

PhotoX - Photo Editor
Oliwia Mlonek

Generated by Doxygen 1.8.15

1 Hierarchical Index	1
1.1 Class Hierarchy	1
2 Class Index	3
2.1 Class List	3
3 File Index	7
3.1 File List	7
4 Class Documentation	11
4.1 _RGB Class Reference	11
4.1.1 Detailed Description	11
4.1.2 Constructor & Destructor Documentation	11
4.1.2.1 _RGB()	12
4.1.2.2 ~_RGB()	12
4.1.3 Member Function Documentation	12
4.1.3.1 change_space()	12
4.2 App Class Reference	12
4.2.1 Detailed Description	13
4.2.2 Member Function Documentation	13
4.2.2.1 OnInit()	13
4.3 BGR Class Reference	13
4.3.1 Detailed Description	14
4.3.2 Constructor & Destructor Documentation	14
4.3.2.1 BGR()	14
4.3.2.2 ~BGR()	14
4.3.3 Member Function Documentation	14
4.3.3.1 change_space()	14
4.4 BlackWhite Class Reference	15
4.4.1 Detailed Description	15
4.4.2 Constructor & Destructor Documentation	15
4.4.2.1 BlackWhite()	15
4.4.2.2 ~BlackWhite()	15
4.4.3 Member Function Documentation	15
4.4.3.1 change_color()	15
4.5 Brightness Class Reference	16
4.5.1 Detailed Description	16
4.5.2 Constructor & Destructor Documentation	16
4.5.2.1 Brightness()	16
4.5.2.2 ~Brightness()	17
4.5.3 Member Function Documentation	17
4.5.3.1 change_shadows()	17
4.6 ChangeColor Class Reference	17

4.6.1 Constructor & Destructor Documentation	18
4.6.1.1 ChangeColor()	18
4.6.1.2 ~ChangeColor()	18
4.6.2 Member Function Documentation	18
4.6.2.1 change_color()	18
4.7 ChangeEffect Class Reference	18
4.7.1 Detailed Description	19
4.7.2 Constructor & Destructor Documentation	19
4.7.2.1 ChangeEffect()	19
4.7.2.2 ~ChangeEffect()	19
4.7.3 Member Function Documentation	19
4.7.3.1 change_effect()	19
4.8 ChangeFlip Class Reference	20
4.8.1 Detailed Description	20
4.8.2 Constructor & Destructor Documentation	20
4.8.2.1 ChangeFlip()	21
4.8.2.2 ~ChangeFlip()	21
4.8.3 Member Function Documentation	21
4.8.3.1 change_flip()	21
4.9 ChangeHistogram Class Reference	21
4.9.1 Constructor & Destructor Documentation	22
4.9.1.1 ChangeHistogram()	22
4.9.1.2 ~ChangeHistogram()	22
4.9.2 Member Function Documentation	22
4.9.2.1 change_histogram()	22
4.10 ChangeRotation Class Reference	23
4.10.1 Detailed Description	23
4.10.2 Constructor & Destructor Documentation	23
4.10.2.1 ChangeRotation()	23
4.10.2.2 ~ChangeRotation()	23
4.10.3 Member Function Documentation	23
4.10.3.1 change_rotation()	23
4.11 ChangeShadows Class Reference	24
4.11.1 Detailed Description	24
4.11.2 Constructor & Destructor Documentation	24
4.11.2.1 ChangeShadows()	25
4.11.2.2 ~ChangeShadows()	25
4.11.3 Member Function Documentation	25
4.11.3.1 change_shadows()	25
4.12 ChangeSpace Class Reference	25
4.12.1 Detailed Description	26
4.12.2 Constructor & Destructor Documentation	26

4.12.2.1 ChangeSpace()	26
4.12.2.2 ~ChangeSpace()	26
4.12.3 Member Function Documentation	26
4.12.3.1 change_space()	26
4.13 ChangeUndo Class Reference	27
4.13.1 Detailed Description	27
4.13.2 Constructor & Destructor Documentation	27
4.13.2.1 ChangeUndo()	27
4.13.2.2 ~ChangeUndo()	28
4.13.3 Member Function Documentation	28
4.13.3.1 change_undo()	28
4.14 Dilate Class Reference	28
4.14.1 Detailed Description	29
4.14.2 Constructor & Destructor Documentation	29
4.14.2.1 Dilate()	29
4.14.2.2 ~Dilate()	29
4.14.3 Member Function Documentation	29
4.14.3.1 change_effect()	29
4.15 Editor Class Reference	30
4.15.1 Detailed Description	30
4.15.2 Constructor & Destructor Documentation	30
4.15.2.1 Editor()	30
4.15.2.2 ~Editor()	30
4.15.3 Member Function Documentation	30
4.15.3.1 CheckMatEquality()	30
4.15.3.2 SetPhoto()	31
4.15.3.3 which_color()	31
4.15.3.4 which_effect()	31
4.15.3.5 which_face()	33
4.15.3.6 which_flip()	33
4.15.3.7 which_histogram()	33
4.15.3.8 which_rotation()	34
4.15.3.9 which_shadow()	34
4.15.3.10 which_space()	35
4.15.3.11 which_undo()	35
4.16 Erode Class Reference	35
4.16.1 Detailed Description	36
4.16.2 Constructor & Destructor Documentation	36
4.16.2.1 Erode()	36
4.16.2.2 ~Erode()	36
4.16.3 Member Function Documentation	36
4.16.3.1 change_effect()	36

4.17 FaceDetection Class Reference	37
4.17.1 Detailed Description	37
4.17.2 Constructor & Destructor Documentation	37
4.17.2.1 FaceDetection()	38
4.17.2.2 ~FaceDetection()	38
4.17.3 Member Function Documentation	38
4.17.3.1 detectAndDraw()	38
4.17.3.2 find_face()	38
4.18 FlipX Class Reference	39
4.18.1 Detailed Description	39
4.18.2 Constructor & Destructor Documentation	39
4.18.2.1 FlipX()	39
4.18.2.2 ~FlipX()	40
4.18.3 Member Function Documentation	40
4.18.3.1 change_flip()	40
4.19 FlipXY Class Reference	40
4.19.1 Constructor & Destructor Documentation	40
4.19.1.1 FlipXY()	41
4.19.1.2 ~FlipXY()	41
4.19.2 Member Function Documentation	41
4.19.2.1 change_flip()	41
4.20 FlipY Class Reference	41
4.20.1 Detailed Description	42
4.20.2 Constructor & Destructor Documentation	42
4.20.2.1 FlipY()	42
4.20.2.2 ~FlipY()	42
4.20.3 Member Function Documentation	42
4.20.3.1 change_flip()	42
4.21 Gamma Class Reference	43
4.21.1 Detailed Description	43
4.21.2 Constructor & Destructor Documentation	43
4.21.2.1 Gamma()	43
4.21.2.2 ~Gamma()	43
4.21.3 Member Function Documentation	44
4.21.3.1 change_shadows()	44
4.22 Grey Class Reference	44
4.22.1 Detailed Description	45
4.22.2 Constructor & Destructor Documentation	45
4.22.2.1 Grey()	45
4.22.2.2 ~Grey()	45
4.22.3 Member Function Documentation	45
4.22.3.1 change_color()	45

4.23 HistogramEq Class Reference	46
4.23.1 Detailed Description	46
4.23.2 Constructor & Destructor Documentation	46
4.23.2.1 HistogramEq()	46
4.23.2.2 ~HistogramEq()	46
4.23.3 Member Function Documentation	46
4.23.3.1 change_histogram()	46
4.24 HLS Class Reference	47
4.24.1 Detailed Description	47
4.24.2 Constructor & Destructor Documentation	47
4.24.2.1 HLS()	47
4.24.2.2 ~HLS()	48
4.24.3 Member Function Documentation	48
4.24.3.1 change_space()	48
4.25 HSV Class Reference	48
4.25.1 Detailed Description	49
4.25.2 Constructor & Destructor Documentation	49
4.25.2.1 HSV()	49
4.25.2.2 ~HSV()	49
4.25.3 Member Function Documentation	49
4.25.3.1 change_space()	49
4.26 IsNull Class Reference	50
4.26.1 Detailed Description	50
4.26.2 Constructor & Destructor Documentation	50
4.26.2.1 IsNull()	50
4.26.2.2 ~IsNull()	51
4.26.3 Member Function Documentation	51
4.26.3.1 what()	51
4.27 IsOneElementOnly Class Reference	51
4.27.1 Detailed Description	52
4.27.2 Constructor & Destructor Documentation	52
4.27.2.1 IsOneElementOnly()	52
4.27.2.2 ~IsOneElementOnly()	52
4.27.3 Member Function Documentation	52
4.27.3.1 what()	52
4.28 MainWindow Class Reference	52
4.28.1 Detailed Description	53
4.28.2 Constructor & Destructor Documentation	53
4.28.2.1 MainWindow()	53
4.28.2.2 ~MainWindow()	53
4.29 NoFaceFound Class Reference	53
4.29.1 Detailed Description	54

4.29.2 Constructor & Destructor Documentation	54
4.29.2.1 NoFaceFound()	54
4.29.2.2 ~NoFaceFound()	54
4.29.3 Member Function Documentation	54
4.29.3.1 what()	54
4.30 Photo Class Reference	55
4.30.1 Detailed Description	55
4.30.2 Constructor & Destructor Documentation	55
4.30.2.1 Photo()	55
4.30.2.2 ~Photo()	55
4.30.3 Member Function Documentation	55
4.30.3.1 GetBrightness()	56
4.30.3.2 GetImageList()	56
4.30.3.3 GetSpaces()	56
4.30.3.4 LoadImageFromFile()	56
4.31 RotateLeft Class Reference	57
4.31.1 Detailed Description	57
4.31.2 Constructor & Destructor Documentation	57
4.31.2.1 RotateLeft()	57
4.31.2.2 ~RotateLeft()	57
4.31.3 Member Function Documentation	57
4.31.3.1 change_rotation()	57
4.32 RotateRight Class Reference	58
4.32.1 Detailed Description	58
4.32.2 Constructor & Destructor Documentation	58
4.32.2.1 RotateRight()	58
4.32.2.2 ~RotateRight()	59
4.32.3 Member Function Documentation	59
4.32.3.1 change_rotation()	59
4.33 SearchFace Class Reference	59
4.33.1 Detailed Description	60
4.33.2 Constructor & Destructor Documentation	60
4.33.2.1 SearchFace()	60
4.33.2.2 ~SearchFace()	60
4.33.3 Member Function Documentation	60
4.33.3.1 find_face()	60
4.34 Sepia Class Reference	61
4.34.1 Detailed Description	61
4.34.2 Constructor & Destructor Documentation	61
4.34.2.1 Sepia()	61
4.34.2.2 ~Sepia()	61
4.34.3 Member Function Documentation	61

