

Ohm

0.1

Generated by Doxygen 1.8.12

Contents

1	Module Index	1
1.1	Modules	1
2	Namespace Index	3
2.1	Packages	3
3	Hierarchical Index	5
3.1	Class Hierarchy	5
4	Class Index	7
4.1	Class List	7
5	Module Documentation	9
5.1	UserInterface	9
5.1.1	Detailed Description	9
5.2	ColourMapping	10
5.2.1	Detailed Description	10
5.3	ValueCalculator	11
5.3.1	Detailed Description	11
5.4	BandLocation	12
5.4.1	Detailed Description	12
5.5	ResistorID	13
5.5.1	Detailed Description	13
5.6	CameraInput	14
5.6.1	Detailed Description	14
5.7	ImageInput	15
5.7.1	Detailed Description	15

6	Namespace Documentation	17
6.1	Package ImageProcessing	17
6.1.1	Detailed Description	17
6.2	Package Input	17
6.2.1	Detailed Description	17
6.3	Package UserInterface	17
6.3.1	Detailed Description	17
6.4	Package ValueIdentification	17
6.4.1	Detailed Description	17
7	Class Documentation	19
7.1	ohm.ImageProcessing.BandReader Class Reference	19
7.1.1	Detailed Description	19
7.1.2	Member Function Documentation	20
7.1.2.1	boxSample()	20
7.1.2.2	componentwiseMean()	21
7.1.2.3	diff()	21
7.1.2.4	dist()	21
7.1.2.5	mag()	23
7.1.2.6	read()	23
7.1.2.7	rollingAverageFilter()	23
7.1.2.8	sample()	24
7.2	ohm.Input.CameraInput Class Reference	24
7.2.1	Detailed Description	25
7.2.2	Member Function Documentation	25
7.2.2.1	getImage()	25
7.2.2.2	getMat()	25
7.3	ohm.Helpers Class Reference	25
7.3.1	Detailed Description	26
7.4	ohm.Input.ImageInput Class Reference	26
7.4.1	Detailed Description	26

7.4.2	Constructor & Destructor Documentation	26
7.4.2.1	ImageInput()	26
7.4.3	Member Function Documentation	27
7.4.3.1	getImage()	27
7.4.3.2	getMat()	27
7.5	ohm.Input.Input Interface Reference	27
7.5.1	Detailed Description	28
7.5.2	Member Function Documentation	28
7.5.2.1	getImage()	28
7.5.2.2	getMat()	28
7.6	ohm.userinterface.LiveImageView Class Reference	28
7.6.1	Detailed Description	29
7.7	ohm.Ohm Class Reference	29
7.8	ohm.ImageProcessing.OhmImageProcessor Class Reference	29
7.8.1	Detailed Description	30
7.9	ohm.userinterface.OhmViewController Class Reference	30
7.9.1	Detailed Description	30
7.9.2	Member Function Documentation	30
7.9.2.1	initialize()	30
7.10	ohm.userinterface.ResistorAxisPickerView Class Reference	31
7.10.1	Detailed Description	31
7.10.2	Member Function Documentation	31
7.10.2.1	setImage()	31
7.10.2.2	setListener()	32
7.11	ohm.ImageProcessing.ResistorBodyID Class Reference	32
7.11.1	Detailed Description	32
7.12	ohm.ValueIdentification.ResistorColour Enum Reference	32
7.12.1	Detailed Description	33
7.12.2	Constructor & Destructor Documentation	33
7.12.2.1	ResistorColour()	33
7.12.3	Member Function Documentation	33
7.12.3.1	fit() [1/2]	33
7.12.3.2	fit() [2/2]	34
7.13	ohm.ValueIdentification.ValueCalculator Class Reference	34
7.13.1	Detailed Description	34

Chapter 1

Module Index

1.1 Modules

Here is a list of all modules:

UserInterface	9
ColourMapping	10
ValueCalculator	11
BandLocation	12
ResistorID	13
CameraInput	14
ImageInput	15

Chapter 2

Namespace Index

2.1 Packages

Here are the packages with brief descriptions (if available):

