CONSTRUCTORS

- SimDoubleArray
  public SimDoubleArray()
  
  - Usage
    * constructor makes an untitled array of size 3

- SimDoubleArray
  public SimDoubleArray(java.lang.String title, int size)
  
  - Usage
    * constructor makes an array with a title the size is array will be size+1

METHODS

- displayIndexOverflowErr
  private void displayIndexOverflowErr(int index, int order)
  
  - Usage
    * overflow error message that is sent to the stdout

- getPanel
  public JPanel getPanel()

- getValue
  public double getValue()

- getValue
  public double getValue(int index)

- makeEditableArrayViaTextField
  public JTextField makeEditableArrayViaTextField()
  
  - Usage
    * static function to generate a text field that is linked to an array. the text field is linked to a double array that is displayed in the text field as separated by commas = ',' for now if there is an overflow error no error is signaled it will just fill the array so there is no error

- refreshFields
  public void refreshFields()
• **setValue**
  
  public void setValue( double [] new_array )

• **setValue**
  
  public void setValue( int n, double val )
  
  ---
  
  Usage
  
  * selects a particular zordered parameter (n) and sets the coefficient to val

• **setZeros**
  
  public void setZeros( )

  ---
  
  Usage
  
  * this method is for convenience in setting up z transfer functions it makes sure that the function is reset to zero from the initial conditions

• **toCommaString**
  
  public String toCommaString( )

  ---
  
  Usage
  
  * toCommaString

• **toString**
  
  public String toString( )

  ---
  
  Usage
  
  * toString in the form of: title, length, vector output

**Methods inherited from class simtk.variable.SimVariable**

• **getPanel**
  
  public abstract JPanel getPanel( )

  ---
  
  Usage
  
  * define the adjustment panel that needs to be displayed

• **refreshFields**
  
  public abstract void refreshFields( )

  ---
  
  Usage
  
  * requires a function that knows how to refresh the data to the panel

• **variableChanged**
  
  public void variableChanged( )

  ---
  
  Usage
  
  * this function is called whenever an internal variable changes
Methods inherited from class simtk.SimBase

- getDescription
  public String getDescription( )
- getReloadBlock
  public Simulatable getReloadBlock( )
  - Usage
    * set the block that gets called to resimulate when the a variable is changed.
- getTitle
  public String getTitle( )
  - Usage
    * get the user defined title for the entire block
- setDescription
  public void setDescription( java.lang.String desc )
- setReloadBlock
  public void setReloadBlock( simtk.Simulatable reloadBlock )
  - Usage
    * set the block that gets called to resimulate when the a variable is changed.
- setTitle
  public void setTitle( java.lang.String title )
  - Usage
    * set the user defined title for the entire block

1.1.6 Class SimInteger

description

Declaration

```java
public class SimInteger
extends simtk.variable.SimNumber
```

Constructors

- SimInteger
  public SimInteger( )
- SimInteger
  public SimInteger( java.lang.String title, double x )
**Methods**

- **getDoubleValue**
  
  ```java
  public double getDoubleValue()
  ```

- **getValue**
  
  ```java
  public double getValue()
  ```

- **main**
  
  ```java
  public static void main( java.lang.String [] args )
  ```
  
  **Usage**
  
  * simply test out this adjustment block

- **setDoubleValue**
  
  ```java
  public void setDoubleValue( double number )
  ```

- **setValue**
  
  ```java
  public void setValue( double number )
  ```

**Methods inherited from class simtk.variable.SimNumber**

- **doAnimationIteration**
  
  ```java
  public void doAnimationIteration()
  ```

  **Usage**
  
  * varies the slider automatically to investigate the control problem

- **getPanel**
  
  ```java
  public JPanel getPanel()
  ```

- **getValue**
  
  ```java
  public double getValue()
  ```

- **main**
  
  ```java
  public static void main( java.lang.String [] args )
  ```

  **Usage**
  
  * simply test out this adjustment block

- **refreshFields**
  
  ```java
  public void refreshFields()
  ```

- **setAnimationSpeed**
  
  ```java
  public void setAnimationSpeed( int time )
  ```

  **Usage**
  
  * Set the animation speed in ms

- **setValue**
  
  ```java
  public void setValue( double number )
  ```

- **toString**
  
  ```java
  public String toString()
  ```
Methods inherited from class `simtk.variable.SimVariable`

- `getPanel`
  
  ```java
  public abstract JPanel getPanel( )
  ```
  
  * Usage
  
  * define the adjustment panel that needs to be displayed

- `refreshFields`
  
  ```java
  public abstract void refreshFields( )
  ```
  
  * Usage
  
  * requires a function that knows how to refresh the data to the panel

- `variableChanged`
  
  ```java
  public void variableChanged( )
  ```
  
  * Usage
  
  * this function is called whenever an internal variable changes

Methods inherited from class `simtk.SimBase`

- `getDescription`
  
  ```java
  public String getDescription( )
  ```

- `getReloadBlock`
  
  ```java
  public Simulatable getReloadBlock( )
  ```
  
  * Usage
  
  * set the block that gets called to resimulate when the a variable is changed.

- `getTitle`
  
  ```java
  public String getTitle( )
  ```
  
  * Usage
  
  * get the user defined title for the entire block

- `setDescription` (method) (`java.lang.String desc`)

- `setReloadBlock` (method) (`simtk.Simulatable reloadBlock`)
  
  * Usage
  
  * set the block that gets called to resimulate when the a variable is changed.

- `setTitle` (method) (`java.lang.String title`)
  
  * Usage
  
  * set the user defined title for the entire block
1.1.7 Class SimNumber

this is a SimNumber that for now holds a double type

Declaration

```java
public class SimNumber
extends simtk.variable.SimVariable
```

Fields

- private double last_x
  - Stores that last number before the value was changed

Constructors

- `SimNumber()`
- `SimNumber(double x)`
- `SimNumber(java.lang.String title, double x)`

Methods

- `doAnimationIteration()`
  - Usage
    - * varies the slider automatically to investigate the control problem
- `getPanel()`
- `getValue()`
simtk.variable – SimNumber

- **main**
  ```java
class SimNumber {
    public static void main(String[] args) {
      // Usage
      // * simply test out this adjustment block
    }
  }
```

- **refreshFields**
  ```java
  public void refreshFields() {
  }
  ```

- **setAnimationSpeed**
  ```java
  public void setAnimationSpeed(int time) {
    // Usage
    // * Set the animation speed in ms
  }
  ```

- **setValue**
  ```java
  public void setValue(double number) {
  }
  ```

- **toString**
  ```java
  public String toString() {
  }
  ```

**Methods inherited from class simtk.variable.SimVariable**

- **getPanel**
  ```java
  public abstract JPanel getPanel() {
    // Usage
    // * define the adjustment panel that needs to be displayed
  }
  ```

- **refreshFields**
  ```java
  public abstract void refreshFields() {
    // Usage
    // * requires a function that knows how to refresh the data to the panel
  }
  ```

- **variableChanged**
  ```java
  public void variableChanged() {
    // Usage
    // * this functions is called whenever an internal variable changes
  }
  ```
Methods inherited from class simtk.SimBase

- `getDescription`
  ```java
  public String getDescription()
  ```
  - Usage
    * set the block that gets called to resimulate when the a variable is changed.

- `getReloadBlock`
  ```java
  public Simulatable getReloadBlock()
  ```
  - Usage
    * set the block that gets called to resimulate when the a variable is changed.

- `getTitle`
  ```java
  public String getTitle()
  ```
  - Usage
    * get the user defined title for the entire block

- `setDescription`
  ```java
  public void setDescription(java.lang.String desc)
  ```

- `setReloadBlock`
  ```java
  public void setReloadBlock(simtk.Simulatable reloadBlock)
  ```
  - Usage
    * set the block that gets called to resimulate when the a variable is changed.

- `setTitle`
  ```java
  public void setTitle(java.lang.String title)
  ```
  - Usage
    * set the user defined title for the entire block

1.1.8 Class SimStaticNumber

this is a SimNumber that has overloaded the getPanel to simple display the value of the number

**Declaration**

```java
public class SimStaticNumber
extends simtk.variable.SimNumber
```

**Constructors**

- `SimStaticNumber`
  ```java
  public SimStaticNumber(java.lang.String title, int val)
  ```
  - Usage
    * calls the constructor of the dynamic number
**Methods**

- **getPanel**
  
  ```java
  public JPanel getPanel()
  ```
  
  - Usage
    
    * overload the get panel of the dynamic number to only allow the number to be viewed

**Methods inherited from class simtk.variable.SimNumber**

- **doAnimationIteration**
  
  ```java
  public void doAnimationIteration()
  ```
  
  - Usage
    
    * varies the slider automatically to investigate the control problem

- **getValue**
  
  ```java
  public double getValue()
  ```

- **main**
  
  ```java
  public static void main( java.lang.String [] args )
  ```
  
  - Usage
    
    * simply test out this adjustment block

- **refreshFields**
  
  ```java
  public void refreshFields()
  ```

- **setAnimationSpeed**
  
  ```java
  public void setAnimationSpeed( int time )
  ```
  
  - Usage
    
    * Set the animation speed in ms

- **setValue**
  
  ```java
  public void setValue( double number )
  ```

- **toString**
  
  ```java
  public String toString()
  ```
Methods inherited from class simtk.variable.SimVariable

- **getPanel**
  public abstract JPanel getPanel()
  - Usage
    * define the adjustment panel that needs to be displayed

- **refreshFields**
  public abstract void refreshFields()
  - Usage
    * requires a function that knows how to refresh the data to the panel

- **variableChanged**
  public void variableChanged()
  - Usage
    * this functions is called whenever an internal variable changes

Methods inherited from class simtk.SimBase

- **getDescription**
  public String getDescription()

- **getReloadBlock**
  public Simulatable getReloadBlock()
  - Usage
    * set the block that gets called to resimulate when the a variable is changed.

- **getTitle**
  public String getTitle()
  - Usage
    * get the user defined title for the entire block

- **setDescription**
  public void setDescription(java.lang.String desc)

- **setReloadBlock**
  public void setReloadBlock(simtk.Simulatable reloadBlock)
  - Usage
    * set the block that gets called to resimulate when the a variable is changed.

- **setTitle**
  public void setTitle(java.lang.String title)
  - Usage
    * set the user defined title for the entire block
1.1.9 Class SimString

this is a SimNumber that for now holds a double type

DECLARATION

```java
public class SimString
extends simtk.variable.SimVariable
```

CONSTRUCTORS

- SimString
  ```java
  public SimString()
  ```

- SimString
  ```java
  public SimString(java.lang.String title, java.lang.String value)
  ```

METHODS

- getPanel
  ```java
  public JPanel getPanel()
  ```

- getValue
  ```java
  public String getValue()
  ```

- refreshFields
  ```java
  public void refreshFields()
  ```

- setValue
  ```java
  public void setValue(java.lang.String value)
  ```

- toString
  ```java
  public String toString()
  ```

METHODS INHERITED FROM CLASS simtk.variable.SimVariable

- getPanel
  ```java
  public abstract JPanel getPanel()
  ```
  - Usage
    ```markdown
    * define the adjustment panel that needs to be displayed
    ```
• **refreshFields**
  
  ```java
  public abstract void refreshFields()
  ```
  
  **Usage**
  
  * requires a function that knows how to refresh the data to the panel

• **variableChanged**

  ```java
  public void variableChanged()
  ```
  
  **Usage**
  
  * this functions is called whenever an internal variable changes

### METHODS INHERITED FROM CLASS `simtk.SimBase`

• **getDescription**

  ```java
  public String getDescription()
  ```

• **getReloadBlock**

  ```java
  public Simulatable getReloadBlock()
  ```
  
  **Usage**
  
  * set the block that gets called to resimulate when the a variable is changed.

• **getTitle**

  ```java
  public String getTitle()
  ```
  
  **Usage**
  
  * get the user defined title for the entire block

• **setDescription**

  ```java
  public void setDescription( java.lang.String desc)
  ```

• **setReloadBlock**

  ```java
  public void setReloadBlock( simtk.Simulatable reloadBlock)
  ```
  
  **Usage**
  
  * set the block that gets called to resimulate when the a variable is changed.

• **setTitle**

  ```java
  public void setTitle( java.lang.String title)
  ```
  
  **Usage**
  
  * set the user defined title for the entire block

### 1.1.10 CLASS SimTest

this is a SimNumber that for now holds a double type
**Declaration**

```java
public class SimTest
extends java.lang.Object
```

**Constructors**

- `SimTest`
  ```java
  public SimTest()
  ```

**Methods**

- `main`
  ```java
  public static void main( java.lang.String[] args )
  ```

**1.1.11 Class SimTextBox**

This is a should be the text box variable, but it is not fully implemented yet

**Declaration**

```java
public class SimTextBox
extends simtk.variable.SimVariable
```

**Constructors**

- `SimTextBox`
  ```java
  public SimTextBox()
  ```

**Methods**

- `append`
  ```java
  public void append( java.lang.String value )
  ```
- `getPanel`
  ```java
  public JPanel getPanel()
  ```
• **getValue**
  ```java
generic getValue()
```

• **main**
  ```java
public static void main(java.lang.String[] args)
  
  - Usage
    * simply test out this adjustment block
  ```

• **refreshFields**
  ```java
public void refreshFields()
```

• **setValue**
  ```java
public void setValue(java.lang.String value)
```

• **toString**
  ```java
public String toString()
```

**Methods inherited from class simtk.variable.SimVariable**

• **getPanel**
  ```java
public abstract JPanel getPanel()
  
  - Usage
    * define the adjustment panel that needs to be displayed
  ```

• **refreshFields**
  ```java
public abstract void refreshFields()
  
  - Usage
    * requires a function that knows how to refresh the data to the panel
  ```

• **variableChanged**
  ```java
public void variableChanged()
  
  - Usage
    * this functions is called whenever an internal variable changes
  ```

**Methods inherited from class simtk.SimBase**

• **getDescription**
  ```java
public String getDescription()
```

• **getReloadBlock**
  ```java
public Simulatable getReloadBlock()
  
  - Usage
    * set the block that gets called to resimulate when the a variable is changed.
  ```
• getTitle
  public String getTitle()
  
  Usage
  * get the user defined title for the entire block

• setDescription
  public void setDescription(java.lang.String desc)

• setReloadBlock
  public void setReloadBlock(simtk.Simulatable reloadBlock)
  
  Usage
  * set the block that gets called to resimulate when the a variable is changed.

• setTitle
  public void setTitle(java.lang.String title)
  
  Usage
  * set the user defined title for the entire block

### 1.1.12 Class SimVariable

This is the main abstract block for all types of Sim

---

### Declaration

```java
public abstract class SimVariable
  extends simtk.SimBase
  implements simtk.SimPanelable
```

---

### Constructors

- **SimVariable**
  ```java
  public SimVariable()
  ```

- **SimVariable**
  ```java
  public SimVariable(simtk.Simulatable reloadBlock)
  ```
Methods

- **getPanel**
  public abstract JPanel getPanel()
  
  - **Usage**
    * define the adjustment panel that needs to be displayed

- **refreshFields**
  public abstract void refreshFields()
  
  - **Usage**
    * requires a function that knows how to refresh the data to the panel

- **variableChanged**
  public void variableChanged()
  
  - **Usage**
    * this function is called whenever an internal variable changes

Methods inherited from class simtk.SimBase

- **getDescription**
  public String getDescription()

- **getReloadBlock**
  public Simulatable getReloadBlock()

  - **Usage**
    * set the block that gets called to resimulate when a variable is changed.

- **getTitle**
  public String getTitle()

  - **Usage**
    * get the user defined title for the entire block

- **setDescription**
  public void setDescription(java.lang.String desc)

- **setReloadBlock**
  public void setReloadBlock(simtk.Simulatable reloadBlock)

  - **Usage**
    * set the block that gets called to resimulate when a variable is changed.

- **setTitle**
  public void setTitle(java.lang.String title)

  - **Usage**
    * set the user defined title for the entire block
1.1.13  **CLASS VariableContainer**

This is the class that holds all of the variables for a block.

**DECLARATION**

```java
public class VariableContainer extends simtk.SimBase implements java.util.Iterator, simtk.SimPanelable
```

**FIELDS**

- private Vector varVec
  - the vector that contains all the simulation blocks
- private Iterator it
  - the iterator that contains all the simulation blocks

**CONSTRUCTORS**

- `VariableContainer`
  ```java
  public VariableContainer()
  ```
  -  *Usage*
    - the constructor makes the first vectors

- `VariableContainer`
  ```java
  public VariableContainer(simtk.Simulatable reloadBlock)
  ```
  -  *Usage*
    - the constructor makes the first vectors

**METHODS**

- `add`
  ```java
  public void add(simtk.variable.SimVariable sv)
  ```
  -  *Usage*
• * Adds the block to the container vector to iterate

• **createAdjustmentBox**
  public JPanel createAdjustmentBox( boolean createButtons )
  
  – Usage
  * create a box with all of the sim buttons

• **createSimButton**
  public JButton createSimButton( simtk.SimPanelable sp )
  
  – Usage
  * creates a sim button with action listener and sets up the dialog panel with the
    passed parameters

• **empty**
  public void empty( )
  
  – Usage
  * empty the variable container of simulation blocks

• **getPanel**
  public JPanel getPanel( )

• **getPanelableCount**
  public int getPanelableCount( )
  
  – Usage
  * iterate through the list and count the number of panelable blocks

• **hasNext**
  public boolean hasNext( )
  
  – Usage
  * iterator function

• **iterator**
  public Iterator iterator( )
  
  – Usage
  * iterator function

• **next**
  public Object next( )
  
  – Usage
• * iterator function

• remove
  public void remove( )
  – Usage
  * iterator function

• setAllReloadBlocks
  public void setAllReloadBlocks( )
  – Usage
  * iterate through the list and set all of the reload objects and set them all to the
  reload object defined for this container

• setAllReloadBlocks
  public void setAllReloadBlocks( simtk.Simulatable reloadBlock )
  – Usage
  * iterate through the list and set all of the reload objects and set them all to the
given reload object

Methods inherited from class simtk.SimBase

• getDescription
  public String getDescription( )

• getReloadBlock
  public Simulatable getReloadBlock( )
  – Usage
  * set the block that gets called to resimulate when the a variable is changed.

• getTitle
  public String getTitle( )
  – Usage
  * get the user defined title for the entire block

• setDescription
  public void setDescription( java.lang.String desc )

• setReloadBlock
  public void setReloadBlock( simtk.Simulatable reloadBlock )
  – Usage
  * set the block that gets called to resimulate when the a variable is changed.

• setTitle
  public void setTitle( java.lang.String title )
  – Usage
  * set the user defined title for the entire block
Chapter 2

Package simtk

Package Contents Page

Interfaces

ContainsFactory
This interface is simply make sure that the proper functions are implemented
in order to contain a factory

HasInput
This interface is just here for any Block that has an input to interface this
way it is easy to test if a block has an input with instanceof.

HasOutput
This interface is just here for any Block that has an output to interface this
way it is easy to test if a block has an output.

SimAdjustable
this is the adjustable abstract type, a new pattern is used for adjusting the
blocks for get Panel.

SimPanelable
this is the panelable abstract interface type.

SimReLoadable
this is the interface that ensures that a method has a calculateAndPlot method

Simulatable
this is the interface that ensures that a method has a doSimulation() method
and the getSimParams()

Classes

NoParamException
This block is to store and keep all the global type simulation parameters in
their own class

ScrollablePicture
This was implemented from sun’s java example on ScrollPanes

Sim1stOrder
\[ \frac{1}{s+a} \]

**Sim2ndOrder** ........................................... ??

\[ \frac{(w^2)}{(s^2 + 2wz^2 + w^2)} \]

**SimAdd** ........................................... ??

*adds multiple inputs to a single output*

**SimADRC** ........................................... ??

*compilation of sim blocks to create the ADRC block*

**SimBase** ........................................... ??

*This is the main abstract block for many of the classes*

**SimBlock** ........................................... ??

*This is the main abstract block for all types of Sim*

**SimBuffer** ........................................... ??

*This is the buffer block for the Sim environment.*

**SimContinuousProfile** ........................................... ??

*compilation of sim blocks to create the profile*

**SimDialogManager** ........................................... ??

*this is used to create JDialogs for the tuning of SimBlocks this generates a special JDialog that has simple sliders for a one to one ratio of variables*

**SimDiff** ........................................... ??

*This differentiator is defined by a Stf and is a double pole order diff*

**SimDiffEq** ........................................... ??

*...no description...*

**SimDisturbance** ........................................... ??

*Disturbance sources, the first will be a sine wave.*

**SimDoubleIntegrator** ........................................... ??

*this is simply a double integrator function*

**SimDoubleTriangle** ........................................... ??

*Double triangle profile function.*

**SimFactory** ........................................... ??

*This is the factory class that makes and controls the simulation environment.*

**SimFilter** ........................................... ??

*This filter contains multiple types of butterworth filters*

**SimFunction** ........................................... ??

*defines the bare bones to a Function block, now multi-input*

**SimFunction2Input** ........................................... ??

*general class for 2 inputs and 1 output*

**SimFunctionFactory** ........................................... ??

*compilation of sim blocks to create the complex blocks such as NPID from simple building blocks.*

**SimFunctionSelector** ........................................... ??

*this is one of the most simple function blocks to select different usable functions.*

**SimFunctionTest** ........................................... ??


compilation of sim blocks to create the NPID

SimFunctionXInput .......................................................... ??
defines the bare bones to a Sink block

SimGain ................................................................. ??
this is one of the most simple function blocks

SimGeneralClasicController ........................................ ??
compilation of sim blocks to create a a simple controller with the clasic r-y
= e input to the controller

SimGFunc ................................................................. ??
a block for nonlinearities, this block hat not been tested yet

SimImage ................................................................. ??
is a block which simply loads a picture at each iteration

SimImageFactory ........................................................ ??
this class has several functions to help get images from files

SimIntegrator ............................................................... ??
this is the block that simulates ztranfer functions a number of general types
of functions can be derived from this block

SimMultiply ............................................................... ??
Multiplys two input values

SimNoise ................................................................. ??
is a source that after each iteration it just outputs a random value from
-1 to 1 a gain is also available

SimNothing .............................................................. ??
Nothing function, just outputs a zero.

SimNPID ................................................................. ??
compilation of sim blocks to create the NPID

SimNPIDwc .............................................................. ??
class which makes the SimNPID parameterized

SimParameters ........................................................... ??
...no description...

SimPid ................................................................. ??
This is the differential pid sim block defined as u/e = kp + ki/s +
kd*s/(tau*s+1)^2.

SimPolyProfile ........................................................... ??
Polynomial Profile provides a smother response for position, velocity, and
acceleration and Jerk in place of a step function or trapezoidal function.

SimProfile ............................................................... ??
compilation of sim blocks to create the profile

SimPtPlot ............................................................... ??
this is a Sink block that continually stores and plots the next value

SimSaturation ........................................................... ??
Simple function for Saturation

SimSink ................................................................. ??
defines the bare bones to a Sink block

SimSource .................................................................
...no description...

SimSourceFactory ....................................................
compilation of sim blocks to create the complex blocks such as NPID from simple building blocks

SimSourceSelector ..................................................
this is one of the most simple source blocks to select different usable sources.

SimStat .................................................................
This is one of the most simple function blocks.

SimStdout ..............................................................
this is a Sink block that simply send the current output to the standard output

SimStep ..............................................................
Step function

SimStf .................................................................
this is the block that simulates s domain transfer functions!

SimSubtract ............................................................
Subtracts input 2 from input 1 out = in1 - in2.

SimTest ...............................................................
compilation of sim blocks to create the ADRC block

SimTextBoxOutput ....................................................
This is a Sink block that simply send the current output to a formated output in swing text box.

SimTimeIndex ........................................................
Step function

SimTrapezoid ..........................................................
Trapezodal profile function.

SimTrapezoidalProfile ................................................
compilation of sim blocks to create the profile

SimZtf .................................................................
this is the block that simulates ztranfer functions a number of general types of functions can be derived from this block

SimZtfDirect ...........................................................
this is the block that simulates ztranfer functions directly to the output without using the diff eq class a number of general types of functions can be derived from this block
2.1 Interfaces

2.1.1 INTERFACE ContainsFactory

This interface is simply made sure that the proper functions are implemented inorder to contain a factory

Declaration

public interface ContainsFactory

Methods

- `iterateBuffers`
  public void iterateBuffers()

2.1.2 INTERFACE HasInput

This interface is just here for any Block that has an input to interface this way it is easy to test if a block has an input with instanceof.

Declaration

public interface HasInput

Methods

- `addInput`
  public void addInput(simtk.SimBuffer bufref)
- `getInput`
  public SimBuffer getInput()
- `removeInputs`
  public void removeInputs()
- `setInput`
  public void setInput(simtk.SimBuffer bufref)
### 2.1.3 INTERFACE HasOutput

This interface is just here for any Block that has an output to interface this way it is easy to test if a block has an output.

**Declaration**

```java
public interface HasOutput
```

**Methods**

- `getOutput`
  ```java
  public SimBuffer getOutput()
  ```

- `setOutput`
  ```java
  public void setOutput( simtk.SimBuffer bufref )
  ```

### 2.1.4 INTERFACE SimAdjustable

This is the adjustable abstract type, a new pattern is used for adjusting the blocks for get Panel. If a class extends this class it has the ability to be adjusted.

**Declaration**

```java
public interface SimAdjustable
```

**Methods**

- `setupAdjustments`
  ```java
  public void setupAdjustments( simtk.SimReLoadable reloadable )
  ```

  **Usage**
  ```java
  * setup function for a reloadable object. Example body for a SimStep block that needs to be in the any adjustable class
    
    `dlg_maker = new SimDialogManager(null, 2, "Step parameters",this,reloadable);
    dlgs_maker.setupVariable(0,"Gain normalized at unity","gain");
    dlgs_maker.setupVariable(1,"Time delay from zero","delay");
  ```
2.1.5 INTERFACE SimPanelable

this is the panelable abstract interface type. If a class extends this class it has the ability to make an adjustment panel and more importantly generate its' own sliders and information

**Declaration**

```
public interface SimPanelable
```

**Methods**

- `getPanel`
  ```java
  public JPanel getPanel()
  ```

  **Usage**
  
  * The functions says the the function must be able to create a panel for itself. This allows information about a certain type of a block to be opened and shown. For example, a PtPlot can make a panel that holds the actual plot. Step functions can make panels that have adjustment sliders on them.

2.1.6 INTERFACE SimReLoadable

this is the interface that ensures that a method has a calculateAndPlot method

**Declaration**

```
public interface SimReLoadable
```

**Methods**

- `calculatePlotAndLoad`
  ```java
  public void calculatePlotAndLoad()
  ```

2.1.7 INTERFACE Simulatable

this is the interface that ensures that a method has a doSimulation() method and the getSimParams()
2.2 Classes

2.2.1 Class NoParamException

This block is to store and keep all the global type simulation parameters in their own class

Declaration

class NoParamException
extends java.lang.Exception

Constructors

- NoParamException
  public NoParamException()
Methods inherited from class java.lang.Throwable

- fillInStackTrace
  public synchronized native Throwable fillInStackTrace()

- getCause
  public Throwable getCause()

- getLocalizedMessage
  public String getLocalizedMessage()

- getMessage
  public String getMessage()

- getOurStackTrace
  private synchronized StackTraceElement getOurStackTrace()

- getStackTrace
  public StackTraceElement getStackTrace()

- getStackTraceDepth
  private native int getStackTraceDepth()

- getStackTraceElement
  private native StackTraceElement getStackTraceElement(int)

- initCause
  public synchronized Throwable initCause(java.lang.Throwable)

- printStackTrace
  public void printStackTrace()

- printStackTrace
  public void printStackTrace(java.io.PrintStream)

- printStackTrace
  public void printStackTrace(java.io.PrintWriter)

- printStackTraceAsCause
  private void printStackTraceAsCause(java.io.PrintStream,
                                      java.lang.StackTraceElement[])

- printStackTraceAsCause
  private void printStackTraceAsCause(java.io.PrintWriter,
                                       java.lang.StackTraceElement[])

- setStackTrace
  public void setStackTrace(java.lang.StackTraceElement[])

- toString
  public String toString()

- writeObject
  private synchronized void writeObject(java.io.ObjectOutputStream)

2.2.2 Class ScrollablePicture

This was implemented from sun’s java example on ScrollPanes
public class ScrollablePicture
extends javax.swing.JLabel
implements javax.swing.Scrollable

Serializable Fields

- private int maxUnitIncrement

Fields

- private int maxUnitIncrement

Constructors

- ScrollablePicture
  public ScrollablePicture( javax.swing.ImageIcon i, int m )

Methods

- getPreferredScrollableViewportSize
  public Dimension getPreferredScrollableViewportSize( )

- getScrollableBlockIncrement
  public int getScrollableBlockIncrement( java.awt.Rectangle visibleRect, int orientation, int direction )

- getScrollableTracksViewportHeight
  public boolean getScrollableTracksViewportHeight( )

- getScrollableTracksViewportWidth
  public boolean getScrollableTracksViewportWidth( )
• `getScrollableUnitIncrement`
  public int getScrollableUnitIncrement( java.awt.Rectangle visibleRect, int orientation, int direction )

• `setMaxUnitIncrement`
  public void setMaxUnitIncrement( int pixels )

Methods inherited from class `javax.swing.JLabel`

• `checkHorizontalKey`
  protected int checkHorizontalKey( int , java.lang.String )

• `checkVerticalKey`
  protected int checkVerticalKey( int , java.lang.String )

• `getAccessibleContext`
  public AccessibleContext getAccessibleContext( )

• `getDisabledIcon`
  public Icon getDisabledIcon( )

• `getDisplayedMnemonic`
  public int getDisplayedMnemonic( )

• `getDisplayedMnemonicIndex`
  public int getDisplayedMnemonicIndex( )

• `getHorizontalAlignment`
  public int getHorizontalAlignment( )

• `getHorizontalTextPosition`
  public int getHorizontalTextPosition( )

• `getIcon`
  public Icon getIcon( )

• `getIconTextGap`
  public int getIconTextGap( )

• `getLabelFor`
  public Component getLabelFor( )

• `getText`
  public String getText( )

• `getUI`
  public LabelUI getUI( )

• `getUIClassID`
  public String getUIClassID( )

• `getVerticalAlignment`
  public int getVerticalAlignment( )

• `getVerticalTextPosition`
  public int getVerticalTextPosition( )

• `imageUpdate`
  public boolean imageUpdate( java.awt.Image , int , int , int , int , int )
ParamString

protected String paramString()

setDisabledIcon

public void setDisabledIcon( javax.swing.Icon )

setDisplayedMnemonic

public void setDisplayedMnemonic( char )

setDisplayedMnemonic

public void setDisplayedMnemonic( int )

setDisplayedMnemonicIndex

public void setDisplayedMnemonicIndex( int )

setHorizontalAlignment

public void setHorizontalAlignment( int )

setHorizontalTextPosition

public void setHorizontalTextPosition( int )

setIcon

public voidsetIcon( javax.swing.Icon )

setIconTextGap

public voidsetIconTextGap( int )

setLabelFor

public voidsetLabelFor( java.awt.Component )

setText

public voidsetText( java.lang.String )

setUI

public voidsetUI( javax.swing.plaf.LabelUI )

setVerticalAlignment

public voidsetVerticalAlignment( int )

setVerticalTextPosition

public voidsetVerticalTextPosition( int )

updateUI

public void updateUI( )

writeObject

private void writeObject( java.io.ObjectOutputStream )

Methods inherited from class javax.swing.JComponent

_paintImmediately

void _paintImmediately( int , int , int , int )

<clinit>

static void <clinit>( )

addAncestorListener

public void addAncestorListener( javax.swing.event.AncestorListener )

addNotify

public void addNotify( )
• addPropertyChangeListener
  public synchronized void addPropertyChangeListener(
    java.beans.PropertyChangeListener  )
• addPropertyChangeListener
  public synchronized void addPropertyChangeListener( java.lang.String ,
    java.beans.PropertyChangeListener  )
• addVetoableChangeListener
  public synchronized void addVetoableChangeListener(
    java.beans.VetoableChangeListener  )
• adjustPaintFlags
  private void adjustPaintFlags(  )
• alwaysOnTop
  boolean alwaysOnTop(  )
• checkIfChildObscuredBySibling
  boolean checkIfChildObscuredBySibling(  )
• componentInputMapChanged
  void componentInputMapChanged( javax.swing.ComponentInputMap  )
• computeVisibleRect
  static final void computeVisibleRect( java.awt.Component , java.awt.Rectangle  )
• computeVisibleRect
  public void computeVisibleRect( java.awt.Rectangle  )
• compWriteObjectNotify
  void compWriteObjectNotify(  )
• contains
  public boolean contains( int , int  )
• createToolTip
  public JToolTip createToolTip(  )
• deregisterNextFocusableComponent
  private void deregisterNextFocusableComponent(  )
• disable
  public void disable(  )
• enable
  public void enable(  )
• enableSerialization
  void enableSerialization(  )
• firePropertyChange
  public void firePropertyChange( java.lang.String , boolean , boolean  )
• firePropertyChange
  public void firePropertyChange( java.lang.String , byte , byte  )
• firePropertyChange
  public void firePropertyChange( java.lang.String , char , char  )
• firePropertyChange
  public void firePropertyChange( java.lang.String , double , double  )
- `firePropertyChange`
  ```java
  public void firePropertyChange(java.lang.String, float, float)
  ```
- `firePropertyChange`
  ```java
  public void firePropertyChange(java.lang.String, int, int)
  ```
- `firePropertyChange`
  ```java
  public void firePropertyChange(java.lang.String, long, long)
  ```
- `firePropertyChange`
  ```java
  protected void firePropertyChange(java.lang.String, java.lang.Object, java.lang.Object)
  ```
- `firePropertyChange`
  ```java
  public void firePropertyChange(java.lang.String, short, short)
  ```
- `fireVetoableChange`
  ```java
  protected void fireVetoableChange(java.lang.String, java.lang.Object, java.lang.Object)
  ```
- `getAccessibleContext`
  ```java
  public AccessibleContext getAccessibleContext()
  ```
- `getActionForKeyStroke`
  ```java
  public ActionListener getActionForKeyStroke(javax.swing.KeyStroke)
  ```
- `getActionMap`
  ```java
  public final ActionMap getActionMap()
  ```
- `getActionMap`
  ```java
  final ActionMap getActionMap(boolean)
  ```
- `getAlignmentX`
  ```java
  public float getAlignmentX()
  ```
- `getAlignmentY`
  ```java
  public float getAlignmentY()
  ```
- `getAncestorListeners`
  ```java
  public AncestorListener getAncestorListeners()
  ```
- `getAutoscrolls`
  ```java
  public boolean getAutoscrolls()
  ```
- `getBorder`
  ```java
  public Border getBorder()
  ```
- `getBounds`
  ```java
  public Rectangle getBounds(java.awt.Rectangle)
  ```
- `getClientProperties`
  ```java
  private Dictionary getClientProperties()
  ```
- `getClientProperty`
  ```java
  public final Object getClientProperty(java.lang.Object)
  ```
- `getComponentGraphics`
  ```java
  ```
- `getConditionForKeyStroke`
  ```java
  public int getConditionForKeyStroke(javax.swing.KeyStroke)
  ```
- `getCreatedDoubleBuffer`
  ```java
  boolean getCreatedDoubleBuffer(boolean)
  ```
• `getDebugGraphicsOptions`
  public int getDebugGraphicsOptions()  
• `getDefaultLocale`
  public static Locale getDefaultLocale()  
• `getFlag`
  private boolean getFlag( int )  
• `getGraphics`
  public Graphics getGraphics()  
• `getHeight`
  public int getHeight()  
• `getInputMap`
  public final InputMap getInputMap()  
• `getInputMap`
  public final InputMap getInputMap( int )  
• `getInputMap`
  final InputMap getInputMap( int , boolean )  
• `getInputVerifier`
  public InputVerifier getInputVerifier()  
• `getInsets`
  public Insets getInsets()  
• `getInsets`
  public Insets getInsets( java.awt.Insets )  
• `getListeners`
  public EventListener getListeners( java.lang.Class )  
• `getLocation`
  public Point getLocation( java.awt.Point )  
• `getManagingFocusBackwardTraversalKeys`
  static Set getManagingFocusBackwardTraversalKeys()  
• `getManagingFocusForwardTraversalKeys`
  static Set getManagingFocusForwardTraversalKeys()  
• `getMaximumSize`
  public Dimension getMaximumSize()  
• `getMinimumSize`
  public Dimension getMinimumSize()  
• `getNextFocusableComponent`
  public Component getNextFocusableComponent()  
• `getPreferredSize`
  public Dimension getPreferredSize()  
• `getPropertyChangeListeners`
  public synchronized PropertyChangeListener getPropertyChangeListeners()  
• `getPropertyChangeListeners`
  public synchronized PropertyChangeListener getPropertyChangeListeners( java.lang.String )
• getRegisteredKeyStrokes
  public KeyStroke getRegisteredKeyStrokes( )
• getRootPane
  public JRootPane getRootPane( )
• getSize
  public Dimension getSize( java.awt.Dimension )
• getSuppressDropTarget
  private static boolean getSuppressDropTarget( )
• getToolTipLocation
  public Point getToolTipLocation( java.awt.event.MouseEvent )
• getToolTipText
  public String getToolTipText( )
• getToolTipText
  public String getToolTipText( java.awt.event.MouseEvent )
• getTopLevelAncestor
  public Container getTopLevelAncestor( )
• getTransferHandler
  public TransferHandler getTransferHandler( )
• getUIClassID
  public String getUIClassID( )
• getVerifyInputWhenFocusTarget
  public boolean getVerifyInputWhenFocusTarget( )
• getVetoableChangeListeners
  public synchronized VetoableChangeListener getVetoableChangeListeners( )
• getVisibleRect
  public Rectangle getVisibleRect( )
• getWidth
  public int getWidth( )
• getWriteObjCounter
  static byte getWriteObjCounter( javax.swing.JComponent )
• getX
  public int getX( )
• getY
  public int getY( )
• grabFocus
  public void grabFocus( )
• isDoubleBuffered
  public boolean isDoubleBuffered( )
• isLightweightComponent
  public static boolean isLightweightComponent( java.awt.Component )
• isManagingFocus
  public boolean isManagingFocus( )
• isMaximumSizeSet
  public boolean isMaximumSizeSet()
• isMinimumSizeSet
  public boolean isMinimumSizeSet()
• isOpaque
  public boolean isOpaque()
• isOptimizedDrawingEnabled
  public boolean isOptimizedDrawingEnabled()
• isPaintingTile
  public boolean isPaintingTile()
• isPreferredSizeSet
  public boolean isPreferredSizeSet()
• isRequestFocusEnabled
  public boolean isRequestFocusEnabled()
• isValidateRoot
  public boolean isValidateRoot()
• paint
  public void paint(java.awt.Graphics )
• paintBorder
  protected void paintBorder(java.awt.Graphics )
• paintChildren
  protected void paintChildren(java.awt.Graphics )
• paintComponent
  protected void paintComponent(java.awt.Graphics )
• paintDoubleBuffered
  private boolean paintDoubleBuffered(javax.swing.JComponent , java.awt.Component , java.awt.Graphics , int , int , int , int )
• paintImmediately
  public void paintImmediately(int , int , int , int )
• print
  public void print(java.awt.Graphics )
• printAll
  public void printAll(java.awt.Graphics )
• printBorder
  protected void printBorder(java.awt.Graphics )
• printChildren
  protected void printChildren(java.awt.Graphics )
- `printComponent`
  ```java
  protected void printComponent( java.awt.Graphics )
  ```

- `processComponentKeyEvent`
  ```java
  protected void processComponentKeyEvent( java.awt.event.KeyEvent )
  ```

- `processKeyBinding`
  ```java
  protected boolean processKeyBinding( javax.swing.KeyStroke , java.awt.event.KeyEvent , int , boolean )
  ```

- `processKeyBindings`
  ```java
  boolean processKeyBindings( java.awt.event.KeyEvent , boolean )
  ```

- `processKeyBindingsForAllComponents`
  ```java
  static boolean processKeyBindingsForAllComponents( java.awt.event.KeyEvent , java.awt.Container , boolean )
  ```

- `processKeyEvent`
  ```java
  protected void processKeyEvent( java.awt.event.KeyEvent )
  ```

- `processMouseMotionEvent`
  ```java
  protected void processMouseMotionEvent( java.awt.event.MouseEvent )
  ```

- `putClientProperty`
  ```java
  public final void putClientProperty( java.lang.Object , java.lang.Object )
  ```

- `readObject`
  ```java
  private void readObject( java.io.ObjectInputStream )
  ```

- `rectangleIsObscured`
  ```java
  boolean rectangleIsObscured( int , int , int , int )
  ```

- `rectangleIsObscuredBySibling`
  ```java
  boolean rectangleIsObscuredBySibling( int , int , int , int , int )
  ```

- `registerKeyboardAction`
  ```java
  public void registerKeyboardAction( java.awt.event.ActionListener , javax.swing.KeyStroke , int )
  ```

- `registerKeyboardAction`
  ```java
  public void registerKeyboardAction( java.awt.event.ActionListener , java.lang.String , javax.swing.KeyStroke , int )
  ```

- `registerNextFocusableComponent`
  ```java
  private void registerNextFocusableComponent( )
  ```

- `registerNextFocusableComponent`
  ```java
  private void registerNextFocusableComponent( java.awt.Component )
  ```

- `registerWithKeyboardManager`
  ```java
  private void registerWithKeyboardManager( boolean )
  ```

- `registerWithKeyboardManager`
  ```java
  private void registerWithKeyboardManager( javax.swing.KeyStroke )
  ```

- `removeAncestorListener`
  ```java
  public void removeAncestorListener( java.awt.event.AncestorListener )
  ```

- `removeNotify`
  ```java
  public void removeNotify( )
  ```
• `removePropertyChangeListener`
  public synchronized void `removePropertyChangeListener`
  `java.beans.PropertyChangeListener`

• `removePropertyChangeListener`
  public synchronized void `removePropertyChangeListener`
  `java.lang.String`,
  `java.beans.PropertyChangeListener`

• `removeVetoableChangeListener`
  public synchronized void `removeVetoableChangeListener`
  `java.beans.VetoableChangeListener`

• `repaint`
  public void `repaint`
  `long`, `int`, `int`, `int`, `int`

• `repaint`
  public void `repaint`
  `java.awt.Rectangle`

• `requestDefaultFocus`
  public boolean `requestDefaultFocus`

• `requestFocus`
  public void `requestFocus`

• `requestFocus`
  public boolean `requestFocus`
  `boolean`

• `requestFocusInWindow`
  public boolean `requestFocusInWindow`

• `resetKeyboardActions`
  public void `resetKeyboardActions`

• `reshape`
  public void `reshape`
  `int`, `int`, `int`, `int`

• `revalidate`
  public void `revalidate`

• `runInputVerifier`
  private boolean `runInputVerifier`

• `scrollRectToVisible`
  public void `scrollRectToVisible`
  `java.awt.Rectangle`

• `setActionMap`
  public final void `setActionMap`
  `javax.swing.ActionMap`

• `setAlignmentX`
  public void `setAlignmentX`
  `float`

• `setAlignmentY`
  public void `setAlignmentY`
  `float`

• `setAutoscrolls`
  public void `setAutoscrolls`
  `boolean`

• `setBackground`
  public void `setBackground`
  `java.awt.Color`
• setBorder
  public void setBorder( javax.swing.border.Border )
• setCreatedDoubleBuffer
  void setCreatedDoubleBuffer( boolean , boolean )
• setDebugGraphicsOptions
  public void setDebugGraphicsOptions( int )
• setDefaultLocale
  public static void setDefaultLocale( java.util.Locale )
• setDoubleBuffered
  public void setDoubleBuffered( boolean )
• setEnabled
  public void setEnabled( boolean )
• setFlag
  private void setFlag( int , boolean )
• setFont
  public void setFont( java.awt.Font )
• setForeground
  public void setForeground( java.awt.Color )
• setInputMap
  public final void setInputMap( int , javax.swing.InputMap )
• setInputVerifier
  public void setInputVerifier( javax.swing.InputVerifier )
• setMaximumSize
  public void setMaximumSize( java.awt.Dimension )
• setMinimumSize
  public void setMinimumSize( java.awt.Dimension )
• setNextFocusableComponent
  public void setNextFocusableComponent( java.awt.Component )
• setOpaque
  public void setOpaque( boolean )
• setPaintingChild
  void setPaintingChild( java.awt.Component )
• setPreferredSize
  public void setPreferredSize( java.awt.Dimension )
• setRequestFocusEnabled
  public void setRequestFocusEnabled( boolean )
• setToolTipText
  public void setToolTipText( java.lang.String )
• setTransferHandler
  public void setTransferHandler( javax.swing.TransferHandler )
• setUI
  protected void setUI( javax.swing.plaf.ComponentUI )
- setVerifyInputWhenFocusTarget
  public void setVerifyInputWhenFocusTarget( boolean )
- setVisible
  public void setVisible( boolean )
- setWriteObjCounter
  static void setWriteObjCounter( javax.swing.JComponent , byte )
- shouldDebugGraphics
  int shouldDebugGraphics( )
- superProcessMouseMotionEvent
  void superProcessMouseMotionEvent( java.awt.event.MouseEvent )
- unregisterKeyboardAction
  public void unregisterKeyboardAction( javax.swing.KeyStroke )
- unregisterWithKeyboardManager
  private void unregisterWithKeyboardManager( )
- unregisterWithKeyboardManager
  private void unregisterWithKeyboardManager( javax.swing.KeyStroke )
- update
  public void update( java.awt.Graphics )
- updateUI
  public void updateUI( )
- writeObject
  private void writeObject( java.io.ObjectOutputStream )

Methods inherited from class java.awt.Container

- <clinit>
  static void <clinit>( )
- add
  public Component add( java.awt.Component )
- add
  public Component add( java.awt.Component , int )
- add
  public void add( java.awt.Component , java.lang.Object )
- add
  public void add( java.awt.Component , java.lang.Object , int )
- add
  public Component add( java.lang.String , java.awt.Component )
- addContainerListener
  public synchronized void addContainerListener( java.awt.event.ContainerListener )
- addImpl
  protected void addImpl( java.awt.Component , java.lang.Object , int )
- addNotify
  public void addNotify( )
• addPropertyChangeListener
  public void addPropertyChangeListener( java.beans.PropertyChangeListener )

• addPropertyChangeListener
  public void addPropertyChangeListener( java.lang.String ,
  java.beans.PropertyChangeListener )

• adjustDescendantsOnParent
  void adjustDescendantsOnParent( int )

• adjustDescendants
  void adjustDescendants( int )

• adjustListeningChildren
  void adjustListeningChildren( long , int )

• applyComponentOrientation
  public void applyComponentOrientation( java.awt.ComponentOrientation )

• areFocusTraversalKeysSet
  public boolean areFocusTraversalKeysSet( int )

• checkGD
  void checkGD( java.lang.String )

• countComponents
  public int countComponents( )

• countHierarchyMembers
  int countHierarchyMembers( )

• createChildHierarchyEvents
  void createChildHierarchyEvents( int , long , boolean )

• createHierarchyEvents
  int createHierarchyEvents( int , java.awt.Component , java.awt.Container , long , boolean )

• deliverEvent
  public void deliverEvent( java.awt.Event )

• dispatchEventImpl
  void dispatchEventImpl( java.awt.AWTEvent )

• dispatchEventToSelf
  void dispatchEventToSelf( java.awt.AWTEvent )

• doLayout
  public void doLayout( )

• eventEnabled
  boolean eventEnabled( java.awt.AWTEvent )

• findComponentAt
  public Component findComponentAt( int , int )

• findComponentAt
  final Component findComponentAt( int , int , boolean )

• findComponentAt
  public Component findComponentAt( java.awt.Point )

• findTraversalRoot
  private Container findTraversalRoot( )
• getAccessibleAt
  Accessible getAccessibleAt(java.awt.Point)

• getAccessibleChild
  Accessible getAccessibleChild(int)

• getAccessibleChildrenCount
  int getAccessibleChildrenCount()

• getAlignmentX
  public float getAlignmentX()

• getAlignmentY
  public float getAlignmentY()

• getComponent
  public Component getComponent(int)

• getComponentAt
  public Component getComponentAt(int, int)

• getComponentAt
  public Component getComponentAt(java.awt.Point)

• getComponentCount
  public int getComponentCount()

• getComponents
  public Component getComponents()

• getContainerListeners
  public synchronized ContainerListener getContainerListeners()

• getDropTargetEventTarget
  Component getDropTargetEventTarget(int, int, boolean)

• getFocusTraversalKeys
  public Set getFocusTraversalKeys(int)

• getFocusTraversalPolicy
  public FocusTraversalPolicy getFocusTraversalPolicy()

• getInsets
  public Insets getInsets()

• getLayout
  public LayoutManager getLayout()

• getListeners
  public EventListener getListeners(java.lang.Class)

• getMaximumSize
  public Dimension getMaximumSize()

• getMinimumSize
  public Dimension getMinimumSize()