4.34.3.1 change_color()	61
4.35 ShowHistogram Class Reference	62
4.35.1 Detailed Description	62
4.35.2 Constructor & Destructor Documentation	62
4.35.2.1 ShowHistogram()	62
4.35.2.2 ~ShowHistogram()	63
4.35.3 Member Function Documentation	63
4.35.3.1 change_histogram()	63
4.36 UndoAll Class Reference	63
4.36.1 Detailed Description	64
4.36.2 Constructor & Destructor Documentation	64
4.36.2.1 UndoAll()	64
4.36.2.2 ~UndoAll()	64
4.36.3 Member Function Documentation	64
4.36.3.1 change_undo()	64
4.37 UndoOne Class Reference	65
4.37.1 Detailed Description	65
4.37.2 Constructor & Destructor Documentation	65
4.37.2.1 UndoOne()	65
4.37.2.2 ~UndoOne()	65
4.37.3 Member Function Documentation	65
4.37.3.1 change_undo()	65
4.38 YCrCb Class Reference	66
4.38.1 Detailed Description	66
4.38.2 Constructor & Destructor Documentation	66
4.38.2.1 YCrCb()	66
4.38.2.2 ~YCrCb()	67
4.38.3 Member Function Documentation	67
4.38.3.1 change_space()	67
5 File Documentation	69
5.1 App.h File Reference	69
5.1.1 Detailed Description	69
5.2 BGR.h File Reference	69
5.2.1 Detailed Description	70
5.3 BlackWhite.h File Reference	70
5.3.1 Detailed Description	70
5.4 Brightness.h File Reference	70
5.4.1 Detailed Description	71
5.5 ChangeColor.h File Reference	71
5.5.1 Detailed Description	71
5.6 ChangeEffect.h File Reference	71

5.6.1 Detailed Description	72
5.7 ChangeFlip.h File Reference	72
5.7.1 Detailed Description	72
5.8 ChangeHistogram.h File Reference	72
5.8.1 Detailed Description	73
5.9 ChangeRotation.h File Reference	73
5.9.1 Detailed Description	73
5.10 ChangeShadows.h File Reference	73
5.10.1 Detailed Description	74
5.11 ChangeSpace.h File Reference	74
5.11.1 Detailed Description	74
5.12 ChangeUndo.h File Reference	74
5.12.1 Detailed Description	75
5.13 Dilate.h File Reference	75
5.13.1 Detailed Description	75
5.14 Editor.h File Reference	76
5.14.1 Detailed Description	76
5.15 Erode.h File Reference	77
5.15.1 Detailed Description	77
5.16 Exceptions.h File Reference	77
5.16.1 Detailed Description	77
5.17 FaceDetection.h File Reference	78
5.17.1 Detailed Description	78
5.18 FlipX.h File Reference	78
5.18.1 Detailed Description	78
5.19 FlipXY.h File Reference	79
5.19.1 Detailed Description	79
5.20 Gamma.h File Reference	79
5.20.1 Detailed Description	79
5.21 Grey.h File Reference	80
5.21.1 Detailed Description	80
5.22 HistogramEq.h File Reference	80
5.22.1 Detailed Description	80
5.23 HLS.h File Reference	81
5.23.1 Detailed Description	81
5.24 HSV.h File Reference	81
5.24.1 Detailed Description	81
5.25 MainWindow.h File Reference	82
5.25.1 Detailed Description	82
5.26 Photo.h File Reference	83
5.26.1 Detailed Description	83
5.27 RBG.h File Reference	83

5.27.1 Detailed Description	83
5.28 RotateLeft.h File Reference	84
5.28.1 Detailed Description	84
5.29 RotateRight.h File Reference	84
5.29.1 Detailed Description	84
5.30 SearchFace.h File Reference	85
5.30.1 Detailed Description	85
5.31 Sepia.h File Reference	85
5.31.1 Detailed Description	86
5.32 ShowHistogram.h File Reference	86
5.32.1 Detailed Description	86
5.33 UndoAll.h File Reference	86
5.33.1 Detailed Description	87
5.34 UndoOne.h File Reference	87
5.34.1 Detailed Description	87
5.35 YCrCb.h File Reference	87
5.35.1 Detailed Description	88

Chapter 1

Hierarchical Index

1.1 Class Hierarchy

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

ChangeColor	17
BlackWhite	15
Grey	44
Sepia	61
ChangeEffect	18
Dilate	28
Erode	35
ChangeFlip	20
FlipX	39
FlipXY	40
FlipY	41
ChangeHistogram	21
HistogramEq	46
ShowHistogram	62
ChangeRotation	23
RotateLeft	57
RotateRight	58
ChangeShadows	24
Brightness	16
Gamma	43
ChangeSpace	25
_RGB	11
BGR	13
HLS	47
HSV	48
YCrCb	66
ChangeUndo	27
UndoAll	63
UndoOne	65
Editor	30
exception	
IsNull	50
IsOneElementOnly	51

NoFaceFound	53
Photo	55
SearchFace	59
FaceDetection	37
wxApp	
App	12
wxFrame	
MainWindow	52

Chapter 2

Class Index

2.1 Class List

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

_RGB	Responsible for changing the color space of the image to RGB space. Implements operations while following the base strategy (SearchFace) interface	11
App	Responsible for application-wide settings for GUI-only apps	12
BGR	Responsible for changing the color space of the image to BGR space. Implements operations while following the base strategy (SearchFace) interface	13
BlackWhite	Responsible for applying black and white colour to the image. Implements operations while following the base strategy (ChangeColor) interface	15
Brightness	Responsible for brightening the image. Implements operations while following the base strategy (ChangeShadows) interface	16
ChangeColor	17
ChangeEffect	An abstract class responsible for applying chosen operation to the image. The class interface declares operations common to all supported versions	18
ChangeFlip	An abstract class responsible for flipping the image. The class interface declares operations common to all supported versions	20
ChangeHistogram	21
ChangeRotation	An abstract class responsible for rotating the given image. The class interface declares operations common to all supported versions	23
ChangeShadows	An abstract class responsible for applying chosen colours to the image. The class interface declares operations common to all supported versions	24
ChangeSpace	An abstract class responsible for changing the color space of the image. The class interface declares operations common to all supported versions	25
ChangeUndo	An abstract class responsible for getting rid of images with unwanted changes. The class interface declares operations common to all supported versions	27

Dilate	Responsible for applying the dilation effect to the image. Implements operations while following the base strategy (ChangeEffect) interface	28
Editor	Class is responsible of maintaining a reference to one of the concrete strategies and communicates with this object only via the strategy interface	30
Erode	Responsible for applying the erosion effect to the image. Implements operations while following the base strategy (ChangeEffect) interface	35
FaceDetection	Responsible for finding human faces in the given image and outlining the area in which they are located. Implements operations while following the base strategy (SearchFace) interface . . .	37
FlipX	Responsible for flipping the image horizontally. Implements operations while following the base strategy (ChangeFlip) interface	39
FlipXY	40
FlipY	Responsible for flipping the image vertically. Implements operations while following the base strategy (ChangeFlip) interface	41
Gamma	Responsible for changing the gamma correction of the image. Implements operations while following the base strategy (ChangeShadows) interface	43
Grey	Responsible for applying grey colour to the image. Implements operations while following the base strategy (ChangeColor) interface	44
HistogramEq	Responsible for histogram equalization. Implements operations while following the base strategy (ChangeFlip) interface	46
HLS	Responsible for changing the color space of the image to HLS space. Implements operations while following the base strategy (SearchFace) interface	47
HSV	Responsible for changing the color space of the image to HSV space. Implements operations while following the base strategy (SearchFace) interface	48
IsNull	Responsible for throwing a message when the used list is empty	50
IsOneElementOnly	Responsible for throwing a message when the used list has only one element	51
MainWindow	Creates and manages the project GUI	52
NoFaceFound	Responsible for throwing a message when face detector cannot recognize any face	53
Photo	Class is responsible for holding the image and the list with all with all versions of the image that was created, as well as stores data about the properties of the photo	55
RotateLeft	An abstract class responsible for rotating the given image 90 degrees to the left. Implements operations while following the base strategy (ChangeFlip) interface	57
RotateRight	An abstract class responsible for rotating the given image 90 degrees to the right. Implements operations while following the base strategy (ChangeFlip) interface	58
SearchFace	An abstract class responsible for finding human faces in the given image. The class interface declares operations common to all supported versions	59
Sepia	Responsible for applying sepia colour to the image. Implements operations while following the base strategy (ChangeColor) interface	61

[ShowHistogram](#)

Responsible for displaying histogram. Implements operations while following the base strategy ([ChangeFlip](#)) interface 62

[UndoAll](#)

Responsible for getting rid of all but the last one images from the list. Implements operations while following the base strategy ([SearchFace](#)) interface 63

[UndoOne](#)

Responsible for getting rid of the last created image with unwanted change. Implements operations while following the base strategy ([SearchFace](#)) interface 65

[YCrCb](#)

Responsible for changing the color space of the image to [YCrCb](#) space. Implements operations while following the base strategy ([SearchFace](#)) interface 66

Chapter 3

File Index

3.1 File List

Here is a list of all documented files with brief descriptions:

App.h	The file contains class which represents the application itself	69
BGR.h	The file contains class which manipulates the image represented by an n-dimensional dense numerical multi-channel array (Mat) by changing the color space of the image to BGR	69
BlackWhite.h	The file contains class which manipulates the image represented by an n-dimensional dense numerical multi-channel array (Mat) by changing the colours of the image to black and white only	70
Brightness.h	The file contains class which manipulates the image represented by an n-dimensional dense numerical multi-channel array (Mat) by changing the lighting intensity of the image	70
ChangeColor.h	The file contains class which manipulates the image represented by an n-dimensional dense numerical multi-channel array (Mat) by changing the colours of the image	71
ChangeEffect.h	The file contains class which manipulates the image represented by an n-dimensional dense numerical multi-channel array (Mat) by applying morphology operations	71
ChangeFlip.h	The file contains class which manipulates the image represented by an n-dimensional dense numerical multi-channel array (Mat) by flipping it in three ways	72
ChangeHistogram.h	The file contains class which manipulates the image represented by an n-dimensional dense numerical multi-channel array (Mat) by executing some operations related to image histogram	72
ChangeRotation.h	The file contains class which manipulates the image represented by an n-dimensional dense numerical multi-channel array (Mat) by rotating the image	73
ChangeShadows.h	The file contains class which manipulates the image represented by an n-dimensional dense numerical multi-channel array (Mat) by changing the lighting intensity of the image	73
ChangeSpace.h	The file contains class which manipulates the image represented by an n-dimensional dense numerical multi-channel array (Mat) by changing the color space of the image	74
ChangeUndo.h	The file contains class which manipulates the list of pointers to the images represented by an n-dimensional dense numerical multi-channel array (Mat) by deleting unwanted nodes	74

Dilate.h	The file contains class which manipulates the image represented by an n-dimensional dense numerical multi-channel array (Mat) by applying dilation effect on it	75
Editor.h	The file contains class which maintains a reference to one of the concrete strategies and communicates with this object only via the strategy interface	76
Erode.h	The file contains class which manipulates the image represented by an n-dimensional dense numerical multi-channel array (Mat) by applying erosion effect on it	77
Exceptions.h	The file contains three classes which are thrown as exceptions when the need arises	77
FaceDetection.h	The file contains class which is designed to detect all human faces that are in a given image (represented by an n-dimensional dense numerical multi-channel array- Mat)	78
FlipX.h	The file contains class which manipulates the image represented by an n-dimensional dense numerical multi-channel array (Mat) by flipping it horizontally	78
FlipXY.h	The file contains class which manipulates the image represented by an n-dimensional dense numerical multi-channel array (Mat) by flipping it horizontally and vertically at once	79
FlipY.h	??
Gamma.h	The file contains class which manipulates the image represented by an n-dimensional dense numerical multi-channel array (Mat) by changing the lighting of the image using gamma correction	79
Grey.h	The file contains class which manipulates the image represented by an n-dimensional dense numerical multi-channel array (Mat) by changing the colours of the image to greyscale	80
HistogramEq.h	The file contains class which manipulates the image represented by an n-dimensional dense numerical multi-channel array (Mat) by executing histogram equalization in the image	80
HLS.h	The file contains class which manipulates the image represented by an n-dimensional dense numerical multi-channel array (Mat) by changing the color space of the image to HLS	81
HSV.h	The file contains class which manipulates the image represented by an n-dimensional dense numerical multi-channel array (Mat) by changing the color space of the image to HSV	81
MainWindow.h	The file contains class which is responsible for creating and managing the entire graphical user interface and calling appropriate actions related to image processing after actions on the part of the user in the GUI	82
Photo.h	The file contains class which manages the loaded image during the procces of edition	83
RBG.h	The file contains class which manipulates the image represented by an n-dimensional dense numerical multi-channel array (Mat) by changing the color space of the image to _RGB	83
RotateLeft.h	The file contains class which manipulates the image represented by an n-dimensional dense numerical multi-channel array (Mat) by rotating the image to the left	84
RotateRight.h	The file contains class which manipulates the image represented by an n-dimensional dense numerical multi-channel array (Mat) by rotating the image to the left	84
SearchFace.h	The file contains class which is designed to detect all human faces that are in a given image (represented by an n-dimensional dense numerical multi-channel array- Mat)	85
Sepia.h	The file contains class which manipulates the image represented by an n-dimensional dense numerical multi-channel array (Mat) by changing the colours of the image to black and white only	85

[ShowHistogram.h](#)

The file contains class which manipulates the image represented by an n-dimensional dense numerical multi-channel array (Mat) by plotting and displaying image histogram 86

[UndoAll.h](#)

The file contains class which manipulates the list of pointers to the images represented by an n-dimensional dense numerical multi-channel array (Mat) by deleting all but the last one elements from the list 86

[UndoOne.h](#)

The file contains class which manipulates the list of pointers to the images represented by an n-dimensional dense numerical multi-channel array (Mat) by deleting unwanted (first) node . . . 87

[YCrCb.h](#)

The file contains class which manipulates the image represented by an n-dimensional dense numerical multi-channel array (Mat) by changing the color space of the image to [YCrCb](#) 87

Chapter 4

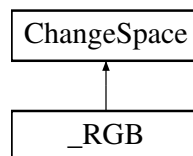
Class Documentation

4.1 `_RGB` Class Reference

responsible for changing the color space of the image to RGB space. Implements operations while following the base strategy ([SearchFace](#)) interface.

```
#include <RBG.h>
```

Inheritance diagram for `_RGB`:



Public Member Functions

- [_RGB](#) ()
- [~_RGB](#) ()
- void [change_space](#) (std::list< std::shared_ptr< Mat >> &lista, std::vector< int > Vector)

4.1.1 Detailed Description

responsible for changing the color space of the image to RGB space. Implements operations while following the base strategy ([SearchFace](#)) interface.

4.1.2 Constructor & Destructor Documentation

4.1.2.1 `_RGB()`

```
_RGB::_RGB ( )
```

Create the `_RGB` object.

4.1.2.2 `~_RGB()`

```
_RGB::~~_RGB ( )
```

Destroys objects.

4.1.3 Member Function Documentation

4.1.3.1 `change_space()`

```
void _RGB::change_space (
    std::list< std::shared_ptr< Mat >> & lista,
    std::vector< int > Vector ) [virtual]
```

Function performs conversion from `BGR` color space to `RGB`.

Parameters

<i>lista</i>	is a list containing pointers to previously loaded images in the form of Mats.
<i>wektor</i>	is a vector of integers which holds the information about the previous changes of color space made to the given image.

Reimplemented from `ChangeSpace`.

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

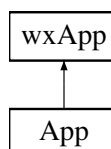
- `RBG.h`
- `RBG.cpp`

4.2 App Class Reference

Responsible for application-wide settings for GUI-only apps.

```
#include <App.h>
```

Inheritance diagram for `App`:



Public Member Functions

- virtual bool [OnInit](#) ()

4.2.1 Detailed Description

Responsible for application-wide settings for GUI-only apps.

4.2.2 Member Function Documentation

4.2.2.1 OnInit()

```
bool App::OnInit ( ) [virtual]
```

Function checks if wxOK = 1 what is crucial for creating main window, initializing image handlers and launching the app.

Returns

bool value of wxOK.

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

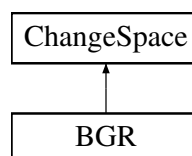
- [App.h](#)
- [App.cpp](#)

4.3 BGR Class Reference

responsible for changing the color space of the image to [BGR](#) space. Implements operations while following the base strategy ([SearchFace](#)) interface.

```
#include <BGR.h>
```

Inheritance diagram for BGR:



Public Member Functions

- [BGR](#) ()
- [~BGR](#) ()
- void [change_space](#) (std::list< std::shared_ptr< Mat >> &lista, std::vector< int > Vector)

4.3.1 Detailed Description

responsible for changing the color space of the image to [BGR](#) space. Implements operations while following the base strategy ([SearchFace](#)) interface.

4.3.2 Constructor & Destructor Documentation

4.3.2.1 BGR()

```
BGR::BGR ( )
```

Create the [BGR](#) object.

4.3.2.2 ~BGR()

```
BGR::~~BGR ( )
```

Destroys objects.

4.3.3 Member Function Documentation

4.3.3.1 change_space()

```
void BGR::change_space (
    std::list< std::shared_ptr< Mat >> & lista,
    std::vector< int > Vector ) [virtual]
```

Function performs coversion from current color space to [BGR](#).

Parameters

<i>lista</i>	is a list containing pointers to previously loaded images in the form of Mats.
<i>wektor</i>	is a vector of integers which holds the information about the previous changes of color space made to the given image.

Reimplemented from [ChangeSpace](#).

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

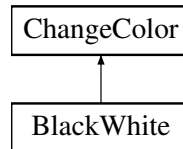
- [BGR.h](#)
- [BGR.cpp](#)

4.4 BlackWhite Class Reference

Responsible for applying black and white colour to the image. Implements operations while following the base strategy ([ChangeColor](#)) interface.

```
#include <BlackWhite.h>
```

Inheritance diagram for BlackWhite:



Public Member Functions

- [BlackWhite](#) ()
- [~BlackWhite](#) ()
- void [change_color](#) (std::list< std::shared_ptr< Mat >> &lista)

4.4.1 Detailed Description

Responsible for applying black and white colour to the image. Implements operations while following the base strategy ([ChangeColor](#)) interface.

4.4.2 Constructor & Destructor Documentation

4.4.2.1 BlackWhite()

```
BlackWhite::BlackWhite ( )
```

Create the [BlackWhite](#) object.

4.4.2.2 ~BlackWhite()

```
BlackWhite::~~BlackWhite ( )
```

Destroys object.

4.4.3 Member Function Documentation

4.4.3.1 change_color()

```
void BlackWhite::change_color (
    std::list< std::shared_ptr< Mat >> & lista ) [virtual]
```

Function applies the black and white colours to the first Mat found on the list .

Parameters

<i>lista</i>	is a list containing pointers to previously loaded images in the form of Mats.
--------------	--------------------------------------------------------------------------------

Reimplemented from [ChangeColor](#).

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

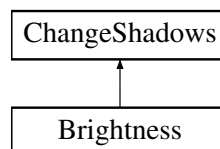
- [BlackWhite.h](#)
- BlackWhite.cpp

4.5 Brightness Class Reference

Responsible for brightening the image. Implements operations while following the base strategy ([ChangeShadows](#)) interface.

```
#include <Brightness.h>
```

Inheritance diagram for Brightness:



Public Member Functions

- [Brightness](#) ()
- [~Brightness](#) ()
- void [change_shadows](#) (std::list< std::shared_ptr< Mat >> &lista, double FromSlider, std::vector< int > wektor)

4.5.1 Detailed Description

Responsible for brightening the image. Implements operations while following the base strategy ([ChangeShadows](#)) interface.