ImageProcessing	17
Input	17
UserInterface	17
ValueIdentification	17

Chapter 3

Hierarchical Index

3.1 Class Hierarchy

This inheritance list is sorted roughly, but not completely, alphabetically:

ohm.ImageProcessing.BandReader	19
ohm.Helpers	25
ohm.Input.Input	27
ohm.Input.CameraInput	24
ohm.Input.ImageInput	26
ohm.ImageProcessing.OhmImageProcessor	29
ohm.ImageProcessing.ResistorBodyID	32
ohm.ValueIdentification.ResistorColour	32
ohm.ValueIdentification.ValueCalculator	34
Application	
ohm.Ohm	29
EventHandler	
ohm.userinterface.ResistorAxisPickerView	31
ImageView	
ohm.userinterface.LiveImageView	28
ohm.userinterface.ResistorAxisPickerView	31
Initializable	
ohm.userinterface.OhmViewController	30

Chapter 4

Class Index

4.1 Class List

Here are the classes, structs, unions and interfaces with brief descriptions:

ohm.ImageProcessing.BandReader	Module used to analyze the line of pixels selected by the user through the UI. It uses high values of the differential of the RGB colours to detect edges of bands	19
ohm.Input.CameraInput	Instances of this class are to be used to receive input from device camera. Currently unimplemented	24
ohm.Helpers	25
ohm.Input.ImageInput	A source of input data, uses static images	26
ohm.Input.Input	27
ohm.userinterface.LiveImageView	The LiveImageView is an ImageView that periodically calls a Renderer to produce a new image that the ImageView then displays. This is useful to create images from real time updating images/data	28
ohm.Ohm	29
ohm.ImageProcessing.OhmImageProcessor	Module that will be used as a means of combining the algorithms outlined in BandReader and ResistorBodyID . Not yet implemented	29
ohm.userinterface.OhmViewController	Coordinates the algorithms present in the model with user input through the view	30
ohm.userinterface.ResistorAxisPickerView	This class displays an opencv Mat as an image and allows the user to pick two endpoints of a line. Once two valid points, a listener is called	31
ohm.ImageProcessing.ResistorBodyID	32
ohm.ValueIdentification.ResistorColour	Enum containing all of the possible colours that a resistor can take on. Also features member functions used to map the colours of bands to values used in the calculation process	32
ohm.ValueIdentification.ValueCalculator	Object used to calculate the resistance of the resistor based on the mapped colours	34

Chapter 5

Module Documentation

5.1 `UserInterface`

Classes

- class [ohm.userinterface.LiveImageView](#)

The [LiveImageView](#) is an `ImageView` that periodically calls a `Renderer` to produce a new image that the `ImageView` then displays. This is useful to create images from real time updating images/data.

- class [ohm.userinterface.OhmViewController](#)

Coordinates the algorithms present in the model with user input through the view.

- class [ohm.userinterface.ResistorAxisPickerView](#)

This class displays an opencv `Mat` as an image and allows the user to pick two endpoints of a line. Once two valid points, a listener is called.

5.1.1 Detailed Description

5.2 ColourMapping

Classes

- enum [ohm.ValueIdentification.ResistorColour](#)

Enum containing all of the possible colours that a resistor can take on. Also features member functions used to map the colours of bands to values used in the calculation process.

5.2.1 Detailed Description

Author

Jonathan Brown

5.3 ValueCalculator

Classes

- class [ohm.ValueIdentification.ValueCalculator](#)

Object used to calculate the resistance of the resistor based on the mapped colours.

5.3.1 Detailed Description

Author

Jonathan Brown

5.4 BandLocation

Classes

- class [ohm.ImageProcessing.BandReader](#)

Module used to analyze the line of pixels selected by the user through the UI. It uses high values of the differential of the RGB colours to detect edges of bands.

5.4.1 Detailed Description

Module used to analyze the line of pixels selected by the user through the UI. It uses high values of the differential of the RGB colours to detect edges of bands.

5.5 ResistorID

Classes

- class [ohm.ImageProcessing.ResistorBodyID](#)

5.5.1 Detailed Description

5.6 CameraInput

Classes

- class [ohm.Input.CameraInput](#)

Instances of this class are to be used to receive input from device camera. Currently unimplemented.

5.6.1 Detailed Description

5.7 ImageInput

Classes

- class [ohm.Input.ImageInput](#)
A source of input data, uses static images.