• getMouseEventTarget
  Component getMouseEventTarget(int, int, boolean)
• getMouseEventTarget
  private Component getMouseEventTarget( int , int , boolean ,
  java.awt.Container.EventTargetFilter , boolean )
• getMouseEventTargetImpl
  private Component getMouseEventTargetImpl( int , int , boolean ,
  java.awt.Container.EventTargetFilter , boolean , boolean )
• getPreferredSize
  public Dimension getPreferredSize( )
• initializeFocusTraversalKeys
  void initializeFocusTraversalKeys( )
• initIDs
  private static native void initIDs( )
• insets
  public Insets insets( )
• invalidate
  public void invalidate( )
• invalidateTree
  void invalidateTree( )
• isAncestorOf
  public boolean isAncestorOf( java.awt.Component )
• isFocusCycleRoot
  public boolean isFocusCycleRoot( )
• isFocusCycleRoot
  public boolean isFocusCycleRoot( java.awt.Container )
• isFocusTraversalPolicySet
  public boolean isFocusTraversalPolicySet( )
• layout
  public void layout( )
• lightweightPaint
  void lightweightPaint( java.awt.Graphics )
• lightweightPrint
  void lightweightPrint( java.awt.Graphics )
• list
  public void list( java.io.PrintStream , int )
• list
  public void list( java.io.PrintWriter , int )
• locate
  public Component locate( int , int )
• minimumSize
  public Dimension minimumSize( )
• nextFocusHelper
  boolean nextFocusHelper( )
• numListening
  int numListening( long )
- `paint`
  ```java
  public void paint( java.awt.Graphics )
  ```
- `paintComponents`
  ```java
  public void paintComponents( java.awt.Graphics )
  ```
- `paintHeavyweightComponents`
  ```java
  void paintHeavyweightComponents( java.awt.Graphics )
  ```
- `paramString`
  ```java
  protected String paramString()
  ```
- `postProcessKeyEvent`
  ```java
  void postProcessKeyEvent( java.awt.event.KeyEvent )
  ```
- `postsOldMouseEvents`
  ```java
  boolean postsOldMouseEvents()
  ```
- `preferredSize`
  ```java
  public Dimension preferredSize()
  ```
- `preProcessKeyEvent`
  ```java
  void preProcessKeyEvent( java.awt.event.KeyEvent )
  ```
- `print`
  ```java
  public void print( java.awt.Graphics )
  ```
- `printComponents`
  ```java
  public void printComponents( java.awt.Graphics )
  ```
- `printHeavyweightComponents`
  ```java
  void printHeavyweightComponents( java.awt.Graphics )
  ```
- `processContainerEvent`
  ```java
  protected void processContainerEvent( java.awt.event.ContainerEvent )
  ```
- `processEvent`
  ```java
  protected void processEvent( java.awt.AWTEvent )
  ```
- `proxyEnableEvents`
  ```java
  void proxyEnableEvents( long )
  ```
- `readObject`
  ```java
  private void readObject( java.io.ObjectInputStream )
  ```
- `remove`
  ```java
  public void remove( java.awt.Component )
  ```
- `remove`
  ```java
  public void remove( int )
  ```
- `removeAll`
  ```java
  public void removeAll()
  ```
- `removeContainerListener`
  ```java
  public synchronized void removeContainerListener( java.awt.event.ContainerListener )
  ```
- `removeNotify`
  ```java
  public void removeNotify()
  ```
- `setFocusCycleRoot`
  ```java
  public void setFocusCycleRoot( boolean )
  ```
setFocusTraversalKeys
public void setFocusTraversalKeys(int, java.util.Set)

setFocusTraversalPolicy
public void setFocusTraversalPolicy(java.awt.FocusTraversalPolicy)

setFont
public void setFont(java.awt.Font)

setLayout
public void setLayout(java.awt.LayoutManager)

setZOrder
void setZOrder(java.awt.Component, int)

transferFocusBackward
public void transferFocusBackward()

transferFocusDownCycle
public void transferFocusDownCycle()

update
public void update(java.awt.Graphics)

validate
public void validate()

validateTree
protected void validateTree()

writeObject
private void writeObject(java.io.ObjectOutputStream)

Methods inherited from class java.awt.Component

<clinit>
static void <clinit>()

action
public boolean action(java.awt.Event, java.lang.Object)

add
public synchronized void add(java.awt.PopupMenu)

addComponentListener
public synchronized void addComponentListener(java.awt.event.ComponentListener)

addFocusListener
public synchronized void addFocusListener(java.awt.event.FocusListener)

addHierarchyBoundsListener
public void addHierarchyBoundsListener(java.awt.event.HierarchyBoundsListener)

addHierarchyListener
public void addHierarchyListener(java.awt.event.HierarchyListener)

addInputMethodListener
public synchronized void addInputMethodListener(java.awt.event.InputMethodListener)
- `addKeyListener`
  ```java
  public synchronized void addKeyListener(java.awt.event.KeyListener)
  ```

- `addMouseListener`
  ```java
  public synchronized void addMouseListener(java.awt.event.MouseListener)
  ```

- `addMouseMotionListener`
  ```java
  public synchronized void addMouseMotionListener(java.awt.event.MouseMotionListener)
  ```

- `addMouseWheelListener`
  ```java
  public synchronized void addMouseWheelListener(java.awt.event.MouseWheelListener)
  ```

- `addNotify`
  ```java
  public void addNotify()
  ```

- `addPropertyChangeListener`
  ```java
  public synchronized void addPropertyChangeListener(java.beans.PropertyChangeListener)
  ```

- `addPropertyChangeListener`
  ```java
  public synchronized void addPropertyChangeListener(java.lang.String, java.beans.PropertyChangeListener)
  ```

- `adjustListeningChildrenOnParent`
  ```java
  void adjustListeningChildrenOnParent(long, int)
  ```

- `applyComponentOrientation`
  ```java
  public void applyComponentOrientation(java.awt.ComponentOrientation)
  ```

- `areFocusTraversalKeysSet`
  ```java
  public boolean areFocusTraversalKeysSet(int)
  ```

- `areInputMethodsEnabled`
  ```java
  boolean areInputMethodsEnabled()
  ```

- `autoProcessMouseWheel`
  ```java
  public void autoProcessMouseWheel(java.awt.event.MouseWheelEvent)
  ```

- `autoTransferFocus`
  ```java
  final void autoTransferFocus(boolean)
  ```

- `bounds`
  ```java
  public Rectangle bounds()
  ```

- `checkGD`
  ```java
  void checkGD(java.lang.String)
  ```

- `checkImage`
  ```java
  public int checkImage(java.awt.Image, java.awt.image.ImageObserver)
  ```

- `checkImage`
  ```java
  public int checkImage(java.awt.Image, int, int, java.awt.image.ImageObserver)
  ```

- `checkWindowClosingException`
  ```java
  boolean checkWindowClosingException()
  ```

- `coalesceEvents`
  ```java
  protected AWTEvent coalesceEvents(java.awt.AWTEvent, java.awt.AWTEvent)
  ```
• `constructComponentName`
  
  `String constructComponentName( )`

• `contains`
  
  `public boolean contains( int , int )`

• `contains`
  
  `public boolean contains( java.awt.Point )`

• `countHierarchyMembers`
  
  `int countHierarchyMembers( )`

• `createBufferStrategy`
  
  `void createBufferStrategy( int )`

• `createBufferStrategy`
  
  `void createBufferStrategy( int , java.awt.BufferCapabilities )`

• `createChildHierarchyEvents`
  
  `void createChildHierarchyEvents( int , long , boolean )`

• `createHierarchyEvents`
  
  `int createHierarchyEvents( int , java.awt.Component , java.awt.Container , long , boolean )`

• `createImage`
  
  `public Image createImage( java.awt.image.ImageProducer )`

• `createImage`
  
  `public Image createImage( int , int )`

• `createVolatileImage`
  
  `public VolatileImage createVolatileImage( int , int )`

• `createVolatileImage`
  
  `public VolatileImage createVolatileImage( int , int , java.awt.ImageCapabilities )`

• `deliverEvent`
  
  `public void deliverEvent( java.awt.Event )`

• `disable`
  
  `public void disable( )`

• `disableEvents`
  
  `protected final void disableEvents( long )`

• `dispatchEvent`
  
  `public final void dispatchEvent( java.awt.AWTEvent )`

• `dispatchEventImpl`
  
  `void dispatchEventImpl( java.awt.AWTEvent )`

• `dispatchMouseWheelToAncestor`
  
  `boolean dispatchMouseWheelToAncestor( java.awt.event.MouseEvent )`

• `doAutoTransfer`
  
  `private void doAutoTransfer( boolean )`

• `doLayout`
  
  `public void doLayout( )`

• `enable`
  
  `public void enable( )`
enable
public void enable( boolean )

enableEvents
protected final void enableEvents( long )

enableInputMethods
public void enableInputMethods( boolean )

eventEnabled
boolean eventEnabled( java.awt.AWTEvent )

eventTypeEnabled
boolean eventTypeEnabled( int )

firePropertyChange
protected void firePropertyChange( java.lang.String , boolean , boolean )

firePropertyChange
protected void firePropertyChange( java.lang.String , int , int )

firePropertyChange
protected void firePropertyChange( java.lang.String , java.lang.Object ,
java.lang.Object )

getAccessibleContext
public AccessibleContext getAccessibleContext( )

getAccessibleIndexInParent
int getAccessibleIndexInParent( )

getAccessibleStateSet
AccessibleStateSet getAccessibleStateSet( )

getAlignmentX
public float getAlignmentX( )

getAlignmentY
public float getAlignmentY( )

getBackground
public Color getBackground( )

getBounds
public Rectangle getBounds( )

getBounds
public Rectangle getBounds( java.awt.Rectangle )

getBufferStrategy
BufferStrategy getBufferStrategy( )

getColorModel
public ColorModel getColorModel( )

getComponentAt
public Component getComponentAt( int , int )

getComponentAt
public Component getComponentAt( java.awt.Point )

getComponentListeners
public synchronized ComponentListener getComponentListeners( )
• `getComponentOrientation`
  public `ComponentOrientation` `getComponentOrientation()`

• `getCursor`
  public `Cursor` `getCursor()`

• `getDropTarget`
  public synchronized `DropTarget` `getDropTarget()`

• `getFocusCycleRootAncestor`
  public `Container` `getFocusCycleRootAncestor()`

• `getFocusListeners`
  public synchronized `FocusListener` `getFocusListeners()`

• `getFocusTraversalKeys_NoIDCheck`
  final `Set` `getFocusTraversalKeys_NoIDCheck(int)`

• `getFocusTraversalKeys`
  public `Set` `getFocusTraversalKeys(int)`

• `getFocusTraversalKeysEnabled`
  public `boolean` `getFocusTraversalKeysEnabled()`

• `getFont_NoClientCode`
  final `Font` `getFont_NoClientCode()`

• `getFont`
  public `Font` `getFont()`

• `getFontMetrics`
  public `FontMetrics` `getFontMetrics(java.awt.Font)`

• `getForeground`
  public `Color` `getForeground()`

• `getGraphics`
  public `Graphics` `getGraphics()`

• `getGraphicsConfiguration`
  public `GraphicsConfiguration` `getGraphicsConfiguration()`

• `getHeight`
  public `int` `getHeight()`

• `getHierarchyBoundsListeners`
  public synchronized `HierarchyBoundsListener` `getHierarchyBoundsListeners()`

• `getHierarchyListeners`
  public synchronized `HierarchyListener` `getHierarchyListeners()`

• `getIgnoreRepaint`
  public `boolean` `getIgnoreRepaint()`

• `getInputContext`
  public `InputContext` `getInputContext()`

• `getInputMethodListeners`
  public synchronized `InputMethodListener` `getInputMethodListeners()`

• `getInputMethodRequests`
  public `InputMethodRequests` `getInputMethodRequests()`
- **getKeyListeners**
  ```java
  public synchronized KeyListener getKeyListeners()
  ```

- **getListeners**
  ```java
  public EventListener getListeners(java.lang.Class)
  ```

- **getLocale**
  ```java
  public Locale getLocale()
  ```

- **getLocation**
  ```java
  public Point getLocation()
  ```

- **getLocation**
  ```java
  public Point getLocation(java.awt.Point)
  ```

- **getLocationOnScreen**
  ```java
  public Point getLocationOnScreen()
  ```

- **getLocationOnScreen**
  ```java
  final Point getLocationOnScreen_NoTreeLock()
  ```

- **getMaximumSize**
  ```java
  public Dimension getMaximumSize()
  ```

- **getMinimumSize**
  ```java
  public Dimension getMinimumSize()
  ```

- **getMouseListeners**
  ```java
  public synchronized MouseListener getMouseListeners()
  ```

- **getMouseMotionListeners**
  ```java
  public synchronized MouseMotionListener getMouseMotionListeners()
  ```

- **getMouseWheelListeners**
  ```java
  public synchronized MouseWheelListener getMouseWheelListeners()
  ```

- **getName**
  ```java
  public String getName()
  ```

- **getNativeContainer**
  ```java
  Container getNativeContainer()
  ```

- **getParent**
  ```java
  public Container getParent()
  ```

- **getPeer**
  ```java
  public ComponentPeer getPeer()
  ```

- **getPreferredSize**
  ```java
  public Dimension getPreferredSize()
  ```

- **getPropertyChangeListeners**
  ```java
  public synchronized PropertyChangeListener getPropertyChangeListeners()
  ```

- **getPropertyChangeListeners**
  ```java
  public synchronized PropertyChangeListener getPropertyChangeListeners(java.lang.String)
  ```

- **getSize**
  ```java
  public Dimension getSize()
  ```
• getSize
  public Dimension getSize( java.awt.Dimension )
• getToolkit
  public Toolkit getToolkit( )
• getToolkitImpl
  final Toolkit getToolkitImpl( )
• getTreeLock
  public final Object getTreeLock( )
• getWidth
  public int getWidth( )
• getX
  public int getX( )
• getY
  public int getY( )
• gotFocus
  public boolean gotFocus( java.awt.Event , java.lang.Object )
• handleEvent
  public boolean handleEvent( java.awt.Event )
• hasFocus
  public boolean hasFocus( )
• hide
  public void hide( )
• imageUpdate
  public boolean imageUpdate( java.awt.Image , int , int , int , int , int )
• initializeFocusTraversalKeys
  void initializeFocusTraversalKeys( )
• initIDs
  private static native void initIDs( )
• inside
  public boolean inside( int , int )
• invalidate
  public void invalidate( )
• isBackgroundSet
  public boolean isBackgroundSet( )
• isCursorSet
  public boolean isCursorSet( )
• isDisplayable
  public boolean isDisplayable( )
• isDoubleBuffered
  public boolean isDoubleBuffered( )
• isEnabled
  public boolean isEnabled( )
- isEnabledImpl
  final boolean isEnabledImpl()
- isFocusable
  public boolean isFocusable()
- isFocusCycleRoot
  public boolean isFocusCycleRoot( java.awt.Container )
- isFocusOwner
  public boolean isFocusOwner()
- isFocusTraversable
  public boolean isFocusTraversable()
- isFocusTraversableOverridden
  final boolean isFocusTraversableOverridden()
- isFontSet
  public boolean isFontSet()
- isForegroundSet
  public boolean isForegroundSet()
- isLightweight
  public boolean isLightweight()
- isOpaque
  public boolean isOpaque()
- isRecursivelyVisible
  boolean isRecursivelyVisible()
- isShowing
  public boolean isShowing()
- isValid
  public boolean isValid()
- isVisible
  public boolean isVisible()
- keyDown
  public boolean keyDown( java.awt.Event , int )
- keyUp
  public boolean keyUp( java.awt.Event , int )
- layout
  public void layout()
- lightweightPaint
  void lightweightPaint( java.awt.Graphics )
- lightweightPrint
  void lightweightPrint( java.awt.Graphics )
- list
  public void list()
- list
  public void list( java.io.PrintStream )
• list
  public void list( java.io.PrintStream , int )
• list
  public void list( java.io.PrintWriter )
• list
  public void list( java.io.PrintWriter , int )
• locate
  public Component locate( int , int )
• location
  public Point location( )
• lostFocus
  public boolean lostFocus( java.awt.Event , java.lang.Object )
• minimumSize
  public Dimension minimumSize( )
• mouseDown
  public boolean mouseDown( java.awt.Event , int , int )
• mouseDrag
  public boolean mouseDrag( java.awt.Event , int , int )
• mouseEnter
  public boolean mouseEnter( java.awt.Event , int , int )
• mouseExit
  public boolean mouseExit( java.awt.Event , int , int )
• mouseMove
  public boolean mouseMove( java.awt.Event , int , int )
• mouseUp
  public boolean mouseUp( java.awt.Event , int , int )
• move
  public void move( int , int )
• nextFocus
  public void nextFocus( )
• nextFocusHelper
  boolean nextFocusHelper( )
• numListening
  int numListening( long )
• paint
  public void paint( java.awt.Graphics )
• paintAll
  public void paintAll( java.awt.Graphics )
• paintHeavyweightComponents
  void paintHeavyweightComponents( java.awt.Graphics )
• paramString
  protected String paramString( )
• \textit{postEvent}
  \texttt{public boolean postEvent( java.awt.Event )}

• \textit{postsOldMouseEvents}
  \texttt{boolean postsOldMouseEvents()}

• \textit{preferredSize}
  \texttt{public Dimension preferredSize()}

• \textit{prepareImage}
  \texttt{public boolean prepareImage( java.awt.Image , java.awt.image.ImageObserver )}

• \texttt{prepareImage}
  \texttt{public boolean prepareImage( java.awt.Image , int , int , java.awt.image.ImageObserver )}

• \textit{print}
  \texttt{public void print( java.awt.Graphics )}

• \texttt{printAll}
  \texttt{public void printAll( java.awt.Graphics )}

• \textit{printHeavyweightComponents}
  \texttt{void printHeavyweightComponents( java.awt.Graphics )}

• \textit{processComponentEvent}
  \texttt{protected void processComponentEvent( java.awt.event.ComponentEvent )}

• \texttt{processEvent}
  \texttt{protected void processEvent( java.awt.AWTEvent )}

• \texttt{processFocusEvent}
  \texttt{protected void processFocusEvent( java.awt.event.FocusEvent )}

• \texttt{processHierarchyBoundsEvent}
  \texttt{protected void processHierarchyBoundsEvent( java.awt.event.HierarchyEvent )}

• \texttt{processHierarchyEvent}
  \texttt{protected void processHierarchyEvent( java.awt.event.HierarchyEvent )}

• \texttt{processInputMethodEvent}
  \texttt{protected void processInputMethodEvent( java.awt.event.InputMethodEvent )}

• \texttt{processKeyEvent}
  \texttt{protected void processKeyEvent( java.awt.event.KeyEvent )}

• \texttt{processMouseEvent}
  \texttt{protected void processMouseEvent( java.awt.event.MouseEvent )}

• \texttt{processMouseMotionEvent}
  \texttt{protected void processMouseMotionEvent( java.awt.event.MouseEvent )}

• \texttt{processMouseWheelEvent}
  \texttt{protected void processMouseWheelEvent( java.awt.event.MouseWheelEvent )}

• \textit{readObject}
  \texttt{private void readObject( java.io.ObjectInputStream )}

• \texttt{remove}
  \texttt{public synchronized void remove( java.awt.MenuComponent )}

• \texttt{removeComponentListener}
  \texttt{public synchronized void removeComponentListener( java.awt.event.ComponentListener )}
• `removeFocusListener`
  public synchronized void `removeFocusListener` ( `java.awt.event.FocusListener` )

• `removeHierarchyBoundsListener`
  public void `removeHierarchyBoundsListener` ( `java.awt.event.HierarchyBoundsListener` )

• `removeHierarchyListener`
  public void `removeHierarchyListener` ( `java.awt.event.HierarchyListener` )

• `removeInputMethodListener`
  public synchronized void `removeInputMethodListener` ( `java.awt.event.InputMethodListener` )

• `removeKeyListener`
  public synchronized void `removeKeyListener` ( `java.awt.event.KeyListener` )

• `removeMouseListener`
  public synchronized void `removeMouseListener` ( `java.awt.event.MouseListener` )

• `removeMouseMotionListener`
  public synchronized void `removeMouseMotionListener` ( `java.awt.event.MouseMotionListener` )

• `removeMouseWheelListener`
  public synchronized void `removeMouseWheelListener` ( `java.awt.event.MouseWheelListener` )

• `removeNotify`
  public void `removeNotify` ( )

• `removePropertyChangeListener`
  public synchronized void `removePropertyChangeListener` ( `java.beans.PropertyChangeListener` )

• `removePropertyChangeListener`
  public synchronized void `removePropertyChangeListener` ( `java.lang.String`, `java.beans.PropertyChangeListener` )

• `repaint`
  public void `repaint` ( )

• `repaint`
  public void `repaint` ( `int`, `int`, `int`, `int` )

• `repaint`
  public void `repaint` ( `long` )

• `repaint`
  public void `repaint` ( `long`, `int`, `int`, `int`, `int` )

• `requestFocus`
  public void `requestFocus` ( )

• `requestFocus`
  protected boolean `requestFocus` ( `boolean` )

• `requestFocusHelper`
  final boolean `requestFocusHelper` ( `boolean`, `boolean` )

• `requestFocusInWindow`
  public boolean `requestFocusInWindow` ( )
• requestFocusInWindow
  protected boolean requestFocusInWindow( boolean )
• resetGC
  void resetGC( )
• reshape
  public void reshape( int, int, int, int )
• resize
  public void resize( java.awt.Dimension )
  public void resize( int, int )
• setBackground
  public void setBackground( java.awt.Color )
• setBounds
  public void setBounds( int, int, int, int )
  public void setBounds( java.awt.Rectangle )
• setComponentOrientation
  public void setComponentOrientation( java.awt.ComponentOrientation )
• setCursor
  public void setCursor( java.awt.Cursor )
• setDropTarget
  public synchronized void setDropTarget( java.awt.dnd.DropTarget )
• setEnabled
  public void setEnabled( boolean )
• setFocusable
  public void setFocusable( boolean )
• setFocusTraversalKeys_NoIDCheck
  final void setFocusTraversalKeys_NoIDCheck( int, java.util.Set )
• setFocusTraversalKeys
  public void setFocusTraversalKeys( int, java.util.Set )
• setFocusTraversalKeysEnabled
  public void setFocusTraversalKeysEnabled( boolean )
• setFont
  public void setFont( java.awt.Font )
• setForeground
  public void setForeground( java.awt.Color )
• setGCFromPeer
  void setGCFromPeer( )
• setIgnoreRepaint
  public void setIgnoreRepaint( boolean )
• setLocale
  public void setLocale( java.util.Locale )
• setLocation
  public void setLocation( int , int )

• setLocation
  public void setLocation( java.awt.Point )

• setName
  public void setName( java.lang.String )

• setSize
  public void setSize( java.awt.Dimension )

• setSize
  public void setSize( int , int )

• setVisible
  public void setVisible( boolean )

• show
  public void show( )

• show
  public void show( boolean )

• size
  public Dimension size( )

• toString
  public String toString( )

• transferFocus
  public void transferFocus( )

• transferFocusBackward
  public void transferFocusBackward( )

• transferFocusUpCycle
  public void transferFocusUpCycle( )

• update
  public void update( java.awt.Graphics )

• updateCursorImmediately
  final void updateCursorImmediately( )

• validate
  public void validate( )

• writeObject
  private void writeObject( java.io.ObjectOutputStream )

\[ 1/(s+a) \]
### Declaration

```java
public class Sim1stOrder extends simtk.SimZtf implements SimPanelable
```

### Constructors

- **Sim1stOrder**
  ```java
  public Sim1stOrder()
  ```
  - **Usage**
    - * constructor set a to 1 so default of 1/(s+a) is 1/(s+1)

### Methods

- **reDefine**
  ```java
  public void reDefine()
  ```
  - **Usage**
    - * this is a method that might be good to have defined in the sim Block as abstract
      this is made to recalculate with new parameters, step sizes parameters overrides the
doIteration of SimZtf and loads in the new parameters then it calls the overridden
doIteration from Ztf

- **setA**
  ```java
  public void setA(double a)
  ```
  - **Usage**
    - * set the parameter a in 1/(s+a)

### Methods inherited from class simtk.SimZtf

- **doIteration**
  ```java
  public void doIteration()
  ```
  - **Usage**
    - * this method overrides the doIteration of the diffeq. this way it first redefines the variables
      and then iterates with the inherited structure
• **doIterationDirect**

  public void doIterationDirect( )

  - **Usage**
    * this is the really cool simple method that converts ztf to diffEQ. The function does the operation in the following order:
      1. parses the numerator and denominator
      2. creates the difference equation
      3. does one step operation.

• **reDefine**

  public void reDefine( )

  - **Usage**
    * this is a method that might be good to have defined in the sim Block as abstract this is made to recalculate with new parameters, step sizes parameters

• **setDen**

  public void setDen( double [] n )

  - **Usage**
    * method to set the denominator to the array value Example: setDen([1,2,3]); would set the denominator to z^2 + 2*z + 3 an error message is displayed if the array is lager than the z order

• **setNum**

  public void setNum( double [] n )

  - **Usage**
    * method to set the numerator to the array value Example: setNum([1,2,3]); would set the numerator to z^2 + 2*z + 3 an error message is displayed if the array is lager than the z order

• **toString**

  public String toString( )

  - **Usage**
    * toString in the form of: zorder = x num = [x] den = [x]

• **zeroNumAndDen**

  public void zeroNumAndDen( )

  - **Usage**
    * this method is for convenience in setting up z transfer functions it makes sure that the function is reset to zero from the initial conditions
Methods inherited from class simtk.SimDiffEq

- **clear**
  public void clear( )
  - Usage
    * clears the text field, sets up headers and resets the index counter

- **doIteration**
  public void doIteration( )

- **getInCoef**
  public SimDoubleArray getInCoef( )
  - Usage
    * return the input coefficients

- **getOutCoef**
  public SimDoubleArray getOutCoef( )
  - Usage
    * return the output coefficients

- **setZeros**
  public void setZeros( )
  - Usage
    * zero all the coefficients for easy settings

- **toDiffEqString**
  public String toDiffEqString( )
  - Usage
    * output the string format of the difference equation. special function to iterate through all of the parameters and create the difference equation. This function is also used for the textBox for the dynamic pretty print viewing of the diff eq

- **toString**
  public String toString( )
  - Usage
    * general to String equation for the diff eq. this uses the toDiffEqString() function for the special formatting so that one doesn’t overloaded
Methods inherited from class simtk.SimFunction

- **addInput**
  public void addInput( simtk.HasOutput outputBlock )
  - Usage
    * this AddInput adds from a HasOutput block an input to the vector function and increases the inputIndex count

- **addInput**
  public void addInput( simtk.SimBuffer bufref )
  - Usage
    * AddInput adds an input to the vector function and increases the inputIndex count

- **doIteration**
  abstract void doIteration( )
  - Usage
    * this is the method that should be implemented in all SimBlock’s

- **getInput**
  public SimBuffer getInput( )
  - Usage
    * get the first input. there might be a problem here because the old method of the implicit in buffer is used

- **getInput**
  public SimBuffer getInput( int index )
  - Usage
    * get the indexed input, referenced from as 1 is the first.
    If the reference is out of range it will return the the single input value.
    The code is not so clean here where it will still print an erroo message when the double input functions are just beginning to set themselves up and before the.???

- **getInputCount**
  public int getInputCount( )
  - Usage
    * get the inputCount, or the number of elements on the list

- **getOutput**
  public SimBuffer getOutput( )
  - Usage
    * Gets the output buffer

- **iterateBuffers**
  public void iterateBuffers( )
- Usage
  * overload the do nothing iterateBuffers and iterate the output buffers

- removeInputs
  public void removeInputs()
  
  - Usage
    * remove the all of the inputs from the block so a new system can be set up

- setInput
  public void setInput(int index, simtk.SimBuffer bufref)
  
  - Usage
    * Set the indexed input to the passed buffer. note: this has out of index problems

- setInput
  public void setInput(simtk.SimBuffer bufref)
  
  - Usage
    * Set the input to the passed buffer, this is replaced with the newer addInput function which
      sounds more like what it does

- setInput1
  public void setInput1(simtk.SimBuffer bufref)

  - Usage
    * This is here to support old systems, it just calls addInput(bufref). so it is not exactly
      input 2

- setInput2
  public void setInput2(simtk.SimBuffer bufref)

  - Usage
    * This is here to support old systems, it just calls addInput(bufref). so it is not exactly
      input 2

- setOutput
  public void setOutput(simtk.SimBuffer bufref)

  - Usage
    * sets the output to a passed buffer

Methods inherited from class simtk.SimBlock

- clear
  public void clear()

  - Usage
    * this is the general function to clear a block. this is just an empty method but if more
      specific functionality is needed this should be overridden
• **doIteration**  
  abstract void **doIteration**();  
  
  - **Usage**  
    * this is the method that should be implemented in all Simblock’s

• **doOutputBufferIteration**  
  public void **doOutputBufferIteration**();  
  
  - **Usage**  
    * iterate the output buffer(s), or shift the history of it

• **getPanel**  
  public JPanel **getPanel**();  
  
  - **Usage**  
    * generates the default panel and the title

• **getSimParams**  
  public SimParameters **getSimParams**();  
  
  - **Usage**  
    * get the parameters from a parameter class

• **getVariableContainer**  
  public VariableContainer **getVariableContainer**();  
  
  - **Usage**  
    * returns the container for all the variables. this is mainly used for the operations done on the varContainer

• **isHidden**  
  public boolean **isHidden**();  
  
  - **Usage**  
    * test if the hidden flag has been set

• **iterateBuffers**  
  public void **iterateBuffers**();  
  
  - **Usage**  
    * This function does nothing, but blocks with outputs or internal buffers such as SimFunctionFactory should overload it for it’s own special purposes

• **reDefine**  
  public void **reDefine**();  
  
  - **Usage**  
    * transfer upper level adjustments to the lower level. this method is overloaded when it is to be used by a lower function

• **setHidden**  
  public void **setHidden**();
– **Usage**
  * keep the block from being loaded to the factory display if it is panelable

- **setHidden**
  ```java
  public void setHidden( boolean hide )
  ```
  – **Usage**
  * keep the block from being loaded to the factory display if it is panelable

- **setSimParams**
  ```java
  public void setSimParams( simtk.SimParameters sp )
  ```
  – **Usage**
  * set the parameters from a parameter class

### Methods inherited from class `simtk.SimBase`

- **getDescription**
  ```java
  public String getDescription( )
  ```

- **getReloadBlock**
  ```java
  public Simulatable getReloadBlock( )
  ```
  – **Usage**
  * set the block that gets called to resimulate when the a variable is changed.

- **getTitle**
  ```java
  public String getTitle( )
  ```
  – **Usage**
  * get the user defined title for the entire block

- **setDescription**
  ```java
  public void setDescription( java.lang.String desc )
  ```

- **setReloadBlock**
  ```java
  public void setReloadBlock( simtk.Simulatable reloadBlock )
  ```
  – **Usage**
  * set the block that gets called to resimulate when the a variable is changed.

- **setTitle**
  ```java
  public void setTitle( java.lang.String title )
  ```
  – **Usage**
  * set the user defined title for the entire block

#### 2.2.4 Class Sim2ndOrder

\[(w^2)/(s^2 + 2wzs + w^2)\]
public class Sim2ndOrder
extends simtk.SimZtf
implements SimPanelable

Constructors

- **Sim2ndOrder**
  public Sim2ndOrder(  )
  
  - Usage
    * constructor set \( \omega_0 = 1 \) and \( \zeta = 1 \) \( \frac{(w^2)}{(s^2 + 2wz + w^2)} \) set a to 1
      so default of \( \frac{1}{(s+a)} \) is \( \frac{1}{(s+1)} \)

Methods

- **reDefine**
  public void reDefine(  )
  
  - Usage
    * this is a method that might be good to have defined in the sim Block as abstract
      this is made to recalculate with new parameters, step sizes parameters

- **setOmega**
  public void setOmega( double omega )
  
  - Usage
    * set omega

- **setZeta**
  public void setZeta( double zeta )
  
  - Usage
    * set zeta
Methods inherited from class simtk.SimZtf

- **doIteration**
  public void doIteration()
  - Usage
    * this method overrides the doIteration of the diffeq. this way it first redefines the variables and then iterates with the inherited structure

- **doIterationDirect**
  public void doIterationDirect()
  - Usage
    * this is the really cool simple method that converts ztf to diffEQ. The function does the operation in the following order:
      1. parses the numerator and denominator
      2. creates the difference equation
      3. does one step operation.

- **reDefine**
  public void reDefine()
  - Usage
    * this is a method that might be good to have defined in the sim Block as abstract this is made to recalculate with new parameters, step sizes parameters

- **setDen**
  public void setDen(double [] n)
  - Usage
    * method to set the denominator to the array value Example: setDen([1,2,3]); would set the denominator to \(z^2 + 2z + 3\) an error message is displayed if the array is larger than the \(z\) order

- **setNum**
  public void setNum(double [] n)
  - Usage
    * method to set the numerator to the array value Example: setNum([1,2,3]); would set the numerator to \(z^2 + 2z + 3\) an error message is displayed if the array is larger than the \(z\) order

- **toString**
  public String toString()
  - Usage
    * toString in the form of: \(z\)order = \(x\) num = \([x]\) den = \([x]\)

- **zeroNumAndDen**
  public void zeroNumAndDen()
  - Usage
    * this method is for convenience in setting up z transfer functions it makes sure that the function is reset to zero from the initial conditions
Methods inherited from class simtk.SimDiffEq

- **clear**
  public void clear( )
  - Usage
    * clears the text field, sets up headers and resets the index counter

- **doIteration**
  public void doIteration( )

- **getInCoef**
  public SimDoubleArray getInCoef( )
  - Usage
    * return the input coefficients

- **getOutCoef**
  public SimDoubleArray getOutCoef( )
  - Usage
    * return the output coefficients

- **setZeros**
  public void setZeros( )
  - Usage
    * zero all the coefficients for easy settings

- **toDiffEqString**
  public String toDiffEqString( )
  - Usage
    * output the string format of the difference equation. special function to iterate through all of the parameters and create the difference equation. This function is also used for the textBox for the dynamic pretty print viewing of the diff eq

- **toString**
  public String toString( )
  - Usage
    * general to String equation for the diff eq. this uses the toDiffEqString() function for the special formatting so that one doesn’t overloaded
Methods inherited from class simtk.SimFunction

- **addInput**
  ```java
  public void addInput( simtk.HasOutput outputBlock )
  ```
  - **Usage**
    * this AddInput adds from a HasOutput block an input to the vector function and increases the inputIndex count

- **addInput**
  ```java
  public void addInput( simtk.SimBuffer bufref )
  ```
  - **Usage**
    * AddInput adds an input to the vector function and increases the inputIndex count

- **doIteration**
  ```java
  abstract void doIteration( )
  ```
  - **Usage**
    * this is the method that should be implemented in all SimBlock’s

- **getInput**
  ```java
  public SimBuffer getInput( )
  ```
  - **Usage**
    * get the first input. there might be a problem here because the old method of the implicit in buffer is used

- **getInput**
  ```java
  public SimBuffer getInput( int index )
  ```
  - **Usage**
    * get the indexed input, referenced from as 1 is the first.
    If the reference is out of range it will return the the single input value.
    The code is not so clean here where it will still print an erroo message when the double input functions are just beginning to set themselves up and before the.???

- **getInputCount**
  ```java
  public int getInputCount( )
  ```
  - **Usage**
    * get the inputCount, or the number of elements on the list

- **getOutput**
  ```java
  public SimBuffer getOutput( )
  ```
  - **Usage**
    * Gets the output buffer

- **iterateBuffers**
  ```java
  public void iterateBuffers( )
  ```
Methods inherited from class simtk.SimBlock

• clear
  public void clear( )
  – Usage
    * this is the general function to clear a block. this is just an empty method but if more specific functionality is needed this should be overridden
• **doIteration**
  
  ```
  abstract void doIteration()
  ```
  
  - **Usage**
    * this is the method that should be implemented in all Simblock’s

• **doOutputBufferIteration**

  ```
  public void doOutputBufferIteration()
  ```
  
  - **Usage**
    * iterate the output buffer(s), or shift the history of it

• **getPanel**

  ```
  public JPanel getPanel()
  ```
  
  - **Usage**
    * generates the default panel and the title

• **getSimParams**

  ```
  public SimParameters getSimParams()
  ```
  
  - **Usage**
    * get the parameters from a parameter class

• **getVariableContainer**

  ```
  public VariableContainer getVariableContainer()
  ```
  
  - **Usage**
    * returns the container for all the variables. this is mainly used for the operations done on the varContainer

• **isHidden**

  ```
  public boolean isHidden()
  ```
  
  - **Usage**
    * test if the hidden flag has been set

• **iterateBuffers**

  ```
  public void iterateBuffers()
  ```
  
  - **Usage**
    * This function does nothing, but blocks with outputs or internal buffers such as SimFunctionFactory should overload it for it’s own special purposes

• **reDefine**

  ```
  public void reDefine()
  ```
  
  - **Usage**
    * transfer upper level adjustments to the lower level. this method is overloaded when it is to be used by a lower function

• **setHidden**

  ```
  public void setHidden()
  ```
- Usage
  * keep the block from being loaded to the factory display if it is panelable

- `setHidden`
  ```java
  public void setHidden(boolean hide)
  ```
  - Usage
    * keep the block from being loaded to the factory display if it is panelable

- `setSimParams`
  ```java
  public void setSimParams(simtk.SimParameters sp)
  ```
  - Usage
    * set the parameters from a parameter class

**Methods inherited from class `simtk.SimBase`**

- `getDescription`
  ```java
  public String getDescription()
  ```
- `getReloadBlock`
  ```java
  public Simulatable getReloadBlock()
  ```
  - Usage
    * set the block that gets called to resimulate when the a variable is changed.

- `getTitle`
  ```java
  public String getTitle()
  ```
  - Usage
    * get the user defined title for the entire block

- `setDescription`
  ```java
  public void setDescription(java.lang.String desc)
  ```
- `setReloadBlock`
  ```java
  public void setReloadBlock(simtk.Simulatable reloadBlock)
  ```
  - Usage
    * set the block that gets called to resimulate when the a variable is changed.

- `setTitle`
  ```java
  public void setTitle(java.lang.String title)
  ```
  - Usage
    * set the user defined title for the entire block

## 2.2.5 Class SimAdd

adds multiple inputs to a single output
public class SimAdd
extends simtk.SimFunction

Constructors

- SimAdd
  public SimAdd( )

Methods

- doIteration
  public void doIteration( )
  - Usage
    * required iteration to define for this class type which does the work

Methods inherited from class simtk.SimFunction

- addInput
  public void addInput( simtk.HasOutput outputBlock )
  - Usage
    * this AddInput adds from a HasOutput block an input to the vector function and increases the inputIndex count

- addInput
  public void addInput( simtk.SimBuffer bufref )
  - Usage
    * AddInput adds an input to the vector function and increases the inputIndex count

- doIteration
  abstract void doIteration( )
  - Usage
    * this is the method that should be implemented in all SimBlock’s

- getInput
  public SimBuffer getInput( )
  - Usage
get the first input, there might be a problem here because the old method of the implicit in buffer is used

• `getInput`
  public SimBuffer getInput( int index )
  - Usage
    * get the indexed input, referenced from as 1 is the first.
    If the reference is out of range it will return the single input value.
    The code is not so clean here where it will still print an error message when the double input functions are just beginning to set themselves up and before the.

• `getInputCount`
  public int getInputCount( )
  - Usage
    * get the inputCount, or the number of elements on the list

• `getOutput`
  public SimBuffer getOutput( )
  - Usage
    * Gets the output buffer

• `iterateBuffers`
  public void iterateBuffers( )
  - Usage
    * overload the do nothing iterateBuffers and iterate the output buffers

• `removeInputs`
  public void removeInputs( )
  - Usage
    * remove the all of the inputs from the block so a new system can be set up

• `setInput`
  public void setInput( int index, simtk.SimBuffer bufref )
  - Usage
    * Set the indexed input to the passed buffer. note: this has out of index problems

• `setInput`
  public void setInput( simtk.SimBuffer bufref )
  - Usage
    * Set the input to the passed buffer, this is replaced with the newer addInput function which sounds more like what it does

• `setInput1`
  public void setInput1( simtk.SimBuffer bufref )
  - Usage
* This is here to support old systems, it just calls addInput(bufref). so it is not exactly input 2

- **setInput2**
  public void setInput2( simtk.SimBuffer bufref )
  
  Usage
  * This is here to support old systems, it just calls addInput(bufref). so it is not exactly input 2

- **setOutput**
  public void setOutput( simtk.SimBuffer bufref )
  
  Usage
  * sets the output to a passed buffer

**Methods inherited from class simtk.SimBlock**

- **clear**
  public void clear( )
  
  Usage
  * this is the general function to clear a block. this is just an empty method but if more specific functionality is needed this should be overridden

- **doIteration**
  abstract void doIteration( )
  
  Usage
  * this is the method that should be implemented in all Simblock's

- **doOutputBufferIteration**
  public void doOutputBufferIteration( )
  
  Usage
  * iterate the output buffer(s), or shift the history of it

- **getPanel**
  public JPanel getPanel( )
  
  Usage
  * generates the default panel and the title

- **getSimParams**
  public SimParameters getSimParams( )
  
  Usage
  * get the parameters from a parameter class

- **getVariableContainer**
  public VariableContainer getVariableContainer( )
- **Usage**  
  * returns the container for all the variables. This is mainly used for the operations done on the varContainer

- **isHidden**  
  public boolean isHidden()  
  - **Usage**  
    * test if the hidden flag has been set

- **iterateBuffers**  
  public void iterateBuffers()  
  - **Usage**  
    * This function does nothing, but blocks with outputs or internal buffers such as SimFunctionFactory should overload it for its own special purposes

- **reDefine**  
  public void reDefine()  
  - **Usage**  
    * transfer upper level adjustments to the lower level. This method is overloaded when it is to be used by a lower function

- **setHidden**  
  public void setHidden()  
  - **Usage**  
    * keep the block from being loaded to the factory display if it is panelable

- **setHidden**  
  public void setHidden(boolean hide)  
  - **Usage**  
    * keep the block from being loaded to the factory display if it is panelable

- **setSimParams**  
  public void setSimParams(simtk.SimParameters sp)  
  - **Usage**  
    * set the parameters from a parameter class

**Methods inherited from class** simtk.SimBase

- **getDescription**  
  public String getDescription()  

- **getReloadBlock**  
  public Simulatable getReloadBlock()  
  - **Usage**  
    * set the block that gets called to resimulate when the a variable is changed.
• **getTitle**
  
  public String getTitle()
  
  - Usage
  
  * get the user defined title for the entire block

• **setDescription**
  
  public void setDescription( java.lang.String desc )

• **setReloadBlock**
  
  public void setReloadBlock( simtk.Simulatable reloadBlock )
  
  - Usage
  
  * set the block that gets called to resimulate when the a variable is changed.

• **setTitle**
  
  public void setTitle( java.lang.String title )
  
  - Usage
  
  * set the user defined title for the entire block

### 2.2.6 Class SimADRC

Compilation of sim blocks to create the ADRC block

#### Declaration

```java
public class SimADRC
    extends simtk.SimFunctionFactory
    implements SimPanelable, ContainsFactory
```

#### Fields

• private SimNumber omega_c
  
• private SimNumber omega_o
  
• private SimNumber b_zero
  
• private boolean debug
Constructors

- **SimADRC**
  
  ```java
  public SimADRC()
  ```

  - **Usage**
    *
    * constructor sets gain to default of unity

Methods

- **connectBlocks**
  
  ```java
  public void connectBlocks()
  ```

  - **Usage**
    *
    * Make the connections for this block. The setInput of this factory calls this to setup. It turns out to be a rather interesting system the setInput to this factory then in turn setups up this internal system.

    This function has problems due to the multiple inputs, which each call this connectBlocks() function, and in turn multiple connectons are made to each of the contained Blocks

    The factory.epmty() function should probably also clear the inputs from the contained functions aswell.

- **getPanel**
  
  ```java
  public JPanel getPanel()
  ```

  - **Usage**
    *
    * override the default and previous dialog panels

- **reDefine**
  
  ```java
  public void reDefine()
  ```

  - **Usage**
    *
    * the super magic code. These transfer functions were derived by Rob Milosovic from the state space seperated controller to a combined transfer funtion

Methods inherited from class simtk.SimFunctionFactory

- **addInput**
  
  ```java
  public void addInput( simtk.HasOutput inputBlock )
  ```
### `addInput`

```java
public void addInput( simtk.SimBuffer sb )
```

- **Usage**
  * set the input of this factory. this is the interesting one

### `clear`

```java
public void clear( )
```

- **Usage**
  * this function is required to reset the buffers and values for the next iterations to set up the factory class

### `connectBlocks`

```java
abstract void connectBlocks( )
```

- **Usage**
  * Make the connections for this block. The setInput of this factory calls this to setup. It turns out to be a rather interesting system. The setInput() or addInput() to this factory then in turn set's up this internal system. this is to be implemented.

### `doIteration`

```java
public void doIteration( )
```

- **Usage**
  * do a calculation for the new output

### `getPanel`

```java
public JPanel getPanel( )
```

### `iterateBuffers`

```java
public void iterateBuffers( )
```

- **Usage**
  * cycle the buffers

### `setInput`

```java
public void setInput( simtk.SimBuffer sb )
```

- **Usage**
  * set the input of this factory. this is to be implemented in the Simblock to call the connect blocks

### `toString`

```java
public String toString( )
```
Methods inherited from class `simtk.SimFunction`

- **addInput**
  ```java
  public void addInput( simtk.HasOutput outputBlock )
  ```
  - **Usage**
    * this AddInput adds from a HasOutput block an input to the vector function and increases the inputIndex count

- **addInput**
  ```java
  public void addInput( simtk.SimBuffer bufref )
  ```
  - **Usage**
    * AddInput adds an input to the vector function and increases the inputIndex count

- **doIteration**
  ```java
  abstract void doIteration( )
  ```
  - **Usage**
    * this is the method that should be implemented in all SimBlock’s

- **getInput**
  ```java
  public SimBuffer getInput( )
  ```
  - **Usage**
    * get the first input. there might be a problem here because the old method of the implicit in buffer is used

- **getInput**
  ```java
  public SimBuffer getInput( int index )
  ```
  - **Usage**
    * get the indexed input, referenced from as 1 is the first.
      If the reference is out of range it will return the the single input value.
      The code is not so clean here where it will still print an error message when the double input functions are just beginning to set themselves up and before the.???

- **getInputCount**
  ```java
  public int getInputCount( )
  ```
  - **Usage**
    * get the inputCount, or the number of elements on the list

- **getOutput**
  ```java
  public SimBuffer getOutput( )
  ```
  - **Usage**
    * Gets the output buffer

- **iterateBuffers**
  ```java
  public void iterateBuffers( )
  ```
• **Usage**
  * overload the do nothing iterateBuffers and iterate the output buffers

**removeInputs**
public void removeInputs( )

• **Usage**
  * remove the all of the inputs from the block so a new system can be set up

**setInput**
public void setInput( int index, simtk.SimBuffer bufref )

• **Usage**
  * Set the indexed input to the passed buffer. note: this has out of index problems

**setInput**
public void setInput( simtk.SimBuffer bufref )

• **Usage**
  * Set the input to the passed buffer, this is replaced with the newer addInput function which sounds more like what it does

**setInput1**
public void setInput1( simtk.SimBuffer bufref )

• **Usage**
  * This is here to support old systems, it just calls addInput(bufref). so it is not exactly input

**setInput2**
public void setInput2( simtk.SimBuffer bufref )

• **Usage**
  * This is here to support old systems, it just calls addInput(bufref). so it is not exactly input

**setOutput**
public void setOutput( simtk.SimBuffer bufref )

• **Usage**
  * sets the output to a passed buffer

**Methods inherited from class simtk.SimBlock**

• **clear**
  public void clear( )

• **Usage**
  * this is the general function to clear a block. this is just an empty method but if more specific functionality is needed this should be overridden
- **doIteration**
  abstract void doIteration( )
  
  - **Usage**
    * this is the method that should be implemented in all Simblock’s

- **doOutputBufferIteration**
  public void doOutputBufferIteration( )
  
  - **Usage**
    * iterate the output buffer(s), or shift the history of it

- **getPanel**
  public JPanel getPanel( )
  
  - **Usage**
    * generates the default panel and the title

- **getSimParams**
  public SimParameters getSimParams( )
  
  - **Usage**
    * get the parameters from a parameter class

- **getVariableContainer**
  public VariableContainer getVariableContainer( )
  
  - **Usage**
    * returns the container for all the variables. this is mainly used for the operations done on the varContainer

- **isHidden**
  public boolean isHidden( )
  
  - **Usage**
    * test if the hidden flag has been set

- **iterateBuffers**
  public void iterateBuffers( )
  
  - **Usage**
    * This function does nothing, but blocks with outputs or internal buffers such as SimFunctionFactory should overload it for it’s own special purposes

- **reDefine**
  public void reDefine( )
  
  - **Usage**
    * transfer upper level adjustments to the lower level. this method is overloaded when it is to be used by a lower function

- **setHidden**
  public void setHidden( )
- **Usage**
  * keep the block from being loaded to the factory display if it is panelable

- **setHidden**
  public void setHidden( boolean hide )
  - **Usage**
    * keep the block from being loaded to the factory display if it is panelable

- **setSimParams**
  public void setSimParams( simtk.SimParameters sp )
  - **Usage**
    * set the parameters from a parameter class

**Methods inherited from class simtk.SimBase**

- **getDescription**
  public String getDescription( )

- **getReloadBlock**
  public Simulatable getReloadBlock( )
  - **Usage**
    * set the block that gets called to resimulate when the a variable is changed.