4.5.2 Constructor & Destructor Documentation

4.5.2.1 Brightness()

```
Brightness::Brightness ( )
```

Create the [Brightness](#) object.

4.5.2.2 ~Brightness()

```
Brightness::~~Brightness ( )
```

Destroys object.

4.5.3 Member Function Documentation

4.5.3.1 change_shadows()

```
void Brightness::change_shadows (
    std::list< std::shared_ptr< Mat >> & lista,
    double FromSlider,
    std::vector< int > wektor ) [virtual]
```

Function brightens or darkens the first Mat found on the list with the selected intensity.

Parameters

<i>lista</i>	is a list containing pointers to previously loaded images in the form of Mats.
<i>FromSlider</i>	is a double variable that stores the value set by the user on the light intensity slider.
<i>wektor</i>	is a vector of integers which holds previously set values from slider.

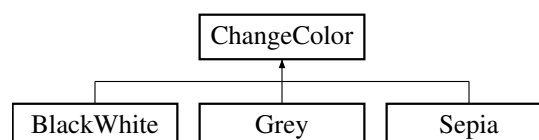
Reimplemented from [ChangeShadows](#).

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

- [Brightness.h](#)
- [Brightness.cpp](#)

4.6 ChangeColor Class Reference

Inheritance diagram for ChangeColor:



Public Member Functions

- [ChangeColor](#) ()
- [~ChangeColor](#) ()
- virtual void [change_color](#) (std::list< std::shared_ptr< Mat >> &lista)

4.6.1 Constructor & Destructor Documentation

4.6.1.1 ChangeColor()

```
ChangeColor::ChangeColor ( )
```

Create the [ChangeColor](#) object.

4.6.1.2 ~ChangeColor()

```
ChangeColor::~~ChangeColor ( )
```

Destroys object.

4.6.2 Member Function Documentation

4.6.2.1 change_color()

```
void ChangeColor::change_color (
    std::list< std::shared_ptr< Mat >> & lista ) [virtual]
```

Function applies the chosen colours to the first Mat found on the list.

Parameters

<i>lista</i>	is a list containing pointers to previously loaded images in the form of Mats.
--------------	--------------------------------------------------------------------------------

Reimplemented in [BlackWhite](#), [Grey](#), and [Sepia](#).

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

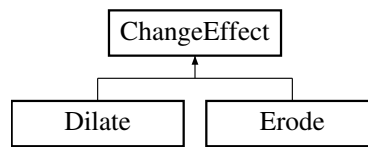
- [ChangeColor.h](#)
- [ChangeColor.cpp](#)

4.7 ChangeEffect Class Reference

An abstract class responsible for applying chosen operation to the image. The class interface declares operations common to all supported versions.

```
#include <ChangeEffect.h>
```

Inheritance diagram for ChangeEffect:



Public Member Functions

- [ChangeEffect](#) ()
- [~ChangeEffect](#) ()
- virtual void [change_effect](#) (std::list< std::shared_ptr< Mat >> &lista, int morph)

4.7.1 Detailed Description

An abstract class responsible for applying chosen operation to the image. The class interface declares operations common to all supported versions.

4.7.2 Constructor & Destructor Documentation

4.7.2.1 ChangeEffect()

```
ChangeEffect::ChangeEffect ( )
```

Create the [ChangeEffect](#) object.

4.7.2.2 ~ChangeEffect()

```
ChangeEffect::~~ChangeEffect ( )
```

Destroys object.

4.7.3 Member Function Documentation

4.7.3.1 change_effect()

```
void ChangeEffect::change_effect (
    std::list< std::shared_ptr< Mat >> & lista,
    int morph ) [virtual]
```

Function applies the chosen operation to the first Mat found on the list.

Parameters

<i>lista</i>	is a list containing pointers to previously loaded images in the form of Mats.
<i>morph</i>	informs which of the three shapes was chosen for our kernel (a fixed size array of numerical coefficients) which is responsible for carrying out the operations..

Reimplemented in [Erode](#), and [Dilate](#).

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

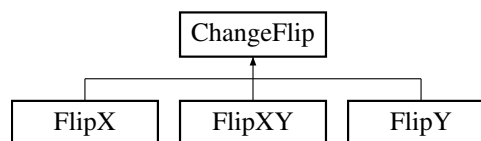
- [ChangeEffect.h](#)
- [ChangeEffect.cpp](#)

4.8 ChangeFlip Class Reference

An abstract class responsible for flipping the image. The class interface declares operations common to all supported versions.

```
#include <ChangeFlip.h>
```

Inheritance diagram for ChangeFlip:



Public Member Functions

- [ChangeFlip](#) ()
- [~ChangeFlip](#) ()
- virtual void [change_flip](#) (std::list< std::shared_ptr< Mat >> &lista)

4.8.1 Detailed Description

An abstract class responsible for flipping the image. The class interface declares operations common to all supported versions.

An abstract class responsible for operations related to the image histogram. The class interface declares operations common to all supported versions.

4.8.2 Constructor & Destructor Documentation

4.8.2.1 ChangeFlip()

```
ChangeFlip::ChangeFlip ( )
```

Create the [FaceDetection](#) object.

4.8.2.2 ~ChangeFlip()

```
ChangeFlip::~~ChangeFlip ( )
```

Destroys objects.

4.8.3 Member Function Documentation

4.8.3.1 change_flip()

```
void ChangeFlip::change_flip (
    std::list< std::shared_ptr< Mat >> & lista ) [virtual]
```

Function flips the image.

Parameters

<i>lista</i>	is a list containing pointers to previously loaded images in the form of Mats.
--------------	--------------------------------------------------------------------------------

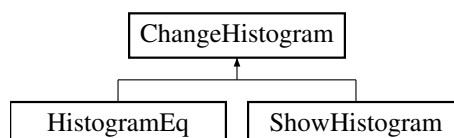
Reimplemented in [FlipY](#), [FlipX](#), and [FlipXY](#).

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

- [ChangeFlip.h](#)
- [ChangeFlip.cpp](#)

4.9 ChangeHistogram Class Reference

Inheritance diagram for ChangeHistogram:



Public Member Functions

- [ChangeHistogram](#) ()
- [~ChangeHistogram](#) ()
- virtual void [change_histogram](#) (std::list< std::shared_ptr< Mat >> &lista)

4.9.1 Constructor & Destructor Documentation

4.9.1.1 ChangeHistogram()

`ChangeHistogram::ChangeHistogram ()`

Create the [ChangeHistogram](#) object.

4.9.1.2 ~ChangeHistogram()

`ChangeHistogram::~~ChangeHistogram ()`

Destroys objects.

4.9.2 Member Function Documentation

4.9.2.1 change_histogram()

```
void ChangeHistogram::change_histogram (
    std::list< std::shared_ptr< Mat >> & lista ) [virtual]
```

Function performs histogram-related operations.

Parameters

<i>lista</i>	is a list containing pointers to previously loaded images in the form of Mats.
--------------	--------------------------------------------------------------------------------

Reimplemented in [HistogramEq](#), and [ShowHistogram](#).

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

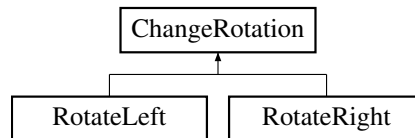
- [ChangeHistogram.h](#)
- [ChangeHistogram.cpp](#)

4.10 ChangeRotation Class Reference

An abstract class responsible for rotating the given image. The class interface declares operations common to all supported versions.

```
#include <ChangeRotation.h>
```

Inheritance diagram for ChangeRotation:



Public Member Functions

- [ChangeRotation](#) ()
- [~ChangeRotation](#) ()
- virtual void [change_rotation](#) (std::list< std::shared_ptr< Mat >> &lista)

4.10.1 Detailed Description

An abstract class responsible for rotating the given image. The class interface declares operations common to all supported versions.

4.10.2 Constructor & Destructor Documentation

4.10.2.1 ChangeRotation()

```
ChangeRotation::ChangeRotation ( )
```

Create the [ChangeHistogram](#) object.

4.10.2.2 ~ChangeRotation()

```
ChangeRotation::~~ChangeRotation ( )
```

Destroys objects.

4.10.3 Member Function Documentation

4.10.3.1 change_rotation()

```
void ChangeRotation::change_rotation (
    std::list< std::shared_ptr< Mat >> & lista ) [virtual]
```

Function performs image rotation.

Parameters

<i>lista</i>	is a list containing pointers to previously loaded images in the form of Mats.
--------------	--------------------------------------------------------------------------------

Reimplemented in [RotateRight](#), and [RotateLeft](#).

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

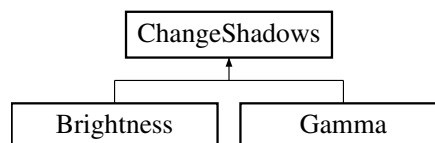
- [ChangeRotation.h](#)
- [ChangeRotation.cpp](#)

4.11 ChangeShadows Class Reference

An abstract clas responsible for applying chosen colours to the image. The class interface declares operations common to all supported versions.

```
#include <ChangeColor.h>
```

Inheritance diagram for ChangeShadows:



Public Member Functions

- [ChangeShadows](#) ()
- [~ChangeShadows](#) ()
- virtual void [change_shadows](#) (std::list< std::shared_ptr< Mat >> &lista, double FromSlider, std::vector< int > wektor)

4.11.1 Detailed Description

An abstract clas responsible for applying chosen colours to the image. The class interface declares operations common to all supported versions.

An abstract clas responsible for applying chosen brightness corrections to the image. The class interface declares operations common to all supported versions.

4.11.2 Constructor & Destructor Documentation

4.11.2.1 ChangeShadows()

```
ChangeShadows::ChangeShadows ( )
```

Create the [ChangeShadows](#) object.

4.11.2.2 ~ChangeShadows()

```
ChangeShadows::~~ChangeShadows ( )
```

Destroys object.