5.7.1 Detailed Description

Chapter 6

Namespace Documentation

6.1 Package ImageProcessing

6.1.1 Detailed Description

Contains the Band Location and Resistor Body Identification modules.

6.2 Package Input

6.2.1 Detailed Description

This package contains the components of the Camera and ImageFile modules. They hide the applications access to the hardware (camera or file system).

6.3 Package UserInterface

6.3.1 Detailed Description

Contains the views and the controller of the application. This is a Behavioural Module who's secret is the UI components of the application.

6.4 Package ValueIdentification

6.4.1 Detailed Description

Contains the Colour Mapping and Value Identification Modules

Chapter 7

Class Documentation

7.1 ohm.ImageProcessing.BandReader Class Reference

Module used to analyze the line of pixels selected by the user through the UI. It uses high values of the differential of the RGB colours to detect edges of bands.

Static Public Member Functions

- static List< Point > **read** (Mat frame, Point p1, Point p2)
- static double [][] **boxSample** (Mat mat, Point start, Point end, int nSamples, double boxWidth)
- static double [][] **rollingAverageFilter** (double[][] **sample**, int windowRadius)
- static double [] **componentwiseMean** (double[][] samples)
- static double [][] **sample** (Mat mat, Point start, Point end, int nSamples)
- static double [][] **diff** (double[][] y)
- static double **mag** (double[] vect)
- static double **dist** (Point p1, Point p2)
- static double [] **groupTerms** (double[] terms, int binSize)
- static List< Pair< Integer, Double > > **findLocalMaxima** (double[] values)
- static List< Pair< Integer, Double > > **findGlobalMaxima** (double[] values, int nMaxima)
- static Point **onLine** (Point start, Point end, double fraction)

7.1.1 Detailed Description

Module used to analyze the line of pixels selected by the user through the UI. It uses high values of the differential of the RGB colours to detect edges of bands.

Author

Ryan Marks & Jonathan Brown

7.1.2 Member Function Documentation

7.1.2.1 boxSample()

```
static double [][] ohm.ImageProcessing.BandReader.boxSample (
    Mat mat,
    Point start,
    Point end,
    int nSamples,
    double boxWidth ) [static]
```

This method takes something Ryan made up called a box sample. At *nSamples* points along the line from *start* to *end*, a line of samples are take along a line of length *boxWidth*, normal to and centred on the sampling line

Parameters

<i>mat</i>	The matrix being sampled
<i>start</i>	The start of the sampling line
<i>end</i>	The end of the sampling line
<i>nSamples</i>	The number or linear samples to take along the sampling line
<i>boxWidth</i>	the length of the normal sampling line.

Returns

An array containing the average rgb values (represented as double[]) in the sampling window.

7.1.2.2 componentwiseMean()

```
static double [] ohm.ImageProcessing.BandReader.componentwiseMean (
    double samples[][] ) [static]
```

Calculate the mean of an array of vectors.

Parameters

<i>samples</i>	Array of RGB vectors represented as double[]
----------------	----------------------------------------------

Returns

The average of all vectors within the input array (represented as a double[])

7.1.2.3 diff()

```
static double [][] ohm.ImageProcessing.BandReader.diff (
    double y[][] ) [static]
```

Take the discrete derivative of a series of vectors.

Parameters

<i>y</i>	A series of vectors
----------	---------------------

Returns

y' , the derivative of the input represented as a array of vectors (double[][]).

7.1.2.4 dist()

```
static double ohm.ImageProcessing.BandReader.dist (
    Point p1,
    Point p2 ) [static]
```

Calcualte the distance between two points.

Parameters

<i>p1</i>	Point 1.
<i>p2</i>	Point 2.

Returns

Distance between p1 and p2.

7.1.2.5 mag()

```
static double ohm.ImageProcessing.BandReader.mag (
    double [] vect ) [static]
```

Calculate the magnitude of a given vector

Parameters

<i>vect</i>	Input vector
-------------	--------------

Returns

The magnitude of the input vector

7.1.2.6 read()

```
static List<Point> ohm.ImageProcessing.BandReader.read (
    Mat frame,
    Point p1,
    Point p2 ) [static]
```

Parameters

<i>frame</i>	Image to Sample.
<i>p1</i>	Starting point of the sampling line.
<i>p2</i>	Ending point of the sampling line.