- **getTitle**
  public String getTitle( )
  - **Usage**
    * get the user defined title for the entire block

- **setDescription**
  public void setDescription( java.lang.String desc )

- **setReloadBlock**
  public void setReloadBlock( simtk.Simulatable reloadBlock )
  - **Usage**
    * set the block that gets called to resimulate when the a variable is changed.

- **setTitle**
  public void setTitle( java.lang.String title )
  - **Usage**
    * set the user defined title for the entire block

**2.2.7 Class SimBase**

This is the main abstract block for many of the classes
simtk– SimBase

Declaration

public class SimBase
extends java.lang.Object

Fields

• private String title
  – holds the title of this block

• private String description
  –

• private Simulatable reloadBlock
  – the reload block for a simulation to call the do simulation of

Constructors

• SimBase
  public SimBase( )

• SimBase
  public SimBase( simtk.Simulatable reloadBlock )

Methods

• getDescription
  public String getDescription( )

• getReloadBlock
  public Simulatable getReloadBlock( )

  – Usage
    * set the block that gets called to resimulate when the a variable is changed.

• getTitle
  public String getTitle( )

  – Usage
2.2.8 **CLASS SimBlock**

This is the main abstract block for all types of Sim

**DECLARATION**

```java
public abstract class SimBlock
extends simtk.SimBase
```

**FIELDS**

- private boolean multi_changed
  - flag to let the system know that a multiparameter has changed such as wc
- private boolean hidden
  - flag to keep the block from the Panel being loaded to a factory display
- private SimParameters simParams
- private VariableContainer varContainer
• private JTabbedPane tabPane
  
• public SimDialogManager dlg_maker
  
CONSTRUCTORS

• SimBlock
  
SimBlock( )
  
  – Usage
  * used for the deprecated functions initialize defaluts zorder = 4, tfinal = 10, stepsize = 1

METHODS

• clear
  
public void clear( )
  
  – Usage
  * this is the general function to clear a block. this is just an empty method but if more specific functionality is needed this should be overridden

• doIteration
  
abstract void doIteration( )
  
  – Usage
  * this is the method that should be implemented in all Simblock’s

• doOutputBufferIteration
  
public void doOutputBufferIteration( )
  
  – Usage
  * iterate the output buffer(s), or shift the history of it

• getPanel
  
public JPanel getPanel( )
  
  – Usage
  * generates the default panel and the title
• **getSimParams**
  public SimParameters getSimParams( )
  
  — Usage
  * get the parameters from a parameter class

• **getVariableContainer**
  public VariableContainer getVariableContainer( )
  
  — Usage
  * returns the container for all the variables. this is mainly used for the operations done on the varContainer

• **isHidden**
  public boolean isHidden( )
  
  — Usage
  * test if the hidden flag has been set

• **iterateBuffers**
  public void iterateBuffers( )
  
  — Usage
  * This function does nothing, but blocks with outputs or internal buffers such as SimFunctionFactory should overload it for its own special purposes

• **reDefine**
  public void reDefine( )
  
  — Usage
  * transfer upper level adjustments to the lower level. this method is overloaded when it is to be used by a lower function

• **setHidden**
  public void setHidden( )
  
  — Usage
  * keep the block from being loaded to the factory display if it is panelable

• **setHidden**
  public void setHidden( boolean hide )
  
  — Usage
  * keep the block from being loaded to the factory display if it is panelable
• **setSimParams**
  
  public void setSimParams( simtk.SimParameters sp )
  
  - Usage
    * set the parameters from a parameter class

### Methods inherited from class `simtk.SimBase`

- **getDescription**
  
  public String getDescription( )

- **getReloadBlock**
  
  public Simulatable getReloadBlock( )
  
  - Usage
    * set the block that gets called to resimulate when the a variable is changed.

- **getTitle**
  
  public String getTitle( )
  
  - Usage
    * get the user defined title for the entire block

- **setDescription**
  
  public void setDescription( java.lang.String desc )

- **setReloadBlock**
  
  public void setReloadBlock( simtk.Simulatable reloadBlock )
  
  - Usage
    * set the block that gets called to resimulate when the a variable is changed.

- **setTitle**
  
  public void setTitle( java.lang.String title )
  
  - Usage
    * set the user defined title for the entire block

### 2.2.9 Class SimBuffer

This is the buffer block for the Sim environment. This is the glue that connects all the blocks together. It also has the doIteration function to shift the history of the buffer.

#### Declaration

```java
public class SimBuffer
  extends simtk.SimBlock
```
Fields

- public double val
  - the double array that holds the history. Example val[0] is the current time and val[2] would be the value of the buffer 2 iterations ago.

Constructors

- SimBuffer
  public SimBuffer()
  - Usage
    * default constructor calls the previous constructor

- SimBuffer
  public SimBuffer( int zorder )
  - Usage
    * constructor no output not even void I am not sure about the implementation of the order deal here Sets the size of the history buffer

Methods

- clear
  public void clear()
  - Usage
    * clears the buffer and resets the values to beginning

- doIteration
  public void doIteration()
  - Usage
    * Shifts the history of the buffer 1 example ym3 = ym2, ym2 = ym1, ym1 = ym0; order is important time slot

- main
  public static void main( java.lang.String [] args )
  - Usage
    * main to test out the functionality of this class
**toString**

`public String toString()`

- **Usage**
  - * print out the zorder and the buffer formatted as the java.util.vector output

**Methods inherited from class simtk.SimBlock**

- **clear**
  `public void clear()`

  - **Usage**
    - * this is the general function to clear a block. this is just an empty method but if more specific functionality is needed this should be overridden

- **doIteration**
  `abstract void doIteration()`

  - **Usage**
    - * this is the method that should be implemented in all Simblock’s

- **doOutputBufferIteration**
  `public void doOutputBufferIteration()`

  - **Usage**
    - * iterate the output buffer(s), or shift the history of it

- **getPanel**
  `public JPanel getPanel()`

  - **Usage**
    - * generates the default panel and the title

- **getSimParams**
  `public SimParameters getSimParams()`

  - **Usage**
    - * get the parameters from a parameter class

- **getVariableContainer**
  `public VariableContainer getVariableContainer()`

  - **Usage**
    - * returns the conatainer for all the variables. this is mainly used for the operations done on the varContainer

- **isHidden**
  `public boolean isHidden()`

  - **Usage**
* test if the hidden flag has been set

- **iterateBuffers**
  
  public void iterateBuffers()
  
  - **Usage**
  
  * This function does nothing, but blocks with outputs or internal buffers such as SimFunctionFactory should overload it for its own special purposes

- **reDefine**
  
  public void reDefine()
  
  - **Usage**
  
  * transfer upper level adjustments to the lower level. this method is overloaded when it is to be used by a lower function

- **setHidden**
  
  public void setHidden()
  
  - **Usage**
  
  * keep the block from being loaded to the factory display if it is panelable

- **setHidden**
  
  public void setHidden(boolean hide)
  
  - **Usage**
  
  * keep the block from being loaded to the factory display if it is panelable

- **setSimParams**
  
  public void setSimParams(simtk.SimParameters sp)
  
  - **Usage**
  
  * set the parameters from a parameter class

**Methods inherited from class simtk.SimBase**

- **getDescription**
  
  public String getDescription()

- **getReloadBlock**
  
  public Simulatable getReloadBlock()
  
  - **Usage**
  
  * set the block that gets called to resimulate when the a variable is changed.

- **getTitle**
  
  public String getTitle()
  
  - **Usage**
  
  * get the user defined title for the entire block
• `setDescription`
  
  ```java
  public void setDescription( java.lang.String desc )
  ```

• `setReloadBlock`
  
  ```java
  public void setReloadBlock( simtk.Simulatable reloadBlock )
  ```
  
  **Usage**
  
  * set the block that gets called to resimulate when the a variable is changed.

• `setTitle`
  
  ```java
  public void setTitle( java.lang.String title )
  ```
  
  **Usage**
  
  * set the user defined title for the entire block

### 2.2.10 Class `SimContinuousProfile`

Compilation of sim blocks to create the profile

**Declaration**

```java
public class SimContinuousProfile
extends simtk.SimSourceFactory
implements SimPanelable, ContainsFactory
```

**Constructors**

- `SimContinuousProfile`
  
  ```java
  public SimContinuousProfile()
  ```
  
  **Usage**
  
  * constructor sets gain to default of unity

**Methods**

- `connectBlocks`
  
  ```java
  public void connectBlocks()
  ```
  
  **Usage**
  
  * Make the connections for this block. The `setInput` of this factory calls this to setup. It turns out to be a rather interesting system the `setInput` to this factory then in turn setups up this internal system
Methods inherited from class simtk.SimSourceFactory

- **clear**
  public void clear()  
  - Usage  
  * this function is required to reset the buffers and values for the next iterations to set up the factory class

- **connectBlocks**
  abstract void connectBlocks()  
  - Usage  
  * Make the connections for this block. The setInput of this factory calls this to setup. It turns out to be a rather interesting system the setInput to this factory then in turn set’s up this internal system. this is to be implemented.

- **doIteration**
  public void doIteration()  
  - Usage  
  * do a calculation for the new output

- **getPanel**
  public JPanel getPanel()  

- **iterateBuffers**
  public void iterateBuffers()  
  - Usage  
  * cycle the buffers

- **toString**
  public String toString()  

Methods inherited from class simtk.SimSource

- **doIteration**
  abstract void doIteration()  
  - Usage  
  * this is the method that should be implemented in all SimBlock’s

- **getOutput**
  public SimBuffer getOutput()  

- **iterateBuffers**
  public void iterateBuffers()  
  - Usage  
  * overload the do nothing iterateBuffers and iterate the output buffers

- **setOutput**
  public void setOutput( simtk.SimBuffer bufref )
Methods inherited from class `simtk.SimBlock`

- `clear`
  ```java
  public void clear()
  ```
  - **Usage**
    * this is the general function to clear a block. this is just an empty method but if more
      specific functionality is needed this should be overridden

- `doIteration`
  ```java
  abstract void doIteration()
  ```
  - **Usage**
    * this is the method that should be implemented in all Simblock’s

- `doOutputBufferIteration`
  ```java
  public void doOutputBufferIteration()
  ```
  - **Usage**
    * iterate the output buffer(s), or shift the history of it

- `getPanel`
  ```java
  public JPanel getPanel()
  ```
  - **Usage**
    * generates the default panel and the title

- `getSimParams`
  ```java
  public SimParameters getSimParams()
  ```
  - **Usage**
    * get the parameters from a parameter class

- `getVariableContainer`
  ```java
  public VariableContainer getVariableContainer()
  ```
  - **Usage**
    * returns the container for all the variables. this is mainly used for the operations done on
      the varContainer

- `isHidden`
  ```java
  public boolean isHidden()
  ```
  - **Usage**
    * test if the hidden flag has been set

- `iterateBuffers`
  ```java
  public void iterateBuffers()
  ```
  - **Usage**
    * This function does nothing, but blocks with outputs or internal buffers such as
      SimFunctionFactory should overload it for it’s own special purposes
reDefine
public void reDefine( )
  
  Usual
  * transfer upper level adjustments to the lower level. this method is overloaded when it is to
be used by a lower function

setHidden
public void setHidden( )

  Usual
  * keep the block from being loaded to the factory display if it is panelable

setHidden
public void setHidden( boolean hide )

  Usual
  * keep the block from being loaded to the factory display if it is panelable

setSimParams
public void setSimParams( simtk.SimParameters sp )

  Usual
  * set the parameters from a parameter class

Methods inherited from class simtk.SimBase

getDescription
public String getDescription( )

getReloadBlock
public Simulatable getReloadBlock( )

  Usual
  * set the block that gets called to resimulate when the a variable is changed.

ggetTitle
public String getTitle( )

  Usual
  * get the user defined title for the entire block

setDescription
public void setDescription( java.lang.String desc )

setReloadBlock
public void setReloadBlock( simtk.Simulatable reloadBlock )

  Usage
  * set the block that gets called to resimulate when the a variable is changed.

setTitle
public void setTitle( java.lang.String title )

  Usage
  * set the user defined title for the entire block
2.2.11 Class SimDialogManager

This is used to create JDialogs for the tuning of SimBlocks. This generates a special JDialog that has simple sliders for a one to one ratio of variables.

Declaration

```java
public class SimDialogManager
    extends java.lang.Object
```

Constructors

- `SimDialogManager`
  ```java
  public SimDialogManager( javax.swing.JFrame parent, int number_of_variables, java.lang.String sname, java.lang.Object sim_object, simtk.SimReLoadable reloadable )
  ```

  * Usage
  ```java
  new SimDialogManager(null, 2, "Step parameters", this, reloadable);
  ```

  Would create a window titled Step parameters with 2 variables having sliders, labels and text fields for each.

  Note: after this is invoked you must make sure you first `setupVariables` for all of the number of variables you asked for.

Methods

- `getDialog`
  ```java
  public JDialog getDialog() )
  ```

  * Usage
  ```java
  * creates a Jdialog window with all of the created components and returns it to the user
  ```

- `getSimField`
  ```java
  public double getSimField( java.lang.String paramName, boolean globalType )
  ```

  * Usage
• gets double field from the object, sim_object with name, name Note: the field must be public for this to read it. this function is used by the setup of the listeners for dynamic setup of sliders and parameter changes

---

• makeEditableArrayViaTextField
public static JTextField makeEditableArrayViaTextField( double[] snum, simtk.SimStf tempStf, simtk.SimReLoadable reloadable )

  – Usage
  * static function to genetate a text field that is linked to an array. the text field is linked to a double array that is displayed in the text field as seperated by commas = ',' for now if there is an overflow error no error is signaled it will just fill the array so there is no error

---

• makePanel
public JPanel makePanel( )

  – Usage
  * creates a JPanel with all of the created components and returns it to the user

---

• refreshFields
public void refreshFields( )

  – Usage
  * calls the refreshFields with a boolean true as default

---

• refreshFields
public void refreshFields( boolean loadLast )

  – Usage
  * Refresh the text fields to display the current field values. This is especially used for the multiparameter adjustements were other variables are effected that need to be updated.

  The loadLast operater says if the loadLast register should also be updated

---

• setReLoadableObject
public void setReLoadableObject( simtk.SimReLoadable reloadable )

  – Usage
  * just sets the reloadable variable

---

• setSimField
public void setSimField( java.lang.String paramName, double d )
simtk – SimDiff

2.2.12 Class SimDiff

This differentiator is defined by a Stf and is a double pole order diff

Declaration

```java
public class SimDiff
    extends simtk.SimStf
    implements SimPanelable
```
CONSTRUCTORS

- \textit{SimDiff}
  
  \begin{verbatim}
  public SimDiff()
  
  Usage
  * constructor for the Diff, I am still a little bit unsure how to implement the zorder
  of the system.
  \end{verbatim}

METHODS

- \textit{reDefine}
  
  \begin{verbatim}
  public void reDefine()
  
  Usage
  * This is made to recalculate with new parameters, step sizes parameters. This is a
  method that might be good to have defined in the sim Block as abstract overrides
  the redefine of SimZtf and SimStf and loads in the new parameters then it calls the
  overridden redefine from Stf which then overrides the Ztf for the final difference
  equation, 3 levels of abstraction
  \end{verbatim}

METHODS INHERITED FROM CLASS \texttt{simtk.SimStf}

- \textit{displayNotProperErr}
  
  \begin{verbatim}
  private void displayNotProperErr()
  
  Usage
  * improper functions error
  \end{verbatim}

- \textit{generateZPoly}
  
  \begin{verbatim}
  public double generateZPoly( double [] s )
  
  Usage
  * generate the polynomial as an array from the s polynomial the order is extracted by the size
  of s this is the magic function for the conversion of continuous to discrete see the
  derivations done on 4/26/02 with mathematica and the papers solution I need to type up
  these derivations
  \end{verbatim}

- \textit{makeNewStf}
  
  \begin{verbatim}
  public void makeNewStf( int snum_order, int sdem_order )
  
  Usage
  * creates the num and den checking if improper else make default
  \end{verbatim}
• **reDefine**
  public void reDefine()
  
  – **Usage**
  * this is a method that might be good to have defined in the sim Block as abstract this is made to recalculate with new parameters, step sizes parameters

• **setSDen**
  public void setSDen( double [] a )

  – **Usage**
  * simply sets the s denominator to an array value

• **setSNum**
  public void setSNum( double [] a )

  – **Usage**
  * simply sets the s numerator to an array value

• **toString**
  public String toString()

  – **Usage**
  * toString in the form of: num_order = x den_order = y num = [x] den = [y]

• **zeroStf**
  public void zeroStf()

  – **Usage**
  * this zeroSTF method is for convienience in setting up z transfer functions it makes sure that the function is reset to zero from the initial conditions

**Methods inherited from class simtk.SimZtf**

• **doIteration**
  public void doIteration()

  – **Usage**
  * this method overrides the doIteration of the diffeq. this way it first redefines the variables and then iterates with the inherited structure

• **doIterationDirect**
  public void doIterationDirect()

  – **Usage**
  * this is the really cool simple method that converts ztf to diffEQ. The function does the operation in the following order:
  1. parses the numerator and denominator
  2. creates the difference equation
  3. does one step operation.
• **reDefine**
  public void reDefine( )
  
  – **Usage**
  * this is a method that might be good to have defined in the sim Block as abstract this is made to recalculate with new parameters, step sizes parameters

• **setDen**
  public void setDen( double [] n )
  
  – **Usage**
  * method to set the denominator to the array value Example: setDen([1,2,3]); would set the denominator to $z^2 + 2z + 3$ an error message is displayed if the array is larger than the z order

• **setNum**
  public void setNum( double [] n )
  
  – **Usage**
  * method to set the numerator to the array value Example: setNum([1,2,3]); would set the numerator to $z^2 + 2z + 3$ an error message is displayed if the array is larger than the z order

• **toString**
  public String toString( )
  
  – **Usage**
  * toString in the form of: zorder = $x$ num = [x] den = [x]

• **zeroNumAndDen**
  public void zeroNumAndDen( )
  
  – **Usage**
  * this method is for convenience in setting up z transfer functions it makes sure that the function is reset to zero from the initial conditions

**Methods inherited from class simtk.SimDiffEq**

• **clear**
  public void clear( )
  
  – **Usage**
  * clears the text field, sets up headers and resets the index counter

• **doIteration**
  public void doIteration( )

• **getInCoef**
  public SimDoubleArray getInCoef( )
  
  – **Usage**
* return the input coefficients

• *return the input coefficients
  public SimDoubleArray getOutCoef()

  – Usage
  * return the output coefficients

• *setZeros
  public void setZeros()

  – Usage
  * zero all the coefficients for easy settings

• *toDiffEqString
  public String toDiffEqString()

  – Usage
  * output the string format of the difference equation. special function to iterate through all
  of the parameters and create the difference equation. This function is also used for the
  textBox for the dynamic pretty print viewing of the diff eq

• *toString
  public String toString()

  – Usage
  * general to String equation for the diff eq. this uses the toDiffEqString() function for the
  special formatting so that one doesn’t overloaded

**Methods inherited from class simtk.SimFunction**

• *addInput
  public void addInput( simtk.HasOutput outputBlock )

  – Usage
  * this AddInput adds from a HasOutput block an input to the vector function and increases
  the inputIndex count

• *addInput
  public void addInput( simtk.SimBuffer bufref )

  – Usage
  * AddInput adds an input to the vector function and increases the inputIndex count

• *doIteration
  abstract void doIteration()

  – Usage
  * this is the method that should be implemented in all SimBlock’s

• *getInput
  public SimBuffer getInput()
- **Usage**
  * get the first input. there might be a problem here because the old method of the implicit
  in buffer is used

• **getInput**
  public SimBuffer getInput( int index )
  - **Usage**
    * get the indexed input, referenced from as 1 is the first.
      If the reference is out of range it will return the single input value.
      The code is not so clean here where it will still print an erro message when the double
      input functions are just beginning to set themselves up and before the.???

• **getInputCount**
  public int getInputCount( )
  - **Usage**
    * get the inputCount, or the number of elements on the list

• **getOutput**
  public SimBuffer getOutput( )
  - **Usage**
    * Gets the output buffer

• **iterateBuffers**
  public void iterateBuffers( )
  - **Usage**
    * overload the do nothing iterateBuffers and iterate the output buffers

• **removeInputs**
  public void removeInputs( )
  - **Usage**
    * remove the all of the inputs from the block so a new system can be set up

• **setInput**
  public void setInput( int index, simtk.SimBuffer bufref )
  - **Usage**
    * Set the indexed input to the passed buffer. note: this has out of index problems

• **setInput**
  public void setInput( simtk.SimBuffer bufref )
  - **Usage**
    * Set the input to the passed buffer, this is replaced with the newer addInput function which
      sounds more like what it does

• **setInput1**
  public void setInput1( simtk.SimBuffer bufref )
Usage
* This is here to support old systems, it just calls addInput(bufref). so it is not exactly input 2

• setInput2
public void setInput2( simtk.SimBuffer bufref )
  – Usage
    * This is here to support old systems, it just calls addInput(bufref). so it is not exactly input 2

• setOutput
public void setOutput( simtk.SimBuffer bufref )
  – Usage
    * sets the output to a passed buffer

Methods inherited from class simtk.SimBlock

• clear
  public void clear( )
  – Usage
    * this is the general function to clear a block. this is just an empty method but if more specific functionality is needed this should be overridden

• doIteration
  abstract void doIteration( )
  – Usage
    * this is the method that should be implemented in all Simblock’s

• doOutputBufferIteration
  public void doOutputBufferIteration( )
  – Usage
    * iterate the output buffer(s), or shift the history of it

• getPanel
  public JPanel getPanel( )
  – Usage
    * generates the default panel and the title

• getSimParams
  public SimParameters getSimParams( )
  – Usage
    * get the parameters from a parameter class
• **getVariableContainer**
  public VariableContainer getVariableContainer()  
  – **Usage**  
  * returns the container for all the variables. this is mainly used for the operations done on  
  the varContainer

• **isHidden**
  public boolean isHidden()  
  – **Usage**  
  * test if the hidden flag has been set

• **iterateBuffers**
  public void iterateBuffers()  
  – **Usage**  
  * This function does nothing, but blocks with outputs or internal buffers such as  
  SimFunctionFactory should overload it for its own special purposes

• **reDefine**
  public void reDefine()  
  – **Usage**  
  * transfer upper level adjustments to the lower level. this method is overloaded when it is to  
  be used by a lower function

• **setHidden**
  public void setHidden()  
  – **Usage**  
  * keep the block from being loaded to the factory display if it is panelable

• **setHidden**
  public void setHidden( boolean hide )  
  – **Usage**  
  * keep the block from being loaded to the factory display if it is panelable

• **setSimParams**
  public void setSimParams( simtk.SimParameters sp )  
  – **Usage**  
  * set the parameters from a parameter class
Methods inherited from class simtk.SimBase

- **getDescription**
  ```java
  public String getDescription()
  ```
  - Usage
    * set the block that gets called to resimulate when the a variable is changed.

- **getReloadBlock**
  ```java
  public Simulatable getReloadBlock()
  ```
  - Usage
    * set the block that gets called to resimulate when the a variable is changed.

- **getTitle**
  ```java
  public String getTitle()
  ```
  - Usage
    * get the user defined title for the entire block

- **setDescription**
  ```java
  public void setDescription(java.lang.String desc)
  ```

- **setReloadBlock**
  ```java
  public void setReloadBlock(simtk.Simulatable reloadBlock)
  ```
  - Usage
    * set the block that gets called to resimulate when the a variable is changed.

- **setTitle**
  ```java
  public void setTitle(java.lang.String title)
  ```
  - Usage
    * set the user defined title for the entire block

### 2.2.13 Class SimDiffEq

**Declaration**

```java
public class SimDiffEq
extends simtk.SimFunction
implements SimPanelable
```

**Fields**

- private int k

- private SimTextBox textBox
  - holds text box output for the difference equation in pretty print format
Constructors

- SimDiffEq
  public SimDiffEq()
  
  - Usage
  * constructor for the DiffEq,

Methods

- clear
  public void clear()
  
  - Usage
  * clears the text field, sets up headers and resets the index counter

- doIteration
  public void doIteration()

- getInCoef
  public SimDoubleArray getInCoef()
  
  - Usage
  * return the input coefficients

- getOutCoef
  public SimDoubleArray getOutCoef()
  
  - Usage
  * return the output coefficients

- setZeros
  public void setZeros()
  
  - Usage
  * zero all the coefficients for easy settings

- toDiffEqString
  public String toDiffEqString()
  
  - Usage
  * output the string format of the difference equation. special function to iterate through all of the parameters and create the difference equation. This function is also used for the text box for the dynamic pretty print viewing of the diff eq
- **toString**
  ```java
  public String toString()
  ```
  - **Usage**
  * general to String equation for the diff eq. this uses the toDiffEqString() function for the special formatting so that one doesn’t overloaded

**Methods inherited from class simtk.SimFunction**

- **addInput**
  ```java
  public void addInput( simtk.HasOutput outputBlock )
  ```
  - **Usage**
  * this AddInput adds from a HasOutput block an input to the vector function and increases the inputIndex count

- **addInput**
  ```java
  public void addInput( simtk.SimBuffer bufref )
  ```
  - **Usage**
  * AddInput adds an input to the vector function and increases the inputIndex count

- **doIteration**
  ```java
  abstract void doIteration()
  ```
  - **Usage**
  * this is the method that should be implemented in all SimBlock’s

- **getInput**
  ```java
  public SimBuffer getInput()
  ```
  - **Usage**
  * get the first input. there might be a problem here because the old method of the implicit in buffer is used

- **getInput**
  ```java
  public SimBuffer getInput( int index )
  ```
  - **Usage**
  * get the indexed input, referenced from as 1 is the first.
  If the reference is out of range it will return the the single input value.
  The code is not so clean here where it will still print an erro message when the double input functions are just beginning to set themselves up and before the.???

- **getInputCount**
  ```java
  public int getInputCount()
  ```
  - **Usage**
  * get the inputCount, or the number of elements on the list
**getOutput**

```java
public SimBuffer getOutput()
```

- **Usage**
  * Gets the output buffer

**iterateBuffers**

```java
public void iterateBuffers()
```

- **Usage**
  * overload the do nothing iterateBuffers and iterate the output buffers

**removeInputs**

```java
public void removeInputs()
```

- **Usage**
  * remove the all of the inputs from the block so a new system can be set up

**setInput**

```java
public void setInput( int index, simtk.SimBuffer bufref )
```

- **Usage**
  * Set the indexed input to the passed buffer. note: this has out of index problems

**setInput**

```java
public void setInput( simtk.SimBuffer bufref )
```

- **Usage**
  * Set the input to the passed buffer, this is replaced with the newer addInput function which sounds more like what it does

**setInput1**

```java
public void setInput1( simtk.SimBuffer bufref )
```

- **Usage**
  * This is here to support old systems, it just calls addInput(bufref). so it is not exactly input 2

**setInput2**

```java
public void setInput2( simtk.SimBuffer bufref )
```

- **Usage**
  * This is here to support old systems, it just calls addInput(bufref). so it is not exactly input 2

**setOutput**

```java
public void setOutput( simtk.SimBuffer bufref )
```

- **Usage**
  * sets the output to a passed buffer
Methods inherited from class simtk.SimBlock

- **clear**
  public void clear()
  
  - Usage
    * this is the general function to clear a block. this is just an empty method but if more specific functionality is needed this should be overridden

- **doIteration**
  abstract void doIteration()
  
  - Usage
    * this is the method that should be implemented in all Simblock's

- **doOutputBufferIteration**
  public void doOutputBufferIteration()
  
  - Usage
    * iterate the output buffer(s), or shift the history of it

- **getPanel**
  public JPanel getPanel()
  
  - Usage
    * generates the default panel and the title

- **getSimParams**
  public SimParameters getSimParams()
  
  - Usage
    * get the parameters from a parameter class

- **getVariableContainer**
  public VariableContainer getVariableContainer()
  
  - Usage
    * returns the conatainer for all the variables. this is mainly used for the operations done on the varContainer

- **isHidden**
  public boolean isHidden()
  
  - Usage
    * test if the hidden flag has been set

- **iterateBuffers**
  public void iterateBuffers()
  
  - Usage
    * This function does nothing, but blocks with outputs or internal buffers such as SimFunctionFactory should overload it for it’s own special pursposes
- **reDefine**
  public void reDefine( )
  
  - **Usage**
  * transfer upper level adjustments to the lower level. This method is overloaded when it is to 
    be used by a lower function

- **setHidden**
  public void setHidden( )
  
  - **Usage**
  * keep the block from being loaded to the factory display if it is panelable

- **setHidden**
  public void setHidden( boolean hide )
  
  - **Usage**
  * keep the block from being loaded to the factory display if it is panelable

- **setSimParams**
  public void setSimParams( simtk.SimParameters sp )
  
  - **Usage**
  * set the parameters from a parameter class

**Methods inherited from class simtk.SimBase**

- **getDescription**
  public String getDescription( )

- **getReloadBlock**
  public Simulatable getReloadBlock( )
  
  - **Usage**
  * set the block that gets called to resimulate when the a variable is changed.

- **getTitle**
  public String getTitle( )
  
  - **Usage**
  * get the user defined title for the entire block

- **setDescription**
  public void setDescription( java.lang.String desc )

- **setReloadBlock**
  public void setReloadBlock( simtk.Simulatable reloadBlock )
  
  - **Usage**
  * set the block that gets called to resimulate when the a variable is changed.

- **setTitle**
  public void setTitle( java.lang.String title )
  
  - **Usage**
  * set the user defined title for the entire block
2.2.14 Class SimDisturbance

Disturbance sources, the first will be a sine wave. Waveform types

- sine
- square

**Declaration**

```java
public class SimDisturbance extends simtk.SimSource implements SimPanelable
```

**Fields**

- private int k
  - this is the private index counter to count how many iterations have been done
- private SimComboBoxString waveformSel
  - Holds the type of waveform generated, also used to select

**Constructors**

- `SimDisturbance`
  
  ```java
  public SimDisturbance()
  ```
  
  - **Usage**
    
    * defaults are set to delay = 0 and gain = 1, freq = 1, waveform = ”sine”

**Methods**

- `clear`
  
  ```java
  public void clear()
  ```

- `doIteration`
  
  ```java
  public void doIteration()
  ```
**Usage**

* set the output to the gain value and function after the delay time

- `getDelay`
  ```java
  public double getDelay()
  ```

- `getFreq`
  ```java
  public double getFreq()
  ```

- `getGain`
  ```java
  public double getGain()
  ```

- `getWaveForm`
  ```java
  public String getWaveForm()
  ```

- `setDelay`
  ```java
  public void setDelay(double delay)
  ```

- `setFreq`
  ```java
  public void setFreq(double freq)
  ```

- `setGain`
  ```java
  public void setGain(double gain)
  ```

**Methods inherited from class simtk.SimSource**

- `doIteration`
  ```java
  abstract void doIteration()
  ```

  * Usage
    * this is the method that should be implemented in all SimBlock's

- `getOutput`
  ```java
  public SimBuffer getOutput()
  ```

- `iterateBuffers`
  ```java
  public void iterateBuffers()
  ```

  * Usage
    * overload the do nothing iterateBuffers and iterate the output buffers

- `setOutput`
  ```java
  public void setOutput(simtk.SimBuffer bufref)
  ```
Methods inherited from class simtk.SimBlock

• clear
  public void clear( )
  – Usage
  * this is the general function to clear a block. this is just an empty method but if more
  specific functionality is needed this should be overridden

• doIteration
  abstract void doIteration( )
  – Usage
  * this is the method that should be implemented in all Simblock’s

• doOutputBufferIteration
  public void doOutputBufferIteration( )
  – Usage
  * iterate the output buffer(s), or shift the history of it

• getPanel
  public JPanel getPanel( )
  – Usage
  * generates the default panel and the title

• getSimParams
  public SimParameters getSimParams( )
  – Usage
  * get the parameters from a parameter class

• getVariableContainer
  public VariableContainer getVariableContainer( )
  – Usage
  * returns the container for all the variables. this is mainly used for the operations done on
  the varContainer

• isHidden
  public boolean isHidden( )
  – Usage
  * test if the hidden flag has been set

• iterateBuffers
  public void iterateBuffers( )
  – Usage
  * This function does nothing, but blocks with outputs or internal buffers such as
    SimFunctionFactory should overload it for it’s own special purposes
• **reDefine**
  public void reDefine( )
  
  - **Usage**
    * transfer upper level adjustments to the lower level. this method is overloaded when it is to be used by a lower function

• **setHidden**
  public void setHidden( )
  
  - **Usage**
    * keep the block from being loaded to the factory display if it is panelable

• **setHidden**
  public void setHidden( boolean hide )
  
  - **Usage**
    * keep the block from being loaded to the factory display if it is panelable

• **setSimParams**
  public void setSimParams( simtk.SimParameters sp )
  
  - **Usage**
    * set the parameters from a parameter class

**Methods inherited from class simtk.SimBase**

• **getDescription**
  public String getDescription( )

• **getReloadBlock**
  public Simulatable getReloadBlock( )
  
  - **Usage**
    * set the block that gets called to resimulate when the a variable is changed.

• **getTitle**
  public String getTitle( )
  
  - **Usage**
    * get the user defined title for the entire block

• **setDescription**
  public void setDescription( java.lang.String desc )

• **setReloadBlock**
  public void setReloadBlock( simtk.Simulatable reloadBlock )
  
  - **Usage**
    * set the block that gets called to resimulate when the a variable is changed.

• **setTitle**
  public void setTitle( java.lang.String title )
  
  - **Usage**
    * set the user defined title for the entire block
2.2.15 CLASS SimDoubleIntegrator

this is simply a double integrator function

DECLARATION

public class SimDoubleIntegrator
extends simtk.SimZtf

CONSTRUCTORS

- SimDoubleIntegrator
  public SimDoubleIntegrator( )

METHODS

- reDefine
  public void reDefine( )

  - Usage
    * this is a method that might be good to have defined in the sim Block as abstract
      this is made to recalculate with new parameters, step sizes parameters overrides the
      reDefine of SimZtf and loads in the new parameters then it calls the overridden
      reDefine from Ztf

METHODS INHERITED FROM CLASS simtk.SimZtf

- doIteration
  public void doIteration( )

  - Usage
    * this method overrides the doIteration of the diffeq. this way it first redefines the variables
      and then iterates with the inherited structure

- doIterationDirect
  public void doIterationDirect( )

  - Usage
    * this is the really cool simple method that converts ztf to diffEQ. The function does the
      operation in the following order:
      1. parses the numerator and denominator
2. creates the difference equation
3. does one step operation.

- **reDefine**
  public void reDefine( )
  - **Usage**
    * this is a method that might be good to have defined in the sim Block as abstract this is made to recalculate with new parameters, step sizes parameters

- **setDen**
  public void setDen( double [] n )
  - **Usage**
    * method to set the denominator to the array value Example: setDen([1,2,3]); would set the denominator to \( z^2 + 2z + 3 \) an error message is displayed if the array is larger than the \( z \) order

- **setNum**
  public void setNum( double [] n )
  - **Usage**
    * method to set the numerator to the array value Example: setNum([1,2,3]); would set the numerator to \( z^2 + 2z + 3 \) an error message is displayed if the array is larger than the \( z \) order

- **toString**
  public String toString( )
  - **Usage**
    * toString in the form of: \( z^{\text{order}} = x \) \( \text{num} = [x] \) \( \text{den} = [x] \)

- **zeroNumAndDen**
  public void zeroNumAndDen( )
  - **Usage**
    * this method is for convenience in setting up \( z \) transfer functions it makes sure that the function is reset to zero from the initial conditions

**Methods inherited from class simtk.SimDiffEq**

- **clear**
  public void clear( )
  - **Usage**
    * clears the text field, sets up headers and resets the index counter

- **doIteration**
  public void doIteration( )

- **getInCoef**
  public SimDoubleArray getInCoef( )
• Usage
  * return the input coefficients

• getOutCoeff
  public SimDoubleArray getOutCoeff( )
  – Usage
  * return the output coefficients

• setZeros
  public void setZeros( )
  – Usage
  * zero all the coefficients for easy settings

• toDiffEqString
  public String toDiffEqString( )
  – Usage
  * output the string format of the difference equation. special function to iterate through all of the parameters and create the difference equation. This function is also used for the text box for the dynamic pretty print viewing of the diff eq

• toString
  public String toString( )
  – Usage
  * general to String equation for the diff eq. this uses the toDiffEqString() function for the special formatting so that one doesn’t overloaded

Methods inherited from class simtk.SimFunction

• addInput
  public void addInput( simtk.HasOutput outputBlock )
  – Usage
  * this AddInput adds from a HasOutput block an input to the vector function and increases the inputIndex count

• addInput
  public void addInput( simtk.SimBuffer bufref )
  – Usage
  * AddInput adds an input to the vector function and increases the inputIndex count

• doIteration
  abstract void doIteration( )
  – Usage
  * this is the method that should be implemented in all SimBlock’s
• **getInput**
  
  ```java
  public SimBuffer getInput()
  ```
  
  **Usage**
  
  * get the first input. there might be a problem here because the old method of the implicit in buffer is used

• **getInput**
  
  ```java
  public SimBuffer getInput(int index)
  ```
  
  **Usage**
  
  * get the indexed input, referenced from as 1 is the first.
  
  If the reference is out of range it will return the the single input value.
  
  The code is not so clean here where it will still print an erro message when the double input functions are just beginning to set themselves up and before the.???

• **getInputCount**
  
  ```java
  public int getInputCount()
  ```
  
  **Usage**
  
  * get the inputCount, or the number of elements on the list

• **getOutput**
  
  ```java
  public SimBuffer getOutput()
  ```
  
  **Usage**
  
  * Gets the output buffer

• **iterateBuffers**
  
  ```java
  public void iterateBuffers()
  ```
  
  **Usage**
  
  * overload the do nothing iterateBuffers and iterate the output buffers

• **removeInputs**
  
  ```java
  public void removeInputs()
  ```
  
  **Usage**
  
  * remove the all of the inputs from the block so a new system can be set up

• **setInput**
  
  ```java
  public void setInput(int index, Simtk.SimBuffer bufref)
  ```
  
  **Usage**
  
  * Set the indexed input to the passed buffer. note: this has out of index problems

• **setInput**
  
  ```java
  public void setInput(Simtk.SimBuffer bufref)
  ```
  
  **Usage**
  
  * Set the input to the passed buffer, this is replaced with the newer addInput function which sounds more like what it does
• **setInput1**
  ```java
  public void setInput1( simtk.SimBuffer bufref )
  ```
  - **Usage**
    * This is here to support old systems, it just calls addInput(bufref). so it is not exactly
      input 2

• **setInput2**
  ```java
  public void setInput2( simtk.SimBuffer bufref )
  ```
  - **Usage**
    * This is here to support old systems, it just calls addInput(bufref). so it is not exactly
      input 2

• **setOutput**
  ```java
  public void setOutput( simtk.SimBuffer bufref )
  ```
  - **Usage**
    * sets the output to a passed buffer

**Methods inherited from class simtk.SimBlock**

• **clear**
  ```java
  public void clear( )
  ```
  - **Usage**
    * this is the general function to clear a block. this is just an empty method but if more
      specific functionality is needed this should be overridden

• **doIteration**
  ```java
  abstract void doIteration( )
  ```
  - **Usage**
    * this is the method that should be implemented in all Simblock’s

• **doOutputBufferIteration**
  ```java
  public void doOutputBufferIteration( )
  ```
  - **Usage**
    * iterate the output buffer(s), or shift the history of it

• **getPanel**
  ```java
  public JPanel getPanel( )
  ```
  - **Usage**
    * generates the default panel and the title

• **getSimParams**
  ```java
  public SimParameters getSimParams( )
  ```
  - **Usage**
• **getVariableContainer**

  ```java
  public VariableContainer getVariableContainer()
  ```

  *Usage*
  
  * returns the container for all the variables. This is mainly used for the operations done on the varContainer

• **isHidden**

  ```java
  public boolean isHidden()
  ```

  *Usage*
  
  * test if the hidden flag has been set

• **iterateBuffers**

  ```java
  public void iterateBuffers()
  ```

  *Usage*
  
  * This function does nothing, but blocks with outputs or internal buffers such as SimFunctionFactory should overload it for its own special purposes

• **reDefine**

  ```java
  public void reDefine()
  ```

  *Usage*
  
  * transfer upper level adjustments to the lower level. This method is overloaded when it is to be used by a lower function

• **setHidden**

  ```java
  public void setHidden()
  ```

  *Usage*
  
  * keep the block from being loaded to the factory display if it is panelable

• **setHidden**

  ```java
  public void setHidden(boolean hide)
  ```

  *Usage*
  
  * keep the block from being loaded to the factory display if it is panelable

• **setSimParams**

  ```java
  public void setSimParams(simtk.SimParameters sp)
  ```

  *Usage*
  
  * set the parameters from a parameter class

• **get the parameters from a parameter class**
Methods inherited from class simtk.SimBase

- `getDescription`
  ```java`
  public String getDescription()
  ```
- `getReloadBlock`
  ```java`
  public Simulatable getReloadBlock()
  ```
  - Usage
    * set the block that gets called to resimulate when the a variable is changed.
- `getTitle`
  ```java`
  public String getTitle()
  ```
  - Usage
    * get the user defined title for the entire block
- `setDescription`
  ```java`
  public void setDescription(java.lang.String desc)
  ```
- `setReloadBlock`
  ```java`
  public void setReloadBlock(simtk.Simulatable reloadBlock)
  ```
  - Usage
    * set the block that gets called to resimulate when the a variable is changed.
- `setTitle`
  ```java`
  public void setTitle(java.lang.String title)
  ```
  - Usage
    * set the user defined title for the entire block

2.2.16 Class SimDoubleTriangle

Double triangle profile function. provides a smoother response for position, velocity, and acceleration and Jerk in place of a step function or trapezoidal function. This creates a smoother trapezoidal filter

Declaration

```java`
public class SimDoubleTriangle
extends simtk.SimSource
implements SimPanelable
```
**Fields**

- private int k
- private SimNumber gain
- private SimNumber delay
- private SimNumber width
  - Width of trapezoid

**Constructors**

- *SimDoubleTriangle*
  
  public SimDoubleTriangle()
  
  - Usage
    
    * defaults are set to delay = 0 and gain = 1, totalTime = 1

**Methods**

- clear
  
  public void clear()

- doIteration
  
  public void doIteration()
  
  - Usage
    
    * set the output to the gain value after the delay time

**Methods inherited from class simtk.SimSource**

- doIteration
  
  abstract void doIteration()
  
  - Usage
    
    * this is the method that should be implemented in all SimBlock’s
• **getOutput**
  public SimBuffer getOutput()  

• **iterateBuffers**
  public void iterateBuffers()
  
  – Usage
  * overload the do nothing iterateBuffers and iterate the output buffers

• **setOutput**
  public void setOutput(simtk.SimBuffer bufref)

**Methods inherited from class simtk.SimBlock**

• **clear**
  public void clear()
  
  – Usage
  * this is the general function to clear a block. this is just an empty method but if more specific functionality is needed this should be overridden

• **doIteration**
  abstract void doIteration()
  
  – Usage
  * this is the method that should be implemented in all Simblock’s

• **doOutputBufferIteration**
  public void doOutputBufferIteration()
  
  – Usage
  * iterate the output buffer(s), or shift the history of it

• **getPanel**
  public JPanel getPanel()
  
  – Usage
  * generates the default panel and the title

• **getSimParams**
  public SimParameters getSimParams()
  
  – Usage
  * get the parameters from a parameter class

• **getVariableContainer**
  public VariableContainer getVariableContainer()
  
  – Usage
  * returns the conatainer for all the variables. this is mainly used for the operations done on the varContainer
• isHidden
  public boolean isHidden()

  – Usage
  * test if the hidden flag has been set

• iterateBuffers
  public void iterateBuffers()

  – Usage
  * This function does nothing, but blocks with outputs or internal buffers such as
    SimFunctionFactory should overload it for it’s own special purposes

• reDefine
  public void reDefine()

  – Usage
  * transfer upper level adjustments to the lower level. this method is overloaded when it is to
    be used by a lower function

• setHidden
  public void setHidden()

  – Usage
  * keep the block from being loaded to the factory display if it is panelable

• setHidden
  public void setHidden( boolean hide )

  – Usage
  * keep the block from being loaded to the factory display if it is panelable

• setSimParams
  public void setSimParams( simtk.SimParameters sp )

  – Usage
  * set the parameters from a parameter class

Methods inherited from class simtk.SimBase

• getDescription
  public String getDescription()

• getReloadBlock
  public Simulatable getReloadBlock()

  – Usage
  * set the block that gets called to resimulate when the a variable is changed.

• getTitle
  public String getTitle()
2.2.17 Class SimFactory

This is the factory class that makes and controls the simulation environment. This should setup, and control button boxes, buffers, connections, simulation iterations and all

**Declaration**

```java
public class SimFactory
extends simtk.SimBlock
implements java.util.Iterator, SimPanelable, Simulatable
```

**Fields**

- private Vector blockVec
  - the vector that contains all the simulation blocks
- private Iterator it
- private boolean show_parameters
Constructors

- **SimFactory**
  
  ```java
  public SimFactory()
  ```
  
  - **Usage**
  
  * the constructor makes the first vectors

- **SimFactory**
  
  ```java
  public SimFactory(simtk.SimParameters params)
  ```
  
  - **Usage**
  
  * set the parameters of this factory block the the passed params

- **SimFactory**
  
  ```java
  public SimFactory(simtk.SimParameters params, simtk.Simulatable reloadBlock)
  ```
  
  - **Usage**
  
  * set the parameters of this factory block the the passed params

Methods

- **add**
  
  ```java
  public void add(simtk.SimBlock block)
  ```
  
  - **Usage**
  
  * Adds the block to the factory vector to iterate the simulation

- **add**
  
  ```java
  public void add(simtk.SimBlock block, java.lang.String title)
  ```
  
  - **Usage**
  
  * sets the title of a block and then adds it to the factory vector

- **add**
  
  ```java
  public void add(simtk.SimBlock block, java.lang.String title, simtk.HasOutput outputBlock)
  ```
  
  - **Usage**
  
  * like the other add(b,t) but now adds the setting of the input buffer by a block. we can now check if the SimBlock has inputs with the HasInput interface and that the outputblock has an output with the HasOutput
• add
  public void add( simtk.SimBlock block, java.lang.String title,
  simtk.HasOutput outputBlock1, simtk.HasOutput outputBlock2 )

  – Usage
  * like the other add(b,t,ob) but now adds a second setting of the input buffer by a
    block. we can now check if the SimBlock has inputs with the HasInput interface
    and that the outputblock has an output with the HasOutput

• add
  public void add( simtk.SimBlock block, java.lang.String title,
  simtk.SimBuffer inputBuf )

  – Usage
  * like the other add(b,t) but now adds the setting of the input buffer. we can now
    check if the SimBlock has inputs with the HasInput interface

• createSimButton
  public JButton createSimButton( simtk.SimPanelable sp, java.lang.String title )

  – Usage
  * creates a sim button with action listener and sets up the dialog panel with the
    passed parameters

• createSimButtonBox
  public JPanel createSimButtonBox( )

  – Usage
  * create a box with all of the sim buttons

• createSimTabBox
  public JPanel createSimTabBox( )

  – Usage
  * create a box with all of the internal components, but use tabs

• doIteration
  public void doIteration( )

  – Usage
  * Iterate all the blocks and then all the buffers. To fulfil the definition of the abstract
    SimBlock type
• **doSimulation**
  public void doSimulation( )
  
  – **Usage**
  * run the simulation, make a plot with any changed parameters. this function contains the iteration to step through the whole process and make a plot the params must be set before this is called. clear buffers, clear blocks, make the new parameter changes, iterate blocks, then iterate buffers

• **empty**
  public void empty( )
  
  – **Usage**
  * empty the factory buffer of simulation blocks. before emptying all the blocks first remove their inputs

• **getPanel**
  public JPanel getPanel( )

• **getPanelableCount**
  public int getPanelableCount( )
  
  – **Usage**
  * iterate through the list and count the number of panelable blocks

• **hasNext**
  public boolean hasNext( )
  
  – **Usage**
  * iterator function

• **hideParameters**
  public void hideParameters( )
  
  – **Usage**
  * this method is called to hide the parameters from the button list. this should be called before the factory the panel is made for the factory.

• **initSimulation**
  public void initSimulation( )
  
  – **Usage**
  * initialize the simulation
• `iterateBlocks`
  ```java
  public void iterateBlocks()
  ```
  - `Usage`
  * iterate the blocks this iterates without the buffers being advanced

• `iterateBuffers`
  ```java
  public void iterateBuffers()
  ```
  - `Usage`
  * iterate the bufferes, this checks if it HasOutput

• `iterator`
  ```java
  public Iterator iterator()
  ```
  - `Usage`
  * iterator function

• `next`
  ```java
  public Object next()
  ```
  - `Usage`
  * iterator function

• `remove`
  ```java
  public void remove()
  ```
  - `Usage`
  * iterator function

• `removeInputs`
  ```java
  public void removeInputs()
  ```
  - `Usage`
  * remove the inputs from all of the contained blocks that have inputs.