4.11.3 Member Function Documentation

4.11.3.1 change_shadows()

```
void ChangeShadows::change_shadows (
    std::list< std::shared_ptr< Mat >> & lista,
    double FromSlider,
    std::vector< int > wektor ) [virtual]
```

Function applies the chosen brightness corrections to the first Mat found on the list with the selected intensity.*

Parameters

<i>lista</i>	is a list containing pointers to previously loaded images in the form of Mats.
<i>FromSlider</i>	is a double variable that stores the value set by the user on the light intensity slider.
<i>wektor</i>	is a vector of integers which holds previously set values from slider.

Reimplemented in [Brightness](#), and [Gamma](#).

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

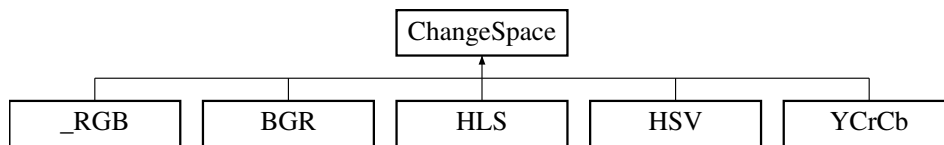
- [ChangeShadows.h](#)
- [ChangeShadows.cpp](#)

4.12 ChangeSpace Class Reference

An abstract class responsible for changing the color space of the image. The class interface declares operations common to all supported versions.

```
#include <ChangeSpace.h>
```

Inheritance diagram for ChangeSpace:



Public Member Functions

- [ChangeSpace](#) ()
- [~ChangeSpace](#) ()
- virtual void [change_space](#) (std::list< std::shared_ptr< Mat >> &lista, std::vector< int > Vector)

4.12.1 Detailed Description

An abstract class responsible for changing the color space of the image. The class interface declares operations common to all supported versions.

4.12.2 Constructor & Destructor Documentation

4.12.2.1 ChangeSpace()

```
ChangeSpace::ChangeSpace ( )
```

Create the [ChangeSpace](#) object.

4.12.2.2 ~ChangeSpace()

```
ChangeSpace::~~ChangeSpace ( )
```

Destroys objects.

4.12.3 Member Function Documentation

4.12.3.1 change_space()

```
void ChangeSpace::change_space (
    std::list< std::shared_ptr< Mat >> & lista,
    std::vector< int > Vector ) [virtual]
```

Function performs coversion from current color space to another.

Parameters

<i>lista</i>	is a list containing pointers to previously loaded images in the form of Mats.
<i>wektor</i>	is a vector of integers which holds the information about the previous changes of color space made to the given image.

Reimplemented in [_RGB](#), [BGR](#), [HSV](#), [HLS](#), and [YCrCb](#).

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

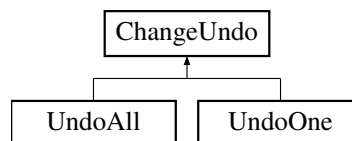
- [ChangeSpace.h](#)
- [ChangeSpace.cpp](#)

4.13 ChangeUndo Class Reference

An abstract class responsible for getting rid of images with unwanted changes. The class interface declares operations common to all supported versions.

```
#include <ChangeUndo.h>
```

Inheritance diagram for ChangeUndo:



Public Member Functions

- [ChangeUndo](#) ()
- [~ChangeUndo](#) ()
- virtual void [change_undo](#) (std::list< std::shared_ptr< Mat >> &lista)

4.13.1 Detailed Description

An abstract class responsible for getting rid of images with unwanted changes. The class interface declares operations common to all supported versions.

4.13.2 Constructor & Destructor Documentation

4.13.2.1 ChangeUndo()

```
ChangeUndo::ChangeUndo ( )
```

Create the [ChangeUndo](#) object.

4.13.2.2 ~ChangeUndo()

ChangeUndo::~~ChangeUndo ()

Destroys objects.

4.13.3 Member Function Documentation

4.13.3.1 change_undo()

```
void ChangeUndo::change_undo (
    std::list< std::shared_ptr< Mat >> & lista ) [virtual]
```

Function erases unwanted images from a list.

Parameters

<i>lista</i>	is a list containing pointers to previously loaded images in the form of Mats.
--------------	--------------------------------------------------------------------------------

Reimplemented in [UndoAll](#), and [UndoOne](#).

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

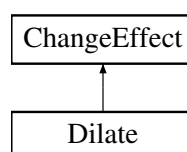
- [ChangeUndo.h](#)
- [ChangeUndo.cpp](#)

4.14 Dilate Class Reference

Responsible for applying the dilation effect to the image. Implements operations while following the base strategy ([ChangeEffect](#)) interface.

```
#include <Dilate.h>
```

Inheritance diagram for Dilate:



Public Member Functions

- [Dilate](#) ()
- [~Dilate](#) ()
- void [change_effect](#) (std::list< std::shared_ptr< Mat >> &lista, int morph)

4.14.1 Detailed Description

Responsible for applying the dilation effect to the image. Implements operations while following the base strategy ([ChangeEffect](#)) interface.

4.14.2 Constructor & Destructor Documentation

4.14.2.1 Dilate()

```
Dilate::Dilate ( )
```

Create the [ChangeEffect](#) object.

4.14.2.2 ~Dilate()

```
Dilate::~~Dilate ( )
```

Destroys object.

4.14.3 Member Function Documentation

4.14.3.1 change_effect()

```
void Dilate::change_effect (
    std::list< std::shared_ptr< Mat >> & lista,
    int morph ) [virtual]
```

Function applies the dilation effect to the first Mat found on the list.

Parameters

<i>lista</i>	is a list containing pointers to previously loaded images in the form of Mats.
<i>morph</i>	informs which of the three shapes was chosen for our kernel (a fixed size array of numerical coefficients) which is responsible for carrying out the operation.

Reimplemented from [ChangeEffect](#).

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

- [Dilate.h](#)
- [Dilate.cpp](#)

4.15 Editor Class Reference

Class is responsible of maintaining a reference to one of the concrete strategies and communicates with this object only via the strategy interface.

```
#include <Editor.h>
```

Public Member Functions

- [Editor](#) ()
- [~Editor](#) ()
- void [SetPhoto](#) (std::shared_ptr< [Photo](#) > X)
- void [which_color](#) (int which)
- void [which_flip](#) (int which)
- void [which_undo](#) (int which)
- void [which_shadow](#) (int which, double HowMuch, std::vector< int > wektor)
- void [which_face](#) ()
- void [which_rotation](#) (int which)
- void [which_histogram](#) (int which)
- void [which_space](#) (int which, std::vector< int > Vector)
- void [which_effect](#) (int which, int morph)
- bool [CheckMatEquality](#) (const cv::Mat Mat1, const cv::Mat Mat2)

4.15.1 Detailed Description

Class is responsible of maintaining a reference to one of the concrete strategies and communicates with this object only via the strategy interface.

4.15.2 Constructor & Destructor Documentation

4.15.2.1 Editor()

```
Editor::Editor ( )
```

Create the [Editor](#) object.

4.15.2.2 ~Editor()

```
Editor::~Editor ( )
```

Destroys objects.

4.15.3 Member Function Documentation

4.15.3.1 CheckMatEquality()

```
bool Editor::CheckMatEquality (
    const cv::Mat Mat1,
    const cv::Mat Mat2 )
```

The function checks if two given Mat arrays are equal (i.e. if they are exactly the same).

Parameters

<i>Mat1</i>	is a first Mat variable to compare with.
<i>Mat2</i>	is a second Mat variable to compare with.

Returns

bool value, true if they are the same, false if they are not.

4.15.3.2 SetPhoto()

```
void Editor::SetPhoto (
    std::shared_ptr< Photo > X )
```

Function assigns the pointer with the [Photo](#) object to the [Editor](#) attribute.

Parameters

<i>X</i>	is a shared pointer to a Photo object.
----------	--------------------------------------------------------

4.15.3.3 which_color()

```
void Editor::which_color (
    int which )
```

Function assigns strategies to the [ChangeColor](#) pointer depending on the user's choice, then calls the appropriate function to perform operations on the photo.

Parameters

<i>which</i>	is an integer variable informing about the user choice.
--------------	---------------------------------------------------------

Exceptions

<i>if</i>	the list of Mats is empty.
-----------	----------------------------

4.15.3.4 which_effect()

```
void Editor::which_effect (
    int which,
    int morph )
```

Function assigns strategies to the [ChangeEffect](#) pointer depending on the user's choice, then calls the appropriate function to perform operations on the photo.

Parameters

<i>which</i>	is an integer variable informing about the user choice.
<i>morph</i>	informs which of the three shapes was chosen for our kernel (a fixed size array of numerical coefficients) which is responsible for carrying out the operations..

Exceptions

<i>IsNull</i>	if the list of images is empty.
---------------	---------------------------------

4.15.3.5 `which_face()`

```
void Editor::which_face ( )
```

Function assigns strategies to the [SearchFace](#) pointer depending on the user's choice, then calls the appropriate function to perform operations on the photo.

Exceptions

<i>IsNull</i>	if the list of images is empty.
<i>NoFaceFound</i>	if no face was detected.

4.15.3.6 `which_flip()`

```
void Editor::which_flip (
    int which )
```

Function assigns strategies to the [ChangeFlip](#) pointer depending on the user's choice, then calls the appropriate function to perform operations on the photo.

Parameters

<i>which</i>	is an integer variable informing about the user choice.
--------------	---------------------------------------------------------

Exceptions

<i>IsNull</i>	if the list of images is empty.
---------------	---------------------------------

4.15.3.7 `which_histogram()`

```
void Editor::which_histogram (
    int which )
```

Function assigns strategies to the [ChangeHistogram](#) pointer depending on the user's choice, then calls the appropriate function to perform operations on the photo.

Parameters

<i>which</i>	is an integer variable informing about the user choice.
--------------	---------------------------------------------------------

Exceptions

<i>IsNull</i>	if the list of images is empty.
---------------	---------------------------------

4.15.3.8 `which_rotation()`

```
void Editor::which_rotation (
    int which )
```

Function assigns strategies to the [ChangeRotation](#) pointer depending on the user's choice, then calls the appropriate function to perform operations on the photo.