Returns

List of points along the line that are likely band edges.

7.1.2.7 rollingAverageFilter()

```
static double [][] ohm.ImageProcessing.BandReader.rollingAverageFilter (
    double sample[][],
    int windowRadius ) [static]
```

This method is used to apply a rolling average filter to vector data.

Parameters

<i>sample</i>	An array of (arrays of doubles representing an individual vector sample)
<i>windowRadius</i>	The number of samples on either side of a given sample to incorporate into the average

Returns

An array containing the average RGB value sampled along the line.

7.1.2.8 sample()

```
static double [][] ohm.ImageProcessing.BandReader.sample (
    Mat mat,
    Point start,
    Point end,
    int nSamples ) [static]
```

Take a linear sampling from a matrix at a given number of points along a line

Parameters

<i>mat</i>	The matrix to be sampled
<i>start</i>	The start of the sampling line
<i>end</i>	The end of the sampling line
<i>nSamples</i>	The number of samples to take along the line.

Returns

The samples taken

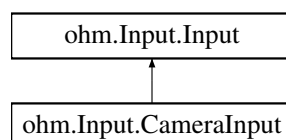
The documentation for this class was generated from the following file:

- java/ohm/ImageProcessing/BandReader.java

7.2 ohm.Input.CameraInput Class Reference

Instances of this class are to be used to receive input from device camera. Currently unimplemented.

Inheritance diagram for ohm.Input.CameraInput:



Public Member Functions

- Mat [getMat](#) ()
- Image [getImage](#) ()

7.2.1 Detailed Description

Instances of this class are to be used to receive input from device camera. Currently unimplemented.

7.2.2 Member Function Documentation

7.2.2.1 [getImage\(\)](#)

```
Image ohm.Input.CameraInput.getImage ( )
```

Used to retrieve a image representation of the input (used by JavaFX).

Returns

Image representation of input.

Implements [ohm.Input.Input](#).

7.2.2.2 [getMat\(\)](#)

```
Mat ohm.Input.CameraInput.getMat ( )
```

Used to retrieve a matrix representation of the input (used by Opencv libraries).

Returns

Matrix representation of input.

Implements [ohm.Input.Input](#).

The documentation for this class was generated from the following file:

- `java/ohm/Input/CameraInput.java`

7.3 ohm.Helpers Class Reference

Static Public Member Functions

- static Image **matToImage** (Mat mat)

7.3.1 Detailed Description

Module featuring static methods used by multiple other modules in order to perform common computations. Will be eliminated in final release.

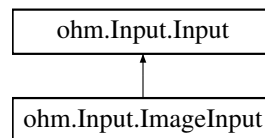
The documentation for this class was generated from the following file:

- `java/ohm/Helpers.java`

7.4 ohm.Input.ImageInput Class Reference

A source of input data, uses static images.

Inheritance diagram for ohm.Input.ImageInput:



Public Member Functions

- [ImageInput](#) ()
- [Mat getMat](#) ()
- [Image getImage](#) ()

7.4.1 Detailed Description

A source of input data, uses static images.

Author

Jonathan Brown

7.4.2 Constructor & Destructor Documentation

7.4.2.1 ImageInput()

```
ohm.Input.ImageInput.ImageInput ( )
```

Default constructor creates an instance using a default image.

7.4.3 Member Function Documentation

7.4.3.1 getImage()

```
Image ohm.Input.ImageInput.getImage ( )
```

Used to retrieve a image representation of the input (used by JavaFX).

Returns

Image representation of input.

Implements [ohm.Input.Input](#).

7.4.3.2 getMat()

```
Mat ohm.Input.ImageInput.getMat ( )
```

Method returns a OpenCV Matrix of the loaded image.

Returns

OpenCV Matrix representation of the image.

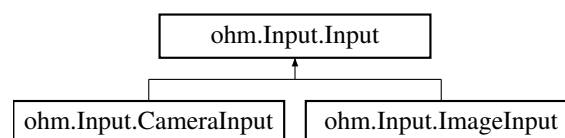
Implements [ohm.Input.Input](#).