• `toString`
  ```java
  public String toString()
  ```
Methods inherited from class `simtk.SimBlock`

- `clear`
  ```java
  public void clear()
  ```
  - **Usage**
    - this is the general function to clear a block. this is just an empty method but if more specific functionality is needed this should be overridden

- `doIteration`
  ```java
  abstract void doIteration()
  ```
  - **Usage**
    - this is the method that should be implemented in all Simblock's

- `doOutputBufferIteration`
  ```java
  public void doOutputBufferIteration()
  ```
  - **Usage**
    - iterate the output buffer(s), or shift the history of it

- `getPanel`
  ```java
  public JPanel getPanel()
  ```
  - **Usage**
    - generates the default panel and the title

- `getSimParams`
  ```java
  public SimParameters getSimParams()
  ```
  - **Usage**
    - get the parameters from a parameter class

- `getVariableContainer`
  ```java
  public VariableContainer getVariableContainer()
  ```
  - **Usage**
    - returns the conatainer for all the variables. this is mainly used for the operations done on the varContainer

- `isHidden`
  ```java
  public boolean isHidden()
  ```
  - **Usage**
    - test if the hidden flag has been set

- `iterateBuffers`
  ```java
  public void iterateBuffers()
  ```
  - **Usage**
    - This function does nothing, but blocks with outputs or internal buffers such as SimFunctionFactory should overload it for it’s own special pursposes
• `reDefine`
  public void `reDefine`
  – Usage
  * transfer upper level adjustments to the lower level. This method is overloaded when it is to
  be used by a lower function.

• ` setHidden`
  public void `setHidden`
  – Usage
  * keep the block from being loaded to the factory display if it is panelable.

• ` setSimParams`
  public void `setSimParams`
  – Usage
  * set the parameters from a parameter class.

**Methods inherited from class `simtk.SimBase`**

• `getDescription`
  public String `getDescription`

• `getReloadBlock`
  public Simulatable `getReloadBlock`
  – Usage
  * set the block that gets called to resimulate when the a variable is changed.

• `getTitle`
  public String `getTitle`
  – Usage
  * get the user defined title for the entire block.

• `setDescription`
  public void `setDescription`
  – Usage
  * set the user defined title for the entire block.

• `setReloadBlock`
  public void `setReloadBlock`
  – Usage
  * set the block that gets called to resimulate when the a variable is changed.

• `setTitle`
  public void `setTitle`
  – Usage
  * set the user defined title for the entire block.
2.2.18 Class SimFilter

This filter contains multiple types of butterworth filters

Declaration

```java
public class SimFilter
    extends simtk.SimStf
    implements SimPanelable
```

Fields

- private SimComboBoxString typeSel
  - Lowpass or high pass selection of filter type
- private SimComboBoxString orderSel
  - 1st 2nd or 3rd order filters
- public static int LOW
  - low pass filter type
- public static int HIGH
  - high pass filter type
- public static int FIRST
  - order size
- public static int SECOND
  - order size
- public static int THIRD
  - order size

Constructors

- SimFilter
  public SimFilter( )
  - Usage
    * Constructor for Filter
Methods

• **reDefine**
  
  ```java
  public void reDefine()
  ```

  **Usage**

  * This is made to recalculate with new parameters, step sizes parameters. This is a method that might be good to have defined in the sim Block as abstract

Methods inherited from class `simtk.SimStf`

• **displayNotProperErr**
  
  ```java
  private void displayNotProperErr()
  ```

  **Usage**

  * improper functions error

• **generateZPoly**
  
  ```java
  public double generateZPoly( double [] s )
  ```

  **Usage**

  * generate the polynomial as an array from the s polynomial the order is extracted by the size of s this is the magic function for the conversion of continuous to discrete see the derivations done on 4/26/02 with mathematica and the papers solution I need to type up these derivations

• **makeNewStf**
  
  ```java
  public void makeNewStf( int snum_order, int sden_order )
  ```

  **Usage**

  * creates the num and den checking if improper else make default

• **reDefine**
  
  ```java
  public void reDefine()
  ```

  **Usage**

  * this is a method that might be good to have defined in the sim Block as abstract this is made to recalculate with new parameters, step sizes parameters

• **setSDen**
  
  ```java
  public void setSDen( double [] a )
  ```

  **Usage**

  * simply sets the s denominator to an array value

• **setSNum**
  
  ```java
  public void setSNum( double [] a )
  ```

  **Usage**
* simply sets the s numerator to an array value

```
public String toString()

Usage
* toString in the form of: num_order = x den_order = y num = [x] den = [y]
```

* zeroStf

```
public void zeroStf()

Usage
* this zeroSTF method is for convinience in setting up z transfer functions it makes sure that the function is reset to zero from the initial conditions
```

Methods inherited from class simtk.SimZtf

```
public void doIteration()

Usage
* this method overides the doIteration of the diffeq. this way it first redefines the variables and then iterates with the inherited structure

public void doIterationDirect()

Usage
* this is the really cool simple method that converts ztf to diffEQ. The function does the operation in the following order:
  1. parses the numerator and denominator
  2. creates the difference equation
  3. does one step operation.

public void reDefine()

Usage
* this is a method that might be good to have defined in the sim Block as abstract this is made to recalculate with new parameters, step sizes parameters

public void setDen(double[] n)

Usage
* method to set the denominator to the array value Example: setDen([1,2,3]); would set the denominator to z^2 + 2z + 3 an error message is displayed if the array is lager than the z order

public void setNum(double[] n)
```
– **Usage**
  * method to set the numerator to the array value Example: setNum([1,2,3]); would set the
    numerator to z^2 + 2*z + 3 an error message is displayed if the array is larger than the z
    order

• **toString**
  public String toString()

  – **Usage**
    * toString in the form of: zorder = x num = [x] den = [x]

• **zeroNumAndDen**
  public void zeroNumAndDen()

  – **Usage**
    * this method is for convenience in setting up z transfer functions it makes sure that the
      function is reset to zero from the initial conditions

**Methods inherited from class simtk.SimDiffEq**

• **clear**
  public void clear()

  – **Usage**
    * clears the text field, sets up headers and resets the index counter

• **doIteration**
  public void doIteration()

• **getInCoef**
  public SimDoubleArray getInCoef()

  – **Usage**
    * return the input coefficients

• **getOutCoef**
  public SimDoubleArray getOutCoef()

  – **Usage**
    * return the output coefficients

• **setZeros**
  public void setZeros()

  – **Usage**
    * zero all the coefficients for easy settings

• **toDiffEqString**
  public String toDiffEqString()

  – **Usage**
* output the string format of the difference equation. special function to iterate through all of the parameters and create the difference equation. This function is also used for the textBox for the dynamic pretty print viewing of the diff eq

- **toString**
  
  ```java
  public String toString()
  ```
  
  **Usage**
  
  * general to String equation for the diff eq. this uses the toDiffEqString() function for the special formatting so that one doesn’t overloaded

**Methods inherited from class simtk.SimFunction**

- **addInput**
  
  ```java
  public void addInput( simtk.HasOutput outputBlock )
  ```
  
  **Usage**
  
  * this AddInput adds from a HasOutput block an input to the vector function and increases the inputIndex count

- **addInput**
  
  ```java
  public void addInput( simtk.SimBuffer bufref )
  ```
  
  **Usage**
  
  * AddInput adds an input to the vector function and increases the inputIndex count

- **doIteration**
  
  ```java
  abstract void doIteration()
  ```
  
  **Usage**
  
  * this is the method that should be implemented in all SimBlock’s

- **getInput**
  
  ```java
  public SimBuffer getInput()
  ```
  
  **Usage**
  
  * get the first input. there might be a problem here because the old method of the implicit in buffer is used

- **getInput**
  
  ```java
  public SimBuffer getInput( int index )
  ```
  
  **Usage**
  
  * get the indexed input, referenced from as 1 is the first. If the reference is out of range it will return the the single input value. The code is not so clean here where it will still print an error message when the double input functions are just beginning to set themselves up and before the.???

- **getInputCount**
  
  ```java
  public int getInputCount()
  ```
Usage
* get the inputCount, or the number of elements on the list

**getOutput**

```java
public SimBuffer getOutput() {
}
```

Usage
* Gets the output buffer

**iterateBuffers**

```java
public void iterateBuffers() {
}
```

Usage
* overload the do nothing iterateBuffers and iterate the output buffers

**removeInputs**

```java
public void removeInputs() {
}
```

Usage
* remove the all of the inputs from the block so a new system can be set up

**setInput**

```java
public void setInput(int index, simtk.SimBuffer bufref) {
}
```

Usage
* Set the indexed input to the passed buffer. note: this has out of index problems

**setInput**

```java
public void setInput(simtk.SimBuffer bufref) {
}
```

Usage
* Set the input to the passed buffer, this is replaced with the newer addInput function which sounds more like what it does

**setInput1**

```java
public void setInput1(simtk.SimBuffer bufref) {
}
```

Usage
* This is here to support old systems, it just calls addInput(bufref). so it is not exactly input 2

**setInput2**

```java
public void setInput2(simtk.SimBuffer bufref) {
}
```

Usage
* This is here to support old systems, it just calls addInput(bufref). so it is not exactly input 2

**setOutput**

```java
public void setOutput(simtk.SimBuffer bufref) {
}
```

Usage
* sets the output to a passed buffer
clear
  public void clear( )
  
  - Usage
    * this is the general function to clear a block. this is just an empty method but if more
    specific functionality is needed this should be overridden

doIteration
  abstract void doIteration( )
  
  - Usage
    * this is the method that should be implemented in all Simblock’s

doOutputBufferIteration
  public void doOutputBufferIteration( )
  
  - Usage
    * iterate the output buffer(s), or shift the history of it

getPanel
  public JPanel getPanel( )
  
  - Usage
    * generates the default panel and the title

getSimParams
  public SimParameters getSimParams( )
  
  - Usage
    * get the parameters from a parameter class

getVariableContainer
  public VariableContainer getVariableContainer( )
  
  - Usage
    * returns the container for all the variables. this is mainly used for the operations done on
    the varContainer

isHidden
  public boolean isHidden( )
  
  - Usage
    * test if the hidden flag has been set

iterateBuffers
  public void iterateBuffers( )
  
  - Usage
    * This function does nothing, but blocks with outputs or internal buffers such as
      SimFunctionFactory should overload it for it’s own special purposes
**reDefine**

```java
public void reDefine()
```

- **Usage**
  
  * transfer upper level adjustments to the lower level. this method is overloaded when it is to be used by a lower function

**setHidden**

```java
public void setHidden()
```

- **Usage**
  
  * keep the block from being loaded to the factory display if it is panelable

```java
public void setHidden( boolean hide )
```

- **Usage**
  
  * keep the block from being loaded to the factory display if it is panelable

**setSimParams**

```java
public void setSimParams( simtk.SimParameters sp )
```

- **Usage**
  
  * set the parameters from a parameter class

**Methods inherited from class simtk.SimBase**

**getDescription**

```java
public String getDescription()
```

**getReloadBlock**

```java
public Simulatable getReloadBlock()
```

- **Usage**
  
  * set the block that gets called to resimulate when the a variable is changed.

**getTitle**

```java
public String getTitle()
```

- **Usage**
  
  * get the user defined title for the entire block

**setDescription**

```java
public void setDescription( java.lang.String desc )
```

**setReloadBlock**

```java
public void setReloadBlock( simtk.Simulatable reloadBlock )
```

- **Usage**
  
  * set the block that gets called to resimulate when the a variable is changed.

**setTitle**

```java
public void setTitle( java.lang.String title )
```

- **Usage**
  
  * set the user defined title for the entire block
2.2.19 Class SimFunction

defines the bare bones to a Function block, now multi-input

DECLARATION

```java
public abstract class SimFunction
extends simtk.SimBlock
implements HasInput, HasOutput
```

FIELDS

- public SimBuffer out
  - the single output buffer
- public SimBuffer in
  - used just for older functions, new ones come from the bufVec
- private int inputIndex
  - initial default input is inputCount = 1

CONSTRUCTORS

- `SimFunction`
  ```java
  public SimFunction()
  ```
  - Usage
    * constructor to make the output buffer

METHODS

- `addInput`
  ```java
  public void addInput( simtk.HasOutput outputBlock )
  ```
  - Usage
    * this AddInput adds from a HasOutput block an input to the vector function and increases the inputIndex count
- **addInput**
  
  ```java
  public void addInput( simtk.SimBuffer bufref )
  ```
  
  **Usage**
  
  * AddInput adds an input to the vector function and increases the inputIndex count

- **doIteration**

  ```java
  abstract void doIteration( )
  ```
  
  **Usage**
  
  * this is the method that should be implemented in all SimBlock's

- **getInput**

  ```java
  public SimBuffer getInput( )
  ```
  
  **Usage**
  
  * get the first input. there might be a problem here because the old method of the implicit in buffer is used

- **getInput**

  ```java
  public SimBuffer getInput( int index )
  ```
  
  **Usage**
  
  * get the indexed input, referenced from as 1 is the first.
  
  If the reference is out of range it will return the single input value.
  
  The code is not so clean here where it will still print an error message when the double input functions are just beginning to set themselves up and before the...

- **getInputCount**

  ```java
  public int getInputCount( )
  ```
  
  **Usage**
  
  * get the inputCount, or the number of elements on the list

- **getOutput**

  ```java
  public SimBuffer getOutput( )
  ```
  
  **Usage**
  
  * Gets the output buffer

- **iterateBuffers**

  ```java
  public void iterateBuffers( )
  ```
  
  **Usage**
  
  * overload the do nothing iterateBuffers and iterate the output buffers
• **removeInputs**
  public void removeInputs()  
  
  – **Usage**  
  * remove the all of the inputs from the block so a new system can be set up

• **setInput**
  public void setInput( int index, simtk.SimBuffer bufref )  
  
  – **Usage**  
  * Set the indexed input to the passed buffer. note: this has out of index problems

• **setInput**
  public void setInput( simtk.SimBuffer bufref )  
  
  – **Usage**  
  * Set the input to the passed buffer, this is replaced with the newer addInput function which sounds more like what it does

• **setInput1**
  public void setInput1( simtk.SimBuffer bufref )  
  
  – **Usage**  
  * This is here to support old systems, it just calls addInput(bufref). so it is not exactly input 2

• **setInput2**
  public void setInput2( simtk.SimBuffer bufref )  
  
  – **Usage**  
  * This is here to support old systems, it just calls addInput(bufref). so it is not exactly input 2

• **setOutput**
  public void setOutput( simtk.SimBuffer bufref )  
  
  – **Usage**  
  * sets the output to a passed buffer
Methods inherited from class `simtk.SimBlock`

- **clear**
  public void clear()
  
  - Usage
    * this is the general function to clear a block. This is just an empty method but if more specific functionality is needed this should be overridden

- **doIteration**
  abstract void doIteration()
  
  - Usage
    * this is the method that should be implemented in all SimBlock's

- **doOutputBufferIteration**
  public void doOutputBufferIteration()
  
  - Usage
    * iterate the output buffer(s), or shift the history of it

- **getPanel**
  public JPanel getPanel()
  
  - Usage
    * generates the default panel and the title

- **getSimParams**
  public SimParameters getSimParams()
  
  - Usage
    * get the parameters from a parameter class

- **getVariableContainer**
  public VariableContainer getVariableContainer()
  
  - Usage
    * returns the container for all the variables. This is mainly used for the operations done on the varContainer

- **isHidden**
  public boolean isHidden()
  
  - Usage
    * test if the hidden flag has been set

- **iterateBuffers**
  public void iterateBuffers()
  
  - Usage
    * This function does nothing, but blocks with outputs or internal buffers such as SimFunctionFactory should overload it for its own special purposes
**reDefine**

```java
public void reDefine() {
    // Usage
    * transfer upper level adjustments to the lower level. This method is overloaded when it is to be used by a lower function
}
```

**setHidden**

```java
public void setHidden() {
    // Usage
    * keep the block from being loaded to the factory display if it is panelable
}
```

**setHidden**

```java
public void setHidden(boolean hide) {
    // Usage
    * keep the block from being loaded to the factory display if it is panelable
}
```

**setSimParams**

```java
public void setSimParams(simtk.SimParameters sp) {
    // Usage
    * set the parameters from a parameter class
}
```

### Methods inherited from class simtk.SimBase

**getDescription**

```java
public String getDescription() {
}
```

**getReloadBlock**

```java
public Simulatable getReloadBlock() {
    // Usage
    * set the block that gets called to resimulate when the a variable is changed.
}
```

**getTitle**

```java
public String getTitle() {
    // Usage
    * get the user defined title for the entire block
}
```

**setDescription**

```java
public void setDescription(java.lang.String desc) {
}
```

**setReloadBlock**

```java
public void setReloadBlock(simtk.Simulatable reloadBlock) {
    // Usage
    * set the block that gets called to resimulate when the a variable is changed.
}
```

**setTitle**

```java
public void setTitle(java.lang.String title) {
    // Usage
    * set the user defined title for the entire block
}
```
2.2.20 Class SimFunction2Input

general class for 2 inputs and 1 output

DECLARATION

```
public abstract class SimFunction2Input
extends simtk.SimBlock
implements HasOutput
```

FIELDS

- public SimBuffer out
- public SimBuffer in1
- public SimBuffer in2

CONSTRUCTORS

- `SimFunction2Input`
  public `SimFunction2Input`()

METHODS

- `doIteration`
  abstract void `doIteration`()
  - Usage
    * this is the method that should be implemented in all SimBlock’s

- `getOutput`
  public SimBuffer `getOutput`()
  - Usage
• setInput1
  public void setInput1( simtk.SimBuffer bufref )
  – Usage
  * set the first Input to the passed buffer

• setInput2
  public void setInput2( simtk.SimBuffer bufref )
  – Usage
  * set the second Input to the passed buffer

• setOutput
  public void setOutput( simtk.SimBuffer bufref )
  – Usage
  * set the output buffer to the passed buffer

Methods inherited from class simtk.SimBlock

• clear
  public void clear( )
  – Usage
  * this is the general function to clear a block. this is just an empty method but if more specific functionality is needed this should be overridden

• doIteration
  abstract void doIteration( )
  – Usage
  * this is the method that should be implemented in all Simblock’s

• doOutputBufferIteration
  public void doOutputBufferIteration( )
  – Usage
  * iterate the output buffer(s), or shift the history of it

• getPanel
  public JPanel getPanel( )
  – Usage
  * generates the default panel and the title
• `getSimParams`
  public SimParameters `getSimParams()`
  
  Usage
  * get the parameters from a parameter class

• `getVariableContainer`
  public VariableContainer `getVariableContainer()`
  
  Usage
  * returns the container for all the variables. this is mainly used for the operations done on
  the varContainer

• `isHidden`
  public boolean `isHidden()`
  
  Usage
  * test if the hidden flag has been set

• `iterateBuffers`
  public void `iterateBuffers()`
  
  Usage
  * This function does nothing, but blocks with outputs or internal buffers such as
    SimFunctionFactory should overload it for its own special purposes

• `reDefine`
  public void `reDefine()`
  
  Usage
  * transfer upper level adjustments to the lower level. this method is overloaded when it is to
  be used by a lower function

• `setHidden`
  public void `setHidden()`
  
  Usage
  * keep the block from being loaded to the factory display if it is panelable

• `setHidden`
  public void `setHidden` (boolean `hide`)
  
  Usage
  * keep the block from being loaded to the factory display if it is panelable

• `setSimParams`
  public void `setSimParams` (simtk.SimParameters `sp`)
  
  Usage
  * set the parameters from a parameter class
Methods inherited from class simtk.SimBase

- `getDescription`
  ```java
  public String getDescription()
  ```

- `getReloadBlock`
  ```java
  public Simulatable getReloadBlock()
  ```
  - **Usage**
    * set the block that gets called to resimulate when the a variable is changed.

- `getTitle`
  ```java
  public String getTitle()
  ```
  - **Usage**
    * get the user defined title for the entire block

- `setDescription`
  ```java
  public void setDescription(java.lang.String desc)
  ```

- `setReloadBlock`
  ```java
  public void setReloadBlock(simtk.Simulatable reloadBlock)
  ```
  - **Usage**
    * set the block that gets called to resimulate when the a variable is changed.

- `setTitle`
  ```java
  public void setTitle(java.lang.String title)
  ```
  - **Usage**
    * set the user defined title for the entire block

2.2.21 Classe **SimFunctionFactory**

compilation of sim blocks to create the complex blocks such as NPID from simple building blocks.

almost every function from SimFunction is overloaded for the special purpose of a contained function factory

???Need to proably add the removeInputs for this implementation of a function

**Declaration**

```java
public abstract class SimFunctionFactory
extends simtk.SimFunction
implements SimPanelable, ContainsFactory
```
Constructors

- **SimFunctionFactory**
  
  ```java
  public SimFunctionFactory()
  ```
  
  - **Usage**
    - * constructor sets gain to default of unity

Methods

- **addInput**
  
  ```java
  public void addInput(simtk.HasOutput inputBlock)
  ```
  
  - **Usage**
    - * set the input of this factory this is the interesting one

- **addInput**
  
  ```java
  public void addInput(simtk.SimBuffer sb)
  ```
  
  - **Usage**
    - * set the input of this factory. this is the interesting one

- **clear**
  
  ```java
  public void clear()
  ```
  
  - **Usage**
    - * this function is required to reset the buffers and values for the next iterations to set up the factory class

- **connectBlocks**
  
  ```java
  abstract void connectBlocks()
  ```
  
  - **Usage**
    - * Make the connections for this block. The setInput of this factory calls this to setup. It turns out to be a rather interesting system. The setInput() or addInput() to this factory then in turn set’s up this internal system. this is to be implemented.

- **doIteration**
  
  ```java
  public void doIteration()
  ```
  
  - **Usage**
    - * do a calculation for the new output
• **getPanel**
  
  ```java
  public JPanel getPanel()
  ```

• **iterateBuffers**
  
  ```java
  public void iterateBuffers()
  ```

  – **Usage**
    
    * cycle the buffers

• **setInput**
  
  ```java
  public void setInput( simtk.SimBuffer sb )
  ```

  – **Usage**
    
    * set the input of this factory. this is to be implemented in the Simblock to call the
      connect blocks

• **toString**
  
  ```java
  public String toString()
  ```

**Methods inherited from class simtk.SimFunction**

• **addInput**
  
  ```java
  public void addInput( simtk.HasOutput outputBlock )
  ```

  – **Usage**
    
    * this AddInput adds from a HasOutput block an input to the vector function and increases
      the inputIndex count

• **addInput**
  
  ```java
  public void addInput( simtk.SimBuffer bufref )
  ```

  – **Usage**
    
    * AddInput adds an input to the vector function and increases the inputIndex count

• **doIteration**
  
  ```java
  abstract void doIteration()
  ```

  – **Usage**
    
    * this is the method that should be implemented in all SimBlock’s

• **getInput**
  
  ```java
  public SimBuffer getInput()
  ```

  – **Usage**
    
    * get the first input. there might be a problem here because the old method of the implicit
      in buffer is used
- **getInput**
  public SimBuffer getInput( int index )
  
  - **Usage**
    * get the indexed input, referenced from as 1 is the first.
      If the reference is out of range it will return the single input value.
      The code is not so clean here where it will still print an error message when the double input functions are just beginning to set themselves up and before the.

- **getInputCount**
  public int getInputCount( )
  
  - **Usage**
    * get the inputCount, or the number of elements on the list

- **getOutput**
  public SimBuffer getOutput( )
  
  - **Usage**
    * Gets the output buffer

- **iterateBuffers**
  public void iterateBuffers( )
  
  - **Usage**
    * overload the do nothing iterateBuffers and iterate the output buffers

- **removeInputs**
  public void removeInputs( )
  
  - **Usage**
    * remove the all of the inputs from the block so a new system can be set up

- **setInput**
  public void setInput( int index, simtk.SimBuffer bufref )
  
  - **Usage**
    * Set the indexed input to the passed buffer. note: this has out of index problems

- **setInput**
  public void setInput( simtk.SimBuffer bufref )
  
  - **Usage**
    * Set the input to the passed buffer, this is replaced with the newer addInput function which sounds more like what it does

- **setInput1**
  public void setInput1( simtk.SimBuffer bufref )
  
  - **Usage**
    * This is here to support old systems, it just calls addInput(bufref). so it is not exactly
• `setInput2`
  public void `setInput2` (simtk.SimBuffer `bufref`)
  
  – Usage
  * This is here to support old systems, it just calls `addInput(bufref)`. so it is not exactly
  input 2

• `setOutput`
  public void `setOutput` (simtk.SimBuffer `bufref`)
  
  – Usage
  * sets the output to a passed buffer

Methods inherited from class `simtk.SimBlock`

• `clear`
  public void `clear`
  
  – Usage
  * this is the general function to clear a block. this is just an empty method but if more
  specific functionality is needed this should be overridden

• `doIteration`
  abstract void `doIteration`
  
  – Usage
  * this is the method that should be implemented in all Simblock’s

• `doOutputBufferIteration`
  public void `doOutputBufferIteration`
  
  – Usage
  * iterate the output buffer(s), or shift the history of it

• `getPanel`
  public JPanel `getPanel`
  
  – Usage
  * generates the default panel and the title

• `getSimParams`
  public SimParameters `getSimParams`
  
  – Usage
  * get the parameters from a parameter class

• `getVariableContainer`
  public VariableContainer `getVariableContainer`
  
  – Usage
* returns the container for all the variables. this is mainly used for the operations done on the varContainer

- **isHidden**
  ```java
  public boolean isHidden()
  ```
  - **Usage**
    * test if the hidden flag has been set

- **iterateBuffers**
  ```java
  public void iterateBuffers()
  ```
  - **Usage**
    * This function does nothing, but blocks with outputs or internal buffers such as SimFunctionFactory should overload it for its own special purposes

- **reDefine**
  ```java
  public void reDefine()
  ```
  - **Usage**
    * transfer upper level adjustments to the lower level. this method is overloaded when it is to be used by a lower function

- **setHidden**
  ```java
  public void setHidden()
  ```
  - **Usage**
    * keep the block from being loaded to the factory display if it is panelable

- **setHidden**
  ```java
  public void setHidden( boolean hide )
  ```
  - **Usage**
    * keep the block from being loaded to the factory display if it is panelable

- **setSimParams**
  ```java
  public void setSimParams( simtk.SimParameters sp )
  ```
  - **Usage**
    * set the parameters from a parameter class

**Methods inherited from class simtk.SimBase**

- **getDescription**
  ```java
  public String getDescription()
  ```

- **getReloadBlock**
  ```java
  public Simulatable getReloadBlock()
  ```
  - **Usage**
    * set the block that gets called to resimulate when the a variable is changed.
• **getTitle**
  ```java
  public String getTitle()
  ```
  — Usage
  * get the user defined title for the entire block

• **setDescription**
  ```java
  public void setDescription(java.lang.String desc)
  ```

• **setReloadBlock**
  ```java
  public void setReloadBlock(simtk.Simulatable reloadBlock)
  ```
  — Usage
  * set the block that gets called to resimulate when the a variable is changed.

• **setTitle**
  ```java
  public void setTitle(java.lang.String title)
  ```
  — Usage
  * set the user defined title for the entire block

### 2.2.22 Class `SimFunctionSelector`

This is one of the most simple function blocks to select different usable functions. Not quite implemented yet, this has lots of problems, I need to think about a better method to provide this functionality of switching function types.

#### Declaration

```java
public class SimFunctionSelector
extends simtk.SimFunction
implements SimPanelable
```

#### Fields

- public `SimComboBoxString cb`
  - the Function to select. To programatically select of function for example, selecting the differentiator use:
    ```java
    SimFunctionSelector ss = new SimFunctionSelector;
    ss.cb.setValue(ss.DIFF)
    ```

- public static `int GAIN`
- used by the fs.cb.setValue to select a **gain** block type

- **public static int DIFF**
  - used by the fs.cb.setValue to select a **differentiator** block type

- **public static int INT**
  - used by the fs.cb.setValue to select a **integrator** block type

- **public static int INTINT**
  - used by the fs.cb.setValue to select a **double integrator** block type

- **public static int FIRST_ORDER**
  - used by the fs.cb.setValue to select a **1st order** block type

- **public static int SECOND_ORDER**
  - used by the fs.cb.setValue to select a **2nd order** block type

- **public static int PID**
  - used by the fs.cb.setValue to select a **PID** block type

**Constructors**

- **SimFunctionSelector**
  
  ```java
  public SimFunctionSelector()
  ```

  - Usage
    
    * constructor sets gain to default of unity

**Methods**

- **checkSFChange**
  
  ```java
  public boolean checkSFChange()
  ```

  - Usage
    
    * Check if the selectable function was changed from the last time, if not don’t do anything

- **clear**
  
  ```java
  public void clear()
  ```

  - Usage
    
    * called before doSimulation, sets up the internal block
• **doIteration**
  
  public void doIteration( )

  – Usage
  * this method overrides the doIteration of the sf, sim function.

• **getPanel**
  
  public JPanel getPanel( )

  – Usage
  * gets the internal sf get panel and the combo selection box

• **iterateBuffers**
  
  public void iterateBuffers( )

  – Usage
  * overload the do nothing iterateBuffers and iterate the output buffers

• **reDefine**
  
  public void reDefine( )

  – Usage
  * this is a method that might be good to have defined in the sim Block as abstract
  this is made to recalculate with new parameters, step sizes parameters

• **refreshPanel**
  
  public void refreshPanel( )

  – Usage
  * refreshes the internal get panel and the combo selection box

• **setupInternalSimBlock**
  
  public void setupInternalSimBlock( )

  – Usage
  * This copies the inherited function selector outputs to the selected function for usage. the steps here are rather important, this function is called by the clear because at the begining of every simulation the clear function is called.
  A rather important easy overlooked step in here is to set the reload blocks of each of the internal Variables
Methods inherited from class simtk.SimFunction

- **addInput**
  public void addInput( simtk.HasOutput outputBlock )
  
  - Usage
    * this AddInput adds from a HasOutput block an input to the vector function and increases the inputIndex count

- **addInput**
  public void addInput( simtk.SimBuffer bufref )
  
  - Usage
    * AddInput adds an input to the vector function and increases the inputIndex count

- **doIteration**
  abstract void doIteration( )
  
  - Usage
    * this is the method that should be implemented in all SimBlock's

- **getInput**
  public SimBuffer getInput( )
  
  - Usage
    * get the first input. there might be a problem here because the old method of the implicit in buffer is used

- **getInput**
  public SimBuffer getInput( int index )
  
  - Usage
    * get the indexed input, referenced from as 1 is the first.
      If the reference is out of range it will return the the single input value.
      The code is not so clean here where it will still print an erroo message when the double input functions are just beginning to set themselves up and before the.???

- **getInputCount**
  public int getInputCount( )
  
  - Usage
    * get the inputCount, or the number of elements on the list

- **getOutput**
  public SimBuffer getOutput( )
  
  - Usage
    * Gets the output buffer

- **iterateBuffers**
  public void iterateBuffers( )
Usage
* overload the do nothing iterateBuffers and iterate the output buffers

**removeInputs**
```java
public void removeInputs()
```

Usage
* remove the all of the inputs from the block so a new system can be set up

**setInput**
```java
public void setInput( int index, simtk.SimBuffer bufref )
```

Usage
* Set the indexed input to the passed buffer. note: this has out of index problems

**setInput**
```java
public void setInput( simtk.SimBuffer bufref )
```

Usage
* Set the input to the passed buffer, this is replaced with the newer addInput function which sounds more like what it does

**setInput1**
```java
public void setInput1( simtk.SimBuffer bufref )
```

Usage
* This is here to support old systems, it just calls addInput(bufref). so it is not exactly input 2

**setInput2**
```java
public void setInput2( simtk.SimBuffer bufref )
```

Usage
* This is here to support old systems, it just calls addInput(bufref). so it is not exactly input 2

**setOutput**
```java
public void setOutput( simtk.SimBuffer bufref )
```

Usage
* sets the output to a passed buffer

Methods inherited from class simtk.SimBlock

**clear**
```java
public void clear()
```

Usage
* this is the general function to clear a block. this is just an empty method but if more specific functionality is needed this should be overridden
• *doIteration*
  abstract void doIteration( )
  – Usage
    * this is the method that should be implemented in all Simblock's

• *doOutputBufferIteration*
  public void doOutputBufferIteration( )
  – Usage
    * iterate the output buffer(s), or shift the history of it

• *getPanel*
  public JPanel getPanel( )
  – Usage
    * generates the default panel and the title

• *getSimParams*
  public SimParameters getSimParams( )
  – Usage
    * get the parameters from a parameter class

• *getVariableContainer*
  public VariableContainer getVariableContainer( )
  – Usage
    * returns the container for all the variables. this is mainly used for the operations done on
    the varContainer

• *isHidden*
  public boolean isHidden( )
  – Usage
    * test if the hidden flag has been set

• *iterateBuffers*
  public void iterateBuffers( )
  – Usage
    * This function does nothing, but blocks with outputs or internal buffers such as
      SimFunctionFactory should overload it for it's own special purposes

• *reDefine*
  public void reDefine( )
  – Usage
    * transfer upper level adjustments to the lower level. this method is overloaded when it is to
      be used by a lower function

• *setHidden*
  public void setHidden( )
Usage
* keep the block from being loaded to the factory display if it is panelable

• setHidden
public void setHidden( boolean hide )

Usage
* keep the block from being loaded to the factory display if it is panelable

• setSimParams
public void setSimParams( simtk.SimParameters sp )

Usage
* set the parameters from a parameter class

Methods inherited from class simtk.SimBase

• getDescription
public String getDescription( )

• getReloadBlock
public Simulatable getReloadBlock( )

Usage
* set the block that gets called to resimulate when the a variable is changed.

• getTitle
public String getTitle( )

Usage
* get the user defined title for the entire block

• setDescription
public void setDescription( java.lang.String desc )

• setReloadBlock
public void setReloadBlock( simtk.Simulatable reloadBlock )

Usage
* set the block that gets called to resimulate when the a variable is changed.

• setTitle
public void setTitle( java.lang.String title )

Usage
* set the user defined title for the entire block

2.2.23 Class SimFunctionTest

compilation of sim blocks to create the NPID
**Declaration**

```java
public class SimFunctionTest
extends simtk.SimFunctionFactory
implements SimPanelable, ContainsFactory
```

**Constructors**

- `SimFunctionTest`

  ```java
  public SimFunctionTest()
  ```

  - Usage
    * constructor

**Methods**

- `connectBlocks`

  ```java
  public void connectBlocks()
  ```

  - Usage
    * Make the connections for this block. The setInput of this factory callls this to setup. It turns out to be a rather interesting system the setInput to this factory then in turn setups up this internal system

**Methods inherited from class simtk.SimFunctionFactory**

- `addInput`

  ```java
  public void addInput( simtk.HasOutput inputBlock )
  ```

  - Usage
    * set the input of this factory this is the interesting one

- `addInput`

  ```java
  public void addInput( simtk.SimBuffer sb )
  ```

  - Usage
    * set the input of this factory. this is the interesting one

- `clear`

  ```java
  public void clear()
  ```

  - Usage
* this function is required to reset the buffers and values for the next iterations to set up the
  factory class

- **connectBlocks**
  abstract void connectBlocks()

  - **Usage**
    * Make the connections for this block. The setInput of this factory calls this to setup. It
turns out to be a rather interesting system. The setInput() or addInput() to this factory
then in turn set’s up this internal system.
  this is to be implemented.

- **doIteration**
  public void doIteration()

  - **Usage**
    * do a calculation for the new output

- **getPanel**
  public JPanel getPanel()

- **iterateBuffers**
  public void iterateBuffers()

  - **Usage**
    * cycle the buffers

- **setInput**
  public void setInput( simtk.SimBuffer sb )

  - **Usage**
    * set the input of this factory. this is to be implemented in the Simblock to call the connect
blocks

- **toString**
  public String toString()

**Methods inherited from class simtk.SimFunction**

- **addInput**
  public void addInput( simtk.HasOutput outputBlock )

  - **Usage**
    * this AddInput adds from a HasOutput block an input to the vector function and increases
the inputIndex count

- **addInput**
  public void addInput( simtk.SimBuffer bufref )

  - **Usage**
    * AddInput adds an input to the vector function and increases the inputIndex count
• **doIteration**
  abstract void doIteration()  
  
  -- Usage  
  * this is the method that should be implemented in all SimBlock's

• **getInput**
  public SimBuffer getInput()  
  
  -- Usage  
  * get the first input. there might be a problem here because the old method of the implicit  
  in buffer is used

• **getInput**
  public SimBuffer getInput(int index)  
  
  -- Usage  
  * get the indexed input, referenced from as 1 is the first.  
  If the reference is out of range it will return the the single input value.  
  The code is not so clean here where it will still print an erro message when the double  
  input functions are just beginning to set themselves up and before the.???

• **getInputCount**
  public int getInputCount()  
  
  -- Usage  
  * get the inputCount, or the number of elements on the list

• **getOutput**
  public SimBuffer getOutput()  
  
  -- Usage  
  * Gets the output buffer

• **iterateBuffers**
  public void iterateBuffers()  
  
  -- Usage  
  * overload the do nothing iterateBuffers and iterate the output buffers

• **removeInputs**
  public void removeInputs()  
  
  -- Usage  
  * remove the all of the inputs from the block so a new system can be set up

• **setInput**
  public void setInput(int index, simtk.SimBuffer bufref)  
  
  -- Usage  
  * Set the indexed input to the passed buffer. note: this has out of index problems
• setInput
   public void setInput( simtk.SimBuffer bufref )
   – Usage
     * Set the input to the passed buffer, this is replaced with the newer addInput function which
       sounds more like what it does

• setInput1
   public void setInput1( simtk.SimBuffer bufref )
   – Usage
     * This is here to support old systems, it just calls addInput(bufref). so it is not exactly
       input 2

• setInput2
   public void setInput2( simtk.SimBuffer bufref )
   – Usage
     * This is here to support old systems, it just calls addInput(bufref). so it is not exactly
       input 2

• setOutput
   public void setOutput( simtk.SimBuffer bufref )
   – Usage
     * sets the output to a passed buffer

METHODS INHERITED FROM CLASS simtk.SimBlock

• clear
   public void clear( )
   – Usage
     * this is the general function to clear a block. this is just an empty method but if more
       specific functionality is needed this should be overridden

• doIteration
   abstract void doIteration( )
   – Usage
     * this is the method that should be implemented in all Simblock’s

• doOutputBufferIteration
   public void doOutputBufferIteration( )
   – Usage
     * iterate the output buffer(s), or shift the history of it

• getPanel
   public JPanel getPanel( )
• **Usage**
  * generates the default panel and the title

• **getSimParams**
  public SimParameters getSimParams()  
  
  • **Usage**
  * get the parameters from a parameter class

• **getVariableContainer**
  public VariableContainer getVariableContainer()  
  
  • **Usage**
  * returns the container for all the variables. this is mainly used for the operations done on the varContainer

• **isHidden**
  public boolean isHidden()  
  
  • **Usage**
  * test if the hidden flag has been set

• **iterateBuffers**
  public void iterateBuffers()  
  
  • **Usage**
  * This function does nothing, but blocks with outputs or internal buffers such as SimFunctionFactory should overload it for its own special purposes

• **reDefine**
  public void reDefine()  
  
  • **Usage**
  * transfer upper level adjustments to the lower level. this method is overloaded when it is to be used by a lower function

• **setHidden**
  public void setHidden()  
  
  • **Usage**
  * keep the block from being loaded to the factory display if it is panelable

• **setHidden**
  public void setHidden( boolean hide )  
  
  • **Usage**
  * keep the block from being loaded to the factory display if it is panelable

• **setSimParams**
  public void setSimParams( simtk.SimParameters sp )  
  
  • **Usage**
  * set the parameters from a parameter class
Methods inherited from class `simtk.SimBase`

- `getDescription`
  ```java
  public String getDescription()
  ```
  
  **Usage**
  * set the block that gets called to resimulate when the a variable is changed.

- `getTitle`
  ```java
  public String getTitle()
  ```
  
  **Usage**
  * get the user defined title for the entire block

- `setDescription`
  ```java
  public void setDescription(java.lang.String desc)
  ```

- `setReloadBlock`
  ```java
  public void setReloadBlock(simtk.Simulatable reloadBlock)
  ```
  
  **Usage**
  * set the block that gets called to resimulate when the a variable is changed.

- `setTitle`
  ```java
  public void setTitle(java.lang.String title)
  ```
  
  **Usage**
  * set the user defined title for the entire block

### 2.2.24 Class `SimFunctionXInput`

defines the bare bones to a Sink block

**Declaration**

```java
public abstract class SimFunctionXInput
extends simtk.SimBlock
```
**Fields**

- public SimBuffer out
  -
- public SimBuffer in
  -
- private int inputIndex
  - initial default input is inputCount = 1

**Constructors**

- `SimFunctionXInput`
  public `SimFunctionXInput()`

**Methods**

- `addInput`
  public void `addInput( simtk.SimBuffer bufref )`
  - Usage
    * addInput adds an input to the vector function and increases the inputIndex count

- `doIteration`
  abstract void `doIteration()`
  - Usage
    * this is the method that should be implemented in all SimBlock’s

- `getInput`
  public SimBuffer `getInput()`
  - Usage
    * get the first input

- `getInput`
  public SimBuffer `getInput( int index )`
  - Usage
    * get the indexed input, referenced from as 1 is the first
• **getInputCount**
  public int getInputCount()

  – Usage
  * get the inputCount, or the number of elements on the list

• **getOutput**
  public SimBuffer getOutput()

  – Usage
  * gets the output buffer

• **setInput**
  public void setInput( int index, simtk.SimBuffer bufref )

  – Usage
  * set the indexed input to the passed buffer this has out of index problems

• **setInput**
  public void setInput( simtk.SimBuffer bufref )

  – Usage
  * set the input to the passed buffer, this is replaced with the newer addInput function which sounds more like what it does

• **setOutput**
  public void setOutput( simtk.SimBuffer bufref )

  – Usage
  * sets the output to a passed buffer

**Methods inherited from class simtk.SimBlock**

• **clear**
  public void clear()

  – Usage
  * this is the general function to clear a block. this is just an empty method but if more specific functionality is needed this should be overridden

• **doIteration**
  abstract void doIteration()

  – Usage
  * this is the method that should be implemented in all Simblock’s
- **doOutputBufferIteration**
  public void doOutputBufferIteration( )
  
      * Usage
      * iterate the output buffer(s), or shift the history of it

- **getPanel**
  public JPanel getPanel( )
  
      * Usage
      * generates the default panel and the title

- **getSimParams**
  public SimParameters getSimParams( )
  
      * Usage
      * get the parameters from a parameter class

- **getVariableContainer**
  public VariableContainer getVariableContainer( )
  
      * Usage
      * returns the conatainer for all the variables. this is mainly used for the operations done on the varContainer

- **isHidden**
  public boolean isHidden( )
  
      * Usage
      * test if the hidden flag has been set

- **iterateBuffers**
  public void iterateBuffers( )
  
      * Usage
      * This function does nothing, but blocks with outputs or internal buffers such as SimFunctionFactory should overload it for it’s own special purposes

- **reDefine**
  public void reDefine( )
  
      * Usage
      * transfer upper level adjustments to the lower level. this method is overloaded when it is to be used by a lower function

- **setHidden**
  public void setHidden( )
  
      * Usage
      * keep the block from being loaded to the factory display if it is panelable

- **setHidden**
  public void setHidden( boolean hide )
− Usage
   * keep the block from being loaded to the factory display if it is panelable

setSimParams
public void setSimParams( simtk.SimParameters sp )
− Usage
   * set the parameters from a parameter class

Methods inherited from class simtk.SimBase

− getDescription
   public String getDescription()

− getReloadBlock
   public Simulatable getReloadBlock()
   − Usage
     * set the block that gets called to resimulate when the a variable is changed.

− getTitle
   public String getTitle()
   − Usage
     * get the user defined title for the entire block

− setDescription
   public void setDescription( java.lang.String desc )

− setReloadBlock
   public void setReloadBlock( simtk.Simulatable reloadBlock )
   − Usage
     * set the block that gets called to resimulate when the a variable is changed.

− setTitle
   public void setTitle( java.lang.String title )
   − Usage
     * set the user defined title for the entire block

2.2.25 Class SimGain

this is one of the most simple function blocks

Declaration

public class SimGain
   extends simtk.SimFunction
   implements SimPanelable
CONSTRUCTORS

- SimGain
  public SimGain()

  - Usage
    * constructor sets gain to default of unity

METHODS

- doIteration
  public void doIteration()

  - Usage
    * this method overrides the doIteration of the diffeq. this way it first redefines the
      variables and then iterates with the inherited structure

- doIterationDirect
  public void doIterationDirect()

  - Usage
    * Do a calculation for the new output without using the diffEq. This is the
      implementation without usign the inherited diffEq.

- setGain
  public void setGain(double g)

  - Usage
    * set the adjustable gain

- toString
  public String toString()

METHODS INHERITED FROM CLASS simtk.SimFunction

- addInput
  public void addInput(simtk.HasOutput outputBlock)

  - Usage
    * this AddInput adds from a HasOutput block an input to the vector function and increases
      the inputIndex count
• addInput
  public void addInput( simtk.SimBuffer bufref )
  
  – Usage
  * AddInput adds an input to the vector function and increases the inputIndex count

• doIteration
  abstract void doIteration( )
  
  – Usage
  * this is the method that should be implemented in all SimBlock’s

• getInput
  public SimBuffer getInput( )
  
  – Usage
  * get the first input. there might be a problem here because the old method of the implicit in buffer is used

• getInput
  public SimBuffer getInput( int index )
  
  – Usage
  * get the indexed input, referenced from as 1 is the first.
  If the reference is out of range it will return the the single input value.
  The code is not so clean here where it will still print an erro message when the double input functions are just beginning to set themselves up and before the.???

• getInputCount
  public int getInputCount( )
  
  – Usage
  * get the inputCount, or the number of elements on the list

• getOutput
  public SimBuffer getOutput( )
  
  – Usage
  * Gets the output buffer

• iterateBuffers
  public void iterateBuffers( )
  
  – Usage
  * overload the do nothing iterateBuffers and iterate the output buffers

• removeInputs
  public void removeInputs( )
  
  – Usage
  * remove the all of the inputs from the block so a new system can be set up
• `setInput`
  public void `setInput`( int index, simtk.SimBuffer bufref )
  – Usage
    * Set the indexed input to the passed buffer. note: this has out of index problems

• `setInput`
  public void `setInput`( simtk.SimBuffer bufref )
  – Usage
    * Set the input to the passed buffer, this is replaced with the newer addInput function which sounds more like what it does

• `setInput1`
  public void `setInput1`( simtk.SimBuffer bufref )
  – Usage
    * This is here to support old systems, it just calls addInput(bufref). so it is not exactly input 2

• `setInput2`
  public void `setInput2`( simtk.SimBuffer bufref )
  – Usage
    * This is here to support old systems, it just calls addInput(bufref). so it is not exactly input 2

• `setOutput`
  public void `setOutput`( simtk.SimBuffer bufref )
  – Usage
    * sets the output to a passed buffer

**Methods inherited from class** simtk.SimBlock

• `clear`
  public void `clear`( )
  – Usage
    * this is the general function to clear a block. this is just an empty method but if more specific functionality is needed this should be overridden

• `doIteration`
  abstract void `doIteration`( )
  – Usage
    * this is the method that should be implemented in all Simblock’s

• `doOutputBufferIteration`
  public void `doOutputBufferIteration`( )
- **Usage**
  * iterate the output buffer(s), or shift the history of it

- **getPanel**
  ```java
  public JPanel getPanel()
  ```
  - **Usage**
    * generates the default panel and the title

- **getSimParams**
  ```java
  public SimParameters getSimParams()
  ```
  - **Usage**
    * get the parameters from a parameter class

- **getVariableContainer**
  ```java
  public VariableContainer getVariableContainer()
  ```
  - **Usage**
    * returns the container for all the variables. this is mainly used for the operations done on the varContainer

- **isHidden**
  ```java
  public boolean isHidden()
  ```
  - **Usage**
    * test if the hidden flag has been set

- **iterateBuffers**
  ```java
  public void iterateBuffers()
  ```
  - **Usage**
    * This function does nothing, but blocks with outputs or internal buffers such as SimFunctionFactory should overload it for it's own special purposes

- **reDefine**
  ```java
  public void reDefine()
  ```
  - **Usage**
    * transfer upper level adjustments to the lower level. this method is overloaded when it is to be used by a lower function

- **setHidden**
  ```java
  public void setHidden()
  ```
  - **Usage**
    * keep the block from being loaded to the factory display if it is panelable

- **setHidden**
  ```java
  public void setHidden(boolean hide)
  ```
  - **Usage**
    * keep the block from being loaded to the factory display if it is panelable
2.2.26 Class SimGeneralClasicController

compilation of sim blocks to create a simple controller with the clasic r-y = e input to the controller

Declaration

```java
public class SimGeneralClasicController
extends simtk.SimFunctionFactory
implements SimPanelable, ContainsFactory
```
* **private boolean debug**

**Constructors**

- **SimGeneralClasicController**
  ```java
  public SimGeneralClasicController()
  ```
  - **Usage**
    - * constructor sets gain to default of unity

**Methods**

- **connectBlocks**
  ```java
  public void connectBlocks()
  ```
  - **Usage**
    - * Make the connections for this block. The setInput of this factory calllls this to setup. It turns out to be a rather interesting system the setInput to this factory then in turn setups up this internal system.