Parameters

<i>which</i>	is an integer variable informing about the user choice.
--------------	---------------------------------------------------------

Exceptions

<i>IsNull</i>	if the list of images is empty.
---------------	---------------------------------

4.15.3.9 `which_shadow()`

```
void Editor::which_shadow (
    int which,
    double HowMuch,
    std::vector< int > wektor )
```

Function assigns strategies to the [ChangeShadows](#) pointer depending on the user's choice, then calls the appropriate function to perform operations on the photo.

Parameters

<i>which</i>	is an integer variable informing about the user choice.
<i>HowMuch</i>	is a double variable that stores information about the intensity of the operation to be performed.
<i>wektor</i>	is a vector of integers which holds previously set values from slider.

Exceptions

<i>IsNull</i>	if the list of images is empty.
---------------	---------------------------------

4.15.3.10 `which_space()`

```
void Editor::which_space (
    int which,
    std::vector< int > Vector )
```

Function assigns strategies to the [ChangeHistogram](#) pointer depending on the user's choice, then calls the appropriate function to perform operations on the photo.

Parameters

<i>which</i>	is an integer variable informing about the user choice.
<i>Vector</i>	is a vector of integers which holds the information about the previous changes of color space made to the given image.

Exceptions

<i>IsNull</i>	if the list of images is empty.
-------------------------------	---------------------------------

4.15.3.11 `which_undo()`

```
void Editor::which_undo (
    int which )
```

Function assigns strategies to the [ChangeUndo](#) pointer depending on the user's choice, then calls the appropriate function to perform operations on the photo.

Parameters

<i>which</i>	is an integer variable informing about the user choice.
--------------	---------------------------------------------------------

Exceptions

<i>IsNull</i>	if the list of images is empty.
<i>IsOneElementOnly</i>	if the list of images has only one element.

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

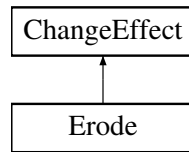
- [Editor.h](#)
- [Editor.cpp](#)

4.16 Erode Class Reference

Responsible for applying the erosion effect to the image. Implements operations while following the base strategy ([ChangeEffect](#)) interface.

```
#include <Erode.h>
```

Inheritance diagram for Erode:



Public Member Functions

- [Erode](#) ()
- [~Erode](#) ()
- void [change_effect](#) (std::list< std::shared_ptr< Mat >> &lista, int morph)

4.16.1 Detailed Description

Responsible for applying the erosion effect to the image. Implements operations while following the base strategy ([ChangeEffect](#)) interface.

4.16.2 Constructor & Destructor Documentation

4.16.2.1 Erode()

```
Erode::Erode ( )
```

Create the [Erode](#) object.

4.16.2.2 ~Erode()

```
Erode::~~Erode ( )
```

Destroys object.

4.16.3 Member Function Documentation

4.16.3.1 change_effect()

```
void Erode::change_effect (
    std::list< std::shared_ptr< Mat >> & lista,
    int morph ) [virtual]
```

Function applies the erosion effect to the first Mat found on the list.

Parameters

<i>lista</i>	is a list containing pointers to previously loaded images in the form of Mats.
<i>morph</i>	informs which of the three shapes was chosen for our kernel (a fixed size array of numerical coefficients) which is responsible for carrying out the operation.

Reimplemented from [ChangeEffect](#).

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

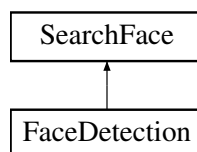
- [Erode.h](#)
- [Erode.cpp](#)

4.17 FaceDetection Class Reference

Responsible for finding human faces in the given image and outlining the area in which they are located. Implements operations while following the base strategy ([SearchFace](#)) interface.

```
#include <FaceDetection.h>
```

Inheritance diagram for FaceDetection:



Public Member Functions

- [FaceDetection](#) ()
- [~FaceDetection](#) ()
- void [find_face](#) (std::list< std::shared_ptr< Mat >> &lista)
- Mat [detectAndDraw](#) (Mat &img, CascadeClassifier &cascade, double scale)

4.17.1 Detailed Description

Responsible for finding human faces in the given image and outlining the area in which they are located. Implements operations while following the base strategy ([SearchFace](#)) interface.

4.17.2 Constructor & Destructor Documentation

4.17.2.1 FaceDetection()

```
FaceDetection::FaceDetection ( )
```

Create the [FaceDetection](#) object.

4.17.2.2 ~FaceDetection()

```
FaceDetection::~~FaceDetection ( )
```

Destroys objects.

4.17.3 Member Function Documentation

4.17.3.1 detectAndDraw()

```
Mat FaceDetection::detectAndDraw (
    Mat & img,
    CascadeClassifier & cascade,
    double scale )
```

Function that actually finds a face in a given image and contours it.

Parameters

<i>img</i>	is a Mat representing the chosen image.
<i>cascade</i>	is a chosen cascade classifier.

Returns

A Mat of changed image.

4.17.3.2 find_face()

```
void FaceDetection::find_face (
    std::list< std::shared_ptr< Mat >> & lista ) [virtual]
```

Function prepares necessary variables and files to perform face detection operation.

Parameters

<i>lista</i>	is a list containing pointers to previously loaded images in the form of Mats.
--------------	--------------------------------------------------------------------------------

Reimplemented from [SearchFace](#).

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

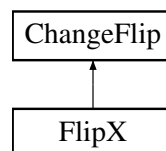
- [FaceDetection.h](#)
- [FaceDetection.cpp](#)

4.18 FlipX Class Reference

Responsible for flipping the image horizontally. Implements operations while following the base strategy ([ChangeFlip](#)) interface.

```
#include <FlipX.h>
```

Inheritance diagram for FlipX:



Public Member Functions

- [FlipX](#) ()
- [~FlipX](#) ()
- void [change_flip](#) (std::list< std::shared_ptr< Mat >> &lista)

4.18.1 Detailed Description

Responsible for flipping the image horizontally. Implements operations while following the base strategy ([ChangeFlip](#)) interface.

Responsible for flipping the image horizontally and vertically. Implements operations while following the base strategy ([ChangeFlip](#)) interface.

4.18.2 Constructor & Destructor Documentation

4.18.2.1 FlipX()

```
FlipX::FlipX ( )
```

Create the [FaceDetection](#) object.

4.18.2.2 ~FlipX()

```
FlipX::~~FlipX ( )
```

Destroys objects.

4.18.3 Member Function Documentation

4.18.3.1 change_flip()

```
void FlipX::change_flip (
    std::list< std::shared_ptr< Mat >> & lista ) [virtual]
```

Function flips the image horizontally.

Parameters

<i>lista</i>	is a list containing pointers to previously loaded images in the form of Mats.
--------------	--------------------------------------------------------------------------------

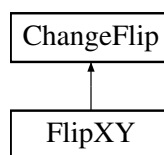
Reimplemented from [ChangeFlip](#).

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

- [FlipX.h](#)
- [FlipX.cpp](#)

4.19 FlipXY Class Reference

Inheritance diagram for FlipXY:



Public Member Functions

- [FlipXY \(\)](#)
- [~FlipXY \(\)](#)
- void [change_flip](#) (std::list< std::shared_ptr< Mat >> &lista)

4.19.1 Constructor & Destructor Documentation

4.19.1.1 FlipXY()

```
FlipXY::FlipXY ( )
```

Create the [FaceDetection](#) object.

4.19.1.2 ~FlipXY()

```
FlipXY::~~FlipXY ( )
```

Destroys objects.

4.19.2 Member Function Documentation

4.19.2.1 change_flip()

```
void FlipXY::change_flip (
    std::list< std::shared_ptr< Mat >> & lista ) [virtual]
```

Function flips the image horizontally and vertically.

Parameters

<i>lista</i>	is a list containing pointers to previously loaded images in the form of Mats.
--------------	--------------------------------------------------------------------------------

Reimplemented from [ChangeFlip](#).

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

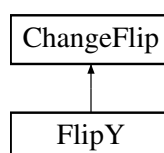
- [FlipXY.h](#)
- [FlipXY.cpp](#)

4.20 FlipY Class Reference

Responsible for flipping the image vertically. Implements operations while following the base strategy ([ChangeFlip](#)) interface.

```
#include <FlipY.h>
```

Inheritance diagram for FlipY:



Public Member Functions

- [FlipY](#) ()
- [~FlipY](#) ()
- void [change_flip](#) (std::list< std::shared_ptr< Mat >> &lista)

4.20.1 Detailed Description

Responsible for flipping the image vertically. Implements operations while following the base strategy ([ChangeFlip](#)) interface.

4.20.2 Constructor & Destructor Documentation

4.20.2.1 FlipY()

```
FlipY::FlipY ( )
```

Create the [FlipY](#) object.

4.20.2.2 ~FlipY()

```
FlipY::~~FlipY ( )
```

Destroys objects.

4.20.3 Member Function Documentation

4.20.3.1 change_flip()

```
void FlipY::change_flip (
    std::list< std::shared_ptr< Mat >> & lista ) [virtual]
```

Function flips the image vertically.

Parameters

<i>lista</i>	is a list containing pointers to previously loaded images in the form of Mats.
--------------	--------------------------------------------------------------------------------

Reimplemented from [ChangeFlip](#).

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

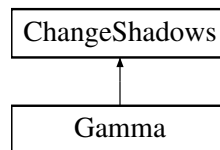
- [FlipY.h](#)
- [FlipY.cpp](#)

4.21 Gamma Class Reference

Responsible for changing the gamma correction of the image. Implements operations while following the base strategy ([ChangeShadows](#)) interface.

```
#include <Gamma.h>
```

Inheritance diagram for Gamma:



Public Member Functions

- [Gamma](#) ()
- [~Gamma](#) ()
- void [change_shadows](#) (std::list< std::shared_ptr< Mat >> &lista, double FromSlider, std::vector< int > wektor)
- bool **matIsEqual** (const cv::Mat Mat1, const cv::Mat Mat2)

4.21.1 Detailed Description

Responsible for changing the gamma correction of the image. Implements operations while following the base strategy ([ChangeShadows](#)) interface.

4.21.2 Constructor & Destructor Documentation

4.21.2.1 Gamma()

```
Gamma::Gamma ( )
```

Create the [Brightness](#) object.

4.21.2.2 ~Gamma()

```
Gamma::~Gamma ( )
```

Destroys object.