The documentation for this class was generated from the following file:

- `java/ohm/Input/ImageInput.java`

7.5 ohm.Input.Input Interface Reference

Inheritance diagram for ohm.Input.Input:



Public Member Functions

- Mat [getMat](#) ()
- Image [getImage](#) ()

7.5.1 Detailed Description

Author

Jonathan Brown

7.5.2 Member Function Documentation

7.5.2.1 getImage()

```
Image ohm.Input.Input.getImage ( )
```

Used to retrieve a image representation of the input (used by JavaFX).

Returns

Image representation of input.

Implemented in [ohm.Input.ImageInput](#), and [ohm.Input.CameraInput](#).

7.5.2.2 getMat()

```
Mat ohm.Input.Input.getMat ( )
```

Used to retrieve a matrix representation of the input (used by Opencv libraries).

Returns

Matrix representation of input.

Implemented in [ohm.Input.ImageInput](#), and [ohm.Input.CameraInput](#).

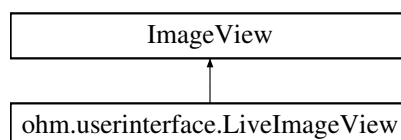
The documentation for this interface was generated from the following file:

- `java/ohm/Input/Input.java`

7.6 ohm.userinterface.LiveImageView Class Reference

The [LiveImageView](#) is an [ImageView](#) that periodically calls a [Renderer](#) to produce a new image that the [ImageView](#) then displays. This is useful to create images from real time updating images/data.

Inheritance diagram for `ohm.userinterface.LiveImageView`:



Classes

- interface **Renderer**

Public Member Functions

- void **setFrameRate** (int frameRate)
- void **setRenderer** (Renderer newRenderer)

7.6.1 Detailed Description

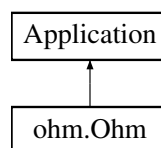
The [LiveImageView](#) is an `ImageView` that periodically calls a `Renderer` to produce a new image that the `ImageView` then displays. This is useful to create images from real time updating images/data.

The documentation for this class was generated from the following file:

- `java/ohm/userinterface/LiveImageView.java`

7.7 ohm.Ohm Class Reference

Inheritance diagram for `ohm.Ohm`:



Public Member Functions

- void **start** (Stage primaryStage) throws Exception

Static Public Member Functions

- static void **main** (String[] args) throws Exception

The documentation for this class was generated from the following file:

- `java/ohm/Ohm.java`

7.8 ohm.ImageProcessing.OhmImageProcessor Class Reference

Module that will be used as a means of combining the algorithms outlined in [BandReader](#) and [ResistorBodyID](#). Not yet implemented.

7.8.1 Detailed Description

Module that will be used as a means of combining the algorithms outlined in [BandReader](#) and [ResistorBodyID](#). Not yet implemented.

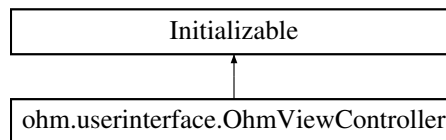
The documentation for this class was generated from the following file:

- `java/ohm/ImageProcessing/OhmImageProcessor.java`

7.9 ohm.userinterface.OhmViewController Class Reference

Coordinates the algorithms present in the model with user input through the view.

Inheritance diagram for ohm.userinterface.OhmViewController:



Public Member Functions

- void [initialize](#) (URL url, ResourceBundle rb)
- void **buttonClicked** (ActionEvent actionEvent)

Public Attributes

- Button **startCameraButton**

7.9.1 Detailed Description

Coordinates the algorithms present in the model with user input through the view.

7.9.2 Member Function Documentation

7.9.2.1 initialize()

```
void ohm.userinterface.OhmViewController.initialize (
    URL url,
    ResourceBundle rb )
```

Method used to "glue" together the front and back end of the application.

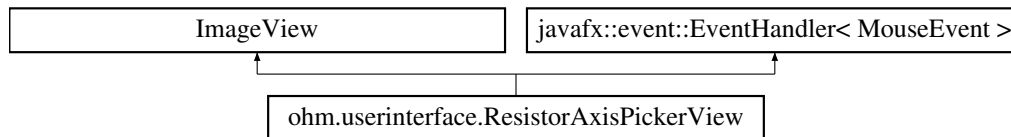
The documentation for this class was generated from the following file:

- `java/ohm/userinterface/OhmViewController.java`

7.10 ohm.userinterface.ResistorAxisPickerView Class Reference

This class displays an opencv Mat as an image and allows the user to pick two endpoints of a line. Once two valid points, a listener is called.