    This function has problems due to the multiple inputs, which each call this connectBlocks() function, and in turn multiple connectons are made to each of the contained Blocks
    The factory.epmty() function should probably also clear the inputs from the contained functions aswell.

- **getPanel**
  ```java
  public JPanel getPanel()
  ```
  - **Usage**
    - * override the default and previous dialog panels

- **reDefine**
  ```java
  public void reDefine()
  ```
  - **Usage**
    - * the super magic code. These transfer functions were derived by Rob Milosovic from the state space seperated controller to a combined transfer funtion
Methods inherited from class `simtk.SimFunctionFactory`

- `addInput`
  ```java
  public void addInput( simtk.HasOutput inputBlock )
  - Usage
    * set the input of this factory this is the interesting one
  ```

- `addInput`
  ```java
  public void addInput( simtk.SimBuffer sb )
  - Usage
    * set the input of this factory. this is the interesting one
  ```

- `clear`
  ```java
  public void clear( )
  - Usage
    * this function is required to reset the buffers and values for the next iterations to set up the factory class
  ```

- `connectBlocks`
  ```java
  abstract void connectBlocks( )
  - Usage
    * Make the connections for this block. The setInput of this factory calls this to setup. It turns out to be a rather interesting system. The setInput() or addInput() to this factory then in turn set’s up this internal system. this is to be implemented.
  ```

- `doIteration`
  ```java
  public void doIteration( )
  - Usage
    * do a calculation for the new output
  ```

- `getPanel`
  ```java
  public JPanel getPanel( )
  ```

- `iterateBuffers`
  ```java
  public void iterateBuffers( )
  - Usage
    * cycle the buffers
  ```

- `setInput`
  ```java
  public void setInput( simtk.SimBuffer sb )
  - Usage
    * set the input of this factory. this is to be implemented in the Simblock to call the connect blocks
  ```

- `toString`
  ```java
  public String toString( )
  ```
Methods inherited from class simtk.SimFunction

- **addInput**
  public void addInput( simtk.HasOutput outputBlock )
  - Usage
    * this AddInput adds from a HasOutput block an input to the vector function and increases the inputIndex count

- **addInput**
  public void addInput( simtk.SimBuffer bufref )
  - Usage
    * AddInput adds an input to the vector function and increases the inputIndex count

- **doIteration**
  abstract void doIteration( )
  - Usage
    * this is the method that should be implemented in all SimBlock’s

- **getInput**
  public SimBuffer getInput( )
  - Usage
    * get the first input. there might be a problem here because the old method of the implicit in buffer is used

- **getInput**
  public SimBuffer getInput( int index )
  - Usage
    * get the indexed input, referenced from as 1 is the first.
      If the reference is out of range it will return the single input value.
      The code is not so clean here where it will still print an error message when the double input functions are just beginning to set themselves up and before the.???

- **getInputCount**
  public int getInputCount( )
  - Usage
    * get the inputCount, or the number of elements on the list

- **getOutput**
  public SimBuffer getOutput( )
  - Usage
    * Gets the output buffer

- **iterateBuffers**
  public void iterateBuffers( )
simtk–SimGeneralClasicController

- **Usage**
  * overload the do nothing iterateBuffers and iterate the output buffers

- **removeInputs**
  ```java
  public void removeInputs()
  ```
  - **Usage**
    * remove the all of the inputs from the block so a new system can be set up

- **setInput**
  ```java
  public void setInput( int index, simtk.SimBuffer bufref )
  ```
  - **Usage**
    * Set the indexed input to the passed buffer. note: this has out of index problems

- **setInput**
  ```java
  public void setInput( simtk.SimBuffer bufref )
  ```
  - **Usage**
    * Set the input to the passed buffer, this is replaced with the newer addInput function which sounds more like what it does

- **setInput1**
  ```java
  public void setInput1( simtk.SimBuffer bufref )
  ```
  - **Usage**
    * This is here to support old systems, it just calls addInput(bufref). so it is not exactly input 2

- **setInput2**
  ```java
  public void setInput2( simtk.SimBuffer bufref )
  ```
  - **Usage**
    * This is here to support old systems, it just calls addInput(bufref). so it is not exactly input 2

- **setOutput**
  ```java
  public void setOutput( simtk.SimBuffer bufref )
  ```
  - **Usage**
    * sets the output to a passed buffer

**Methods inherited from class simtk.SimBlock**

- **clear**
  ```java
  public void clear()
  ```
  - **Usage**
    * this is the general function to clear a block. this is just an empty method but if more specific functionality is needed this should be overridden
• **doIteration**
  abstract void doIteration()  
  – Usage
  * this is the method that should be implemented in all Simblock’s

• **doOutputBufferIteration**
  public void doOutputBufferIteration()  
  – Usage
  * iterate the output buffer(s), or shift the history of it

• **getPanel**
  public JPanel getPanel()  
  – Usage
  * generates the default panel and the title

• **getSimParams**
  public SimParameters getSimParams()  
  – Usage
  * get the parameters from a parameter class

• **getVariableContainer**
  public VariableContainer getVariableContainer()  
  – Usage
  * returns the container for all the variables. this is mainly used for the operations done on the varContainer

• **isHidden**
  public boolean isHidden()  
  – Usage
  * test if the hidden flag has been set

• **iterateBuffers**
  public void iterateBuffers()  
  – Usage
  * This function does nothing, but blocks with outputs or internal buffers such as SimFunctionFactory should overload it for it’s own special purposes

• **reDefine**
  public void reDefine()  
  – Usage
  * transfer upper level adjustments to the lower level. this method is overloaded when it is to be used by a lower function

• **setHidden**
  public void setHidden()
– **Usage**
  * keep the block from being loaded to the factory display if it is panelable

- **setHidden**
  public void setHidden( boolean hide )
  – **Usage**
  * keep the block from being loaded to the factory display if it is panelable

- **setSimParams**
  public void setSimParams( simtk.SimParameters sp )
  – **Usage**
  * set the parameters from a parameter class

---

**Methods inherited from class simtk.SimBase**

- **getDescription**
  public String getDescription( )

- **getReloadBlock**
  public Simulatable getReloadBlock( )
  – **Usage**
  * set the block that gets called to resimulate when the a variable is changed.

- **getTitle**
  public String getTitle( )
  – **Usage**
  * get the user defined title for the entire block

- **setDescription**
  public void setDescription( java.lang.String desc )

- **setReloadBlock**
  public void setReloadBlock( simtk.Simulatable reloadBlock )
  – **Usage**
  * set the block that gets called to resimulate when the a variable is changed.

- **setTitle**
  public void setTitle( java.lang.String title )
  – **Usage**
  * set the user defined title for the entire block

#### 2.2.27 Class SimGFnc

A block for nonlinearities, this block has not been tested yet
public class SimGFunc
extends simtk.SimFunction
implements SimPanelable

• SimGFunc
  public SimGFunc( )

• doIteration
  public void doIteration( )

• getPanel
  public JPanel getPanel( )

  – Usage
  * creates the default dialog window automatically

• setGain
  public void setGain( double g )

• toString
  public String toString( )

Methods inherited from class simtk.SimFunction

• addInput
  public void addInput( simtk.HasOutput outputBlock )

  – Usage
  * this AddInput adds from a HasOutput block an input to the vector function and increases the inputIndex count

• addInput
  public void addInput( simtk.SimBuffer bufref )

  – Usage
  * AddInput adds an input to the vector function and increases the inputIndex count
• `doIteration`
  ```java
  abstract void doIteration()
  ```
  - **Usage**
    * this is the method that should be implemented in all SimBlock’s

• `getInput`
  ```java
  public SimBuffer getInput()
  ```
  - **Usage**
    * get the first input. there might be a problem here because the old method of the implicit in buffer is used

• `getInput`
  ```java
  public SimBuffer getInput( int index )
  ```
  - **Usage**
    * get the indexed input, referenced from as 1 is the first.
      If the reference is out of range it will return the the single input value.
      The code is not so clean here where it will still print an error message when the double input functions are just beginning to set themselves up and before the...??

• `getInputCount`
  ```java
  public int getInputCount()
  ```
  - **Usage**
    * get the inputCount, or the number of elements on the list

• `getOutput`
  ```java
  public SimBuffer getOutput()
  ```
  - **Usage**
    * Gets the output buffer

• `iterateBuffers`
  ```java
  public void iterateBuffers()
  ```
  - **Usage**
    * overload the do nothing iterateBuffers and iterate the output buffers

• `removeInputs`
  ```java
  public void removeInputs()
  ```
  - **Usage**
    * remove the all of the inputs from the block so a new system can be set up

• `setInput`
  ```java
  public void setInput( int index, simtk.SimBuffer bufref )
  ```
  - **Usage**
    * Set the indexed input to the passed buffer. note: this has out of index problems
• **setInput**
  
  public void setInput(simtk.SimBuffer bufref)
  
  – **Usage**
  
  * Set the input to the passed buffer, this is replaced with the newer addInput function which sounds more like what it does

• **setInput1**
  
  public void setInput1(simtk.SimBuffer bufref)
  
  – **Usage**
  
  * This is here to support old systems, it just calls addInput(bufref). so it is not exactly input 2

• **setInput2**
  
  public void setInput2(simtk.SimBuffer bufref)
  
  – **Usage**
  
  * This is here to support old systems, it just calls addInput(bufref). so it is not exactly input 2

• **setOutput**
  
  public void setOutput(simtk.SimBuffer bufref)
  
  – **Usage**
  
  * sets the output to a passed buffer

**Methods inherited from class simtk.SimBlock**

• **clear**
  
  public void clear()
  
  – **Usage**
  
  * this is the general function to clear a block. this is just an empty method but if more specific functionality is needed this should be overridden

• **doIteration**
  
  abstract void doIteration()
  
  – **Usage**
  
  * this is the method that should be implemented in all Simblock’s

• **doOutputBufferIteration**
  
  public void doOutputBufferIteration()
  
  – **Usage**
  
  * iterate the output buffer(s), or shift the history of it

• **getPanel**
  
  public JPanel getPanel()
• **getSimParams**
  
  ```java
  public SimParameters getSimParams()
  ```

  - **Usage**
    * get the parameters from a parameter class

• **getVariableContainer**

  ```java
  public VariableContainer getVariableContainer()
  ```

  - **Usage**
    * returns the container for all the variables. this is mainly used for the operations done on the varContainer

• **isHidden**

  ```java
  public boolean isHidden()
  ```

  - **Usage**
    * test if the hidden flag has been set

• **iterateBuffers**

  ```java
  public void iterateBuffers()
  ```

  - **Usage**
    * This function does nothing, but blocks with outputs or internal buffers such as SimFunctionFactory should overload it for it's own special purposes

• **reDefine**

  ```java
  public void reDefine()
  ```

  - **Usage**
    * transfer upper level adjustments to the lower level. this method is overloaded when it is to be used by a lower function

• **setHidden**

  ```java
  public void setHidden()
  ```

  - **Usage**
    * keep the block from being loaded to the factory display if it is panelable

• **setHidden**

  ```java
  public void setHidden( boolean hide )
  ```

  - **Usage**
    * keep the block from being loaded to the factory display if it is panelable

• **setSimParams**

  ```java
  public void setSimParams( simtk.SimParameters sp )
  ```

  - **Usage**
    * set the parameters from a parameter class
Methods inherited from class simtk.SimBase

- `getDescription`
  ```java
  public String getDescription()
  ```
- `getReloadBlock`
  ```java
  public Simulatable getReloadBlock()
  ```
  - *Usage*
    - *set the block that gets called to resimulate when the a variable is changed.*
- `getTitle`
  ```java
  public String getTitle()
  ```
  - *Usage*
    - *get the user defined title for the entire block*
- `setDescription`
  ```java
  public void setDescription(java.lang.String desc)
  ```
- `setReloadBlock`
  ```java
  public void setReloadBlock(simtk.Simulatable reloadBlock)
  ```
  - *Usage*
    - *set the block that gets called to resimulate when the a variable is changed.*
- `setTitle`
  ```java
  public void setTitle(java.lang.String title)
  ```
  - *Usage*
    - *set the user defined title for the entire block*

### 2.2.28 Class SimImage

This is a block which simply loads a picture at each iteration.

**Declaration**

```java
public class SimImage
  extends simtk.SimBlock
  implements SimPanelable
```

**Fields**

- private boolean loaded
  - first time to load
CONSTRUCTORS

- **SimImage**
  public SimImage()  
  - Usage
    * constructor for now sets the index counter to zero

- **SimImage**
  public SimImage(java.lang.String filename)
  - Usage
    * constructor for now sets the index counter to zero

METHODS

- **clear**
  public void clear()
  - Usage
    * clears the plot, repaints and resets the index counter

- **doIteration**
  public void doIteration()
  - Usage
    * this is the action function that does the iteration of sending the input to the standard output after each iteration

- **getPanel**
  public JPanel getPanel()
  - Usage
    * simply returns the plot. In the future it would be nice if this also set up some button to auto fit and such and some other simple plot functions

- **makeURL**
  public URL makeURL(java.lang.String filename)

- **setFileName**
  public void setFileName(java.lang.String filename)

- **setImage**
  public void setImage()
  - Usage
    * return a JLabel with the icon embedded
Methods inherited from class `simtk.SimBlock`

- `clear`
  
  ```java
  public void clear()
  ```
  
  Usage
  * this is the general function to clear a block. this is just an empty method but if more specific functionality is needed this should be overridden

- `doIteration`
  
  ```java
  abstract void doIteration()
  ```
  
  Usage
  * this is the method that should be implemented in all Simblock’s

- `doOutputBufferIteration`
  
  ```java
  public void doOutputBufferIteration()
  ```
  
  Usage
  * iterate the output buffer(s), or shift the history of it

- `getPanel`
  
  ```java
  public JPanel getPanel()
  ```
  
  Usage
  * generates the default panel and the title

- `getSimParams`
  
  ```java
  public SimParameters getSimParams()
  ```
  
  Usage
  * get the parameters from a parameter class

- `getVariableContainer`
  
  ```java
  public VariableContainer getVariableContainer()
  ```
  
  Usage
  * returns the conatainer for all the variables. this is mainly used for the operations done on the varContainer

- `isHidden`
  
  ```java
  public boolean isHidden()
  ```
  
  Usage
  * test if the hidden flag has been set

- `iterateBuffers`
  
  ```java
  public void iterateBuffers()
  ```
  
  Usage
  * This function does nothing, but blocks with outputs or internal buffers such as SimFunctionFactory should overload it for it’s own special pursposes
• `reDefine`  
  public void `reDefine`()  
  - `Usage`  
    * transfer upper level adjustments to the lower level. this method is overloaded when it is to  
      be used by a lower function

• ` setHidden`  
  public void `setHidden`()  
  - `Usage`  
    * keep the block from being loaded to the factory display if it is panelable

• `setHidden`  
  public void `setHidden` (boolean `hide`)  
  - `Usage`  
    * keep the block from being loaded to the factory display if it is panelable

• `setSimParams`  
  public void `setSimParams` (simtk.SimParameters `sp`)  
  - `Usage`  
    * set the parameters from a parameter class

**Methods inherited from class simtk.SimBase**

• `getDescription`  
  public String `getDescription`()  

• `getReloadBlock`  
  public Simulatable `getReloadBlock`()  
  - `Usage`  
    * set the block that gets called to resimulate when the a variable is changed.

• `getTitle`  
  public String `getTitle`()  
  - `Usage`  
    * get the user defined title for the entire block

• `setDescription`  
  public void `setDescription` (java.lang.String `desc`)  

• `setReloadBlock`  
  public void `setReloadBlock` (simtk.Simulatable `reloadBlock`)  
  - `Usage`  
    * set the block that gets called to resimulate when the a variable is changed.

• `setTitle`  
  public void `setTitle` (java.lang.String `title`)  
  - `Usage`  
    * set the user defined title for the entire block
2.2.29 Class SimImageFactory

This class has several functions to help get images from files.

Declaration

```java
public class SimImageFactory
    extends javax.swing.JApplet
```

Constructors

- `SimImageFactory`
  ```java
  public SimImageFactory()
  ```
  - Usage
    * constructor

Methods

- `makeIconLabel`
  ```java
  public JLabel makeIconLabel(java.lang.String filename, java.lang.String title)
  ```
  - Usage
    * return a JLabel with the icon embedded

- `makeURL`
  ```java
  public URL makeURL(java.lang.String filename)
  ```

Methods inherited from class `javax.swing.JApplet`

- `addImpl`
  ```java
  protected void addImpl(java.awt.Component, java.lang.Object, int)
  ```

- `createRootPane`
  ```java
  protected JRootPane createRootPane()
  ```

- `createRootPaneException`
  ```java
  private Error createRootPaneException(java.lang.String)
  ```

- `getAccessibleContext`
  ```java
  public AccessibleContext getAccessibleContext()
  ```
Methods inherited from class java.applet.Applet

• destroy
  public void destroy( )
• getAccessibleContext
  public AccessibleContext getAccessibleContext( )
• getAppletContext
  public AppletContext getAppletContext( )
• getAppletInfo
  public String getAppletInfo( )
- `getAudioClip`
  ```java
  public AudioClip getAudioClip(java.net.URL )
  ```
- `getAudioClip`
  ```java
  public AudioClip getAudioClip(java.net.URL , java.lang.String )
  ```
- `getCodeBase`
  ```java
  public URL getCodeBase()
  ```
- `getDocumentBase`
  ```java
  public URL getDocumentBase()
  ```
- `getImage`
  ```java
  public Image getImage(java.net.URL )
  ```
- `getImage`
  ```java
  public Image getImage(java.net.URL , java.lang.String )
  ```
- `getLocale`
  ```java
  public Locale getLocale()
  ```
- `getParameter`
  ```java
  public String getParameter(java.lang.String )
  ```
- `getParameterInfo`
  ```java
  public String getParameterInfo()
  ```
- `init`
  ```java
  public void init()
  ```
- `isActive`
  ```java
  public boolean isActive()
  ```
- `newAudioClip`
  ```java
  public static final AudioClip newAudioClip(java.net.URL )
  ```
- `play`
  ```java
  public void play(java.net.URL )
  ```
- `play`
  ```java
  public void play(java.net.URL , java.lang.String )
  ```
- `readObject`
  ```java
  private void readObject(java.io.ObjectInputStream )
  ```
- `resize`
  ```java
  public void resize(java.awt.Dimension )
  ```
- `resize`
  ```java
  public void resize(int , int )
  ```
- `setStub`
  ```java
  public final void setStub(java.applet.AppletStub )
  ```
- `showStatus`
  ```java
  public void showStatus(java.lang.String )
  ```
- `start`
  ```java
  public void start()
  ```
- `stop`
  ```java
  public void stop()
  ```
Methods inherited from class java.awt.Panel

- `<clinit>`
  static void `<clinit>()`
- `addNotify`
  public void `addNotify()`
- `constructComponentName`
  String `constructComponentName()`
- `getAccessibleContext`
  public AccessibleContext `getAccessibleContext()`

Methods inherited from class java.awt.Container

- `<clinit>`
  static void `<clinit>()`
- `add`
  public Component `add( java.awt.Component )`
- `add`
  public Component `add( java.awt.Component , int )`
- `add`
  public void `add( java.awt.Component , java.lang.Object )`
- `add`
  public void `add( java.awt.Component , java.lang.Object , int )`
- `add`
  public Component `add( java.lang.String , java.awt.Component )`
- `addContainerListener`
  public synchronized void `addContainerListener( java.awt.event.ContainerListener )`
- `addImpl`
  protected void `addImpl( java.awt.Component , java.lang.Object , int )`
- `addNotify`
  public void `addNotify()`
- `addPropertyChangeListener`
  public void `addPropertyChangeListener( java.beans.PropertyChangeListener )`
- `addPropertyChangeListener`
  public void `addPropertyChangeListener( java.lang.String , java.beans.PropertyChangeListener )`
- `adjustDescendantsOnParent`
  void `adjustDescendantsOnParent( int )`
- `adjustDescendants`
  void `adjustDescendants( int )`
- `adjustListeningChildren`
  void `adjustListeningChildren( long , int )`
• applyComponentOrientation
  public void applyComponentOrientation(java.awt.ComponentOrientation)

• areFocusTraversalKeysSet
  public boolean areFocusTraversalKeysSet(int)

• checkGD
  void checkGD(java.lang.String)

• countComponents
  public int countComponents()

• countHierarchyMembers
  int countHierarchyMembers()

• createChildHierarchyEvents
  void createChildHierarchyEvents(int, long, boolean)

• createHierarchyEvents
  int createHierarchyEvents(int, java.awt.Component, java.awt.Container, long, boolean)

• deliverEvent
  public void deliverEvent(java.awt.Event)

• dispatchEventImpl
  void dispatchEventImpl(java.awt.AWTEvent)

• dispatchEventToSelf
  void dispatchEventToSelf(java.awt.AWTEvent)

• doLayout
  public void doLayout()

• eventEnabled
  boolean eventEnabled(java.awt.AWTEvent)

• findComponentAt
  public Component findComponentAt(int, int)

• findComponentAt
  final Component findComponentAt(int, int, boolean)

• findComponentAt
  public Component findComponentAt(java.awt.Point)

• findTraversalRoot
  private Container findTraversalRoot()

• getAccessibleAt
  Accessible getAccessibleAt(java.awt.Point)

• getAccessibleChild
  Accessible getAccessibleChild(int)

• getAccessibleChildrenCount
  int getAccessibleChildrenCount()

• getAlignmentX
  public float getAlignmentX()

• getAlignmentY
  public float getAlignmentY()
- `getComponent`
  ```java
  public Component getComponent( int )
  ```
- `getComponentAt`
  ```java
  public Component getComponentAt( int , int )
  ```
- `getComponentAt`
  ```java
  public Component getComponentAt( java.awt.Point )
  ```
- `getComponentCount`
  ```java
  public int getComponentCount()
  ```
- `getComponents_NoClientCode`
  ```java
  final Component getComponents_NoClientCode()
  ```
- `getComponents`
  ```java
  public Component getComponents()
  ```
- `getContainerListeners`
  ```java
  public synchronized ContainerListener getContainerListeners()
  ```
- `getDropTargetEventTarget`
  ```java
  Component getDropTargetEventTarget( int , int , boolean )
  ```
- `getFocusTraversalKeys`
  ```java
  public Set getFocusTraversalKeys( int )
  ```
- `getFocusTraversalPolicy`
  ```java
  public FocusTraversalPolicy getFocusTraversalPolicy()
  ```
- `getInsets`
  ```java
  public Insets getInsets()
  ```
- `getLayout`
  ```java
  public LayoutManager getLayout()
  ```
- `getListeners`
  ```java
  public EventListener getListeners( java.lang.Class )
  ```
- `getMaximumSize`
  ```java
  public Dimension getMaximumSize()
  ```
- `getMinimumSize`
  ```java
  public Dimension getMinimumSize()
  ```
- `getMouseEventTarget`
  ```java
  Component getMouseEventTarget( int , int , boolean )
  ```
- `getMouseEventTarget`
  ```java
  private Component getMouseEventTarget( int , int , boolean , java.awt.Container.EventTargetFilter , boolean )
  ```
- `getPreferredSize`
  ```java
  public Dimension getPreferredSize()
  ```
- `initializeFocusTraversalKeys`
  ```java
  void initializeFocusTraversalKeys()
  ```
- `initIDs`
  ```java
  private static native void initIDs()
  ```
• insets
  public Insets insets()  
• invalidate
  public void invalidate()  
• invalidateTree
  void invalidateTree()  
• isAncestorOf
  public boolean isAncestorOf(java.awt.Component)  
• isFocusCycleRoot
  public boolean isFocusCycleRoot()  
• isFocusCycleRoot
  public boolean isFocusCycleRoot(java.awt.Container)  
• isFocusTraversalPolicySet
  public boolean isFocusTraversalPolicySet()  
• layout
  public void layout()  
• lightweightPaint
  void lightweightPaint(java.awt.Graphics)  
• lightweightPrint
  void lightweightPrint(java.awt.Graphics)  
• list
  public void list(java.io.PrintStream, int)  
• list
  public void list(java.io.PrintWriter, int)  
• locate
  public Component locate(int, int)  
• minimumSize
  public Dimension minimumSize()  
• nextFocusHelper
  boolean nextFocusHelper()  
• numListening
  int numListening(long)  
• paint
  public void paint(java.awt.Graphics)  
• paintComponents
  public void paintComponents(java.awt.Graphics)  
• paintHeavyweightComponents
  void paintHeavyweightComponents(java.awt.Graphics)  
• paramString
  protected String paramString()  
• postProcessKeyEvent
  void postProcessKeyEvent(java.awt.event.KeyEvent)  
• postsOldMouseEvents
  boolean postsOldMouseEvents( )
• preferredSize
  public Dimension preferredSize( )
• preProcessKeyEvent
  void preProcessKeyEvent( java.awt.event.KeyEvent )
• print
  public void print( java.awt.Graphics )
• printComponents
  public void printComponents( java.awt.Graphics )
• printHeavyweightComponents
  void printHeavyweightComponents( java.awt.Graphics )
• processContainerEvent
  protected void processContainerEvent( java.awt.event.ContainerEvent )
• processEvent
  protected void processEvent( java.awt.AWTEvent )
• proxyEnableEvents
  void proxyEnableEvents( long )
• readObject
  private void readObject( java.io.ObjectInputStream )
• remove
  public void remove( java.awt.Component )
• remove
  public void remove( int )
• removeAll
  public void removeAll( )
• removeContainerListener
  public synchronized void removeContainerListener( java.awt.event.ContainerListener )
• removeNotify
  public void removeNotify( )
• setFocusCycleRoot
  public void setFocusCycleRoot( boolean )
• setFocusTraversableKeys
  public void setFocusTraversableKeys( int , java.util.Set )
• setFocusTraversalPolicy
  public void setFocusTraversalPolicy( java.awt.FocusTraversalPolicy )
• setFont
  public void setFont( java.awt.Font )
• setLayout
  public void setLayout( java.awt.LayoutManager )
• setZOrder
  void setZOrder( java.awt.Component , int )
• transferFocusBackward
  public void transferFocusBackward()

• transferFocusDownCycle
  public void transferFocusDownCycle()

• update
  public void update( java.awt.Graphics )

• validate
  public void validate()

• validateTree
  protected void validateTree()

• writeObject
  private void writeObject( java.io.ObjectOutputStream )

Methods inherited from class java.awt.Component

• <clinit>
  static void <clinit>()

• action
  public boolean action( java.awt.Event, java.lang.Object )

• add
  public synchronized void add( java.awt.PopupMenu )

• addComponentListener
  public synchronized void addComponentListener( java.awt.event.ComponentListener )

• addFocusListener
  public synchronized void addFocusListener( java.awt.event.FocusListener )

• addHierarchyBoundsListener
  public void addHierarchyBoundsListener( java.awt.event.HierarchyBoundsListener )

• addHierarchyListener
  public void addHierarchyListener( java.awt.event.HierarchyListener )

• addInputMethodListener
  public synchronized void addInputMethodListener( java.awt.event.InputMethodListener )

• addKeyListener
  public synchronized void addKeyListener( java.awt.event.KeyListener )

• addMouseListener
  public synchronized void addMouseListener( java.awt.event.MouseListener )

• addMouseMotionListener
  public synchronized void addMouseMotionListener( java.awt.event.MouseMotionListener )

• addMouseWheelListener
  public synchronized void addMouseWheelListener( java.awt.event.MouseWheelListener )
- `addNotify`
  ```java
  public void addNotify()
  ```

- `addPropertyChangeListener`
  ```java
  public synchronized void addPropertyChangeListener(
    java.beans.PropertyChangeListener )
  ```

- `addPropertyChangeListener`
  ```java
  public synchronized void addPropertyChangeListener( java.lang.String ,
    java.beans.PropertyChangeListener )
  ```

- `adjustListeningChildrenOnParent`
  ```java
  void adjustListeningChildrenOnParent( long , int )
  ```

- `applyComponentOrientation`
  ```java
  public void applyComponentOrientation( java.awt.ComponentOrientation )
  ```

- `areFocusTraversalKeysSet`
  ```java
  public boolean areFocusTraversalKeysSet( int )
  ```

- `areInputMethodsEnabled`
  ```java
  boolean areInputMethodsEnabled( )
  ```

- `autoProcessMouseWheel`
  ```java
  void autoProcessMouseWheel( java.awt.event.MouseWheelEvent )
  ```

- `autoTransferFocus`
  ```java
  final void autoTransferFocus( boolean )
  ```

- `bounds`
  ```java
  public Rectangle bounds( )
  ```

- `checkGD`
  ```java
  void checkGD( java.lang.String )
  ```

- `checkImage`
  ```java
  public int checkImage( java.awt.Image , java.awt.image.ImageObserver )
  ```

- `checkImage`
  ```java
  public int checkImage( java.awt.Image , int , int , java.awt.image.ImageObserver )
  ```

- `checkWindowClosingException`
  ```java
  boolean checkWindowClosingException( )
  ```

- `coalesceEvents`
  ```java
  protected AWTEvent coalesceEvents( java.awt.AWTEvent , java.awt.AWTEvent )
  ```

- `constructComponentName`
  ```java
  String constructComponentName( )
  ```

- `contains`
  ```java
  public boolean contains( int , int )
  ```

- `contains`
  ```java
  public boolean contains( java.awt.Point )
  ```

- `countHierarchyMembers`
  ```java
  int countHierarchyMembers( )
  ```

- `createBufferStrategy`
  ```java
  void createBufferStrategy( int )
  ```
• createBufferStrategy
  void createBufferStrategy(int, java.awt.BufferCapabilities)

• createChildHierarchyEvents
  void createChildHierarchyEvents(int, long, boolean)

• createHierarchyEvents
  int createHierarchyEvents(int, java.awt.Component, java.awt.Container, long, boolean)

• createImage
  public Image createImage(java.awt.image.ImageProducer)

• createImage
  public Image createImage(int, int)

• createVolatileImage
  public VolatileImage createVolatileImage(int, int)

• createVolatileImage
  public VolatileImage createVolatileImage(int, int, java.awt.ImageCapabilities)

• deliverEvent
  public void deliverEvent(java.awt.Event)

• disable
  public void disable()

• disableEvents
  protected final void disableEvents(long)

• dispatchEvent
  public final void dispatchEvent(java.awt.AWTEvent)

• dispatchEventImpl
  void dispatchEventImpl(java.awt.AWTEvent)

• dispatchMouseWheelToAncestor
  boolean dispatchMouseWheelToAncestor(java.awt.event.MouseWheelEvent)

• doAutoTransfer
  private void doAutoTransfer(boolean)

• doLayout
  public void doLayout()

• enable
  public void enable()

• enable
  public void enable(boolean)

• enableEvents
  protected final void enableEvents(long)

• enableInputMethods
  public void enableInputMethods(boolean)

• eventEnabled
  boolean eventEnabled(java.awt.AWTEvent)

• eventTypeEnabled
  boolean eventTypeEnabled(int)
• firePropertyChange
  protected void firePropertyChange( java.lang.String , boolean , boolean )
• firePropertyChange
  protected void firePropertyChange( java.lang.String , int , int )
• firePropertyChange
  protected void firePropertyChange( java.lang.String , java.lang.Object , java.lang.Object )
• getAccessibleContext
  public AccessibleContext getAccessibleContext( )
• getAccessibleIndexInParent
  int getAccessibleIndexInParent( )
• getAccessibleStateSet
  AccessibleStateSet getAccessibleStateSet( )
• getAlignmentX
  public float getAlignmentX( )
• getAlignmentY
  public float getAlignmentY( )
• getBackground
  public Color getBackground( )
• getBounds
  public Rectangle getBounds( )
• getBounds
  public Rectangle getBounds( java.awt.Rectangle )
• getBufferStrategy
  BufferStrategy getBufferStrategy( )
• getColorModel
  public ColorModel getColorModel( )
• getComponentAt
  public Component getComponentAt( int , int )
• getComponentAt
  public Component getComponentAt( java.awt.Point )
• getComponentListeners
  public synchronized ComponentListener getComponentListeners( )
• getComponentOrientation
  public ComponentOrientation getComponentOrientation( )
• getCursor
  public Cursor getCursor( )
• getDropTarget
  public synchronized DropTarget getDropTarget( )
• getFocusCycleRootAncestor
  public Container getFocusCycleRootAncestor( )
• getFocusListeners
  public synchronized FocusListener getFocusListeners( )
- `getFocusTraversalKeys_NoIDCheck`
  ```java
  public final Set<java.awt.Component> getFocusTraversalKeys_NoIDCheck(int direction)
  ```
- `getFocusTraversalKeys`
  ```java
  public final Set<java.awt.Component> getFocusTraversalKeys(int direction)
  ```
- `getFocusTraversalKeysEnabled`
  ```java
  public boolean getFocusTraversalKeysEnabled()
  ```
- `getLocale`
  ```java
  public Locale getLocale()
  ```
- `getLocation`
  ```java
  public Point getLocation()
  ```
- `getLocation(java.awt.Point)`
  ```java
  public Point getLocation(java.awt.Point p)
  ```
- `getListeners(java.lang.Class)`
  ```java
  public EventListener getListeners(java.lang.Class c)
  ```
- `getListeners()`
  ```java
  public EventListener getListeners()
  ```
- `getKeyListeners()`
  ```java
  public synchronized KeyListener getKeyListeners()
  ```
- `getListeners()`
  ```java
  public synchronized KeyListener getListeners()
  ```
- `getInputMethodListeners()`
  ```java
  public synchronized InputMethodListener getInputMethodListeners()
  ```
- `getInputMethodRequests()`
  ```java
  public InputMethodRequests getInputMethodRequests()
  ```
- `getInputContext()`
  ```java
  public InputContext getInputContext()
  ```
- `getInputContext()`
  ```java
  public synchronized InputContext getInputContext()
  ```
- `getHierarchyListeners()`
  ```java
  public synchronized HierarchyListener getHierarchyListeners()
  ```
- `getHierarchyListeners()`
  ```java
  public synchronized HierarchyListener getHierarchyListeners()
  ```
- `getIgnoreRepaint()`
  ```java
  public boolean getIgnoreRepaint()
  ```
- `getLayout()`
  ```java
  public Layout getLayout()
  ```
- `getLocal()`
  ```java
  public Layout getLocal()
  ```
- `getMetrics()`
  ```java
  public Metrics getMetrics()
  ```
- `getMetrics()`
  ```java
  public synchronized Metrics getMetrics()
  ```
- `getMetrics()`
  ```java
  public synchronized Metrics getMetrics()
  ```
- `getFont()`
  ```java
  public Font getFont()
  ```
- `getFontNoClientCode()`
  ```java
  public final Font getFontNoClientCode()
  ```
- `getFontMetrics()`
  ```java
  public FontMetrics getFontMetrics(java.awt.Font font)
  ```
- `getFontMetrics()`
  ```java
  public FontMetrics getFontMetrics()
  ```
- `getForeground()`
  ```java
  public Color getForeground()
  ```
- `getGraphics()`
  ```java
  public Graphics getGraphics()
  ```
- `getGraphicsConfiguration()`
  ```java
  public GraphicsConfiguration getGraphicsConfiguration()
  ```
- `getHeight()`
  ```java
  public int getHeight()
  ```
- `getHierarchyBoundsListeners()`
  ```java
  public synchronized HierarchyBoundsListener getHierarchyBoundsListeners()
  ```
- `getHierarchyListeners()`
  ```java
  public synchronized HierarchyListener getHierarchyListeners()
  ```
- `getIgnoreRepaint()`
  ```java
  public boolean getIgnoreRepaint()
  ```
- `getInputContext()`
  ```java
  public InputContext getInputContext()
  ```
- `getInputMethodListeners()`
  ```java
  public synchronized InputMethodListener getInputMethodListeners()
  ```
- `getInputMethodRequests()`
  ```java
  public InputMethodRequests getInputMethodRequests()
  ```
- `getKeyListeners()`
  ```java
  public synchronized KeyListener getKeyListeners()
  ```
- `getListeners()`
  ```java
  public EventListener getListeners(java.lang.Class c)
  ```
- `getListeners()`
  ```java
  public EventListener getListeners()
  ```
- `getLocale()`
  ```java
  public Locale getLocale()
  ```
- `getLocation()`
  ```java
  public Point getLocation()
  ```
- `getLocation()`
  ```java
  public Point getLocation(java.awt.Point p)
  ```
• `getLocationOnScreen_NoTreeLock`
  
  ```java
  final Point getLocationOnScreen_NoTreeLock()
  ```

• `getLocationOnScreen`
  
  ```java
  public Point getLocationOnScreen()
  ```

• `getMaximumSize`
  
  ```java
  public Dimension getMaximumSize()
  ```

• `getMinimumSize`
  
  ```java
  public Dimension getMinimumSize()
  ```

• `getMouseListeners`
  
  ```java
  public synchronized MouseListener getMouseListeners()
  ```

• `getMouseMotionListeners`
  
  ```java
  public synchronized MouseMotionListener getMouseMotionListeners()
  ```

• `getMouseWheelListeners`
  
  ```java
  public synchronized MouseWheelListener getMouseWheelListeners()
  ```

• `getName`
  
  ```java
  public String getName()
  ```

• `getNativeContainer`
  
  ```java
  Container getNativeContainer()
  ```

• `getParent_NoClientCode`
  
  ```java
  final Container getParent_NoClientCode()
  ```

• `getParent`
  
  ```java
  public Container getParent()
  ```

• `getPeer`
  
  ```java
  public ComponentPeer getPeer()
  ```

• `getPreferredSize`
  
  ```java
  public Dimension getPreferredSize()
  ```

• `getPropertyChangeListeners`
  
  ```java
  public synchronized PropertyChangeListener getPropertyChangeListeners()
  ```

• `getPropertyChangeListeners` (with `java.lang.String`)
  
  ```java
  public synchronized PropertyChangeListener getPropertyChangeListeners(java.lang.String)
  ```

• `getSize`
  
  ```java
  public Dimension getSize()
  ```

• `getSize` (with `java.awt.Dimension`)
  
  ```java
  public Dimension getSize(java.awt.Dimension)
  ```

• `getToolkit`
  
  ```java
  public Toolkit getToolkit()
  ```

• `getToolkitImpl`
  
  ```java
  final Toolkit getToolkitImpl()
  ```

• `getTreeLock`
  
  ```java
  public final Object getTreeLock()
  ```

• `getWidth`
  
  ```java
  public int getWidth()
  ```
• \texttt{getX}
  
  \begin{verbatim}
  public int getX()
  \end{verbatim}

• \texttt{getY}
  
  \begin{verbatim}
  public int getY()
  \end{verbatim}

• \texttt{gotFocus}
  
  \begin{verbatim}
  public boolean gotFocus( java.awt.Event, java.lang.Object )
  \end{verbatim}

• \texttt{handleEvent}
  
  \begin{verbatim}
  public boolean handleEvent( java.awt.Event )
  \end{verbatim}

• \texttt{hasFocus}
  
  \begin{verbatim}
  public boolean hasFocus()
  \end{verbatim}

• \texttt{hide}
  
  \begin{verbatim}
  public void hide()
  \end{verbatim}

• \texttt{imageUpdate}
  
  \begin{verbatim}
  public boolean imageUpdate( java.awt.Image, int, int, int, int, int )
  \end{verbatim}

• \texttt{initializeFocusTraversalKeys}
  
  \begin{verbatim}
  void initializeFocusTraversalKeys()
  \end{verbatim}

• \texttt{initIDs}
  
  \begin{verbatim}
  private static native void initIDs()
  \end{verbatim}

• \texttt{inside}
  
  \begin{verbatim}
  public boolean inside( int, int )
  \end{verbatim}

• \texttt{invalidate}
  
  \begin{verbatim}
  public void invalidate()
  \end{verbatim}

• \texttt{isBackgroundSet}
  
  \begin{verbatim}
  public boolean isBackgroundSet()
  \end{verbatim}

• \texttt{isCursorSet}
  
  \begin{verbatim}
  public boolean isCursorSet()
  \end{verbatim}

• \texttt{isDisplayable}
  
  \begin{verbatim}
  public boolean isDisplayable()
  \end{verbatim}

• \texttt{isDoubleBuffered}
  
  \begin{verbatim}
  public boolean isDoubleBuffered()
  \end{verbatim}

• \texttt{isEnabled}
  
  \begin{verbatim}
  public boolean isEnabled()
  \end{verbatim}

• \texttt{isEnabledImpl}
  
  \begin{verbatim}
  final boolean isEnabledImpl()
  \end{verbatim}

• \texttt{isFocusable}
  
  \begin{verbatim}
  public boolean isFocusable()
  \end{verbatim}

• \texttt{isFocusCycleRoot}
  
  \begin{verbatim}
  public boolean isFocusCycleRoot( java.awt.Container )
  \end{verbatim}

• \texttt{isFocusOwner}
  
  \begin{verbatim}
  public boolean isFocusOwner()
  \end{verbatim}

• \texttt{isFocusTraversable}
  
  \begin{verbatim}
  public boolean isFocusTraversable()
  \end{verbatim}
simtk – SimImageFactory

- `isFocusTraversableOverridden`
  ```java
  final boolean isFocusTraversableOverridden()
  ```
- `isFontSet`
  ```java
  public boolean isFontSet()
  ```
- `isForegroundSet`
  ```java
  public boolean isForegroundSet()
  ```
- `isLightweight`
  ```java
  public boolean isLightweight()
  ```
- `isOpaque`
  ```java
  public boolean isOpaque()
  ```
- `isRecursivelyVisible`
  ```java
  boolean isRecursivelyVisible()
  ```
- `isShowing`
  ```java
  public boolean isShowing()
  ```
- `isValid`
  ```java
  public boolean isValid()
  ```
- `isVisible`
  ```java
  public boolean isVisible()
  ```
- `keyDown`
  ```java
  public boolean keyDown(java.awt.Event, int)
  ```
- `keyUp`
  ```java
  public boolean keyUp(java.awt.Event, int)
  ```
- `layout`
  ```java
  public void layout()
  ```
- `lightweightPaint`
  ```java
  void lightweightPaint(java.awt.Graphics)
  ```
- `lightweightPrint`
  ```java
  void lightweightPrint(java.awt.Graphics)
  ```
- `list`
  ```java
  public void list()
  ```
- `list`
  ```java
  public void list(java.io.PrintStream)
  ```
- `list`
  ```java
  public void list(java.io.PrintStream, int)
  ```
- `list`
  ```java
  public void list(java.io.PrintWriter)
  ```
- `list`
  ```java
  public void list(java.io.PrintWriter, int)
  ```
- `locate`
  ```java
  public Component locate(int, int)
  ```
- `location`
  ```java
  public Point location()
  ```
- **lostFocus**
  public boolean lostFocus( java.awt.Event , java.lang.Object )

- **minimumSize**
  public Dimension minimumSize( )

- **mouseDown**
  public boolean mouseDown( java.awt.Event , int , int )

- **mouseDrag**
  public boolean mouseDrag( java.awt.Event , int , int )

- **mouseEnter**
  public boolean mouseEnter( java.awt.Event , int , int )

- **mouseExit**
  public boolean mouseExit( java.awt.Event , int , int )

- **mouseMove**
  public boolean mouseMove( java.awt.Event , int , int )

- **mouseUp**
  public boolean mouseUp( java.awt.Event , int , int )

- **move**
  public void move( int , int )

- **nextFocus**
  public void nextFocus( )

- **nextFocusHelper**
  boolean nextFocusHelper( )

- **numListening**
  int numListening( long )

- **paint**
  public void paint( java.awt.Graphics )

- **paintAll**
  public void paintAll( java.awt.Graphics )

- **paintHeavyweightComponents**
  void paintHeavyweightComponents( java.awt.Graphics )

- **paramString**
  protected String paramString( )

- **postEvent**
  public boolean postEvent( java.awt.Event )

- **postsOldMouseEvents**
  boolean postsOldMouseEvents( )

- **preferredSize**
  public Dimension preferredSize( )

- **prepareImage**
  public boolean prepareImage( java.awt.Image , java.awt.image.ImageObserver )

- **prepareImage**
  public boolean prepareImage( java.awt.Image , int , int , java.awt.image.ImageObserver )
• print
  public void print( java.awt.Graphics )
• printAll
  public void printAll( java.awt.Graphics )
• printHeavyweightComponents
  void printHeavyweightComponents( java.awt.Graphics )
• processComponentEvent
  protected void processComponentEvent( java.awt.event.ComponentEvent )
• processEvent
  protected void processEvent( java.awt.AWTEvent )
• processFocusEvent
  protected void processFocusEvent( java.awt.event.FocusEvent )
• processHierarchyBoundsEvent
  protected void processHierarchyBoundsEvent( java.awt.event.HierarchyEvent )
• processHierarchyEvent
  protected void processHierarchyEvent( java.awt.event.HierarchyEvent )
• processInputMethodEvent
  protected void processInputMethodEvent( java.awt.event.InputMethodEvent )
• processKeyEvent
  protected void processKeyEvent( java.awt.event.KeyEvent )
• processMouseEvent
  protected void processMouseEvent( java.awt.event.MouseEvent )
• processMouseMotionEvent
  protected void processMouseMotionEvent( java.awt.event.MouseEvent )
• processMouseWheelEvent
  protected void processMouseWheelEvent( java.awt.event.MouseWheelEvent )
• readObject
  private void readObject( java.io.ObjectInputStream )
• remove
  public synchronized void remove( java.awt.MenuComponent )
• removeComponentListener
  public synchronized void removeComponentListener( java.awt.event.ComponentListener )
• removeFocusListener
  public synchronized void removeFocusListener( java.awt.event.FocusListener )
• removeHierarchyBoundsListener
  public void removeHierarchyBoundsListener( java.awt.event.HierarchyBoundsListener )
• removeHierarchyListener
  public void removeHierarchyListener( java.awt.event.HierarchyListener )
• removeInputMethodListener
  public synchronized void removeInputMethodListener( java.awt.event.InputMethodListener )
- **removeKeyListener**
  
  ```java
  public synchronized void removeKeyListener( java.awt.event.KeyListener )
  ```

- **removeMouseListener**
  
  ```java
  public synchronized void removeMouseListener( java.awt.event.MouseAdapter )
  ```

- **removeMouseMotionListener**
  
  ```java
  public synchronized void removeMouseMotionListener( java.awt.event.MouseMotionListener )
  ```

- **removeMouseWheelListener**
  
  ```java
  public synchronized void removeMouseWheelListener( java.awt.event.MouseWheelListener )
  ```

- **removeNotify**
  
  ```java
  public void removeNotify( )
  ```

- **removePropertyChangeListener**
  
  ```java
  public synchronized void removePropertyChangeListener( java.beans.PropertyChangeListener )
  ```

- **removePropertyChangeListener**
  
  ```java
  public synchronized void removePropertyChangeListener( java.lang.String , java.beans.PropertyChangeListener )
  ```

- **repaint**
  
  ```java
  public void repaint( )
  ```

- **repaint**
  
  ```java
  public void repaint( int , int , int , int )
  ```

- **repaint**
  
  ```java
  public void repaint( long )
  ```

- **repaint**
  
  ```java
  public void repaint( long , int , int , int , int )
  ```

- **requestFocus**
  
  ```java
  public void requestFocus( )
  ```

- **requestFocus**
  
  ```java
  protected boolean requestFocus( boolean )
  ```

- **requestFocusHelper**
  
  ```java
  final boolean requestFocusHelper( boolean , boolean )
  ```

- **requestFocusInWindow**
  
  ```java
  public boolean requestFocusInWindow( )
  ```

- **requestFocusInWindow**
  
  ```java
  protected boolean requestFocusInWindow( boolean )
  ```

- **resetGC**
  
  ```java
  void resetGC( )
  ```

- **reshape**
  
  ```java
  public void reshape( int , int , int , int )
  ```

- **resize**
  
  ```java
  public void resize( java.awt.Dimension )
  ```

- **resize**
  
  ```java
  public void resize( int , int )
  ```
• `setBackground`
  
  ```java
  public void setBackground( java.awt.Color )
  ```

• `setBounds`

  ```java
  public void setBounds( int , int , int , int )
  ```

• `setBounds`

  ```java
  public void setBounds( java.awt.Rectangle )
  ```

• `setComponentOrientation`

  ```java
  public void setComponentOrientation( java.awt.ComponentOrientation )
  ```

• `setCursor`

  ```java
  public void setCursor( java.awt.Cursor )
  ```

• `setDropTarget`

  ```java
  public synchronized void setDropTarget( java.awt.dnd.DropTarget )
  ```

• `setEnabled`

  ```java
  public void setEnabled( boolean )
  ```

• `setFocusable`

  ```java
  public void setFocusable( boolean )
  ```

• `setFocusTraversalKeys_NoIDCheck`

  ```java
  final void setFocusTraversalKeys_NoIDCheck( int , java.util.Set )
  ```

• `setFocusTraversalKeys`

  ```java
  public void setFocusTraversalKeys( int , java.util.Set )
  ```

• `setFocusTraversalKeysEnabled`

  ```java
  public void setFocusTraversalKeysEnabled( boolean )
  ```

• `setFont`

  ```java
  public void setFont( java.awt.Font )
  ```

• `setForeground`

  ```java
  public void setForeground( java.awt.Color )
  ```

• `setGCFromPeer`

  ```java
  void setGCFromPeer()
  ```

• `setIgnoreRepaint`

  ```java
  public void setIgnoreRepaint( boolean )
  ```

• `setLocale`

  ```java
  public void setLocale( java.util.Locale )
  ```

• `setLocation`

  ```java
  public void setLocation( int , int )
  ```

• `setLocation`

  ```java
  public void setLocation( java.awt.Point )
  ```

• `setName`

  ```java
  public void setName( java.lang.String )
  ```

• `setSize`

  ```java
  public void setSize( java.awt.Dimension )
  ```

• `setSize`

  ```java
  public void setSize( int , int )
  ```
• **setVisible**  
  ```java
  public void setVisible(boolean )
  ```

• **show**  
  ```java
  public void show()
  ```

• **show**  
  ```java
  public void show(boolean )
  ```

• **size**  
  ```java
  public Dimension size()
  ```

• **toString**  
  ```java
  public String toString()
  ```

• **transferFocus**  
  ```java
  public void transferFocus()
  ```

• **transferFocusBackward**  
  ```java
  public void transferFocusBackward()
  ```

• **transferFocusUpCycle**  
  ```java
  public void transferFocusUpCycle()
  ```

• **update**  
  ```java
  public void update(java.awt.Graphics )
  ```

• **updateCursorImmediately**  
  ```java
  final void updateCursorImmediately()
  ```

• **validate**  
  ```java
  public void validate()
  ```

• **writeObject**  
  ```java
  private void writeObject(java.io.ObjectOutputStream )
  ```

### 2.2.30 Class SimIntegrator

this is the block that simulates ztranfer functions a number of general types of functions can be derived from this block

#### Declaration

```java
public class SimIntegrator  
extends simtk.SimZtf
```

#### Constructors

• **SimIntegrator**  
  ```java
  public SimIntegrator()
  ```
Methods

- **reDefine**
  public void **reDefine**()  
  
  - **Usage**
    * this is a method that might be good to have defined in the sim Block as abstract this is made to recalculate with new parameters, step sizes parameters this function overloads any other functions overrides the reDefine of SimZtf and loads in the new parameters then it calls the overriden reDefine from Ztf

Methods inherited from class simtk.SimZtf

- **doIteration**
  public void **doIteration**()  
  
  - **Usage**
    * this method overrides the doIteration of the diffeq this way it first redefines the variables and then iterates with the inherited structure

- **doIterationDirect**
  public void **doIterationDirect**()  
  
  - **Usage**
    * this is the really cool simple method that converts ztf to diffEQ. The function does the operation in the following order:
      1. parses the numerator and denominator
      2. creates the difference equation
      3. does one step operation.