4.21.3 Member Function Documentation

4.21.3.1 change_shadows()

```
void Gamma::change_shadows (
    std::list< std::shared_ptr< Mat >> & lista,
    double FromSlider,
    std::vector< int > wektor ) [virtual]
```

Function brightens or darkens the first Mat found on the list with the selected intensity by adjusting gamma correction.

Parameters

<i>lista</i>	is a list containing pointers to previously loaded images in the form of Mats.
<i>FromSlider</i>	is a double variable that stores the intensity value set by the user.
<i>wektor</i>	is a vector of integers which holds previously set values from slider.

Reimplemented from [ChangeShadows](#).

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

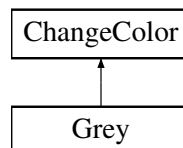
- [Gamma.h](#)
- [Gamma.cpp](#)

4.22 Grey Class Reference

Responsible for applying grey colour to the image. Implements operations while following the base strategy ([ChangeColor](#)) interface.

```
#include <Grey.h>
```

Inheritance diagram for Grey:



Public Member Functions

- [Grey \(\)](#)
- [~Grey \(\)](#)
- void [change_color](#) (std::list< std::shared_ptr< Mat >> &lista)

4.22.1 Detailed Description

Responsible for applying grey colour to the image. Implements operations while following the base strategy ([ChangeColor](#)) interface.

4.22.2 Constructor & Destructor Documentation

4.22.2.1 Grey()

```
Grey::Grey ( )
```

Create the [Grey](#) object.

4.22.2.2 ~Grey()

```
Grey::~~Grey ( )
```

Destroys object.

4.22.3 Member Function Documentation

4.22.3.1 change_color()

```
void Grey::change_color (
    std::list< std::shared_ptr< Mat >> & lista ) [virtual]
```

Function applies the grey colour to the first Mat found on the list .

Parameters

<i>lista</i>	is a list containing pointers to previously loaded images in the form of Mats.
--------------	--------------------------------------------------------------------------------

Reimplemented from [ChangeColor](#).

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

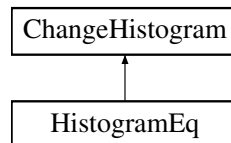
- [Grey.h](#)
- Grey.cpp

4.23 HistogramEq Class Reference

Responsible for histogram equalization. Implements operations while following the base strategy ([ChangeFlip](#)) interface.

```
#include <HistogramEq.h>
```

Inheritance diagram for HistogramEq:



Public Member Functions

- [HistogramEq](#) ()
- [~HistogramEq](#) ()
- void [change_histogram](#) (std::list< std::shared_ptr< Mat >> &lista)

4.23.1 Detailed Description

Responsible for histogram equalization. Implements operations while following the base strategy ([ChangeFlip](#)) interface.

4.23.2 Constructor & Destructor Documentation

4.23.2.1 HistogramEq()

```
HistogramEq::HistogramEq ( )
```

Create the [HistogramEq](#) object.

4.23.2.2 ~HistogramEq()

```
HistogramEq::~~HistogramEq ( )
```

Destroys objects.

4.23.3 Member Function Documentation

4.23.3.1 change_histogram()

```
void HistogramEq::change_histogram (
    std::list< std::shared_ptr< Mat >> & lista ) [virtual]
```

Function performs histogram equalization.

Parameters

<i>lista</i>	is a list containing pointers to previously loaded images in the form of Mats.
--------------	--------------------------------------------------------------------------------

Reimplemented from [ChangeHistogram](#).

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

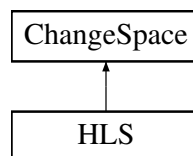
- [HistogramEq.h](#)
- [HistogramEq.cpp](#)

4.24 HLS Class Reference

responsible for changing the color space of the image to [HLS](#) space. Implements operations while following the base strategy ([SearchFace](#)) interface.

```
#include <HLS.h>
```

Inheritance diagram for HLS:



Public Member Functions

- [HLS](#) ()
- [~HLS](#) ()
- void [change_space](#) (std::list< std::shared_ptr< Mat >> &lista, std::vector< int > Vector)

4.24.1 Detailed Description

responsible for changing the color space of the image to [HLS](#) space. Implements operations while following the base strategy ([SearchFace](#)) interface.

4.24.2 Constructor & Destructor Documentation

4.24.2.1 HLS()

```
HLS::HLS ( )
```

Create the [HLS](#) object.

4.24.2.2 ~HLS()

```
HLS::~HLS ( )
```

Destroys objects.

4.24.3 Member Function Documentation

4.24.3.1 change_space()

```
void HLS::change_space (
    std::list< std::shared_ptr< Mat >> & lista,
    std::vector< int > Vector ) [virtual]
```

Function performs coversion from [BGR](#) color space to [HLS](#).

Parameters

<i>lista</i>	is a list containing pointers to previously loaded images in the form of Mats.
<i>wektor</i>	is a vector of integers which holds the information about the previous changes of color space made to the given image.

Reimplemented from [ChangeSpace](#).

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

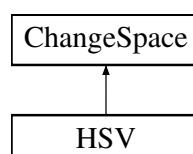
- [HLS.h](#)
- [HLS.cpp](#)

4.25 HSV Class Reference

responsible for changing the color space of the image to [HSV](#) space. Implements operations while following the base strategy ([SearchFace](#)) interface.

```
#include <HSV.h>
```

Inheritance diagram for HSV:



Public Member Functions

- [HSV](#) ()
- [~HSV](#) ()
- void [change_space](#) (std::list< std::shared_ptr< Mat >> &lista, std::vector< int > Vector)

4.25.1 Detailed Description

responsible for changing the color space of the image to [HSV](#) space. Implements operations while following the base strategy ([SearchFace](#)) interface.

4.25.2 Constructor & Destructor Documentation

4.25.2.1 HSV()

```
HSV::HSV ( )
```

Create the [HSV](#) object.

4.25.2.2 ~HSV()

```
HSV::~HSV ( )
```

Destroys objects.

4.25.3 Member Function Documentation

4.25.3.1 change_space()

```
void HSV::change_space (
    std::list< std::shared_ptr< Mat >> & lista,
    std::vector< int > Vector ) [virtual]
```

Function performs coversion from [BGR](#) color space to [HSV](#).

Parameters

<i>lista</i>	is a list containing pointers to previously loaded images in the form of Mats.
<i>wektor</i>	is a vector of integers which holds the information about the previous changes of color space made to the given image.

Reimplemented from [ChangeSpace](#).

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

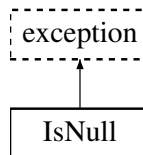
- [HSV.h](#)
- [HSV.cpp](#)

4.26 IsNull Class Reference

Responsible for throwing a message when the used list is empty.

```
#include <Exceptions.h>
```

Inheritance diagram for IsNull:



Public Member Functions

- [IsNull](#) ()
- virtual [~IsNull](#) () throw ()
- virtual const char * [what](#) () const throw ()

Protected Attributes

- std::string [error_message](#) = "You did not upload any photo!!!"
Error message.

4.26.1 Detailed Description

Responsible for throwing a message when the used list is empty.

4.26.2 Constructor & Destructor Documentation

4.26.2.1 IsNull()

```
IsNull::IsNull ( ) [inline], [explicit]
```

Create the [IsNull](#) object.

4.26.2.2 ~IsNull()

```
virtual IsNull::~~IsNull ( ) throw ( )    [inline], [virtual]
```

Destructor. Virtual to allow for subclassing.

4.26.3 Member Function Documentation

4.26.3.1 what()

```
virtual const char* IsNull::what ( ) const throw ( )    [inline], [virtual]
```

Returns a pointer to the (constant) error description.

Returns

A pointer to a const char*.

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

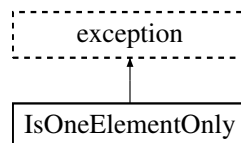
- [Exceptions.h](#)

4.27 IsOneElementOnly Class Reference

Responsible for throwing a message when the used list has only one element.

```
#include <Exceptions.h>
```

Inheritance diagram for IsOneElementOnly:



Public Member Functions

- [IsOneElementOnly](#) ()
- virtual [~IsOneElementOnly](#) () throw ()
- virtual const char * [what](#) () const throw ()

Protected Attributes

- std::string [error_message](#) = "Nothing more to undo!"
Error message.

4.27.1 Detailed Description

Responsible for throwing a message when the used list has only one element.

4.27.2 Constructor & Destructor Documentation

4.27.2.1 IsOneElementOnly()

```
IsOneElementOnly::IsOneElementOnly ( ) [inline], [explicit]
```

Create the [IsOneElementOnly](#) object.

4.27.2.2 ~IsOneElementOnly()

```
virtual IsOneElementOnly::~~IsOneElementOnly ( ) throw ( ) [inline], [virtual]
```

Destructor. Virtual to allow for subclassing.

4.27.3 Member Function Documentation

4.27.3.1 what()

```
virtual const char* IsOneElementOnly::what ( ) const throw ( ) [inline], [virtual]
```

Returns a pointer to the (constant) error description.

Returns

A pointer to a const char*.

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

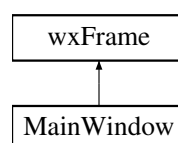
- [Exceptions.h](#)

4.28 MainWindow Class Reference

Creates and manages the project GUI.

```
#include <MainWindow.h>
```

Inheritance diagram for MainWindow:



Public Member Functions

- [MainWindow](#) (wxWindow *parent, wxWindowID id=-1)
- virtual [~MainWindow](#) ()

4.28.1 Detailed Description

Creates and manages the project GUI.

4.28.2 Constructor & Destructor Documentation

4.28.2.1 MainWindow()

```
MainWindow::MainWindow (
    wxWindow * parent,
    wxWindowID id = -1 )
```

Constructor. Create the [MainWindow](#) object.

Parameters

<i>parent</i>	The window parent.
<i>id</i>	The window identifier.

4.28.2.2 ~MainWindow()

```
MainWindow::~MainWindow ( ) [virtual]
```

Destructor. Destroys all child windows and menu bar if present.