Inheritance diagram for ohm.userinterface.ResistorAxisPickerView:



Classes

- interface **Listener**

Public Member Functions

- void **handle** (MouseEvent event)
- void [setImage](#) (Mat newImage)
- void [setListener](#) (Listener listener)

Static Public Member Functions

- static double **dist** (Point p1, Point p2)

7.10.1 Detailed Description

This class displays an opencv Mat as an image and allows the user to pick two endpoints of a line. Once two valid points, a listener is called.

7.10.2 Member Function Documentation

7.10.2.1 setImage()

```
void ohm.userinterface.ResistorAxisPickerView.setImage (
    Mat newImage )
```

Set the matrix to display

Parameters

<i>newImage</i>	Image to be set as the image to be displayed in the UI
-----------------	--------------------------------------------------------

7.10.2.2 `setListener()`

```
void ohm.userinterface.ResistorAxisPickerView.setListener (
    Listener listener )
```

Assign a line listener to be updated when new points are selected

Parameters

<i>listener</i>	Object that triggers responds to a mouse click inside the view.
-----------------	-----------------------------------------------------------------

The documentation for this class was generated from the following file:

- `java/ohm/userinterface/ResistorAxisPickerView.java`

7.11 `ohm.ImageProcessing.ResistorBodyID` Class Reference

7.11.1 Detailed Description

Module to be used for identifying arbitrarily placed resistors in an image. Not yet implemented. Expected time is by Rev-1 Demo.

The documentation for this class was generated from the following file:

- `java/ohm/ImageProcessing/ResistorBodyID.java`

7.12 `ohm.ValueIdentification.ResistorColour` Enum Reference

Enum containing all of the possible colours that a resistor can take on. Also features member functions used to map the colours of bands to values used in the calculation process.

Public Member Functions

- [ResistorColour](#) (int v, double r, double g, double b)

Static Public Member Functions

- static [ResistorColour fit](#) (int r, int g, int b)
- static [ResistorColour fit](#) (double r, double g, double b)

Public Attributes

- **BLACK** =(0,0,0,0)
- **BROWN** =(1,70,45,35)
- **RED** =(2,155,0,35)
- **ORANGE** =(3,245,105,30)
- **YELLOW** =(4,200,200,55)
- **GREEN** =(5,20,75,40)
- **BLUE** =(6,45,45,140)
- **VIOLET** =(7,50,33,80)
- **GREY** =(8,125,125,125)
- **WHITE** =(9,200,200,200)
- **BASE** =(-1,200,170,145)
- int **value**
- double **red**
- double **green**
- double **blue**

7.12.1 Detailed Description

Enum containing all of the possible colours that a resistor can take on. Also features member functions used to map the colours of bands to values used in the calculation process.

7.12.2 Constructor & Destructor Documentation

7.12.2.1 ResistorColour()

```
ohm.ValueIdentification.ResistorColour.ResistorColour (
    int v,
    double r,
    double g,
    double b )
```

Parameters

<i>v</i>	The number represented by the colour in the calculation of the resistor's ohmage.
<i>r</i>	The red value of the colour in RGB space. Ranges between 0 and 255.
<i>g</i>	The green value of the colour in RGB space. Ranges between 0 and 255.
<i>b</i>	The blue value of the colour in RGB space. Ranges between 0 and 255.

7.12.3 Member Function Documentation

7.12.3.1 fit() [1/2]

```
static ResistorColour ohm.ValueIdentification.ResistorColour.fit (
    int r,
    int g,
    int b ) [static]
```

Function takes in a sampled colour from the images and attempts to fit it to the closest known colour a resistor can possess.

Parameters

<i>r</i>	The red colour value of the colour to be fit.
<i>g</i>	The green colour value of the colour to be fit.
<i>b</i>	The blue colour value of the colour to be fit.

Returns

The known colour that best represents the sampled colour.

7.12.3.2 fit() [2/2]

```
static ResistorColour ohm.ValueIdentification.ResistorColour.fit (
    double r,
    double g,
    double b ) [static]
```

Function takes in a sampled colour from the images and attempts to fit it to the closest known colour a resistor can possess.