- **reDefine**
  public void **reDefine**()  
  
  - **Usage**
    * this is a method that might be good to have defined in the sim Block as abstract this is made to recalculate with new parameters, step sizes parameters

- **setDen**
  public void **setDen**( double [] n )  
  
  - **Usage**
    * method to set the denominator to the array value Example: setDen([1,2,3]); would set the denominator to $z^2 + 2z + 3$ an error message is displayed if the array is lager than the z order

- **setNum**
  public void **setNum**( double [] n )  
  
  - **Usage**
* method to set the numerator to the array value Example: setNum([1,2,3]); would set the
numerator to $z^2 + 2z + 3$ an error message is displayed if the array is larger than the $z$
order

* **toString**
public String toString()

  - Usage
    * toString in the form of: $z$order = x num = [x] den = [x]

* **zeroNumAndDen**
public void zeroNumAndDen()

  - Usage
    * this method is for convenience in setting up $z$ transfer functions it makes sure that the
      function is reset to zero from the initial conditions

Methods inherited from class simtk.SimDiffEq

* **clear**
public void clear()

  - Usage
    * clears the text field, sets up headers and resets the index counter

* **doIteration**
public void doIteration()

* **getInCoef**
public SimDoubleArray getInCoef()

  - Usage
    * return the input coefficients

* **getOutCoef**
public SimDoubleArray getOutCoef()

  - Usage
    * return the output coefficients

* **setZeros**
public void setZeros()

  - Usage
    * zero all the coefficients for easy settings

* **toDiffEqString**
public String toDiffEqString()

  - Usage
* output the string format of the difference equation. special function to iterate through all
of the parameters and create the difference equation. This function is also used for the
textBox for the dynamic pretty print viewing of the diff eq

- **toString**
  ```java
  public String toString()
  ```
  
  - **Usage**
    
    * general to String equation for the diff eq. this uses the toDiffEqString() function for the
    special formatting so that one doesn’t overloaded

Methods inherited from class simtk.SimFunction

- **addInput**
  ```java
  public void addInput(simtk.HasOutput outputBlock)
  ```
  
  - **Usage**
    
    * this AddInput adds from a HasOutput block an input to the vector function and increases
    the inputIndex count

- **addInput**
  ```java
  public void addInput(simtk.SimBuffer bufref)
  ```
  
  - **Usage**
    
    * AddInput adds an input to the vector function and increases the inputIndex count

- **doIteration**
  ```java
  abstract void doIteration()
  ```
  
  - **Usage**
    
    * this is the method that should be implemented in all SimBlock’s

- **getInput**
  ```java
  public SimBuffer getInput()
  ```
  
  - **Usage**
    
    * get the first input. there might be a problem here because the old method of the implicit
    in buffer is used

- **getInput**
  ```java
  public SimBuffer getInput(int index)
  ```
  
  - **Usage**
    
    * get the indexed input, referenced from as 1 is the first.
    
    If the reference is out of range it will return the the single input value.

    The code is not so clean here where it will still print an error message when the double
    input functions are just beginning to set themselves up and before the.???

- **getInputCount**
  ```java
  public int getInputCount()
  ```
- **Usage**
  * get the inputCount, or the number of elements on the list

- **getOutput**
  
  ```java
  public SimBuffer getOutput()
  ```
  
  - **Usage**
    * Gets the output buffer

- **iterateBuffers**
  
  ```java
  public void iterateBuffers()
  ```
  
  - **Usage**
    * overload the do nothing iterateBuffers and iterate the output buffers

- **removeInputs**
  
  ```java
  public void removeInputs()
  ```
  
  - **Usage**
    * remove the all of the inputs from the block so a new system can be set up

- **setInput**
  
  ```java
  public void setInput( int index, simtk.SimBuffer bufref)
  ```
  
  - **Usage**
    * Set the indexed input to the passed buffer. note: this has out of index problems

- **setInput**
  
  ```java
  public void setInput( simtk.SimBuffer bufref)
  ```
  
  - **Usage**
    * Set the input to the passed buffer, this is replaced with the newer addInput function which sounds more like what it does

- **setInput1**
  
  ```java
  public void setInput1( simtk.SimBuffer bufref)
  ```
  
  - **Usage**
    * This is here to support old systems, it just calls addInput(bufref). so it is not exactly input 2

- **setInput2**
  
  ```java
  public void setInput2( simtk.SimBuffer bufref)
  ```
  
  - **Usage**
    * This is here to support old systems, it just calls addInput(bufref). so it is not exactly input 2

- **setOutput**
  
  ```java
  public void setOutput( simtk.SimBuffer bufref)
  ```
  
  - **Usage**
    * sets the output to a passed buffer
Methods inherited from class simtk.SimBlock

- **clear**
  public void clear()
  
  - **Usage**
  
  * this is the general function to clear a block. this is just an empty method but if more specific functionality is needed this should be overridden

- **doIteration**
  abstract void doIteration()
  
  - **Usage**
  
  * this is the method that should be implemented in all Simblock’s

- **doOutputBufferIteration**
  public void doOutputBufferIteration()
  
  - **Usage**
  
  * iterate the output buffer(s), or shift the history of it

- **getPanel**
  public JPanel getPanel()
  
  - **Usage**
  
  * generates the default panel and the title

- **getSimParams**
  public SimParameters getSimParams()
  
  - **Usage**
  
  * get the parameters from a parameter class

- **getVariableContainer**
  public VariableContainer getVariableContainer()
  
  - **Usage**
  
  * returns the conatainer for all the variables. this is mainly used for the operations done on the varContainer

- **isHidden**
  public boolean isHidden()
  
  - **Usage**
  
  * test if the hidden flag has been set

- **iterateBuffers**
  public void iterateBuffers()
  
  - **Usage**
  
  * This function does nothing, but blocks with outputs or internal buffers such as SimFunctionFactory should overload it for it’s own special pursposes
• **Redefine**
  public void redefine()
  
  - Usage
    * transfer upper level adjustments to the lower level. This method is overloaded when it is to
    be used by a lower function

• **SetHidden**
  public void setHidden()
  
  - Usage
    * keep the block from being loaded to the factory display if it is panelable

• **SetHidden**
  public void setHidden( boolean hide )
  
  - Usage
    * keep the block from being loaded to the factory display if it is panelable

• **SetSimParams**
  public void setSimParams( simtk.SimParameters sp )
  
  - Usage
    * set the parameters from a parameter class

**Methods inherited from class simtk.SimBase**

• **getDescription**
  public String getDescription()

• **getReloadBlock**
  public Simulatable getReloadBlock()
  
  - Usage
    * set the block that gets called to resimulate when the a variable is changed.

• **getTitle**
  public String getTitle()
  
  - Usage
    * get the user defined title for the entire block

• **setDescription**
  public void setDescription( java.lang.String desc )

• **setReloadBlock**
  public void setReloadBlock( simtk.Simulatable reloadBlock )
  
  - Usage
    * set the block that gets called to resimulate when the a variable is changed.

• **setTitle**
  public void setTitle( java.lang.String title )
  
  - Usage
    * set the user defined title for the entire block
2.2.31 CLASS SimMultiply

Multiplies two input values

**Declaration**

```
public class SimMultiply
extends simtk.SimFunction2Input
```

**Constructors**

- `SimMultiply`
  ```
  public SimMultiply()
  ```

**Methods**

- `doIteration`
  ```
  public void doIteration()
  ```

**Methods inherited from class simtk.SimFunction2Input**

- `doIteration`
  ```
  abstract void doIteration()
  ```
  - Usage
    * this is the method that should be implemented in all SimBlock’s

- `getOutput`
  ```
  public SimBuffer getOutput()
  ```
  - Usage
    * get the output buffer of the passed buffer

- `setInput1`
  ```
  public void setInput1( simtk.SimBuffer bufref )
  ```
  - Usage
    * set the first Input to the passed buffer

- `setInput2`
  ```
  public void setInput2( simtk.SimBuffer bufref )
  ```
  - Usage
**setOutput**

```java
public void setOutput( SimBuffer bufref )
```

- **Usage**
  
  * set the output buffer to the passed buffer

---

**Methods inherited from class simtk.SimBlock**

- **clear**
  
  ```java
  public void clear( )
  ```
  
  - **Usage**
    
    * this is the general function to clear a block. this is just an empty method but if more specific functionality is needed this should be overridden

- **doIteration**
  
  ```java
  abstract void doIteration( )
  ```
  
  - **Usage**
    
    * this is the method that should be implemented in all Simblock’s

- **doOutputBufferIteration**
  
  ```java
  public void doOutputBufferIteration( )
  ```
  
  - **Usage**
    
    * iterate the output buffer(s), or shift the history of it

- **getPanel**
  
  ```java
  public JPanel getPanel( )
  ```
  
  - **Usage**
    
    * generates the default panel and the title

- **getSimParams**
  
  ```java
  public SimParameters getSimParams( )
  ```
  
  - **Usage**
    
    * get the parameters from a parameter class

- **getVariableContainer**
  
  ```java
  public VariableContainer getVariableContainer( )
  ```
  
  - **Usage**
    
    * returns the container for all the variables. this is mainly used for the operations done on the varContainer

- **isHidden**
  
  ```java
  public boolean isHidden( )
  ```
  
  - **Usage**
* test if the hidden flag has been set

- **iterateBuffers**
  ```java
  public void iterateBuffers()
  ```
  - **Usage**
  * This function does nothing, but blocks with outputs or internal buffers such as SimFunctionFactory should overload it for its own special purposes

- **reDefine**
  ```java
  public void reDefine()
  ```
  - **Usage**
  * transfer upper level adjustments to the lower level. This method is overloaded when it is to be used by a lower function

- **setHidden**
  ```java
  public void setHidden()
  ```
  - **Usage**
  * keep the block from being loaded to the factory display if it is panelable

- **setHidden**
  ```java
  public void setHidden( boolean hide )
  ```
  - **Usage**
  * keep the block from being loaded to the factory display if it is panelable

- **setSimParams**
  ```java
  public void setSimParams( simtk.SimParameters sp )
  ```
  - **Usage**
  * set the parameters from a parameter class

**Methods inherited from class simtk.SimBase**

- **getDescription**
  ```java
  public String getDescription()
  ```

- **getReloadBlock**
  ```java
  public Simulatable getReloadBlock()
  ```
  - **Usage**
  * set the block that gets called to resimulate when the a variable is changed.

- **getTitle**
  ```java
  public String getTitle()
  ```
  - **Usage**
  * get the user defined title for the entire block
- **setDescription**
  
  ```java
  public void setDescription( java.lang.String desc )
  ```

- **setReloadBlock**
  
  ```java
  public void setReloadBlock( simtk.Simulatable reloadBlock )
  ```
  
  **Usage**
  
  * set the block that gets called to resimulate when the a variable is changed.

- **setTitle**
  
  ```java
  public void setTitle( java.lang.String title )
  ```
  
  **Usage**
  
  * set the user defined title for the entire block

### 2.2.32 Class **SimNoise**

This is a source that after each iteration it just outputs a random value from -1 to 1. A gain is also available.

#### Declaration

```java
public class SimNoise
extends simtk.SimSource
implements SimPanelable
```

#### Fields

- private SimNumber gain

#### Constructors

- **SimNoise**
  
  ```java
  public SimNoise( )
  ```
  
  **Usage**
  
  * default sets gain = 1
Methods

- `doIteration`
  ```java
  public void doIteration()
  ```

- `setGain`
  ```java
  public void setGain(double gain)
  ```

Methods inherited from class `simtk.SimSource`

- `doIteration`
  ```java
  abstract void doIteration()
  ```
  - Usage
    * this is the method that should be implemented in all SimBlock’s

- `getOutput`
  ```java
  public SimBuffer getOutput()
  ```

- `iterateBuffers`
  ```java
  public void iterateBuffers()
  ```
  - Usage
    * overload the do nothing iterateBuffers and iterate the output buffers

- `setOutput`
  ```java
  public void setOutput(simtk.SimBuffer bufref)
  ```

Methods inherited from class `simtk.SimBlock`

- `clear`
  ```java
  public void clear()
  ```
  - Usage
    * this is the general function to clear a block. this is just an empty method but if more specific functionality is needed this should be overridden

- `doIteration`
  ```java
  abstract void doIteration()
  ```
  - Usage
    * this is the method that should be implemented in all SimBlock’s

- `doOutputBufferIteration`
  ```java
  public void doOutputBufferIteration()
  ```
  - Usage
    * iterate the output buffer(s), or shift the history of it
- **getPanel**
  public JPanel getPanel( )
  - Usage
    * generates the default panel and the title

- **getSimParams**
  public SimParameters getSimParams( )
  - Usage
    * get the parameters from a parameter class

- **getVariableContainer**
  public VariableContainer getVariableContainer( )
  - Usage
    * returns the container for all the variables. this is mainly used for the operations done on the varContainer

- **isHidden**
  public boolean isHidden( )
  - Usage
    * test if the hidden flag has been set

- **iterateBuffers**
  public void iterateBuffers( )
  - Usage
    * This function does nothing, but blocks with outputs or internal buffers such as SimFunctionFactory should overload it for its own special purposes

- **reDefine**
  public void reDefine( )
  - Usage
    * transfer upper level adjustments to the lower level. this method is overloaded when it is to be used by a lower function

- **setHidden**
  public void setHidden( )
  - Usage
    * keep the block from being loaded to the factory display if it is panelable

- **setHidden**
  public void setHidden( boolean hide )
  - Usage
    * keep the block from being loaded to the factory display if it is panelable

- **setSimParams**
  public void setSimParams( simtk.SimParameters sp )
  - Usage
    * set the parameters from a parameter class
Methods inherited from class simtk.SimBase

- **getDescription**
  ```java
  public String getDescription()
  ```
  - Usage
    * set the block that gets called to resimulate when the a variable is changed.

- **getReloadBlock**
  ```java
  public Simulatable getReloadBlock()
  ```
  - Usage
    * set the block that gets called to resimulate when the a variable is changed.

- **getTitle**
  ```java
  public String getTitle()
  ```
  - Usage
    * get the user defined title for the entire block

- **setDescription**
  ```java
  public void setDescription(java.lang.String desc)
  ```

- **setReloadBlock**
  ```java
  public void setReloadBlock(simtk.Simulatable reloadBlock)
  ```
  - Usage
    * set the block that gets called to resimulate when the a variable is changed.

- **setTitle**
  ```java
  public void setTitle(java.lang.String title)
  ```
  - Usage
    * set the user defined title for the entire block

2.2.33 Class SimNothing

Nothing function, just outputs a zero. This can be used for test cases of zero noise instead of having some noise

Declaration

```java
public class SimNothing
extends simtk.SimSource
implements SimPanelable
```
CONSTRUCTORS

- **SimNothing**
  
  public SimNothing()

  - Usage
    
    * defaults are set to delay = 0 and gain = 1

METHODS

- **doIteration**
  
  public void doIteration()

  - Usage
    
    * set the output to the gain value after the delay time

METHODS INHERITED FROM CLASS simtk.SimSource

- **doIteration**
  
  abstract void doIteration()

  - Usage
    
    * this is the method that should be implemented in all SimBlock’s

- **getOutput**
  
  public SimBuffer getOutput()

- **iterateBuffers**
  
  public void iterateBuffers()

  - Usage
    
    * overload the do nothing iterateBuffers and iterate the output buffers

- **setOutput**
  
  public void setOutput( simtk.SimBuffer bufref )

METHODS INHERITED FROM CLASS simtk.SimBlock

- **clear**
  
  public void clear()

  - Usage
    
    * this is the general function to clear a block. this is just an empty method but if more specific functionality is needed this should be overridden
• **doIteration**
  
  abstract void doIteration( )
  
  − Usage
    * this is the method that should be implemented in all Simblock’s

• **doOutputBufferIteration**
  
  public void doOutputBufferIteration( )
  
  − Usage
    * iterate the output buffer(s), or shift the history of it

• **getPanel**
  
  public JPanel getPanel( )
  
  − Usage
    * generates the default panel and the title

• **getSimParams**
  
  public SimParameters getSimParams( )
  
  − Usage
    * get the parameters from a parameter class

• **getVariableContainer**
  
  public VariableContainer getVariableContainer( )
  
  − Usage
    * returns the container for all the variables. this is mainly used for the operations done on the varContainer

• **isHidden**
  
  public boolean isHidden( )
  
  − Usage
    * test if the hidden flag has been set

• **iterateBuffers**
  
  public void iterateBuffers( )
  
  − Usage
    * This function does nothing, but blocks with outputs or internal buffers such as SimFunctionFactory should overload it for it’s own special purposes

• **reDefine**
  
  public void reDefine( )
  
  − Usage
    * transfer upper level adjustments to the lower level. this method is overloaded when it is to be used by a lower function

• **setHidden**
  
  public void setHidden( )
simtk—SimNPID

- **Usage**
  * keep the block from being loaded to the factory display if it is panelable

- **setHidden**
  
  public void setHidden( boolean hide )

  - **Usage**
    * keep the block from being loaded to the factory display if it is panelable

- **setSimParams**
  
  public void setSimParams( simtk.SimParameters sp )

  - **Usage**
    * set the parameters from a parameter class

**Methods inherited from class simtk.SimBase**

- **getDescription**
  
  public String getDescription( )

- **getReloadBlock**
  
  public Simulatable getReloadBlock( )

  - **Usage**
    * set the block that gets called to resimulate when the a variable is changed.

- **getTitle**
  
  public String getTitle( )

  - **Usage**
    * get the user defined title for the entire block

- **setDescription**
  
  public void setDescription( java.lang.String desc )

- **setReloadBlock**
  
  public void setReloadBlock( simtk.Simulatable reloadBlock )

  - **Usage**
    * set the block that gets called to resimulate when the a variable is changed.

- **setTitle**
  
  public void setTitle( java.lang.String title )

  - **Usage**
    * set the user defined title for the entire block

2.2.34 **Class SimNPID**

compilation of sim blocks to create the NPID
public class SimNPID
extends simtk.SimFunctionFactory
implements SimPanelable, ContainsFactory

• SimNPID
  public SimNPID( )
  – Usage
    * constructor sets gain to default of unity

• connectBlocks
  public void connectBlocks( )
  – Usage
    * Make the connections for this block. The setInput of this factory calls this to
      setup. It turns out to be a rather interesting system the setInput to this factory
      then in turn setups up this internal system

Methods inherited from class simtk.SimFunctionFactory

• addInput
  public void addInput( simtk.HasOutput inputBlock )
  – Usage
    * set the input of this factory this is the interesting one

• addInput
  public void addInput( simtk.SimBuffer sb )
  – Usage
    * set the input of this factory. this is the interesting one

• clear
  public void clear( )
  – Usage
* this function is required to reset the buffers and values for the next iterations to set up the factory class

- **connectBlocks**
  abstract void connectBlocks( )
  - **Usage**
    * Make the connections for this block. The setInput of this factory calls this to setup. It turns out to be a rather interesting system. The setInput() or addInput() to this factory then in turn set’s up this internal system. this is to be implemented.

- **doIteration**
  public void doIteration( )
  - **Usage**
    * do a calculation for the new output

- **getPanel**
  public JPanel getPanel( )

- **iterateBuffers**
  public void iterateBuffers( )
  - **Usage**
    * cycle the buffers

- **setInput**
  public void setInput( simtk.SimBuffer sb )
  - **Usage**
    * set the input of this factory. this is to be implemented in the Simblock to call the connect blocks

- **toString**
  public String toString( )

**Methods inherited from class simtk.SimFunction**

- **addInput**
  public void addInput( simtk.HasOutput outputBlock )
  - **Usage**
    * this AddInput adds from a HasOutput block an input to the vector function and increases the inputIndex count

- **addInput**
  public void addInput( simtk.SimBuffer bufref )
  - **Usage**
    * AddInput adds an input to the vector function and increases the inputIndex count
• **doIteration**
  abstract void doIteration( )
  
  – Usage
  * this is the method that should be implemented in all SimBlock’s

• **getInput**
  public SimBuffer getInput( )
  
  – Usage
  * get the first input. there might be a problem here because the old method of the implicit in buffer is used

• **getInput**
  public SimBuffer getInput( int index )
  
  – Usage
  * get the indexed input, referenced from as 1 is the first.
    If the reference is out of range it will return the the single input value.
    The code is not so clean here where it will still print an error message when the double input functions are just beginning to set themselves up and before the.???

• **getInputCount**
  public int getInputCount( )
  
  – Usage
  * get the inputCount, or the number of elements on the list

• **getOutput**
  public SimBuffer getOutput( )
  
  – Usage
  * Gets the output buffer

• **iterateBuffers**
  public void iterateBuffers( )
  
  – Usage
  * overload the do nothing iterateBuffers and iterate the output buffers

• **removeInputs**
  public void removeInputs( )
  
  – Usage
  * remove all of the inputs from the block so a new system can be set up

• **setInput**
  public void setInput( int index, simtk.SimBuffer bufref )
  
  – Usage
  * Set the indexed input to the passed buffer. note: this has out of index problems
• `setInput`
  public void `setInput`( simtk.SimBuffer `bufref` )
  
  – Usage
  * Set the input to the passed buffer, this is replaced with the newer `addInput` function which sounds more like what it does

• `setInput1`
  public void `setInput1`( simtk.SimBuffer `bufref` )
  
  – Usage
  * This is here to support old systems, it just calls `addInput(bufref)`. so it is not exactly input 2

• `setInput2`
  public void `setInput2`( simtk.SimBuffer `bufref` )
  
  – Usage
  * This is here to support old systems, it just calls `addInput(bufref)`. so it is not exactly input 2

• `setOutput`
  public void `setOutput`( simtk.SimBuffer `bufref` )
  
  – Usage
  * sets the output to a passed buffer

Methods inherited from class `simtk.SimBlock`

• `clear`
  public void `clear`( )
  
  – Usage
  * this is the general function to clear a block. this is just an empty method but if more specific functionality is needed this should be overridden

• `doIteration`
  abstract void `doIteration`( )
  
  – Usage
  * this is the method that should be implemented in all Simblock’s

• `doOutputBufferIteration`
  public void `doOutputBufferIteration`( )
  
  – Usage
  * iterate the output buffer(s), or shift the history of it

• `getPanel`
  public JPanel `getPanel`( )
- **Usage**
  * generates the default panel and the title

- **getSimParams**
  public SimParameters getSimParams()
  - **Usage**
    * get the parameters from a parameter class

- **getVariableContainer**
  public VariableContainer getVariableContainer()
  - **Usage**
    * returns the container for all the variables. This is mainly used for the operations done on the `varContainer`

- **isHidden**
  public boolean isHidden()
  - **Usage**
    * test if the hidden flag has been set

- **iterateBuffers**
  public void iterateBuffers()
  - **Usage**
    * This function does nothing, but blocks with outputs or internal buffers such as `SimFunctionFactory` should overload it for its own special purposes

- **reDefine**
  public void reDefine()
  - **Usage**
    * transfer upper level adjustments to the lower level. This method is overloaded when it is to be used by a lower function

- **setHidden**
  public void setHidden()
  - **Usage**
    * keep the block from being loaded to the factory display if it is panelable

- **setHidden**
  public void setHidden(boolean hide)
  - **Usage**
    * keep the block from being loaded to the factory display if it is panelable

- **setSimParams**
  public void setSimParams(simtk.SimParameters sp)
  - **Usage**
    * set the parameters from a parameter class
Methods inherited from class simtk.SimBase

• getDescription
  public String getDescription()

• getReloadBlock
  public Simulatable getReloadBlock()
    – Usage
      * set the block that gets called to resimulate when the a variable is changed.

• getTitle
  public String getTitle()
    – Usage
      * get the user defined title for the entire block

• setDescription
  public void setDescription(java.lang.String desc)

• setReloadBlock
  public void setReloadBlock(simtk.Simulatable reloadBlock)
    – Usage
      * set the block that gets called to resimulate when the a variable is changed.

• setTitle
  public void setTitle(java.lang.String title)
    – Usage
      * set the user defined title for the entire block

2.2.35 Class SimNPIDwc

class which makes the SimNPID parameterized

Declaration

public class SimNPIDwc
  extends simtk.SimNPID

Fields

• private SimNumber wc
Constructors

- **SimNPIDwc**
  
  ```java
  public SimNPIDwc()
  ```

  - **Usage**
    - * constructor sets gain to default of unity

Methods

- **doIteration**
  
  ```java
  public void doIteration()
  ```

  - **Usage**
    - * do the redefining and then the lower level iterations

- **getPanel**
  
  ```java
  public JPanel getPanel()
  ```

  - **Usage**
    - * override the default and previous dialog panels

- **reDefine**
  
  ```java
  public void reDefine()
  ```

  - **Usage**
    - * transfer upper level adjustments to the lower level

Methods inherited from class simtk.SimPID

- **connectBlocks**
  
  ```java
  public void connectBlocks()
  ```

  - **Usage**
    - * Make the connections for this block. The setInput of this factory calls this to setup. It turns out to be a rather interesting system the setInput to this factory then in turn setups up this internal system
Methods inherited from class Simtk.SimFunctionFactory

- `addInput`
  public void `addInput` ( Simtk.HasOutput `inputBlock` )
  
  - Usage
    * set the input of this factory this is the interesting one

- `addInput`
  public void `addInput` ( Simtk.SimBuffer `sb` )
  
  - Usage
    * set the input of this factory. this is the interesting one

- `clear`
  public void `clear` ( )
  
  - Usage
    * this function is required to reset the buffers and values for the next iterations to set up the factory class

- `connectBlocks`
  abstract void `connectBlocks` ( )
  
  - Usage
    * Make the connections for this block. The setInput of this factory calls this to setup. It turns out to be a rather interesting system. The setInput() or addInput() to this factory then in turn set's up this internal system. this is to be implemented.

- `doIteration`
  public void `doIteration` ( )
  
  - Usage
    * do a calculation for the new output

- `getPanel`
  public JPanel `getPanel` ( )

- `iterateBuffers`
  public void `iterateBuffers` ( )
  
  - Usage
    * cycle the buffers

- `setInput`
  public void `setInput` ( Simtk.SimBuffer `sb` )
  
  - Usage
    * set the input of this factory. this is to be implemented in the Simblock to call the connect blocks

- `toString`
  public String `toString` ( )
Methods inherited from class simtk.SimFunction

- **addInput**
  public void addInput( simtk.HasOutput outputBlock )
  
  - Usage
    * this AddInput adds from a HasOutput block an input to the vector function and increases the inputIndex count

- **addInput**
  public void addInput( simtk.SimBuffer bufref )
  
  - Usage
    * AddInput adds an input to the vector function and increases the inputIndex count

- **doIteration**
  abstract void doIteration( )
  
  - Usage
    * this is the method that should be implemented in all SimBlock’s

- **getInput**
  public SimBuffer getInput( )
  
  - Usage
    * get the first input. there might be a problem here because the old method of the implicit in buffer is used

- **getInput**
  public SimBuffer getInput( int index )
  
  - Usage
    * get the indexed input, referenced from as 1 is the first.
    If the reference is out of range it will return the the single input value.
    The code is not so clean here where it will still print an erorr message when the double input functions are just beginning to set themselves up and before the.???

- **getInputCount**
  public int getInputCount( )
  
  - Usage
    * get the inputCount, or the number of elements on the list

- **getOutput**
  public SimBuffer getOutput( )
  
  - Usage
    * Gets the output buffer

- **iterateBuffers**
  public void iterateBuffers( )
• **removeInputs**
  ```java
  public void removeInputs()
  ```
  **Usage**
  * remove the all of the inputs from the block so a new system can be set up

• **setInput**
  ```java
  public void setInput(int index, simtk.SimBuffer bufref)
  ```
  **Usage**
  * Set the indexed input to the passed buffer. note: this has out of index problems

• **setInput**
  ```java
  public void setInput(simtk.SimBuffer bufref)
  ```
  **Usage**
  * Set the input to the passed buffer, this is replaced with the newer addInput function which sounds more like what it does

• **setInput1**
  ```java
  public void setInput1(simtk.SimBuffer bufref)
  ```
  **Usage**
  * This is here to support old systems, it just calls addInput(bufref). so it is not exactly input 2

• **setInput2**
  ```java
  public void setInput2(simtk.SimBuffer bufref)
  ```
  **Usage**
  * This is here to support old systems, it just calls addInput(bufref). so it is not exactly input 2

• **setOutput**
  ```java
  public void setOutput(simtk.SimBuffer bufref)
  ```
  **Usage**
  * sets the output to a passed buffer

**Methods inherited from class simtk.SimBlock**

• **clear**
  ```java
  public void clear()
  ```
  **Usage**
  * this is the general function to clear a block. this is just an empty method but if more specific functionality is needed this should be overridden
• doIteration
  abstract void doIteration( )
    – Usage
      * this is the method that should be implemented in all Simblock’s

• doOutputBufferIteration
  public void doOutputBufferIteration( )
    – Usage
      * iterate the output buffer(s), or shift the history of it

• getPanel
  public JPanel getPanel( )
    – Usage
      * generates the default panel and the title

• getSimParams
  public SimParameters getSimParams( )
    – Usage
      * get the parameters from a parameter class

• getVariableContainer
  public VariableContainer getVariableContainer( )
    – Usage
      * returns the container for all the variables. this is mainly used for the operations done on
        the varContainer

• isHidden
  public boolean isHidden( )
    – Usage
      * test if the hidden flag has been set

• iterateBuffers
  public void iterateBuffers( )
    – Usage
      * This function does nothing, but blocks with outputs or internal buffers such as
        SimFunctionFactory should overload it for its own special purposes

• reDefine
  public void reDefine( )
    – Usage
      * transfer upper level adjustments to the lower level. this method is overloaded when it is to
        be used by a lower function

• setHidden
  public void setHidden( )
Methods inherited from class `simtk.SimBase`

- `getDescription`
  public String `getDescription`();
  
  Usage
  * get the user defined title for the entire block

- `getTitle`
  public String `getTitle`();
  
  Usage
  * get the user defined title for the entire block

- `setDescription`
  public void `setDescription`( java.lang.String desc );

- `setReloadBlock`
  public void `setReloadBlock`( simtk.Simulatable reloadBlock );
  
  Usage
  * set the block that gets called to resimulate when the a variable is changed.

- `setTitle`
  public void `setTitle`( java.lang.String title );
  
  Usage
  * set the user defined title for the entire block

### 2.2.36 Class `SimParameters`

- `setHidden`
  public void `setHidden`( boolean hide );
  
  Usage
  * keep the block from being loaded to the factory display if it is panelable

- `setSimParams`
  public void `setSimParams`( simtk.SimParameters sp );
  
  Usage
  * set the parameters from a parameter class
public class SimParameters  
extends simtk.SimBase  
implements SimPanelable

• private int zorder

• private HashMap paramHash

• private HashMap paramUpdatedHash

• SimParameters
  public SimParameters( )

• containsParam
  public boolean containsParam( java.lang.String paramName )
    – Usage
      * Has the parameter been added to the global specs yet

• getPanel
  public JPanel getPanel( )
    – Usage
      * get panel from the sim parameters

• getParam
  public double getParam( java.lang.String paramName )
- **Usage**
  - this is the function to get the value of the global parameter

- **getStepSize**
  - public double getStepSize()  

- **getTFinal**
  - public double getTFinal()  

- **getZOrder**
  - public int getZOrder()  

- **isParamUpdated**
  - public boolean isParamUpdated( java.lang.String paramName )  
    - **Usage**
      - * is the specific variable updated?  

- **refreshPanel**
  - public void refreshPanel()  
    - **Usage**
      - * build the panel by iterating over the hash and building a var container which produces a panel  

- **setParameter**
  - public void setParam( java.lang.String paramName, double value )  
    - **Usage**
      - * this is the function to add and edit a global parameter  

- **setParameters**
  - public void setParameters( double tfinal, double stepsize, int zorder )  

- **setParameterNotUpdated**
  - public void setParamNotUpdated( java.lang.String paramName )  
    - **Usage**
      - * set the hash as not updated  

- **setStepSize**
  - public void setStepSize( double s )  

- **setTFinal**
  - public void setTFinal( double t )  

- **setZOrder**
  - public void setZOrder( int z )
Methods inherited from class simtk.SimBase

- `getDescription`
  ```java
  public String getDescription()
  ```
- `getReloadBlock`
  ```java
  public Simulatable getReloadBlock()
  ```
  - Usage
    * set the block that gets called to resimulate when the a variable is changed.
- `getTitle`
  ```java
  public String getTitle()
  ```
  - Usage
    * get the user defined title for the entire block
- `setDescription`
  ```java
  public void setDescription(java.lang.String desc)
  ```
- `setReloadBlock`
  ```java
  public void setReloadBlock(simtk.Simulatable reloadBlock)
  ```
  - Usage
    * set the block that gets called to resimulate when the a variable is changed.
- `setTitle`
  ```java
  public void setTitle(java.lang.String title)
  ```
  - Usage
    * set the user defined title for the entire block

### 2.2.37 Class SimPid

This is the differential pid sim block defined as $u/e = kp + ki/s + kd*s/(tau*s+1)^2$. This is internally defined as a 3rd order s tf. The approximate differentiator is a 2nd order, which was found by Jing’s paper to be more accurate than 1st order.

**Declaration**

```java
public class SimPid
extends simtk.SimStf
implements SimPanelable
```
Constructors

- *SimPid*
  
  public SimPid( )

  - Usage
    * constructor set kp=ki=kd=tau=1 u/e = kp + ki/s + kd*s/(tau*s+1)^2

Methods

- *reDefine*
  
  public void reDefine( )

  - Usage
    * This is made to recalculate with new parameters, step sizes parameters. This is a method that might be good to have defined in the sim Block as abstract this is also where the super mulit parameters are calculated in this function omega_c is a supper of kp and kd

    kp = wc^2

    kd = 2*wc

    overrides the dolteration of SimZtf and SimStf and loads in the new parameters then it calls the overridden dolteration from Stf which then overrides the Ztf for the final difference equation, 3 levels of abstraction

Methods inherited from class simtk.SimStf

- *displayNotProperErr*
  
  private void displayNotProperErr( )

  - Usage
    * improper functions error

- *generateZPoly*
  
  public double generateZPoly( double [] s )

  - Usage
* generate the polynomial as an array from the s polynomial the order is extracted by the size of s this is the magic function for the conversion of continuous to discrete see the derivations done on 4/26/02 with mathematica and the papers solution I need to type up these derivations

- **makeNewStf**
  
  ```java
  public void makeNewStf(int snum_order, int sden_order)
  ```
  
  - **Usage**
    *
    creates the num and den checking if improper else make default

- **reDefine**
  
  ```java
  public void reDefine()
  ```
  
  - **Usage**
    *
    this is a method that might be good to have defined in the sim Block as abstract this is made to recalculate with new parameters, step sizes parameters

- **setSDen**
  
  ```java
  public void setSDen(double[] a)
  ```
  
  - **Usage**
    *
    simply sets the s denominator to an array value

- **setSNum**
  
  ```java
  public void setSNum(double[] a)
  ```
  
  - **Usage**
    *
    simply sets the s numerator to an array value

- **toString**
  
  ```java
  public String toString()
  ```
  
  - **Usage**
    *
    toString in the form of: num_order = x den_order = y num = [x] den = [y]

- **zeroStf**
  
  ```java
  public void zeroStf()
  ```
  
  - **Usage**
    *
    this zeroSTF method is for convenience in setting up z transfer functions it makes sure that the function is reset to zero from the initial conditions

**Methods inherited from class simtk.SimZtf**

- **doIteration**
  
  ```java
  public void doIteration()
  ```
  
  - **Usage**
    *
    this method overrides the doIteration of the diffeq. this way it first redefines the variables and then iterates with the inherited structure
- **doIterationDirect**
  
  public void doIterationDirect()
  
  - **Usage**
  
  * this is the really cool simple method that converts ztf to diffEQ. The function does the operation in the following order:
    1. parses the numerator and denominator
    2. creates the difference equation
    3. does one step operation.

- **reDefine**
  
  public void reDefine()
  
  - **Usage**
  
  * this is a method that might be good to have defined in the sim Block as abstract this is made to recalculate with new parameters, step sizes parameters

- **setDen**
  
  public void setDen( double [] n )
  
  - **Usage**
  
  * method to set the denominator to the array value Example: setDen([1,2,3]); would set the denominator to \( z^2 + 2z + 3 \) an error message is displayed if the array is larger than the \( z \) order

- **setNum**
  
  public void setNum( double [] n )
  
  - **Usage**
  
  * method to set the numerator to the array value Example: setNum([1,2,3]); would set the numerator to \( z^2 + 2z + 3 \) an error message is displayed if the array is larger than the \( z \) order

- **toString**
  
  public String toString()
  
  - **Usage**
  
  * toString in the form of: zorder = x num = [x] den = [x]

- **zeroNumAndDen**
  
  public void zeroNumAndDen()
  
  - **Usage**
  
  * this method is for convenience in setting up z transfer functions it makes sure that the function is reset to zero from the initial conditions
Methods inherited from class simtk.SimDiffEq

- **clear**
  public void clear( )
  
<table>
<thead>
<tr>
<th>Usage</th>
</tr>
</thead>
<tbody>
<tr>
<td>* clears the text field, sets up headers and resets the index counter</td>
</tr>
</tbody>
</table>

- **doIteration**
  public void doIteration( )

- **getInCoef**
  public SimDoubleArray getInCoef( )
  
<table>
<thead>
<tr>
<th>Usage</th>
</tr>
</thead>
<tbody>
<tr>
<td>* return the input coefficients</td>
</tr>
</tbody>
</table>

- **getOutCoef**
  public SimDoubleArray getOutCoef( )
  
<table>
<thead>
<tr>
<th>Usage</th>
</tr>
</thead>
<tbody>
<tr>
<td>* return the output coefficients</td>
</tr>
</tbody>
</table>

- **setZeros**
  public void setZeros( )
  
<table>
<thead>
<tr>
<th>Usage</th>
</tr>
</thead>
<tbody>
<tr>
<td>* zero all the coefficients for easy settings</td>
</tr>
</tbody>
</table>

- **toDiffEqString**
  public String toDiffEqString( )
  
<table>
<thead>
<tr>
<th>Usage</th>
</tr>
</thead>
<tbody>
<tr>
<td>* output the string format of the difference equation. special function to iterate through all</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
</tbody>
</table>

- **toString**
  public String toString( )
  
<table>
<thead>
<tr>
<th>Usage</th>
</tr>
</thead>
<tbody>
<tr>
<td>* general to String equation for the diff eq. this uses the toDiffEqString() function for the</td>
</tr>
<tr>
<td></td>
</tr>
</tbody>
</table>
Methods inherited from class simtk.SimFunction

- **addInput**
  public void addInput( simtk.HasOutput outputBlock )
  
  - Usage
  * this AddInput adds from a HasOutput block an input to the vector function and increases
    the inputIndex count

- **addInput**
  public void addInput( simtk.SimBuffer bufref )
  
  - Usage
  * AddInput adds an input to the vector function and increases the inputIndex count

- **doIteration**
  abstract void doIteration( )
  
  - Usage
  * this is the method that should be implemented in all SimBlock’s

- **getInput**
  public SimBuffer getInput( )
  
  - Usage
  * get the first input. there might be a problem here because the old method of the implicit
    in buffer is used

- **getInput**
  public SimBuffer getInput( int index )
  
  - Usage
  * get the indexed input, referenced from as 1 is the first.
    If the reference is out of range it will return the the single input value.
    The code is not so clean here where it will still print an error message when the double
    input functions are just beginning to set themselves up and before the.???

- **getInputCount**
  public int getInputCount( )
  
  - Usage
  * get the inputCount, or the number of elements on the list

- **getOutput**
  public SimBuffer getOutput( )
  
  - Usage
  * Gets the output buffer

- **iterateBuffers**
  public void iterateBuffers( )
- **Usage**
  * overload the do nothing iterateBuffers and iterate the output buffers

- **removeInputs**
  
  ```java
  public void removeInputs()
  ```

  - **Usage**
    * remove the all of the inputs from the block so a new system can be set up

- **setInput**
  
  ```java
  public void setInput( int index, simtk.SimBuffer bufref )
  ```

  - **Usage**
    * Set the indexed input to the passed buffer. note: this has out of index problems

- **setInput**
  
  ```java
  public void setInput( simtk.SimBuffer bufref )
  ```

  - **Usage**
    * Set the input to the passed buffer, this is replaced with the newer addInput function which sounds more like what it does

- **setInput1**
  
  ```java
  public void setInput1( simtk.SimBuffer bufref )
  ```

  - **Usage**
    * This is here to support old systems, it just calls addInput(bufref). so it is not exactly input 2

- **setInput2**
  
  ```java
  public void setInput2( simtk.SimBuffer bufref )
  ```

  - **Usage**
    * This is here to support old systems, it just calls addInput(bufref). so it is not exactly input 2

- **setOutput**
  
  ```java
  public void setOutput( simtk.SimBuffer bufref )
  ```

  - **Usage**
    * sets the output to a passed buffer

**Methods inherited from class simtk.SimBlock**

- **clear**
  
  ```java
  public void clear()
  ```

  - **Usage**
    * this is the general function to clear a block. this is just an empty method but if more specific functionality is needed this should be overridden
• **doIteration**
  abstract void **doIteration**()

  — **Usage**
  * this is the method that should be implemented in all Simblock’s

• **doOutputBufferIteration**
  public void **doOutputBufferIteration**()

  — **Usage**
  * iterate the output buffer(s), or shift the history of it

• **getPanel**
  public JPanel **getPanel**()

  — **Usage**
  * generates the default panel and the title

• **getSimParams**
  public SimParameters **getSimParams**()

  — **Usage**
  * get the parameters from a parameter class

• **getVariableContainer**
  public VariableContainer **getVariableContainer**()

  — **Usage**
  * returns the container for all the variables. this is mainly used for the operations done on
  the varContainer

• **isHidden**
  public boolean **isHidden**()

  — **Usage**
  * test if the hidden flag has been set

• **iterateBuffers**
  public void **iterateBuffers**()

  — **Usage**
  * This function does nothing, but blocks with outputs or internal buffers such as
  SimFunctionFactory should overload it for it’s own special purposes

• **reDefine**
  public void **reDefine**()

  — **Usage**
  * transfer upper level adjustments to the lower level. this method is overloaded when it is to
  be used by a lower function

• **setHidden**
  public void **setHidden**()
setHidden
public void setHidden(boolean hide)

setSimParams
public void setSimParams(simtk.SimParameters sp)

**Methods inherited from class simtk.SimBase**

getDescription
public String getDescription()

getReloadBlock
public Simulatable getReloadBlock()

getTitle
public String getTitle()

setDescription
public void setDescription(java.lang.String desc)

setReloadBlock
public void setReloadBlock(simtk.Simulatable reloadBlock)

setTitle
public void setTitle(java.lang.String title)

**2.2.38 Class SimPolyProfile**

Polynomial Profile provides a smoother response for position, velocity, and acceleration and Jerk in place of a step function or trapezoidal function.
**DECLARATION**

```java
public class SimPolyProfile
extends simtk.SimSource
implements SimPanelable
```

**FIELDS**

- private int k
- private SimNumber t_{risetime}  
  - settling time for of the profile
- private SimNumber gain
  - gain of the profile

**CONSTRUCTORS**

- `SimPolyProfile`
  ```java
  public SimPolyProfile()
  ```
  - Usage
  * defaults are set to delay = 0 and gain = 1, totalTime = 1

**METHODS**

- `clear`
  ```java
  public void clear()
  ```
- `doIteration`
  ```java
  public void doIteration()
  ```
  - Usage
  * set the output to the gain value after the delay time
**Methods inherited from class simtk.SimSource**

- **doIteration**
  ```java
  abstract void doIteration()
  
  - Usage
    * this is the method that should be implemented in all SimBlock’s
  ```

- **getOutput**
  ```java
  public SimBuffer getOutput()
  ```

- **iterateBuffers**
  ```java
  public void iterateBuffers()
  
  - Usage
    * overload the do nothing iterateBuffers and iterate the output buffers
  ```

- **setOutput**
  ```java
  public void setOutput( simtk.SimBuffer bufref )
  ```

**Methods inherited from class simtk.SimBlock**

- **clear**
  ```java
  public void clear()
  
  - Usage
    * this is the general function to clear a block. this is just an empty method but if more specific functionality is needed this should be overridden
  ```

- **doIteration**
  ```java
  abstract void doIteration()
  
  - Usage
    * this is the method that should be implemented in all Simblock’s
  ```

- **doOutputBufferIteration**
  ```java
  public void doOutputBufferIteration()
  
  - Usage
    * iterate the output buffer(s), or shift the history of it
  ```

- **getPanel**
  ```java
  public JPanel getPanel()
  
  - Usage
    * generates the default panel and the title
  ```

- **getSimParams**
  ```java
  public SimParameters getSimParams()
  
  - Usage
  ```
* get the parameters from a parameter class

### getVariableContainer

```java
public VariableContainer getVariableContainer()
```

- **Usage**
  * returns the container for all the variables. this is mainly used for the operations done on the varContainer

### isHidden

```java
public boolean isHidden()
```

- **Usage**
  * test if the hidden flag has been set

### iterateBuffers

```java
public void iterateBuffers()
```

- **Usage**
  * This function does nothing, but blocks with outputs or internal buffers such as SimFunctionFactory should overload it for it’s own special purposes

### reDefine

```java
public void reDefine()
```

- **Usage**
  * transfer upper level adjustments to the lower level. this method is overloaded when it is to be used by a lower function

### setHidden

```java
public void setHidden()
```

- **Usage**
  * keep the block from being loaded to the factory display if it is panelable

### setHidden

```java
public void setHidden(boolean hide)
```

- **Usage**
  * keep the block from being loaded to the factory display if it is panelable

### setSimParams

```java
public void setSimParams(simtk.SimParameters sp)
```

- **Usage**
  * set the parameters from a parameter class
Methods inherited from class simtk.SimBase

- **getDescription**
  public String getDescription( )

- **getReloadBlock**
  public Simulatable getReloadBlock( )
  
  **Usage**
  * set the block that gets called to resimulate when the a variable is changed.

- **getTitle**
  public String getTitle( )
  
  **Usage**
  * get the user defined title for the entire block

- **setDescription**
  public void setDescription( java.lang.String desc )

- **setReloadBlock**
  public void setReloadBlock( simtk.Simulatable reloadBlock )
  
  **Usage**
  * set the block that gets called to resimulate when the a variable is changed.

- **setTitle**
  public void setTitle( java.lang.String title )
  
  **Usage**
  * set the user defined title for the entire block

### 2.2.39 Class SimProfile

Compilation of sim blocks to create the profile

**Declaration**

```java
public class SimProfile
    extends simtk.SimSourceFactory
    implements SimPanelable, ContainsFactory
```
Constructors

- `SimProfile`
  ```java
  public SimProfile()
  ```
  
  - Usage
    * Constructor sets gain to default of unity

Methods

- `connectBlocks`
  ```java
  public void connectBlocks()
  ```
  
  - Usage
    * Make the connections for this block. The setInput of this factory calls this to setup. It turns out to be a rather interesting system the setInput to this factory then in turn sets up this internal system.