Parameters

<i>r</i>	The red colour value of the colour to be fit.
<i>g</i>	The green colour value of the colour to be fit.
<i>b</i>	The blue colour value of the colour to be fit.

Returns

The known colour that best represents the sampled colour.

The documentation for this enum was generated from the following file:

- `java/ohm/ValueIdentification/ResistorColour.java`

7.13 ohm.ValueIdentification.ValueCalculator Class Reference

Object used to calculate the resistance of the resistor based on the mapped colours.

7.13.1 Detailed Description

Object used to calculate the resistance of the resistor based on the mapped colours.

The documentation for this class was generated from the following file:

- `java/ohm/ValueIdentification/ValueCalculator.java`

Index

- BandLocation, [12](#)
- boxSample
 - ohm::ImageProcessing::BandReader, [20](#)
- CameraInput, [14](#)
- ColourMapping, [10](#)
- componentwiseMean
 - ohm::ImageProcessing::BandReader, [21](#)
- diff
 - ohm::ImageProcessing::BandReader, [21](#)
- dist
 - ohm::ImageProcessing::BandReader, [21](#)
- fit
 - ohm::ValueIdentification::ResistorColour, [33](#), [34](#)
- getImage
 - ohm::Input::CameraInput, [25](#)
 - ohm::Input::ImageInput, [27](#)
 - ohm::Input::Input, [28](#)
- getMat
 - ohm::Input::CameraInput, [25](#)
 - ohm::Input::ImageInput, [27](#)
 - ohm::Input::Input, [28](#)
- ImageInput, [15](#)
 - ohm::Input::ImageInput, [26](#)
- ImageProcessing, [17](#)
- initialize
 - ohm::userinterface::OhmViewController, [30](#)
- Input, [17](#)
- mag
 - ohm::ImageProcessing::BandReader, [23](#)
- ohm.Helpers, [25](#)
- ohm.ImageProcessing.BandReader, [19](#)
- ohm.ImageProcessing.OhmImageProcessor, [29](#)
- ohm.ImageProcessing.ResistorBodyID, [32](#)
- ohm.Input.CameraInput, [24](#)
- ohm.Input.ImageInput, [26](#)
- ohm.Input.Input, [27](#)
- ohm.Ohm, [29](#)
- ohm.userinterface.LiveImageView, [28](#)
- ohm.userinterface.OhmViewController, [30](#)
- ohm.userinterface.ResistorAxisPickerView, [31](#)
- ohm.ValueIdentification.ResistorColour, [32](#)
- ohm.ValueIdentification.ValueCalculator, [34](#)
- ohm::ImageProcessing::BandReader
 - boxSample, [20](#)
 - componentwiseMean, [21](#)
 - diff, [21](#)
 - dist, [21](#)
 - mag, [23](#)
 - read, [23](#)
 - rollingAverageFilter, [23](#)
 - sample, [24](#)
- ohm::Input::CameraInput
 - getImage, [25](#)
 - getMat, [25](#)
- ohm::Input::ImageInput
 - getImage, [27](#)
 - getMat, [27](#)
 - ImageInput, [26](#)
- ohm::Input::Input
 - getImage, [28](#)
 - getMat, [28](#)
- ohm::ValueIdentification::ResistorColour
 - fit, [33](#), [34](#)
 - ResistorColour, [33](#)
- ohm::userinterface::OhmViewController
 - initialize, [30](#)
- ohm::userinterface::ResistorAxisPickerView
 - setImage, [31](#)
 - setListener, [31](#)
- read
 - ohm::ImageProcessing::BandReader, [23](#)
- ResistorColour
 - ohm::ValueIdentification::ResistorColour, [33](#)
- ResistorID, [13](#)
- rollingAverageFilter
 - ohm::ImageProcessing::BandReader, [23](#)
- sample
 - ohm::ImageProcessing::BandReader, [24](#)
- setImage
 - ohm::userinterface::ResistorAxisPickerView, [31](#)
- setListener
 - ohm::userinterface::ResistorAxisPickerView, [31](#)
- UserInterface, [9](#), [17](#)
- ValueCalculator, [11](#)
- ValueIdentification, [17](#)