Methods inherited from class `simtk.SimSourceFactory`

- `clear`
  ```java
  public void clear()
  ```
  
  - Usage
    * This function is required to reset the buffers and values for the next iterations to set up the factory class.

- `connectBlocks`
  ```java
  abstract void connectBlocks()
  ```
  
  - Usage
    * Make the connections for this block. The setInput of this factory calls this to setup. It turns out to be a rather interesting system the setInput to this factory then in turn sets up this internal system. This is to be implemented.

- `doIteration`
  ```java
  public void doIteration()
  ```
  
  - Usage
    * Do a calculation for the new output

- `getPanel`
  ```java
  public JPanel getPanel()
  ```
• **iterateBuffers**  
  public void iterateBuffers( )  
  - Usage  
    * cycle the buffers

• **toString**  
  public String toString( )

**Methods inherited from class simtk.SimSource**

• **doIteration**  
  abstract void doIteration( )  
  - Usage  
    * this is the method that should be implemented in all SimBlock’s

• **getOutput**  
  public SimBuffer getOutput( )

• **iterateBuffers**  
  public void iterateBuffers( )  
  - Usage  
    * overload the do nothing iterateBuffers and iterate the output buffers

• **setOutput**  
  public void setOutput( simtk.SimBuffer bufref )

**Methods inherited from class simtk.SimBlock**

• **clear**  
  public void clear( )  
  - Usage  
    * this is the general function to clear a block. this is just an empty method but if more specific functionality is needed this should be overridden

• **doIteration**  
  abstract void doIteration( )  
  - Usage  
    * this is the method that should be implemented in all SimBlock’s

• **doOutputBufferIteration**  
  public void doOutputBufferIteration( )  
  - Usage  
    * iterate the output buffer(s), or shift the history of it
• `getPanel`
  ```java
  public JPanel getPanel()
  ```
  - Usage
    * generates the default panel and the title

• `getSimParams`
  ```java
  public SimParameters getSimParams()
  ```
  - Usage
    * get the parameters from a parameter class

• `getVariableContainer`
  ```java
  public VariableContainer getVariableContainer()
  ```
  - Usage
    * returns the container for all the variables. this is mainly used for the operations done on the varContainer

• `isHidden`
  ```java
  public boolean isHidden()
  ```
  - Usage
    * test if the hidden flag has been set

• `iterateBuffers`
  ```java
  public void iterateBuffers()
  ```
  - Usage
    * This function does nothing, but blocks with outputs or internal buffers such as SimFunctionFactory should overload it for it’s own special purposes

• `reDefine`
  ```java
  public void reDefine()
  ```
  - Usage
    * transfer upper level adjustments to the lower level. this method is overloaded when it is to be used by a lower function

• `setHidden`
  ```java
  public void setHidden()
  ```
  - Usage
    * keep the block from being loaded to the factory display if it is panelable

• `setHidden` (boolean `hide`)
  ```java
  public void setHidden( boolean hide )
  ```
  - Usage
    * keep the block from being loaded to the factory display if it is panelable

• `setSimParams`
  ```java
  public void setSimParams( simtk.SimParameters sp )
  ```
  - Usage
    * set the parameters from a parameter class
Methods inherited from class simtk.SimBase

- **getDescription**
  public String getDescription()

- **getReloadBlock**
  public Simulatable getReloadBlock()
  - **Usage**
    * set the block that gets called to resimulate when the a variable is changed.

- **getTitle**
  public String getTitle()
  - **Usage**
    * get the user defined title for the entire block

- **setDescription**
  public void setDescription(java.lang.String desc)

- **setReloadBlock**
  public void setReloadBlock(simtk.Simulatable reloadBlock)
  - **Usage**
    * set the block that gets called to resimulate when the a variable is changed.

- **setTitle**
  public void setTitle(java.lang.String title)
  - **Usage**
    * set the user defined title for the entire block

### 2.2.40 Class SimPtPlot

this is a Sink block that continually stores and plots the next value

**Declaration**

```java
public class SimPtPlot
extends simtk.SimSink
implements SimPanelable
```
Fields

• public Plot plot
  – the is the ptPlot that is plotted, it is public for now so it can be edited from without

• public boolean first
  – flag to signal if it is the first point to be plotted

• private boolean auto_fit_mode
  –

• private int k
  – the index counter

• private int export_count
  – the export counter to vary the plot filename

• private String exportDir
  –

Constructors

• SimPtPlot
  public SimPtPlot( )
  – Usage
    * constructor for now sets the the index counter to zero

• SimPtPlot
  public SimPtPlot( int inputCount )
  – Usage
    * constructor sets the number of inputs

• SimPtPlot
  public SimPtPlot( java.lang.String title )
  – Usage
    * constructor sets the title of the plot
Methods

- **addLegend**
  
  ```
  public void addLegend( int dataset, java.lang.String legend )
  ```

  - **Usage**
  
    * calls the add legend to this ptplot box

- **clear**
  
  ```
  public void clear( )
  ```

  - **Usage**
  
    * clears the plot, repaints and resets the index counter

- **doIteration**
  
  ```
  public void doIteration( )
  ```

  - **Usage**
  
    * this is the action function that does the iteration of sending the input to the standard output after each iteration

- **exportToFile**
  
  ```
  public void exportToFile( )
  ```

  - **Usage**
  
    * Export eps data to the default plot.eps file name

- **exportToFile**
  
  ```
  public void exportToFile( java.lang.String filename )
  ```

  - **Usage**
  
    * Export eps data to a filename

- **getPanel**
  
  ```
  public JPanel getPanel( )
  ```

  - **Usage**
  
    * simply returns the plot. in the future it would be nice if this also set up some button to auto fit and such and some other simple plot functions

- **refreshPlot**
  
  ```
  public void refreshPlot( )
  ```

  - **Usage**
simtk – SimPtPlot

* simply use the same data plot data added and refresh the display

**setExportDirectory**

```java
public void setExportDirectory( java.lang.String dir )
```

- **Usage**
  - * Set the directory that the plots eps plots are saved to

**setPlotDefaults**

```java
public void setPlotDefaults( )
```

- **Usage**
  - * sets up some generic default settings for the plot

**setPlotTitle**

```java
public void setPlotTitle( java.lang.String t )
```

- **Usage**
  - * sets the title of the plot box graph

**setXYLabels**

```java
public void setXYLabels( java.lang.String x, java.lang.String y )
```

- **Usage**
  - * set the x and y labels

**Methods inherited from class simtk.SimSink**

**addInput**

```java
public void addInput( simtk.HasOutput outputBlock )
```

- **Usage**
  - * this AddInput adds from a HasOutput block an input to the vector function and increases the inputIndex count

**addInput**

```java
public void addInput( simtk.SimBuffer bufref )
```

- **Usage**
  - * addInput adds an input to the vector function and increases the inputIndex count

**doIteration**

```java
abstract void doIteration( )
```

- **Usage**
  - * this is the method that should be implemented in all SimBlock’s
**getInput**

```java
public SimBuffer getInput()
```

- **Usage**
  - *get the first input*

```java
public SimBuffer getInput( int index )
```

- **Usage**
  - *get the indexed input, referenced from as 1 is the first*

```java
public int getInputCount()
```

- **Usage**
  - *get the inputCount, or the number of elements on the list*

```java
public void removeInputs()
```

- **Usage**
  - *remove all of the inputs from the block so a new system can be set up*

```java
public void setInput( simtk.HasOutput blockref )
```

- **Usage**
  - *set the input to the output buffer of the passed block*

```java
public void setInput( int index, simtk.SimBuffer bufref )
```

- **Usage**
  - *set the indexed input to the passed buffer this has out of index problems*

```java
public void setInput( simtk.SimBuffer bufref )
```

- **Usage**
  - *set the input to the passed buffer, this is replaced with the newer addInput function which sounds more like what it does*

### Methods inherited from class simtk.SimBlock

- **clear**

```java
public void clear()
```

- **Usage**
  - *this is the general function to clear a block. this is just an empty method but if more specific functionality is needed this should be overridden*
• **doIteration**
  abstract void doIteration( )
  
  – Usage
  * this is the method that should be implemented in all Simblock’s

• **doOutputBufferIteration**
  public void doOutputBufferIteration( )
  
  – Usage
  * iterate the output buffer(s), or shift the history of it

• **getPanel**
  public JPanel getPanel( )
  
  – Usage
  * generates the default panel and the title

• **getSimParams**
  public SimParameters getSimParams( )
  
  – Usage
  * get the parameters from a parameter class

• **getVariableContainer**
  public VariableContainer getVariableContainer( )
  
  – Usage
  * returns the container for all the variables. this is mainly used for the operations done on the varContainer

• **isHidden**
  public boolean isHidden( )
  
  – Usage
  * test if the hidden flag has been set

• **iterateBuffers**
  public void iterateBuffers( )
  
  – Usage
  * This function does nothing, but blocks with outputs or internal buffers such as SimFunctionFactory should overload it for its own special purposes

• **reDefine**
  public void reDefine( )
  
  – Usage
  * transfer upper level adjustments to the lower level. this method is overloaded when it is to be used by a lower function

• ** setHidden**
  public void setHidden( )
- **Usage**
  * keep the block from being loaded to the factory display if it is panelable

- **setHidden**
  ```java
  public void setHidden( boolean hide )
  ```
  - **Usage**
    * keep the block from being loaded to the factory display if it is panelable

- **setSimParams**
  ```java
  public void setSimParams( simtk.SimParameters sp )
  ```
  - **Usage**
    * set the parameters from a parameter class

**Methods inherited from class simtk.SimBase**

- **getDescription**
  ```java
  public String getDescription( )
  ```

- **getReloadBlock**
  ```java
  public Simulatable getReloadBlock( )
  ```
  - **Usage**
    * set the block that gets called to resimulate when the a variable is changed.

- **getTitle**
  ```java
  public String getTitle( )
  ```
  - **Usage**
    * get the user defined title for the entire block

- **setDescription**
  ```java
  public void setDescription( java.lang.String desc )
  ```

- **setReloadBlock**
  ```java
  public void setReloadBlock( simtk.Simulatable reloadBlock )
  ```
  - **Usage**
    * set the block that gets called to resimulate when the a variable is changed.

- **setTitle**
  ```java
  public void setTitle( java.lang.String title )
  ```
  - **Usage**
    * set the user defined title for the entire block

**2.2.41 Class SimSaturation**

Simple function for Saturation
public class SimSaturation
extends simtk.SimFunction
implements SimPanelable

Constructors

• SimSaturation
  public SimSaturation()

  – Usage
  * constructor setup the block

Methods

• doIteration
  public void doIteration()

  – Usage
  * Do a calculation for the function

• toString
  public String toString()

Methods inherited from class simtk.SimFunction

• addInput
  public void addInput(simtk.HasOutput outputBlock)

  – Usage
  * this AddInput adds from a HasOutput block an input to the vector function and increases the inputIndex count

• addInput
  public void addInput(simtk.SimBuffer bufref)

  – Usage
  * AddInput adds an input to the vector function and increases the inputIndex count

• doIteration
  abstract void doIteration()
- **Usage**
  * this is the method that should be implemented in all SimBlock’s

**getInput**

```java
public SimBuffer getInput() {
  // Usage
  * get the first input. there might be a problem here because the old method of the implicit
  in buffer is used
```

**getInput**

```java
public SimBuffer getInput(int index) {
  // Usage
  * get the indexed input, referenced from as 1 is the first.
  * If the reference is out of range it will return the the single input value.
  * The code is not so clean here where it will still print an error message when the double
  input functions are just beginning to set themselves up and before the.???
```

**getInputCount**

```java
public int getInputCount() {
  // Usage
  * get the inputCount, or the number of elements on the list
```

**getOutput**

```java
public SimBuffer getOutput() {
  // Usage
  * Gets the output buffer
```

**iterateBuffers**

```java
public void iterateBuffers() {
  // Usage
  * overload the do nothing iterateBuffers and iterate the output buffers
```

**removeInputs**

```java
public void removeInputs() {
  // Usage
  * remove the all of the inputs from the block so a new system can be set up
```

**setInput**

```java
public void setInput(int index, simtk.SimBuffer bufref) {
  // Usage
  * Set the indexed input to the passed buffer. note: this has out of index problems
```

**setInput**

```java
public void setInput(simtk.SimBuffer bufref) {
  // Usage
```
* Set the input to the passed buffer, this is replaced with the newer addInput function which sounds more like what it does

- `setInput1`
  
  ```java
  public void setInput1( simtk.SimBuffer bufref )
  ```
  
  **Usage**
  
  * This is here to support old systems, it just calls addInput(bufref). so it is not exactly input 2

- `setInput2`
  
  ```java
  public void setInput2( simtk.SimBuffer bufref )
  ```
  
  **Usage**
  
  * This is here to support old systems, it just calls addInput(bufref). so it is not exactly input 2

- `setOutput`
  
  ```java
  public void setOutput( simtk.SimBuffer bufref )
  ```
  
  **Usage**
  
  * sets the output to a passed buffer

**Methods inherited from class simtk.SimBlock**

- `clear`
  
  ```java
  public void clear( )
  ```
  
  **Usage**
  
  * this is the general function to clear a block. this is just an empty method but if more specific functionality is needed this should be overridden

- `doIteration`
  
  ```java
  abstract void doIteration( )
  ```
  
  **Usage**
  
  * this is the method that should be implemented in all Simblock’s

- `doOutputBufferIteration`
  
  ```java
  public void doOutputBufferIteration( )
  ```
  
  **Usage**
  
  * iterate the output buffer(s), or shift the history of it

- `getPanel`
  
  ```java
  public JPanel getPanel( )
  ```
  
  **Usage**
  
  * generates the default panel and the title

- `getSimParams`
  
  ```java
  public SimParameters getSimParams( )
  ```
– Usage
  * get the parameters from a parameter class

• getVariableContainer
  public VariableContainer getVariableContainer()

  – Usage
    * returns the conatainer for all the variables. this is mainly used for the operations done on
    the varContainer

• isHidden
  public boolean isHidden()

  – Usage
    * test if the hidden flag has been set

• iterateBuffers
  public void iterateBuffers()

  – Usage
    * This function does nothing, but blocks with outputs or internal buffers such as
      SimFunctionFactory should overload it for it’s own special purposes

• reDefine
  public void reDefine()

  – Usage
    * transfer upper level adjustments to the lower level. this method is overloaded when it is to
      be used by a lower function

• setHidden
  public void setHidden()

  – Usage
    * keep the block from being loaded to the factory display if it is panelable

• setHidden
  public void setHidden( boolean hide )

  – Usage
    * keep the block from being loaded to the factory display if it is panelable

• setSimParams
  public void setSimParams( simtk.SimParameters sp )

  – Usage
    * set the parameters from a parameter class
Methods inherited from class simtk.SimBase

- `getDescription`
  `public String getDescription();`
- `getReloadBlock`
  `public Simulatable getReloadBlock();`
  - Usage
    * set the block that gets called to resimulate when the a variable is changed.
- `getTitle`
  `public String getTitle();`
  - Usage
    * get the user defined title for the entire block
- `setDescription`
  `public void setDescription(java.lang.String desc);`
- `setReloadBlock`
  `public void setReloadBlock(simtk.Simulatable reloadBlock);`
  - Usage
    * set the block that gets called to resimulate when the a variable is changed.
- `setTitle`
  `public void setTitle(java.lang.String title);`
  - Usage
    * set the user defined title for the entire block

2.2.42 Class SimSink

defines the bare bones to a Sink block

Declaration

```
public abstract class SimSink
extends simtk.SimBlock
implements HasInput
```

Fields

- private int inputIndex
  - initial default input is inputCount = 1
CONSTRUCTORS

- **SimSink**
  
  public SimSink()

METHODS

- **addInput**
  
  public void addInput( simtk.HasOutput outputBlock )
  
  - Usage
    * this AddInput adds from a HasOutput block an input to the vector function and increases the inputIndex count

- **addInput**
  
  public void addInput( simtk.SimBuffer bufref )
  
  - Usage
    * addInput adds an input to the vector function and increases the inputIndex count

- **doIteration**
  
  abstract void doIteration()
  
  - Usage
    * this is the method that should be implemented in all SimBlock’s

- **getInput**
  
  public SimBuffer getInput()
  
  - Usage
    * get the first input

- **getInput**
  
  public SimBuffer getInput( int index )
  
  - Usage
    * get the indexed input, referenced from as 1 is the first

- **getInputCount**
  
  public int getInputCount()
  
  - Usage
    * get the inputCount, or the number of elements on the list
• **removeInputs**
  public void removeInputs( )
  
  – **Usage**
  * remove the all of the inputs from the block so a new system can be set up

• **setInput**
  public void setInput( simtk.HasOutput blockref )
  
  – **Usage**
  * set the input to the output buffer of the passed block

• **setInput**
  public void setInput( int index, simtk.SimBuffer bufref )
  
  – **Usage**
  * set the indexed input to the passed buffer this has out of index problems

• **setInput**
  public void setInput( simtk.SimBuffer bufref )
  
  – **Usage**
  * set the input to the passed buffer, this is replaced with the newer addInput function which sounds more like what it does

**Methods inherited from class simtk.SimBlock**

• **clear**
  public void clear( )
  
  – **Usage**
  * this is the general function to clear a block. this is just an empty method but if more specific functionality is needed this should be overridden

• **doIteration**
  abstract void doIteration( )
  
  – **Usage**
  * this is the method that should be implemented in all Simblock’s

• **doOutputBufferIteration**
  public void doOutputBufferIteration( )
  
  – **Usage**
  * iterate the output buffer(s), or shift the history of it
- **getPanel**
  ```java
  public JPanel getPanel()
  ```
  - **Usage**
    * generates the default panel and the title

- **getSimParams**
  ```java
  public SimParameters getSimParams()
  ```
  - **Usage**
    * get the parameters from a parameter class

- **getVariableContainer**
  ```java
  public VariableContainer getVariableContainer()
  ```
  - **Usage**
    * returns the container for all the variables. this is mainly used for the operations done on
      the varContainer

- **isHidden**
  ```java
  public boolean isHidden()
  ```
  - **Usage**
    * test if the hidden flag has been set

- **iterateBuffers**
  ```java
  public void iterateBuffers()
  ```
  - **Usage**
    * This function does nothing, but blocks with outputs or internal buffers such as
      SimFunctionFactory should overload it for its own special purposes

- **reDefine**
  ```java
  public void reDefine()
  ```
  - **Usage**
    * transfer upper level adjustments to the lower level. this method is overloaded when it is to
      be used by a lower function

- **setHidden**
  ```java
  public void setHidden()
  ```
  - **Usage**
    * keep the block from being loaded to the factory display if it is panelable

- **setHidden**
  ```java
  public void setHidden( boolean hide )
  ```
  - **Usage**
    * keep the block from being loaded to the factory display if it is panelable

- **setSimParams**
  ```java
  public void setSimParams( simtk.SimParameters sp )
  ```
  - **Usage**
    * set the parameters from a parameter class
Methods inherited from class simtk.SimBase

- `getDescription`
  public String `getDescription`()

- `getReloadBlock`
  public Simulatable `getReloadBlock`()
  
  - Usage
    * set the block that gets called to resimulate when the a variable is changed.

- `getTitle`
  public String `getTitle`()
  
  - Usage
    * get the user defined title for the entire block

- `setDescription`
  public void `setDescription`(java.lang.String `desc`)

- `setReloadBlock`
  public void `setReloadBlock` (simtk.Simulatable `reloadBlock`)
  
  - Usage
    * set the block that gets called to resimulate when the a variable is changed.

- `setTitle`
  public void `setTitle`(java.lang.String `title`)
  
  - Usage
    * set the user defined title for the entire block

### 2.2.43 Class SimSource

**Declaration**

```
public abstract class SimSource
extends simtk.SimBlock
implements HasOutput
```

**Fields**

- public SimBuffer `out`
**Constructors**

- `SimSource`
  ```java
class SimSource {
public SimSource() {
}
}
```

**Methods**

- `doIteration`
  ```java
  abstract void doIteration()
  ```
  - **Usage**
    * this is the method that should be implemented in all SimBlock's

- `getOutput`
  ```java
  public SimBuffer getOutput()
  ```

- `iterateBuffers`
  ```java
  public void iterateBuffers()
  ```
  - **Usage**
    * overload the do nothing iterateBuffers and iterate the output buffers

- `setOutput`
  ```java
  public void setOutput(simtk.SimBuffer bufref)
  ```

**Methods inherited from class simtk.SimBlock**

- `clear`
  ```java
  public void clear()
  ```
  - **Usage**
    * this is the general function to clear a block. this is just an empty method but if more specific functionality is needed this should be overridden

- `doIteration`
  ```java
  abstract void doIteration()
  ```
  - **Usage**
    * this is the method that should be implemented in all Simblock's

- `doOutputBufferIteration`
  ```java
  public void doOutputBufferIteration()
  ```
  - **Usage**
    * iterate the output buffer(s), or shift the history of it
- **getPanel**
  public JPanel getPanel()  
  - **Usage**
    * generates the default panel and the title

- **getSimParams**
  public SimParameters getSimParams()  
  - **Usage**
    * get the parameters from a parameter class

- **getVariableContainer**
  public VariableContainer getVariableContainer()  
  - **Usage**
    * returns the container for all the variables. this is mainly used for the operations done on the varContainer

- **isHidden**
  public boolean isHidden()  
  - **Usage**
    * test if the hidden flag has been set

- **iterateBuffers**
  public void iterateBuffers()  
  - **Usage**
    * This function does nothing, but blocks with outputs or internal buffers such as SimFunctionFactory should overload it for its own special purposes

- **reDefine**
  public void reDefine()  
  - **Usage**
    * transfer upper level adjustments to the lower level. this method is overloaded when it is to be used by a lower function

- **setHidden**
  public void setHidden()  
  - **Usage**
    * keep the block from being loaded to the factory display if it is panelable

- **setHidden**
  public void setHidden( boolean hide )  
  - **Usage**
    * keep the block from being loaded to the factory display if it is panelable

- **setSimParams**
  public void setSimParams( simtk.SimParameters sp )  
  - **Usage**
    * set the parameters from a parameter class
Methods inherited from class simtk.SimBase

- `getDescription`
  ```java`
  public String getDescription()
  ```
  - Usage
    * set the block that gets called to resimulate when the a variable is changed.

- `getReloadBlock`
  ```java`
  public Simulatable getReloadBlock()
  ```
  - Usage
    * set the block that gets called to resimulate when the a variable is changed.

- `getTitle`
  ```java`
  public String getTitle()
  ```
  - Usage
    * get the user defined title for the entire block

- `setDescription`
  ```java`
  public void setDescription(java.lang.String desc)
  ```

- `setReloadBlock`
  ```java`
  public void setReloadBlock(simtk.Simulatable reloadBlock)
  ```
  - Usage
    * set the block that gets called to resimulate when the a variable is changed.

- `setTitle`
  ```java`
  public void setTitle(java.lang.String title)
  ```
  - Usage
    * set the user defined title for the entire block

2.2.44 Class SimSourceFactory

Compilation of sim blocks to create the complex blocks such as NPID from simple building blocks

Declaration

```java`
public abstract class SimSourceFactory
    extends simtk.SimSource
    implements SimPanelable, ContainsFactory
```
CONSTRUCTORS

- SimSourceFactory
  public SimSourceFactory()
  - Usage
    * constructor sets gain to default of unity

METHODS

- clear
  public void clear()
  - Usage
    * this function is required to reset the buffers and values for the next iterations to
set up the factory class

- connectBlocks
  abstract void connectBlocks()
  - Usage
    * Make the connections for this block. The setInput of this factory calls this to
setup. It turns out to be a rather interesting system the setInput to this factory
then in turn set’s up this internal system.
this is to be implemented.

- doIteration
  public void doIteration()
  - Usage
    * do a calculation for the new output

- getPanel
  public JPanel getPanel()

- iterateBuffers
  public void iterateBuffers()
  - Usage
    * cycle the buffers

- toString
  public String toString()
Methods inherited from class simtk.SimSource

- `doIteration`
  ```
  abstract void doIteration( )
  ```
  - Usage
    * this is the method that should be implemented in all SimBlock’s

- `getOutput`
  ```
  public SimBuffer getOutput( )
  ```

- `iterateBuffers`
  ```
  public void iterateBuffers( )
  ```
  - Usage
    * overload the do nothing iterateBuffers and iterate the output buffers

- `setOutput`
  ```
  public void setOutput( simtk.SimBuffer buref )
  ```

Methods inherited from class simtk.SimBlock

- `clear`
  ```
  public void clear( )
  ```
  - Usage
    * this is the general function to clear a block. this is just an empty method but if more specific functionality is needed this should be overridden

- `doIteration`
  ```
  abstract void doIteration( )
  ```
  - Usage
    * this is the method that should be implemented in all Simblock’s

- `doOutputBufferIteration`
  ```
  public void doOutputBufferIteration( )
  ```
  - Usage
    * iterate the output buffer(s), or shift the history of it

- `getPanel`
  ```
  public JPanel getPanel( )
  ```
  - Usage
    * generates the default panel and the title

- `getSimParams`
  ```
  public SimParameters getSimParams( )
  ```
  - Usage
- get the parameters from a parameter class

- **getVariableContainer**
  ```java
  public VariableContainer getVariableContainer()
  ```
  - **Usage**
  * returns the container for all the variables. this is mainly used for the operations done on
  the varContainer

- **isHidden**
  ```java
  public boolean isHidden()
  ```
  - **Usage**
  * test if the hidden flag has been set

- **iterateBuffers**
  ```java
  public void iterateBuffers()
  ```
  - **Usage**
  * This function does nothing, but blocks with outputs or internal buffers such as
  SimFunctionFactory should overload it for its own special purposes

- **reDefine**
  ```java
  public void reDefine()
  ```
  - **Usage**
  * transfer upper level adjustments to the lower level. this method is overloaded when it is to
  be used by a lower function

- **setHidden**
  ```java
  public void setHidden()
  ```
  - **Usage**
  * keep the block from being loaded to the factory display if it is panelable

- **setHidden**
  ```java
  public void setHidden( boolean hide)
  ```
  - **Usage**
  * keep the block from being loaded to the factory display if it is panelable

- **setSimParams**
  ```java
  public void setSimParams( simtk.SimParameters sp)
  ```
  - **Usage**
  * set the parameters from a parameter class
Methods inherited from class simtk.SimBase

- `getDescription`
  ```java
  public String getDescription()
  ```

- `getReloadBlock`
  ```java
  public Simulatable getReloadBlock()
  ```
  - **Usage**
    - * set the block that gets called to resimulate when the a variable is changed.

- `getTitle`
  ```java
  public String getTitle()
  ```
  - **Usage**
    - * get the user defined title for the entire block

- `setDescription`
  ```java
  public void setDescription(java.lang.String desc)
  ```

- `setReloadBlock`
  ```java
  public void setReloadBlock(simtk.Simulatable reloadBlock)
  ```
  - **Usage**
    - * set the block that gets called to resimulate when the a variable is changed.

- `setTitle`
  ```java
  public void setTitle(java.lang.String title)
  ```
  - **Usage**
    - * set the user defined title for the entire block

### 2.2.45 Class SimSourceSelector

This is one of the most simple source blocks to select different usable sources. Not quite implemented yet, this has lots of problems, I need to think about a better method to provide this functionality of switching source types.

**Declaration**

```java
public class SimSourceSelector
extends simtk.SimSource
implements SimPanelable
```
Fields

- public SimComboBoxString cb
  - the source to select. To programatically select a function for example, selecting the differentiator use:
    ```java
    SimFunctionSelector ss = new SimFunctionSelector;
    ss.cb.setValue(ss.DIFF)
    ```

- public static int STEP
  - used by the fs.cb.setValue to select a step block type

- public static int DIST
  - used by the fs.cb.setValue to select a disturbance block type

- public static int NOISE
  - used by the fs.cb.setValue to select a noise block type

- public static int TRAP
  - used by the fs.cb.setValue to select a Trapezoid block type

- public static int DTRI
  - used by the fs.cb.setValue to select a Double triangle block type

- public static int PROF
  - used by the fs.cb.setValue to select a profile block type

- public static int CONT_PRE
  - used by the fs.cb.setValue to select a continuous pre-filter block type

- public static int TRAP_PRE
  - used by the fs.cb.setValue to select a trapezoidal pre-filter block type

- public static int TIME
  - used by the fs.cb.setValue to select a Time Index block type

- public static int POLY_PROF
  - used by the fs.cb.setValue to select a Polynomial profile block type

- public static int NOTHING
  - used by the fs.cb.setValue to select a Nothing, a zero block type
Constructors

- **SimSourceSelector**
  public SimSourceSelector()
  - Usage
    * constructor sets step to default of unity

Methods

- **checkSFChange**
  public boolean checkSFChange()
  - Usage
    * Check if the selectable source was changed from the last time, if not don’t do anything

- **clear**
  public void clear()
  - Usage
    * called before doSimulation, sets up the internal block

- **doIteration**
  public void doIteration()
  - Usage
    * this method overrides the doIteration of the sf, sim source.

- **getPanel**
  public JPanel getPanel()
  - Usage
    * gets the internal sf get panel and the combo selection box

- **iterateBuffers**
  public void iterateBuffers()
  - Usage
    * overload the do nothing iterateBuffers and iterate the output buffers

- **reDefine**
  public void reDefine()
- **Usage**
  * this is a method that might be good to have defined in the sim Block as abstract
  this is made to recalculate with new parameters, step sizes parameters

- **refreshPanel**
  ```java
  public void refreshPanel()
  ```
  - **Usage**
    * refreshes the internal get panel and the combo selection box

- **setupInternalSimBlock**
  ```java
  public void setupInternalSimBlock()
  ```
  - **Usage**
    * This copies the inherited source selector outputs to the selected source for usage.
      the steps here are rather important, this source is called by the clear because at
      the beginning of every simulation the clear source is called.
      A rather important easy overlooked step in here is to set the reload blocks of each
      of the internal Variables

**Methods inherited from class simtk.SimSource**

- **doIteration**
  ```java
  abstract void doIteration()
  ```
  - **Usage**
    * this is the method that should be implemented in all SimBlock’s

- **getOutput**
  ```java
  public SimBuffer getOutput()
  ```

- **iterateBuffers**
  ```java
  public void iterateBuffers()
  ```
  - **Usage**
    * overload the do nothing iterateBuffers and iterate the output buffers

- **setOutput**
  ```java
  public void setOutput( simtk.SimBuffer bufref )
  ```

**Methods inherited from class simtk.SimBlock**

- **clear**
  ```java
  public void clear()
  ```
  - **Usage**
* this is the general function to clear a block. this is just an empty method but if more specific functionality is needed this should be overridden

- **doIteration**
  abstract void doIteration( )
  - Usage
    * this is the method that should be implemented in all Simblock’s

- **doOutputBufferIteration**
  public void doOutputBufferIteration( )
  - Usage
    * iterate the output buffer(s), or shift the history of it

- **getPanel**
  public JPanel getPanel( )
  - Usage
    * generates the default panel and the title

- **getSimParams**
  public SimParameters getSimParams( )
  - Usage
    * get the parameters from a parameter class

- **getVariableContainer**
  public VariableContainer getVariableContainer( )
  - Usage
    * returns the container for all the variables. this is mainly used for the operations done on the varContainer

- **isHidden**
  public boolean isHidden( )
  - Usage
    * test if the hidden flag has been set

- **iterateBuffers**
  public void iterateBuffers( )
  - Usage
    * This function does nothing, but blocks with outputs or internal buffers such as SimFunctionFactory should overload it for it’s own special purposes

- **reDefine**
  public void reDefine( )
  - Usage
    * transfer upper level adjustments to the lower level. this method is overloaded when it is to be used by a lower function
• `setHidden`
  public void setHidden()
  
  — Usage
  * keep the block from being loaded to the factory display if it is panelable

• `setHidden`
  public void setHidden( boolean hide )
  
  — Usage
  * keep the block from being loaded to the factory display if it is panelable

• `setSimParams`
  public void setSimParams( simtk.SimParameters sp )
  
  — Usage
  * set the parameters from a parameter class

Methods inherited from class `simtk.SimBase`

• `getDescription`
  public String getDescription()

• `getReloadBlock`
  public Simulatable getReloadBlock()
  
  — Usage
  * set the block that gets called to resimulate when the a variable is changed.

• `getTitle`
  public String getTitle()
  
  — Usage
  * get the user defined title for the entire block

• `setDescription`
  public void setDescription( java.lang.String desc )

• `setReloadBlock`
  public void setReloadBlock( simtk.Simulatable reloadBlock )
  
  — Usage
  * set the block that gets called to resimulate when the a variable is changed.

• `setTitle`
  public void setTitle( java.lang.String title )
  
  — Usage
  * set the user defined title for the entire block
2.2.46 Class SimStat

This is one of the most simple function blocks. For now it just does the average of the n samples.

Declaration

```java
public class SimStat
    extends simtk.SimFunction
    implements SimPanelable
```

Fields

- private Vector memory_vec
  - holds the memory of the values
- private SimInteger n
- private SimNumber t
- private SimStaticNumber xbar
- private SimStaticNumber xmin
- private SimStaticNumber xmax
- private SimStaticNumber xsum
- private SimStaticNumber ssx
- private SimStaticNumber s2
• private SimStaticNumber s

• private SimComboBoxNumber outputSel

Constructors

• SimStat
  public SimStat( )
  
  Usage
  * constructor sets gain to default of unity

Methods

• doIteration
  public void doIteration( )
  
  Usage
  * do a calculation for the new output

• toString
  public String toString( )

Methods inherited from class simtk.SimFunction

• addInput
  public void addInput( simtk.HasOutput outputBlock )
  
  Usage
  * this AddInput adds from a HasOutput block an input to the vector function and increases the inputIndex count

• addInput
  public void addInput( simtk.SimBuffer bufref )
  
  Usage
  * AddInput adds an input to the vector function and increases the inputIndex count

• doIteration
  abstract void doIteration( )
  
  Usage
* this is the method that should be implemented in all SimBlock’s

- **getInput**
  ```java
  public SimBuffer getInput()
  ```
  - **Usage**
    * get the first input. there might be a problem here because the old method of the implicit in buffer is used

- **getInput**
  ```java
  public SimBuffer getInput( int index )
  ```
  - **Usage**
    * get the indexed input, referenced from as 1 is the first.
      If the reference is out of range it will return the the single input value.
      The code is not so clean here where it will still print an erro message when the double input functions are just beginning to set themselves up and before the.

- **getInputCount**
  ```java
  public int getInputCount()
  ```
  - **Usage**
    * get the inputCount, or the number of elements on the list

- **getOutput**
  ```java
  public SimBuffer getOutput()
  ```
  - **Usage**
    * Gets the output buffer

- **iterateBuffers**
  ```java
  public void iterateBuffers()
  ```
  - **Usage**
    * overload the do nothing iterateBuffers and iterate the output buffers

- **removeInputs**
  ```java
  public void removeInputs()
  ```
  - **Usage**
    * remove the all of the inputs from the block so a new system can be set up

- **setInput**
  ```java
  public void setInput( int index, simtk.SimBuffer bufref )
  ```
  - **Usage**
    * Set the indexed input to the passed buffer. note: this has out of index problems

- **setInput**
  ```java
  public void setInput( simtk.SimBuffer bufref )
  ```
  - **Usage**
Set the input to the passed buffer, this is replaced with the newer addInput function which sounds more like what it does

* setInput1
public void setInput1( simtk.SimBuffer bufref )
  
  - Usage
  * This is here to support old systems, it just calls addInput(bufref). so it is not exactly input 2

* setInput2
public void setInput2( simtk.SimBuffer bufref )
  
  - Usage
  * This is here to support old systems, it just calls addInput(bufref). so it is not exactly input 2

* setOutput
public void setOutput( simtk.SimBuffer bufref )
  
  - Usage
  * sets the output to a passed buffer

Methods inherited from class simtk.SimBlock

* clear
public void clear( )
  
  - Usage
  * this is the general function to clear a block. this is just an empty method but if more specific functionality is needed this should be overridden

* doIteration
abstract void doIteration( )
  
  - Usage
  * this is the method that should be implemented in all Simblock’s

* doOutputBufferIteration
public void doOutputBufferIteration( )
  
  - Usage
  * iterate the output buffer(s), or shift the history of it

* getPanel
public JPanel getPanel( )
  
  - Usage
  * generates the default panel and the title

* getSimParams
public SimParameters getSimParams( )
- **Usage**
  * get the parameters from a parameter class

- **getVariableContainer**
  ```java
  public VariableContainer getVariableContainer()
  ```
  - **Usage**
    * returns the container for all the variables. this is mainly used for the operations done on the varContainer

- **isHidden**
  ```java
  public boolean isHidden()
  ```
  - **Usage**
    * test if the hidden flag has been set

- **iterateBuffers**
  ```java
  public void iterateBuffers()
  ```
  - **Usage**
    * This function does nothing, but blocks with outputs or internal buffers such as SimFunctionFactory should overload it for its own special purposes

- **reDefine**
  ```java
  public void reDefine()
  ```
  - **Usage**
    * transfer upper level adjustments to the lower level. this method is overloaded when it is to be used by a lower function

- **setHidden**
  ```java
  public void setHidden()
  ```
  - **Usage**
    * keep the block from being loaded to the factory display if it is panelable

- **setHidden**
  ```java
  public void setHidden(boolean hide)
  ```
  - **Usage**
    * keep the block from being loaded to the factory display if it is panelable

- **setSimParams**
  ```java
  public void setSimParams(simtk.SimParameters sp)
  ```
  - **Usage**
    * set the parameters from a parameter class
Methods inherited from class simtk.SimBase

- `getDescription`
  ```java
  public String getDescription()
  ```
- `getReloadBlock`
  ```java
  public Simulatable getReloadBlock()
  ```
  - Usage
    * set the block that gets called to resimulate when the a variable is changed.
- `getTitle`
  ```java
  public String getTitle()
  ```
  - Usage
    * get the user defined title for the entire block
- `setDescription`
  ```java
  public void setDescription(java.lang.String desc)
  ```
- `setReloadBlock`
  ```java
  public void setReloadBlock(simtk.Simulatable reloadBlock)
  ```
  - Usage
    * set the block that gets called to resimulate when the a variable is changed.
- `setTitle`
  ```java
  public void setTitle(java.lang.String title)
  ```
  - Usage
    * set the user defined title for the entire block

2.2.47 Class SimStdout

This is a Sink block that simply send the current output to the standard output

Declaration

```java
public class SimStdout extends simtk.SimSink
```

Fields

- private int k
  - this is the private index counter to count how many iterations have been done
CONSTRUCTORS

- **SimStdout**
  
  `public SimStdout()`

METHODS

- **doIteration**
  
  `public void doIteration()`

  - **Usage**
    
    * this is the action function that does the iteration of sending the input to the standard output after each iteration

METHODS INHERITED FROM CLASS `simtk.SimSink`

- **addInput**
  
  `public void addInput( simtk.HasOutput outputBlock )`

  - **Usage**
    
    * this AddInput adds from a HasOutput block an input to the vector function and increases the inputIndex count

- **addInput**
  
  `public void addInput( simtk.SimBuffer bufref )`

  - **Usage**
    
    * addInput adds an input to the vector function and increases the inputIndex count

- **doIteration**
  
  `abstract void doIteration()`

  - **Usage**
    
    * this is the method that should be implemented in all SimBlock’s

- **getInput**
  
  `public SimBuffer getInput()`

  - **Usage**
    
    * get the first input

- **getInput**
  
  `public SimBuffer getInput( int index )`

  - **Usage**
    
    * get the indexed input, referenced from as 1 is the first
• `getInputCount`

  public int `getInputCount()`

  – **Usage**
  
  * get the inputCount, or the number of elements on the list

• `removeInputs`

  public void `removeInputs()`

  – **Usage**
  
  * remove the all of the inputs from the block so a new system can be set up

• `setInput`

  public void `setInput( simtk.HasOutput blockref )`

  – **Usage**
  
  * set the input to the output buffer of the passed block

• `setInput`

  public void `setInput( int index, simtk.SimBuffer bufref )`

  – **Usage**
  
  * set the indexed input to the passed buffer this has out of index problems

• `setInput`

  public void `setInput( simtk.SimBuffer bufref )`

  – **Usage**
  
  * set the input to the passed buffer, this is replaced with the newer addInput function which sounds more like what it does

**Methods inherited from class simtk.SimBlock**

• `clear`

  public void `clear()`

  – **Usage**
  
  * this is the general function to clear a block. this is just an empty method but if more specific functionality is needed this should be overridden

• `doIteration`

  abstract void `doIteration()`

  – **Usage**
  
  * this is the method that should be implemented in all Simblock’s

• `doOutputBufferIteration`

  public void `doOutputBufferIteration()`

  – **Usage**
  
  * iterate the output buffer(s), or shift the history of it
• getPanel
  public JPanel getPanel()
  – Usage
  * generates the default panel and the title

• getSimParams
  public SimParameters getSimParams()
  – Usage
  * get the parameters from a parameter class

• getVariableContainer
  public VariableContainer getVariableContainer()
  – Usage
  * returns the container for all the variables. this is mainly used for the operations done on the varContainer

• isHidden
  public boolean isHidden()
  – Usage
  * test if the hidden flag has been set

• iterateBuffers
  public void iterateBuffers()
  – Usage
  * This function does nothing, but blocks with outputs or internal buffers such as SimFunctionFactory should overload it for its own special purposes

• reDefine
  public void reDefine()
  – Usage
  * transfer upper level adjustments to the lower level. this method is overloaded when it is to be used by a lower function

• setHidden
  public void setHidden()
  – Usage
  * keep the block from being loaded to the factory display if it is panelable

• setHidden
  public void setHidden(boolean hide)
  – Usage
  * keep the block from being loaded to the factory display if it is panelable

• setSimParams
  public void setSimParams(simtk.SimParameters sp)
  – Usage
  * set the parameters from a parameter class
### Methods inherited from class `simtk.SimBase`

- **getDescription**
  ```java
  public String getDescription()
  ```
  - Usage
    * set the block that gets called to resimulate when the a variable is changed.

- **getReloadBlock**
  ```java
  public Simulatable getReloadBlock()
  ```
  - Usage
    * set the block that gets called to resimulate when the a variable is changed.

- **getTitle**
  ```java
  public String getTitle()
  ```
  - Usage
    * get the user defined title for the entire block

- **setDescription**
  ```java
  public void setDescription(java.lang.String desc)
  ```

- **setReloadBlock**
  ```java
  public void setReloadBlock(simtk.Simulatable reloadBlock)
  ```
  - Usage
    * set the block that gets called to resimulate when the a variable is changed.

- **setTitle**
  ```java
  public void setTitle(java.lang.String title)
  ```
  - Usage
    * set the user defined title for the entire block

### 2.2.48 Class `SimStep`

Step function

**Declaration**

```java
public class SimStep
  extends simtk.SimSource
  implements SimPanelable
```
**Fields**

- private int k
- private SimNumber gain
- private SimNumber delay

**Constructors**

- `SimStep`
  - public `SimStep()`
    - Usage
      - * defaults are set to delay = 0 and gain = 1

**Methods**

- `clear`
  - public void `clear()`

- `doIteration`
  - public void `doIteration()`
    - Usage
      - * set the output to the gain value after the delay time

- `getDelay`
  - public double `getDelay()`

- `getGain`
  - public double `getGain()`

- `setDelay`
  - public void `setDelay(double delay)`

- `setGain`
  - public void `setGain(double gain)`
Methods inherited from class simtk.SimSource

- **doIteration**
  abstract void **doIteration**()
  - **Usage**
    * this is the method that should be implemented in all SimBlock’s

- **getOutput**
  public SimBuffer **getOutput**()

- **iterateBuffers**
  public void **iterateBuffers**()
  - **Usage**
    * overload the do nothing iterateBuffers and iterate the output buffers

- **setOutput**
  public void **setOutput**( simtk.SimBuffer **bufref**)

Methods inherited from class simtk.SimBlock

- **clear**
  public void **clear**()
  - **Usage**
    * this is the general function to clear a block. this is just an empty method but if more specific functionality is needed this should be overridden

- **doIteration**
  abstract void **doIteration**()
  - **Usage**
    * this is the method that should be implemented in all Simblock’s

- **doOutputBufferIteration**
  public void **doOutputBufferIteration**()
  - **Usage**
    * iterate the output buffer(s), or shift the history of it

- **getPanel**
  public JPanel **getPanel**()
  - **Usage**
    * generates the default panel and the title

- **getSimParams**
  public SimParameters **getSimParams**()
  - **Usage**
* get the parameters from a parameter class

- **getVariableContainer**
  
  public VariableContainer getVariableContainer()

  - **Usage**
    * returns the container for all the variables. this is mainly used for the operations done on the varContainer

- **isHidden**
  
  public boolean isHidden()

  - **Usage**
    * test if the hidden flag has been set

- **iterateBuffers**
  
  public void iterateBuffers()

  - **Usage**
    * This function does nothing, but blocks with outputs or internal buffers such as SimFunctionFactory should overload it for its own special purposes

- **reDefine**
  
  public void reDefine()

  - **Usage**
    * transfer upper level adjustments to the lower level. this method is overloaded when it is to be used by a lower function

- **setHidden**
  
  public void setHidden()

  - **Usage**
    * keep the block from being loaded to the factory display if it is panelable

- **setHidden**
  
  public void setHidden(boolean hide)

  - **Usage**
    * keep the block from being loaded to the factory display if it is panelable

- **setSimParams**
  
  public void setSimParams(simtk.SimParameters sp)

  - **Usage**
    * set the parameters from a parameter class
Methods inherited from class simtk.SimBase

- **getDescription**
  ```java
  public String getDescription()
  ```
  - Usage
    * set the block that gets called to resimulate when the a variable is changed.

- **getReloadBlock**
  ```java
  public Simulatable getReloadBlock()
  ```
  - Usage
    * set the block that gets called to resimulate when the a variable is changed.

- **getTitle**
  ```java
  public String getTitle()
  ```
  - Usage
    * get the user defined title for the entire block

- **setDescription**
  ```java
  public void setDescription(java.lang.String desc)
  ```

- **setReloadBlock**
  ```java
  public void setReloadBlock(simtk.Simulatable reloadBlock)
  ```
  - Usage
    * set the block that gets called to resimulate when the a variable is changed.

- **setTitle**
  ```java
  public void setTitle(java.lang.String title)
  ```
  - Usage
    * set the user defined title for the entire block

### 2.2.49 Class SimStf

This is the block that simulates s domain transfer functions!. It actually is an extension of the Ztf, it first converts the s transfer function into a ztransfer function then the ztransfer function is setup in the SimZtf to create the difference equations to plot a number of general types of functions can be derived from this block but the general types should use the directly solved blocks from the ztransfer functions method to set the numerator to the array value Example: setNum([1,2,3]); would set the numerator to \( s^2 + 2s + 3 \) an error message is displayed if the array is larger than the s order **Note:**

1. it only works with 3rd polynomials

**Todo:**

1. make it work with other than 3rd orderd functions (this probably has to do with the fact that the entire system is sort of fixed to a zbuffer size of 4
public class SimStf
extends simtk.SimZtf
implements SimPanelable

Constructors

- **SimStf**
  public SimStf( )
  - **Usage**
    * sets the default to num_order = 3 and den_order = 3 (with a size of 4)

- **SimStf**
  public SimStf( int snum_order, int sden_order )
  - **Usage**
    * constructor for the stf, I am still a little bit unsure how to implement the zorder of the system.

Methods

- **displayNotProperErr**
  private void displayNotProperErr( )
  - **Usage**
    * improper functions error

- **generateZPoly**
  public double generateZPoly( double [] s )
  - **Usage**
    * generate the polynomial as an array from the s polynomial the order is extracted by the size of s this is the magic function for the conversion of continuous to discrete see the derivations done on 4/26/02 with mathematica and the papers solution I need to type up these derivations

- **makeNewStf**
  public void makeNewStf( int snum_order, int sden_order )
- **Usage**  
  * creates the num and den checking if improper else make default

- **reDefine**  
  public void reDefine( )

  - **Usage**  
    * this is a method that might be good to have defined in the sim Block as abstract  
      this is made to recalculate with new parameters, step sizes parameters

- **setSDen**  
  public void setSDen( double [] a )

  - **Usage**  
    * simply sets the s denominator to an array value

- **setSNum**  
  public void setSNum( double [] a )

  - **Usage**  
    * simply sets the s numerator to an array value

- **toString**  
  public String toString( )

  - **Usage**  
    * toString in the form of: num_order = x den_order = y num = [x] den = [y]

- **zeroStf**  
  public void zeroStf( )

  - **Usage**  
    * this zeroSTF method is for convienience in setting up z transfer functions it makes  
      sure that the function is reset to zero from the initial conditions

**Methods inherited from class simtk.SimZtf**

- **doIteration**  
  public void doIteration( )

  - **Usage**  
    * this method overrides the doIteration of the diffeq. this way it first redefines the variables  
      and then iterates with the inherited structure
• **doIterationDirect**
  
  public void doIterationDirect()

  - Usage
    * this is the really cool simple method that converts ztf to diffEQ. The function does the operation in the following order:
      1. parses the numerator and denominator
      2. creates the difference equation
      3. does one step operation.

• **reDefine**
  
  public void reDefine()

  - Usage
    * this is a method that might be good to have defined in the sim Block as abstract this is made to recalculate with new parameters, step sizes parameters

• **setDen**
  
  public void setDen( double [] n )

  - Usage
    * method to set the denominator to the array value Example: setDen([1,2,3]); would set the denominator to \( z^2 + 2z + 3 \) an error message is displayed if the array is larger than the \( z \) order

• **setNum**
  
  public void setNum( double [] n )

  - Usage
    * method to set the numerator to the array value Example: setNum([1,2,3]); would set the numerator to \( z^2 + 2z + 3 \) an error message is displayed if the array is larger than the \( z \) order

• **toString**
  
  public String toString()

  - Usage
    * toString in the form of: \( \text{zorder} = x \text{ num} = [x] \text{ den} = [x] \)

• **zeroNumAndDen**
  
  public void zeroNumAndDen()

  - Usage
    * this method is for convenience in setting up z transfer functions it makes sure that the function is reset to zero from the initial conditions
Methods inherited from class simtk.SimDiffEq

- `clear`
  ```java
  public void clear()
  ```
  - Usage
    * clears the text field, sets up headers and resets the index counter

- `doIteration`
  ```java
  public void doIteration()
  ```

- `getInCoef`
  ```java
  public SimDoubleArray getInCoef()
  ```
  - Usage
    * return the input coefficients

- `getOutCoef`
  ```java
  public SimDoubleArray getOutCoef()
  ```
  - Usage
    * return the output coefficients

- `setZeros`
  ```java
  public void setZeros()
  ```
  - Usage
    * zero all the coefficients for easy settings

- `toDiffEqString`
  ```java
  public String toDiffEqString()
  ```
  - Usage
    * output the string format of the difference equation. special function to iterate through all of the parameters and create the difference equation. This function is also used for the textBox for the dynamic pretty print viewing of the diff eq

- `toString`
  ```java
  public String toString()
  ```
  - Usage
    * general to String equation for the diff eq. this uses the toDiffEqString() function for the special formating so that one doesn’t overloaded
Methods inherited from class simtk.SimFunction

• `addInput`
  public void addInput( simtk.HasOutput outputBlock )
  ─ Usage
  * this AddInput adds from a HasOutput block an input to the vector function and increases
    the inputIndex count

• `addInput`
  public void addInput( simtk.SimBuffer bufref )
  ─ Usage
  * AddInput adds an input to the vector function and increases the inputIndex count

• `doIteration`
  abstract void doIteration()  
  ─ Usage
  * this is the method that should be implemented in all SimBlock’s

• `getInput`
  public SimBuffer getInput()  
  ─ Usage
  * get the first input. there might be a problem here because the old method of the implicit
    in buffer is used

• `getInput`
  public SimBuffer getInput( int index )
  ─ Usage
  * get the indexed input, referenced from as 1 is the first.
    If the reference is out of range it will return the the single input value.
    The code is not so clean here where it will still print an erro message when the double
    input functions are just beginning to set themselves up and before the.???

• `getInputCount`
  public int getInputCount()  
  ─ Usage
  * get the inputCount, or the number of elements on the list

• `getOutput`
  public SimBuffer getOutput()  
  ─ Usage
  * Gets the output buffer

• `iterateBuffers`
  public void iterateBuffers()
- **Usage**
  * overload the do nothing iterateBuffers and iterate the output buffers

- **removeInputs**
  ```java
  public void removeInputs()
  ```
  - **Usage**
    * remove the all of the inputs from the block so a new system can be set up

- **setInput**
  ```java
  public void setInput( int index, simtk.SimBuffer bufref )
  ```
  - **Usage**
    * Set the indexed input to the passed buffer. note: this has out of index problems

- **setInput**
  ```java
  public void setInput( simtk.SimBuffer bufref )
  ```
  - **Usage**
    * Set the input to the passed buffer, this is replaced with the newer addInput function which sounds more like what it does

- **setInput1**
  ```java
  public void setInput1( simtk.SimBuffer bufref )
  ```
  - **Usage**
    * This is here to support old systems, it just calls addInput(bufref). so it is not exactly input 2

- **setInput2**
  ```java
  public void setInput2( simtk.SimBuffer bufref )
  ```
  - **Usage**
    * This is here to support old systems, it just calls addInput(bufref). so it is not exactly input 2

- **setOutput**
  ```java
  public void setOutput( simtk.SimBuffer bufref )
  ```
  - **Usage**
    * sets the output to a passed buffer

**Methods inherited from class simtk.SimBlock**

- **clear**
  ```java
  public void clear()
  ```
  - **Usage**
    * this is the general function to clear a block. this is just an empty method but if more specific functionality is needed this should be overridden
• **doIteration**
  abstract void doIteration( )
  
  -- Usage
  * this is the method that should be implemented in all Simblock’s

• **doOutputBufferIteration**
  public void doOutputBufferIteration( )
  
  -- Usage
  * iterate the output buffer(s), or shift the history of it

• **getPanel**
  public JPanel getPanel( )
  
  -- Usage
  * generates the default panel and the title

• **getSimParams**
  public SimParameters getSimParams( )
  
  -- Usage
  * get the parameters from a parameter class

• **getVariableContainer**
  public VariableContainer getVariableContainer( )
  
  -- Usage
  * returns the conatainer for all the variables. this is mainly used for the operations done on the varContainer

• **isHidden**
  public boolean isHidden( )
  
  -- Usage
  * test if the hidden flag has been set

• **iterateBuffers**
  public void iterateBuffers( )
  
  -- Usage
  * This function does nothing, but blocks with outputs or internal buffers such as SimFunctionFactory should overload it for it’s own special pursposes

• **reDefine**
  public void reDefine( )
  
  -- Usage
  * transfer upper level adjustments to the lower level. this method is overloaded when it is to be used by a lower function

• **setHidden**
  public void setHidden( )
Usage
* keep the block from being loaded to the factory display if it is panelable

```java
public void setHidden( boolean hide )
```

Usage
* keep the block from being loaded to the factory display if it is panelable

```java
public void setSimParams( simtk.SimParameters sp )
```

Usage
* set the parameters from a parameter class

Methods inherited from class simtk.SimBase

```java
public String getDescription( )
```

Usage
* get the user defined title for the entire block

```java
public Simulatable getReloadBlock( )
```

Usage
* set the block that gets called to resimulate when the a variable is changed.

```java
public String getTitle( )
```

Usage
* get the user defined title for the entire block

```java
public void setDescription( java.lang.String desc )
```

```java
public void setReloadBlock( simtk.Simulatable reloadBlock )
```

Usage
* set the block that gets called to resimulate when the a variable is changed.

```java
public void setTitle( java.lang.String title )
```

Usage
* set the user defined title for the entire block

### 2.2.50 Class SimSubtract

Subtracts input 2 from input 1 out = in1 - in2. note: only the first 2 inputs work are used even if more are added
**Declaration**

```java
public class SimSubtract
extends simtk.SimFunction
```

**Constructors**

- `SimSubtract`
  ```java
  public SimSubtract()
  ```

**Methods**

- `doIteration`
  ```java
  public void doIteration()
  ```

- `toString`
  ```java
  public String toString()
  ```
  - Usage
    * figure out the special conditions for the subtract and print out the error situations

**Methods inherited from class `simtk.SimFunction`**

- `addInput`
  ```java
  public void addInput( simtk.HasOutput outputBlock )
  ```
  - Usage
    * this AddInput adds from a HasOutput block an input to the vector function and increases the inputIndex count

- `addInput`
  ```java
  public void addInput( simtk.SimBuffer bufref )
  ```
  - Usage
    * AddInput adds an input to the vector function and increases the inputIndex count

- `doIteration`
  ```java
  abstract void doIteration()
  ```
  - Usage
    * this is the method that should be implemented in all SimBlock’s

- `getInput`
  ```java
  public SimBuffer getInput()
  ```
Simtk – SimSubtract

- Usage
  * get the first input. there might be a problem here because the old method of the implicit in buffer is used

- **getInput**
  public SimBuffer getInput( int index )
  - Usage
    * get the indexed input, referenced from as 1 is the first.
    If the reference is out of range it will return the the single input value.
    The code is not so clean here where it will still print an error message when the double input functions are just beginning to set themselves up and before the.

- **getInputCount**
  public int getInputCount( )
  - Usage
    * get the inputCount, or the number of elements on the list

- **getOutput**
  public SimBuffer getOutput( )
  - Usage
    * Gets the output buffer

- **iterateBuffers**
  public void iterateBuffers( )
  - Usage
    * overload the do nothing iterateBuffers and iterate the output buffers

- **removeInputs**
  public void removeInputs( )
  - Usage
    * remove the all of the inputs from the block so a new system can be set up

- **setInput**
  public void setInput( int index, simtk.SimBuffer bufref )
  - Usage
    * Set the indexed input to the passed buffer. note: this has out of index problems

- **setInput**
  public void setInput( simtk.SimBuffer bufref )
  - Usage
    * Set the input to the passed buffer, this is replaced with the newer addInput function which sounds more like what it does

- **setInput1**
  public void setInput1( simtk.SimBuffer bufref )
Usage

• This is here to support old systems, it just calls addInput(bufref). so it is not exactly input 2

```java
public void setInput2( simtk.SimBuffer bufref )
```

Usage

• This is here to support old systems, it just calls addInput(bufref). so it is not exactly input 2

```java
public void setOutput( simtk.SimBuffer bufref )
```

Usage

• sets the output to a passed buffer

**Methods inherited from class simtk.SimBlock**

• clear

```java
public void clear( )
```

Usage

• this is the general function to clear a block. this is just an empty method but if more specific functionality is needed this should be overridden

• doIteration

```java
abstract void doIteration( )
```

Usage

• this is the method that should be implemented in all Simblock’s

• doOutputBufferIteration

```java
public void doOutputBufferIteration( )
```

Usage

• iterate the output buffer(s), or shift the history of it

• getPanel

```java
public JPanel getPanel( )
```

Usage

• generates the default panel and the title

• getSimParams

```java
public SimParameters getSimParams( )
```

Usage

• get the parameters from a parameter class
• `getVariableContainer`
  public VariableContainer `getVariableContainer`(
  )
  
  – Usage
  * returns the container for all the variables. this is mainly used for the operations done on
  the varContainer

• `isHidden`
  public boolean `isHidden`()
  
  – Usage
  * test if the hidden flag has been set

• `iterateBuffers`
  public void `iterateBuffers`()
  
  – Usage
  * This function does nothing, but blocks with outputs or internal buffers such as
    SimFunctionFactory should overload it for its own special purposes

• `reDefine`
  public void `reDefine`()
  
  – Usage
  * transfer upper level adjustments to the lower level. this method is overloaded when it is to
    be used by a lower function

• `setHidden`
  public void `setHidden`()
  
  – Usage
  * keep the block from being loaded to the factory display if it is panelable

• `setHidden boolean hide`?
  public void `setHidden` ( boolean `hide`)
  
  – Usage
  * keep the block from being loaded to the factory display if it is panelable

• `setSimParams`
  public void `setSimParams` ( simtk.SimParameters `sp`)
  
  – Usage
  * set the parameters from a parameter class
Methods inherited from class simtk.SimBase

- **getDescription**
  public String getDescription()
- **getReloadBlock**
  public Simulatable getReloadBlock()
  - **Usage**
    * set the block that gets called to resimulate when the a variable is changed.
- **getTitle**
  public String getTitle()
  - **Usage**
    * get the user defined title for the entire block
- **setDescription**
  public void setDescription(java.lang.String desc)
- **setReloadBlock**
  public void setReloadBlock(simtk.Simulatable reloadBlock)
  - **Usage**
    * set the block that gets called to resimulate when the a variable is changed.
- **setTitle**
  public void setTitle(java.lang.String title)
  - **Usage**
    * set the user defined title for the entire block

2.2.51 Class SimTest

Compilation of sim blocks to create the ADRC block

Declaration

```java
public class SimTest
  extends simtk.SimFunctionFactory
  implements SimPanelable, ContainsFactory
```

Fields

- private boolean debug
**Constructors**

- *SimTest*
  
  public SimTest()  
  
  - **Usage**  
    * constructor sets gain to default of unity

**Methods**

- *connectBlocks*
  
  public void connectBlocks()  
  
  - **Usage**  
    * Make the connections for this block. The setInput of this factory calls this to setup. It turns out to be a rather interesting system the setInput to this factory then in turn setups up this internal system.

  This function has problems due to the multiple inputs, which each call this connectBlocks() function, and in turn multiple connectons are made to each of the contained Blocks

  The factory.epmty() function should probably also clear the inputs from the contained functions aswell.

- *getPanel*
  
  public JPanel getPanel()  
  
  - **Usage**  
    * override the default and previous dialog panels

- *reDefine*
  
  public void reDefine()  
  
  - **Usage**  
    * the super magic code. These transfer functions were derived by Rob Milosovic from the state space seperated controller to a combined transfer funtion

**Methods inherited from class simtk.SimFunctionFactory**

- *addInput*
  
  public void addInput( simtk.HasOutput inputBlock )
- **Usage**
  * set the input of this factory this is the interesting one

- **addInput**
  ```java
  public void addInput( simtk.SimBuffer sb )
  ```
  - **Usage**
    * set the input of this factory. this is the interesting one

- **clear**
  ```java
  public void clear( )
  ```
  - **Usage**
    * this function is required to reset the buffers and values for the next iterations to set up the factory class

- **connectBlocks**
  ```java
  abstract void connectBlocks( )
  ```
  - **Usage**
    * Make the connections for this block. The setInput of this factory calls this to setup. It turns out to be a rather interesting system. The setInput() or addInput() to this factory then in turn set’s up this internal system. this is to be implemented.

- **doIteration**
  ```java
  public void doIteration( )
  ```
  - **Usage**
    * do a calculation for the new output

- **getPanel**
  ```java
  public JPanel getPanel( )
  ```

- **iterateBuffers**
  ```java
  public void iterateBuffers( )
  ```
  - **Usage**
    * cycle the buffers

- **setInput**
  ```java
  public void setInput( simtk.SimBuffer sb )
  ```
  - **Usage**
    * set the input of this factory. this is to be implemented in the Simblock to call the connect blocks

- **toString**
  ```java
  public String toString( )
  ```
Methods inherited from class simtk.SimFunction

- **addInput**
  public void addInput( simtk.HasOutput outputBlock )
  - **Usage**
    * this AddInput adds from a HasOutput block an input to the vector function and increases the inputIndex count

- **addInput**
  public void addInput( simtk.SimBuffer bufref )
  - **Usage**
    * AddInput adds an input to the vector function and increases the inputIndex count

- **doIteration**
  abstract void doIteration( )
  - **Usage**
    * this is the method that should be implemented in all SimBlock’s

- **getInput**
  public SimBuffer getInput( )
  - **Usage**
    * get the first input. there might be a problem here because the old method of the implicit in buffer is used

- **getInput**
  public SimBuffer getInput( int index )
  - **Usage**
    * get the indexed input, referenced from as 1 is the first.
      If the reference is out of range it will return the the single input value.
      The code is not so clean here where it will still print an erro message when the double input functions are just beginning to set themselves up and before the.???

- **getInputCount**
  public int getInputCount( )
  - **Usage**
    * get the inputCount, or the number of elements on the list

- **getOutput**
  public SimBuffer getOutput( )
  - **Usage**
    * Gets the output buffer

- **iterateBuffers**
  public void iterateBuffers( )
- **Usage**
  * overload the do nothing iterateBuffers and iterate the output buffers

- **removeInputs**
  ```java
  public void removeInputs()
  ```
  - **Usage**
    * remove the all of the inputs from the block so a new system can be set up

- **setInput**
  ```java
  public void setInput( int index, simtk.SimBuffer bufref )
  ```
  - **Usage**
    * Set the indexed input to the passed buffer. note: this has out of index problems

- **setInput**
  ```java
  public void setInput( simtk.SimBuffer bufref )
  ```
  - **Usage**
    * Set the input to the passed buffer, this is replaced with the newer addInput function which sounds more like what it does

- **setInput1**
  ```java
  public void setInput1( simtk.SimBuffer bufref )
  ```
  - **Usage**
    * This is here to support old systems, it just calls addInput(bufref). so it is not exactly input 2

- **setInput2**
  ```java
  public void setInput2( simtk.SimBuffer bufref )
  ```
  - **Usage**
    * This is here to support old systems, it just calls addInput(bufref). so it is not exactly input 2

- **setOutput**
  ```java
  public void setOutput( simtk.SimBuffer bufref )
  ```
  - **Usage**
    * sets the output to a passed buffer

**Methods inherited from class simtk.SimBlock**

- **clear**
  ```java
  public void clear()
  ```
  - **Usage**
    * this is the general function to clear a block. this is just an empty method but if more specific functionality is needed this should be overridden
• doIteration
  abstract void doIteration( )
  – Usage
    * this is the method that should be implemented in all Simblock’s

• doOutputBufferIteration
  public void doOutputBufferIteration( )
  – Usage
    * iterate the output buffer(s), or shift the history of it

• getPanel
  public JPanel getPanel( )
  – Usage
    * generates the default panel and the title

• getSimParams
  public SimParameters getSimParams( )
  – Usage
    * get the parameters from a parameter class

• getVariableContainer
  public VariableContainer getVariableContainer( )
  – Usage
    * returns the container for all the variables. This is mainly used for the operations done on
      the varContainer

• isHidden
  public boolean isHidden( )
  – Usage
    * test if the hidden flag has been set

• iterateBuffers
  public void iterateBuffers( )
  – Usage
    * This function does nothing, but blocks with outputs or internal buffers such as
      SimFunctionFactory should overload it for its own special purposes

• reDefine
  public void reDefine( )
  – Usage
    * transfer upper level adjustments to the lower level. This method is overloaded when it is to
      be used by a lower function

• setHidden
  public void setHidden( )
- **Usage**
  * keep the block from being loaded to the factory display if it is panelable

- **setHidden**
  public void setHidden( boolean hide )
  - **Usage**
  * keep the block from being loaded to the factory display if it is panelable

- **setSimParams**
  public void setSimParams( simtk.SimParameters sp )
  - **Usage**
  * set the parameters from a parameter class

**Methods inherited from class simtk.SimBase**

- **getDescription**
  public String getDescription( )

- **getReloadBlock**
  public Simulatable getReloadBlock( )
  - **Usage**
  * set the block that gets called to resimulate when the a variable is changed.

- **getTitle**
  public String getTitle( )
  - **Usage**
  * get the user defined title for the entire block

- **setDescription**
  public void setDescription( java.lang.String desc )

- **setReloadBlock**
  public void setReloadBlock( simtk.Simulatable reloadBlock )
  - **Usage**
  * set the block that gets called to resimulate when the a variable is changed.

- **setTitle**
  public void setTitle( java.lang.String title )
  - **Usage**
  * set the user defined title for the entire block
2.2.52  **Class SimTextBoxOutput**

This is a Sink block that simply send the current output to a formated output in swing text box.

**work to be done**

- add multiple inputs as the scope input has
- add selection of output options

**Declaration**

```java
public class SimTextBoxOutput
extends simtk.SimSink
implements SimPanelable
```

**Fields**

- private int k
  -
- private SimTextBox textBox
  -

**Constructors**

- `SimTextBoxOutput`
  ```java
  public SimTextBoxOutput()
  ```
  `- Usage
    * Default constructor

**Methods**

- `clear`
  ```java
  public void clear()
  ```
  `- Usage
* clears the text field, sets up headers and resets the index counter

• **doIteration**
  `public void doIteration( )`

  — **Usage**
  * this is the action function that does the iteration of sending the input to the standard output after each iteration

**Methods inherited from class simtk.SimSink**

• **addInput**
  `public void addInput( simtk.HasOutput outputBlock )`

  — **Usage**
  * this AddInput adds from a HasOutput block an input to the vector function and increases the inputIndex count

• **addInput**
  `public void addInput( simtk.SimBuffer bufref )`

  — **Usage**
  * addInput adds an input to the vector function and increases the inputIndex count

• **doIteration**
  `abstract void doIteration( )`

  — **Usage**
  * this is the method that should be implemented in all SimBlock’s

• **getInput**
  `public SimBuffer getInput( )`

  — **Usage**
  * get the first input

• **getInput**
  `public SimBuffer getInput( int index )`

  — **Usage**
  * get the indexed input, referenced from as 1 is the first

• **getInputCount**
  `public int getInputCount( )`

  — **Usage**
  * get the inputCount, or the number of elements on the list

• **removeInputs**
  `public void removeInputs( )`
• **setInput**
  
  ```java
  public void setInput( simtk.HasOutput blockref )
  ```

  _Usage_
  
  * set the input to the output buffer of the passed block

• **setInput**
  
  ```java
  public void setInput( int index, simtk.SimBuffer bufref )
  ```

  _Usage_
  
  * set the indexed input to the passed buffer this has out of index problems

• **setInput**
  
  ```java
  public void setInput( simtk.SimBuffer bufref )
  ```

  _Usage_
  
  * set the input to the passed buffer, this is replaced with the newer addInput function which
    sounds more like what it does

**Methods inherited from class simtk.SimBlock**

• **clear**
  
  ```java
  public void clear( )
  ```

  _Usage_
  
  * this is the general function to clear a block. this is just an empty method but if more
    specific functionality is needed this should be overridden

• **doIteration**
  
  ```java
  abstract void doIteration( )
  ```

  _Usage_
  
  * this is the method that should be implemented in all Simblock’s

• **doOutputBufferIteration**
  
  ```java
  public void doOutputBufferIteration( )
  ```

  _Usage_
  
  * iterate the output buffer(s), or shift the history of it

• **getPanel**
  
  ```java
  public JPanel getPanel( )
  ```

  _Usage_
  
  * generates the default panel and the title

• **getSimParams**
  
  ```java
  public SimParameters getSimParams( )
  ```
- **Usage**
  
  * get the parameters from a parameter class

- **getVariableContainer**
  
  ```java
  public VariableContainer getVariableContainer()
  ```

  - **Usage**
    
    * returns the container for all the variables. this is mainly used for the operations done on the varContainer

- **isHidden**
  
  ```java
  public boolean isHidden()
  ```

  - **Usage**
    
    * test if the hidden flag has been set

- **iterateBuffers**
  
  ```java
  public void iterateBuffers()
  ```

  - **Usage**
    
    * This function does nothing, but blocks with outputs or internal buffers such as SimFunctionFactory should overload it for it's own special purposes

- **reDefine**
  
  ```java
  public void reDefine()
  ```

  - **Usage**
    
    * transfer upper level adjustments to the lower level. this method is overloaded when it is to be used by a lower function

- **setHidden**
  
  ```java
  public void setHidden()
  ```

  - **Usage**
    
    * keep the block from being loaded to the factory display if it is panelable

- **setHidden**
  
  ```java
  public void setHidden( boolean hide )
  ```

  - **Usage**
    
    * keep the block from being loaded to the factory display if it is panelable

- **setSimParams**
  
  ```java
  public void setSimParams( simtk.SimParameters sp )
  ```

  - **Usage**
    
    * set the parameters from a parameter class
Methods inherited from class simtk.SimBase

- **getDescription**
  ```java
  public String getDescription()
  ```

- **getReloadBlock**
  ```java
  public Simulatable getReloadBlock()
  ```
  - **Usage**
    * set the block that gets called to resimulate when the a variable is changed.

- **getTitle**
  ```java
  public String getTitle()
  ```
  - **Usage**
    * get the user defined title for the entire block

- **setDescription**
  ```java
  public void setDescription(java.lang.String desc)
  ```

- **setReloadBlock**
  ```java
  public void setReloadBlock(simtk.Simulatable reloadBlock)
  ```
  - **Usage**
    * set the block that gets called to resimulate when the a variable is changed.

- **setTitle**
  ```java
  public void setTitle(java.lang.String title)
  ```
  - **Usage**
    * set the user defined title for the entire block

### 2.2.53 Class SimTimeIndex

Step function

**Declaration**

```java
public class SimTimeIndex
extends simtk.SimSource
implements SimPanelable
```
Fields

- private int k
- private SimNumber scale
  - Time scaler

Constructors

- SimTimeIndex
  public SimTimeIndex()
  - Usage
    * defaults are set to scale = 1

Methods

- clear
  public void clear()

- doIteration
  public void doIteration()
  - Usage
    * just the output time

Methods inherited from class simtk.SimSource

- doIteration
  abstract void doIteration()
  - Usage
    * this is the method that should be implemented in all SimBlock’s

- getOutput
  public SimBuffer getOutput()

- iterateBuffers
  public void iterateBuffers()
  - Usage
    * overload the do nothing iterateBuffers and iterate the output buffers

- setOutput
  public void setOutput( simtk.SimBuffer bufref )
Methods inherited from class `simtk.SimBlock`

- **clear**
  ```java
  public void clear()
  ```
  - **Usage**
  * this is the general function to clear a block. this is just an empty method but if more specific functionality is needed this should be overridden

- **doIteration**
  ```java
  abstract void doIteration()
  ```
  - **Usage**
  * this is the method that should be implemented in all Simblock’s

- **doOutputBufferIteration**
  ```java
  public void doOutputBufferIteration()
  ```
  - **Usage**
  * iterate the output buffer(s), or shift the history of it

- **getPanel**
  ```java
  public JPanel getPanel()
  ```
  - **Usage**
  * generates the default panel and the title

- **getSimParams**
  ```java
  public SimParameters getSimParams()
  ```
  - **Usage**
  * get the parameters from a parameter class

- **getVariableContainer**
  ```java
  public VariableContainer getVariableContainer()
  ```
  - **Usage**
  * returns the conatainer for all the variables. this is mainly used for the operations done on the varContainer

- **isHidden**
  ```java
  public boolean isHidden()
  ```
  - **Usage**
  * test if the hidden flag has been set

- **iterateBuffers**
  ```java
  public void iterateBuffers()
  ```
  - **Usage**
  * This function does nothing, but blocks with outputs or internal buffers such as SimFunctionFactory should overload it for it’s own special purposes
• reDefine
  public void reDefine( )
  
  Usage
  * transfer upper level adjustments to the lower level. this method is overloaded when it is to
  be used by a lower function

• setHidden
  public void setHidden( )
  
  Usage
  * keep the block from being loaded to the factory display if it is panelable

• setHidden
  public void setHidden( boolean hide )
  
  Usage
  * keep the block from being loaded to the factory display if it is panelable

• setSimParams
  public void setSimParams( simtk.SimParameters sp )
  
  Usage
  * set the parameters from a parameter class

Methods inherited from class simtk.SimBase

• getDescription
  public String getDescription( )

• getReloadBlock
  public Simulatable getReloadBlock( )
  
  Usage
  * set the block that gets called to resimulate when the a variable is changed.

• getTitle
  public String getTitle( )
  
  Usage
  * get the user defined title for the entire block

• setDescription
  public void setDescription( java.lang.String desc )

• setReloadBlock
  public void setReloadBlock( simtk.Simulatable reloadBlock )
  
  Usage
  * set the block that gets called to resimulate when the a variable is changed.

• setTitle
  public void setTitle( java.lang.String title )
  
  Usage
  * set the user defined title for the entire block
2.2.54 Class SimTrapezoid

Trapezoidal profile function. provides a smoother response for position, velocity, and acceleration in place of a step function.

Declaration

```java
public class SimTrapezoid
    extends simtk.SimSource
    implements SimPanelable
```

Fields

- private int k
- private SimNumber gain
- private SimNumber delay
- private SimNumber width
  - Width of trapezoid

Constructors

- **SimTrapezoid**
  public SimTrapezoid()
  - Usage
    * defaults are set to delay = 0 and gain = 1, totalTime = 1
Methods

- **clear**
  ```java
  public void clear()
  ```

- **doIteration**
  ```java
  public void doIteration()
  ```
  - **Usage**
    * set the output to the gain value after the delay time

Methods inherited from class `simtk.SimSource`

- **doIteration**
  ```java
  abstract void doIteration()
  ```
  - **Usage**
    * this is the method that should be implemented in all SimBlock’s

- **getOutput**
  ```java
  public SimBuffer getOutput()
  ```

- **iterateBuffers**
  ```java
  public void iterateBuffers()
  ```
  - **Usage**
    * overload the do nothing iterateBuffers and iterate the output buffers

- **setOutput**
  ```java
  public void setOutput( simtk.SimBuffer bufref)
  ```

Methods inherited from class `simtk.SimBlock`

- **clear**
  ```java
  public void clear()
  ```
  - **Usage**
    * this is the general function to clear a block. this is just an empty method but if more specific functionality is needed this should be overridden

- **doIteration**
  ```java
  abstract void doIteration()
  ```
  - **Usage**
    * this is the method that should be implemented in all Simblock’s

- **doOutputBufferIteration**
  ```java
  public void doOutputBufferIteration()
  ```
- **Usage**
  - *iterate the output buffer(s), or shift the history of it*

- **getPanel**
  public JPanel getPanel()

  - **Usage**
    * generates the default panel and the title

- **getSimParams**
  public SimParameters getSimParams()

  - **Usage**
    * get the parameters from a parameter class

- **getVariableContainer**
  public VariableContainer getVariableContainer()

  - **Usage**
    * returns the conatainer for all the variables. this is mainly used for the operations done on the varContainer

- **isHidden**
  public boolean isHidden()

  - **Usage**
    * test if the hidden flag has been set

- **iterateBuffers**
  public void iterateBuffers()

  - **Usage**
    * This function does nothing, but blocks with outputs or internal buffers such as SimFunctionFactory should overload it for it’s own special purposes

- **reDefine**
  public void reDefine()

  - **Usage**
    * transfer upper level adjustments to the lower level. this method is overloaded when it is to be used by a lower function

- **setHidden**
  public void setHidden()

  - **Usage**
    * keep the block from being loaded to the factory display if it is panelable

- **setHidden**
  public void setHidden(boolean hide)

  - **Usage**
    * keep the block from being loaded to the factory display if it is panelable
**setSimParams**
public void setSimParams(simtk.SimParameters sp)

- Usage
  * set the parameters from a parameter class

**Methods inherited from class simtk.SimBase**

- getDescription
  public String getDescription()

- getReloadBlock
  public Simulatable getReloadBlock()

  - Usage
    * set the block that gets called to resimulate when the a variable is changed.

- getTitle
  public String getTitle()

  - Usage
    * get the user defined title for the entire block

- setDescription
  public void setDescription(java.lang.String desc)

- setReloadBlock
  public void setReloadBlock(simtk.Simulatable reloadBlock)

  - Usage
    * set the block that gets called to resimulate when the a variable is changed.

- setTitle
  public void setTitle(java.lang.String title)

  - Usage
    * set the user defined title for the entire block

### 2.2.55 Class SimTrapezoidalProfile

compilation of sim blocks to create the profile

**Declaration**

```java
public class SimTrapezoidalProfile
extends simtk.SimSourceFactory
implements SimPanelable, ContainsFactory
```
**Constructors**

- `SimTrapezoidalProfile`
  ```java
  public SimTrapezoidalProfile()
  ```
  
  **Usage**
  
  * constructor sets gain to default of unity

**Methods**

- `connectBlocks`
  ```java
  public void connectBlocks()
  ```
  
  **Usage**
  
  * Make the connections for this block. The setInput of this factory calls this to setup. It turns out to be a rather interesting system the setInput to this factory then in turn sets up this internal system

**Methods inherited from class simtk.SimSourceFactory**

- `clear`
  ```java
  public void clear()
  ```
  
  **Usage**
  
  * this function is required to reset the buffers and values for the next iterations to set up the factory class

- `connectBlocks`
  ```java
  abstract void connectBlocks()
  ```
  
  **Usage**
  
  * Make the connections for this block. The setInput of this factory calls this to setup. It turns out to be a rather interesting system the setInput to this factory then in turn sets up this internal system. this is to be implemented.

- `doIteration`
  ```java
  public void doIteration()
  ```
  
  **Usage**
  
  * do a calculation for the new output

- `getPanel`
  ```java
  public JPanel getPanel()
  ```
## SimTrapezoidalProfile

- **iterateBuffers**
  public void iterateBuffers()  
  - Usage  
    * cycle the buffers

- **toString**
  public String toString()  

### Methods inherited from class simtk.SimSource

- **doIteration**
  abstract void doIteration()  
  - Usage  
    * this is the method that should be implemented in all SimBlock's

- **getOutput**
  public SimBuffer getOutput()  

- **iterateBuffers**
  public void iterateBuffers()  
  - Usage  
    * overload the do nothing iterateBuffers and iterate the output buffers

- **setOutput**
  public void setOutput( simtk.SimBuffer bufref )  

### Methods inherited from class simtk.SimBlock

- **clear**
  public void clear()  
  - Usage  
    * this is the general function to clear a block. this is just an empty method but if more specific functionality is needed this should be overridden

- **doIteration**
  abstract void doIteration()  
  - Usage  
    * this is the method that should be implemented in all Simblock's

- **doOutputBufferIteration**
  public void doOutputBufferIteration()  
  - Usage  
    * iterate the output buffer(s), or shift the history of it
• **getPanel**
  ```java
  public JPanel getPanel()
  ```
  ```
  - Usage
  * generates the default panel and the title
  ```

• **getSimParams**
  ```java
  public SimParameters getSimParams()
  ```
  ```
  - Usage
  * get the parameters from a parameter class
  ```

• **getVariableContainer**
  ```java
  public VariableContainer getVariableContainer()
  ```
  ```
  - Usage
  * returns the container for all the variables. this is mainly used for the operations done on the varContainer
  ```

• **isHidden**
  ```java
  public boolean isHidden()
  ```
  ```
  - Usage
  * test if the hidden flag has been set
  ```

• **iterateBuffers**
  ```java
  public void iterateBuffers()
  ```
  ```
  - Usage
  * This function does nothing, but blocks with outputs or internal buffers such as SimFunctionFactory should overload it for it's own special purposes
  ```

• **reDefine**
  ```java
  public void reDefine()
  ```
  ```
  - Usage
  * transfer upper level adjustments to the lower level. this method is overloaded when it is to be used by a lower function
  ```

• **setHidden**
  ```java
  public void setHidden()
  ```
  ```
  - Usage
  * keep the block from being loaded to the factory display if it is panelable
  ```

• **setHidden**
  ```java
  public void setHidden( boolean hide )
  ```
  ```
  - Usage
  * keep the block from being loaded to the factory display if it is panelable
  ```

• **setSimParams**
  ```java
  public void setSimParams( simtk.SimParameters sp )
  ```
  ```
  - Usage
  * set the parameters from a parameter class
  ```
Methods inherited from class simtk.SimBase

- **getDescription**
  public String getDescription()
- **getReloadBlock**
  public Simulatable getReloadBlock()
  - Usage
    * set the block that gets called to resimulate when the a variable is changed.
- **getTitle**
  public String getTitle()
  - Usage
    * get the user defined title for the entire block
- **setDescription**
  public void setDescription(java.lang.String desc)
- **setReloadBlock**
  public void setReloadBlock(simtk.Simulatable reloadBlock)
  - Usage
    * set the block that gets called to resimulate when the a variable is changed.
- **setTitle**
  public void setTitle(java.lang.String title)
  - Usage
    * set the user defined title for the entire block

2.2.56 Class SimZtf

this is the block that simulates ztranfer functions a number of general types of functions can be derived from this block

Declaration

```java
public class SimZtf
extends simtk.SimDiffEq
```
CONSTRUCTORS

- **SimZtf**
  
  public SimZtf()  

- **SimZtf**
  
  public SimZtf(int order)

  - **Usage**
    
    * constructor for the ztf, I am still a little bit unsure how to implement the zorder of the system.

METHODS

- **doIteration**
  
  public void doIteration()  

  - **Usage**
    
    * this method overrides the doIteration of the diffeq. this way it first redefines the variables and then iterates with the inherited structure

- **doIterationDirect**
  
  public void doIterationDirect()  

  - **Usage**
    
    * this is the really cool simple method that converts ztf to diffEQ. The function does the operation in the following order:
      1. parses the numerator and denominator
      2. creates the difference equation
      3. does one step operation.

- **reDefine**
  
  public void reDefine()  

  - **Usage**
    
    * this is a method that might be good to have defined in the sim Block as abstract this is made to recalculate with new parameters, step sizes parameters

- **setDen**
  
  public void setDen(double [] n)

  - **Usage**
    
    * method to set the denominator to the array value Example: setDen([1,2,3]); would set the denominator to $z^2 + 2z + 3$ an error message is displayed if the array is larger than the z order
• **setNum**
  public void setNum( double [] n )

  — **Usage**
  * method to set the numerator to the array value Example: setNum([1,2,3]); would set the numerator to z^2 + 2z + 3 an error message is displayed if the array is lager than the z order

• **toString**
  public String toString( )

  — **Usage**
  * toString in the form of: zorder = x num = [x] den = [x]

• **zeroNumAndDen**
  public void zeroNumAndDen( )

  — **Usage**
  * this method is for convienience in setting up z transfer functions it makes sure that the function is reset to zero from the initial conditions

**Methods inherited from class simtk.SimDiffEq**

• **clear**
  public void clear( )

  — **Usage**
  * clears the text field, sets up headers and resets the index counter

• **doIteration**
  public void doIteration( )

• **getInCoef**
  public SimDoubleArray getInCoef( )

  — **Usage**
  * return the input coeficients

• **getOutCoef**
  public SimDoubleArray getOutCoef( )

  — **Usage**
  * return the output coeficients

• **setZeros**
  public void setZeros( )
Usage
* zero all the coefficients for easy settings

• toDiffEqString
public String toDiffEqString()
– Usage
* output the string format of the difference equation. special function to iterate through all
of the parameters and create the difference equation. This function is also used for the
textBox for the dynamic pretty print viewing of the diff eq

• toString
public String toString()
– Usage
* general to String equation for the diff eq. this uses the toDiffEqString() function for the
special formatting so that one doesn’t overloaded

Methods inherited from class simtk.SimFunction

• addInput
public void addInput( simtk.HasOutput outputBlock)
– Usage
* this AddInput adds from a HasOutput block an input to the vector function and increases
the inputIndex count

• addInput
public void addInput( simtk.SimBuffer bufref)
– Usage
* AddInput adds an input to the vector function and increases the inputIndex count

• doIteration
abstract void doIteration()
– Usage
* this is the method that should be implemented in all SimBlock’s

• getInput
public SimBuffer getInput()
– Usage
* get the first input. there might be a problem here because the old method of the implicit
in buffer is used

• getInput
public SimBuffer getInput( int index)
– Usage
* get the indexed input, referenced from as 1 is the first.  
  If the reference is out of range it will return the single input value.  
  The code is not so clean here where it will still print an error message when the double  
  input functions are just beginning to set themselves up and before the.???

- **getInputCount**
  ```java
  public int getInputCount()
  ```
  - Usage
  * get the inputCount, or the number of elements on the list

- **getOutput**
  ```java
  public SimBuffer getOutput()
  ```
  - Usage
  * Gets the output buffer

- **iterateBuffers**
  ```java
  public void iterateBuffers()
  ```
  - Usage
  * overload the do nothing iterateBuffers and iterate the output buffers

- **removeInputs**
  ```java
  public void removeInputs()
  ```
  - Usage
  * remove the all of the inputs from the block so a new system can be set up

- **setInput**
  ```java
  public void setInput( int index, simtk.SimBuffer bufref )
  ```
  - Usage
  * Set the indexed input to the passed buffer. note: this has out of index problems

- **setInput**
  ```java
  public void setInput( simtk.SimBuffer bufref )
  ```
  - Usage
  * Set the input to the passed buffer, this is replaced with the newer addInput function which  
    sounds more like what it does

- **setInput1**
  ```java
  public void setInput1( simtk.SimBuffer bufref )
  ```
  - Usage
  * This is here to support old systems, it just calls addInput(bufref). so it is not exactly  
    input 2

- **setInput2**
  ```java
  public void setInput2( simtk.SimBuffer bufref )
  ```
  - Usage
* This is here to support old systems, it just calls addInput(bufref). so it is not exactly input 2

• **setOutput**
  public void setOutput( simtk.SimBuffer bufref )
  – Usage
  * sets the output to a passed buffer

**Methods inherited from class simtk.SimBlock**

• **clear**
  public void clear( )
  – Usage
  * this is the general function to clear a block. this is just an empty method but if more specific functionality is needed this should be overridden

• **doIteration**
  abstract void doIteration( )
  – Usage
  * this is the method that should be implemented in all Simblock’s

• **doOutputBufferIteration**
  public void doOutputBufferIteration( )
  – Usage
  * iterate the output buffer(s), or shift the history of it

• **getPanel**
  public JPanel getPanel( )
  – Usage
  * generates the default panel and the title

• **getSimParams**
  public SimParameters getSimParams( )
  – Usage
  * get the parameters from a parameter class

• **getVariableContainer**
  public VariableContainer getVariableContainer( )
  – Usage
  * returns the conatainer for all the variables. this is mainly used for the operations done on the varContainer

• **isHidden**
  public boolean isHidden( )
- **Usage**
  * test if the hidden flag has been set

- **iterateBuffers**
  
  ```java
  public void iterateBuffers()
  ```

  - **Usage**
    * This function does nothing, but blocks with outputs or internal buffers such as `SimFunctionFactory` should overload it for its own special purposes

- **reDefine**
  
  ```java
  public void reDefine()
  ```

  - **Usage**
    * transfer upper level adjustments to the lower level. This method is overloaded when it is to be used by a lower function

- **setHidden**
  
  ```java
  public void setHidden()
  ```

  - **Usage**
    * keep the block from being loaded to the factory display if it is panelable

- **setHidden**
  
  ```java
  public void setHidden( boolean hide )
  ```

  - **Usage**
    * keep the block from being loaded to the factory display if it is panelable

- **setSimParams**
  
  ```java
  public void setSimParams( simtk.SimParameters sp )
  ```

  - **Usage**
    * set the parameters from a parameter class

**Methods inherited from class simtk.SimBase**

- **getDescription**
  
  ```java
  public String getDescription()
  ```

- **getReloadBlock**
  
  ```java
  public Simulatable getReloadBlock()
  ```

  - **Usage**
    * set the block that gets called to resimulate when the a variable is changed.

- **getTitle**
  
  ```java
  public String getTitle()
  ```

  - **Usage**
    * get the user defined title for the entire block
• **setDescription**
  
  ```java
  public void setDescription(java.lang.String desc)
  ```
  
  **Usage**
  
  * set the description of the block.

• **setReloadBlock**
  
  ```java
  public void setReloadBlock(simtk.Simulatable reloadBlock)
  ```
  
  **Usage**
  
  * set the block that gets called to resimulate when a variable is changed.

• **setTitle**
  
  ```java
  public void setTitle(java.lang.String title)
  ```
  
  **Usage**
  
  * set the user defined title for the entire block.

### 2.2.57 Class SimZtfDirect

This is the block that simulates transfer functions directly to the output without using the diff eq class. A number of general types of functions can be derived from this block.

**Declaration**

```java
public class SimZtfDirect
extends simtk.SimFunction
```

**Constructors**

• **SimZtfDirect**
  
  ```java
  public SimZtfDirect()
  ```

• **SimZtfDirect**
  
  ```java
  public SimZtfDirect(int order)
  ```
  
  **Usage**
  
  * constructor for the ztf, I am still a little bit unsure how to implement the zorder of the system.

**Methods**

• **displayIndexOverflowErr**
  
  ```java
  private void displayIndexOverflowErr(int index)
  ```
  
  **Usage**
* overflow error message that is sent to the stdout

- **doIteration**
  public void *doIteration*( )
  
  - **Usage**
    * this is the really cool simple method that converts ztf to diffEQ. The function does
      the operation in the following order:
      1. parses the numerator and denominator
      2. creates the difference equation
      3. does one step operation.

- **setDen**
  public void *setDen*( double [] n )
  
  - **Usage**
    * method to set the denominator to the array value Example: *setDen*([1,2,3]); would
      set the denominator to z^2 + 2*z + 3 an error message is displayed if the array is
      larger than the z order

- **setDen**
  public void *setDen*( int n, double val )

  - **Usage**
    * selects a particular zordered parameter (n) and sets the coefficient to val

- **setNum**
  public void *setNum*( double [] n )

  - **Usage**
    * method to set the numerator to the array value Example: *setNum*([1,2,3]); would
      set the numerator to z^2 + 2*z + 3 an error message is displayed if the array is
      larger than the z order

- **setNum**
  public void *setNum*( int n, double val )

  - **Usage**
    * selects a particular zordered parameter (n) and sets the coefficient to val

- **toString**
  public String *toString*( )

  - **Usage**
    * toString in the form of: zorder = x num = [x] den = [x]
• **zeroNumAndDen**
  public void zeroNumAndDen( )

  – **Usage**
  * this method is for convenience in setting up z transfer functions it makes sure that
  the function is reset to zero from the initial conditions

**Methods inherited from class simtk.SimFunction**

• **addInput**
  public void addInput( simtk.HasOutput outputBlock )

  – **Usage**
  * this AddInput adds from a HasOutput block an input to the vector function and increases
  the inputIndex count

• **addInput**
  public void addInput( simtk.SimBuffer bufref )

  – **Usage**
  * AddInput adds an input to the vector function and increases the inputIndex count

• **doIteration**
  abstract void doIteration( )

  – **Usage**
  * this is the method that should be implemented in all SimBlock’s

• **getInput**
  public SimBuffer getInput( )

  – **Usage**
  * get the first input. there might be a problem here because the old method of the implicit
  in buffer is used

• **getInput**
  public SimBuffer getInput( int index )

  – **Usage**
  * get the indexed input, referenced from as 1 is the first.
  If the reference is out of range it will return the the single input value.
  The code is not so clean here where it will still print an erro message when the double
  input functions are just beginning to set themselves up and before the.???

• **getInputCount**
  public int getInputCount( )

  – **Usage**
  * get the inputCount, or the number of elements on the list
• getOutput
  public SimBuffer getOutput( )
  – Usage
  * Gets the output buffer

• iterateBuffers
  public void iterateBuffers( )
  – Usage
  * overload the do nothing iterateBuffers and iterate the output buffers

• removeInputs
  public void removeInputs( )
  – Usage
  * remove the all of the inputs from the block so a new system can be set up

• setInput
  public void setInput( int index, simtk.SimBuffer bufref )
  – Usage
  * Set the indexed input to the passed buffer. note: this has out of index problems

• setInput
  public void setInput( simtk.SimBuffer bufref )
  – Usage
  * Set the input to the passed buffer, this is replaced with the newer addInput function which
    sounds more like what it does

• setInput1
  public void setInput1( simtk.SimBuffer bufref )
  – Usage
  * This is here to support old systems, it just calls addInput(bufref). so it is not exactly
    input 2

• setInput2
  public void setInput2( simtk.SimBuffer bufref )
  – Usage
  * This is here to support old systems, it just calls addInput(bufref). so it is not exactly
    input 2

• setOutput
  public void setOutput( simtk.SimBuffer bufref )
  – Usage
  * sets the output to a passed buffer
Methods inherited from class `simtk.SimBlock`

- **clear**
  
  ```java
  public void clear()
  ```
  
  - **Usage**
    
    * this is the general function to clear a block. this is just an empty method but if more specific functionality is needed this should be overridden

- **doIteration**
  
  ```java
  abstract void doIteration()
  ```
  
  - **Usage**
    
    * this is the method that should be implemented in all Simblock’s

- **doOutputBufferIteration**
  
  ```java
  public void doOutputBufferIteration()
  ```
  
  - **Usage**
    
    * iterate the output buffer(s), or shift the history of it

- **getPanel**
  
  ```java
  public JPanel getPanel()
  ```
  
  - **Usage**
    
    * generates the default panel and the title

- **getSimParams**
  
  ```java
  public SimParameters getSimParams()
  ```
  
  - **Usage**
    
    * get the parameters from a parameter class

- **getVariableContainer**
  
  ```java
  public VariableContainer getVariableContainer()
  ```
  
  - **Usage**
    
    * returns the container for all the variables. this is mainly used for the operations done on the varContainer

- **isHidden**
  
  ```java
  public boolean isHidden()
  ```
  
  - **Usage**
    
    * test if the hidden flag has been set

- **iterateBuffers**
  
  ```java
  public void iterateBuffers()
  ```
  
  - **Usage**
    
    * This function does nothing, but blocks with outputs or internal buffers such as SimFunctionFactory should overload it for it’s own special purposes
• **reDefine**
  public void reDefine( )
  
  – **Usage**
    * transfer upper level adjustments to the lower level. this method is overloaded when it is to be used by a lower function

• **setHidden**
  public void setHidden( )
  
  – **Usage**
    * keep the block from being loaded to the factory display if it is panelable

• **setHidden**
  public void setHidden( boolean hide )
  
  – **Usage**
    * keep the block from being loaded to the factory display if it is panelable

• **setSimParams**
  public void setSimParams( simtk.SimParameters sp )
  
  – **Usage**
    * set the parameters from a parameter class

**Methods inherited from class simtk.SimBase**

• **getDescription**
  public String getDescription( )

• **getReloadBlock**
  public Simulatable getReloadBlock( )
  
  – **Usage**
    * set the block that gets called to resimulate when the a variable is changed.

• **getTitle**
  public String getTitle( )
  
  – **Usage**
    * get the user defined title for the entire block

• **setDescription**
  public void setDescription( java.lang.String desc )

• **setReloadBlock**
  public void setReloadBlock( simtk.Simulatable reloadBlock )
  
  – **Usage**
    * set the block that gets called to resimulate when the a variable is changed.

• **setTitle**
  public void setTitle( java.lang.String title )
  
  – **Usage**
    * set the user defined title for the entire block