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

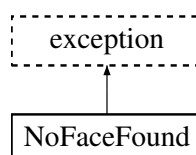
- [MainWindow.h](#)
- [MainWindow.cpp](#)

4.29 NoFaceFound Class Reference

Responsible for throwing a message when face detector cannot recognize any face.

```
#include <Exceptions.h>
```

Inheritance diagram for NoFaceFound:



Public Member Functions

- [NoFaceFound](#) ()
- virtual [~NoFaceFound](#) () throw ()
- virtual const char * [what](#) () const throw ()

Protected Attributes

- std::string [error_message](#) = "No face was detected!"
Error message.

4.29.1 Detailed Description

Responsible for throwing a message when face detector cannot recognize any face.

4.29.2 Constructor & Destructor Documentation

4.29.2.1 NoFaceFound()

```
NoFaceFound::NoFaceFound ( ) [inline], [explicit]
```

Create the [IsOneElementOnly](#) object.

4.29.2.2 ~NoFaceFound()

```
virtual NoFaceFound::~~NoFaceFound ( ) throw ( ) [inline], [virtual]
```

Destructor. Virtual to allow for subclassing.

4.29.3 Member Function Documentation

4.29.3.1 what()

```
virtual const char* NoFaceFound::what ( ) const throw ( ) [inline], [virtual]
```

Returns a pointer to the (constant) error description.

Returns

A pointer to a const char*.

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

- [Exceptions.h](#)

4.30 Photo Class Reference

Class is responsible for holding the image and the list with all with all versions of the image that was created, as well as stores data about the properties of the photo.

```
#include <Photo.h>
```

Public Member Functions

- [Photo](#) ()
- [~Photo](#) ()
- `std::list< std::shared_ptr< Mat > > & GetImageList ()`
- `std::vector< int > & GetBrightness ()`
- `std::vector< int > & GetSpaces ()`
- `void LoadImageFromFile (std::string FileName)`

4.30.1 Detailed Description

Class is responsible for holding the image and the list with all with all versions of the image that was created, as well as stores data about the properties of the photo.

4.30.2 Constructor & Destructor Documentation

4.30.2.1 [Photo](#)()

```
Photo::Photo ( )
```

Create the [Photo](#) object.

4.30.2.2 [~Photo](#)()

```
Photo::~~Photo ( )
```

Destroys object.

4.30.3 Member Function Documentation

4.30.3.1 GetBrightness()

```
std::vector< int > & Photo::GetBrightness ( )
```

Function allows the access to the private [Photo](#)'s vector which stores the previous brightness values.

Returns

the vector.

4.30.3.2 GetImageList()

```
std::list< std::shared_ptr< Mat > > & Photo::GetImageList ( )
```

Function allows the access to the private [Photo](#)'s list of pointers to Mats.

Returns

the list pf pointers to Mats.

4.30.3.3 GetSpaces()

```
std::vector< int > & Photo::GetSpaces ( )
```

Function allows the access to the private [Photo](#)'s vector which stores the previously chosen color spaces.

Returns

the vector.

4.30.3.4 LoadImageFromFile()

```
void Photo::LoadImageFromFile (
    std::string FileName )
```

Function loads and opens chosen by the user image to a Mat array ans sets initial values in list and vectors.

Parameters

<i>FileName</i>	is a string variable which holds name of the image.
-----------------	-----------------------------------------------------

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

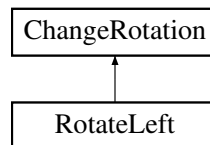
- [Photo.h](#)
- [Photo.cpp](#)

4.31 RotateLeft Class Reference

An abstract class responsible for rotating the given image 90 degrees to the left. Implements operations while following the base strategy ([ChangeFlip](#)) interface.

```
#include <RotateLeft.h>
```

Inheritance diagram for RotateLeft:



Public Member Functions

- [RotateLeft](#) ()
- [~RotateLeft](#) ()
- void [change_rotation](#) (std::list< std::shared_ptr< Mat >> &lista)

4.31.1 Detailed Description

An abstract class responsible for rotating the given image 90 degrees to the left. Implements operations while following the base strategy ([ChangeFlip](#)) interface.

4.31.2 Constructor & Destructor Documentation

4.31.2.1 RotateLeft()

```
RotateLeft::RotateLeft ( )
```

Create the [RotateLeft](#) object.

4.31.2.2 ~RotateLeft()

```
RotateLeft::~~RotateLeft ( )
```

Destroys objects.

4.31.3 Member Function Documentation

4.31.3.1 change_rotation()

```
void RotateLeft::change_rotation (
    std::list< std::shared_ptr< Mat >> & lista ) [virtual]
```

Function performs image rotation to the left.

Parameters

<i>lista</i>	is a list containing pointers to previously loaded images in the form of Mats.
--------------	--------------------------------------------------------------------------------

Reimplemented from [ChangeRotation](#).

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

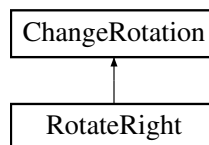
- [RotateLeft.h](#)
- [RotateLeft.cpp](#)

4.32 RotateRight Class Reference

An abstract class responsible for rotating the given image 90 degrees to the right. Implements operations while following the base strategy ([ChangeFlip](#)) interface.

```
#include <RotateRight.h>
```

Inheritance diagram for RotateRight:



Public Member Functions

- [RotateRight](#) ()
- [~RotateRight](#) ()
- void [change_rotation](#) (std::list< std::shared_ptr< Mat >> &lista)

4.32.1 Detailed Description

An abstract class responsible for rotating the given image 90 degrees to the right. Implements operations while following the base strategy ([ChangeFlip](#)) interface.

4.32.2 Constructor & Destructor Documentation

4.32.2.1 RotateRight()

```
RotateRight::RotateRight ( )
```

Create the [RotateLeft](#) object.

4.32.2.2 ~RotateRight()

```
RotateRight::~~RotateRight ( )
```

Destroys objects.

4.32.3 Member Function Documentation

4.32.3.1 change_rotation()

```
void RotateRight::change_rotation (
    std::list< std::shared_ptr< Mat >> & lista ) [virtual]
```

Function performs image rotation to the left.

Parameters

<i>lista</i>	is a list containing pointers to previously loaded images in the form of Mats.
--------------	--------------------------------------------------------------------------------

Reimplemented from [ChangeRotation](#).

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

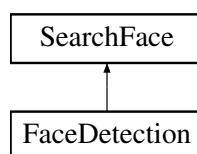
- [RotateRight.h](#)
- [RotateRight.cpp](#)

4.33 SearchFace Class Reference

An abstract clas responsible for finding human faces in the given image. The class interface declares operations common to all supported versions.

```
#include <SearchFace.h>
```

Inheritance diagram for SearchFace:



Public Member Functions

- [SearchFace](#) ()
- [~SearchFace](#) ()
- virtual void [find_face](#) (std::list< std::shared_ptr< Mat >> &lista)

4.33.1 Detailed Description

An abstract clas responsible for finding human faces in the given image. The class interface declares operations common to all supported versions.

4.33.2 Constructor & Destructor Documentation

4.33.2.1 SearchFace()

```
SearchFace::SearchFace ( )
```

Create the [SearchFace](#) object.

4.33.2.2 ~SearchFace()

```
SearchFace::~~SearchFace ( )
```

Destroys objects.

4.33.3 Member Function Documentation

4.33.3.1 find_face()

```
void SearchFace::find_face (
    std::list< std::shared_ptr< Mat >> & lista ) [virtual]
```

Function detects faces on the given image.

Parameters

<i>lista</i>	is a list containing pointers to previously loaded images in the form of Mats.
--------------	--------------------------------------------------------------------------------

Reimplemented in [FaceDetection](#).

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

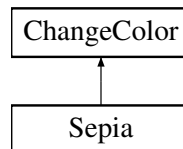
- [SearchFace.h](#)
- [SearchFace.cpp](#)

4.34 Sepia Class Reference

Responsible for applying sepia colour to the image. Implements operations while following the base strategy ([ChangeColor](#)) interface.

```
#include <Sepia.h>
```

Inheritance diagram for Sepia:



Public Member Functions

- [Sepia](#) ()
- [~Sepia](#) ()
- void [change_color](#) (std::list< std::shared_ptr< Mat >> &lista)

4.34.1 Detailed Description

Responsible for applying sepia colour to the image. Implements operations while following the base strategy ([ChangeColor](#)) interface.

4.34.2 Constructor & Destructor Documentation

4.34.2.1 Sepia()

```
Sepia::Sepia ( )
```

Create the [Sepia](#) object.

4.34.2.2 ~Sepia()

```
Sepia::~~Sepia ( )
```

Destroys object.

4.34.3 Member Function Documentation

4.34.3.1 change_color()

```
void Sepia::change_color (
    std::list< std::shared_ptr< Mat >> & lista ) [virtual]
```

Function applies the sepia colour to the first Mat found on the list .

Parameters

<i>lista</i>	is a list containing pointers to previously loaded images in the form of Mats.
--------------	--------------------------------------------------------------------------------

Reimplemented from [ChangeColor](#).

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

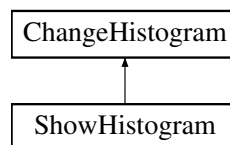
- [Sepia.h](#)
- [Sepia.cpp](#)

4.35 ShowHistogram Class Reference

Responsible for displaying histogram. Implements operations while following the base strategy ([ChangeFlip](#)) interface.

```
#include <ShowHistogram.h>
```

Inheritance diagram for ShowHistogram:



Public Member Functions

- [ShowHistogram](#) ()
- [~ShowHistogram](#) ()
- void [change_histogram](#) (std::list< std::shared_ptr< Mat >> &lista)

4.35.1 Detailed Description

Responsible for displaying histogram. Implements operations while following the base strategy ([ChangeFlip](#)) interface.

4.35.2 Constructor & Destructor Documentation

4.35.2.1 ShowHistogram()

```
ShowHistogram::ShowHistogram ( )
```

Create the [ChangeHistogram](#) object.

4.35.2.2 ~ShowHistogram()

```
ShowHistogram::~~ShowHistogram ( )
```

Destroys objects.

4.35.3 Member Function Documentation

4.35.3.1 change_histogram()

```
void ShowHistogram::change_histogram (
    std::list< std::shared_ptr< Mat >> & lista ) [virtual]
```

Function plots and displays histogram in separate window.

Parameters

<i>lista</i>	is a list containing pointers to previously loaded images in the form of Mats.
--------------	--------------------------------------------------------------------------------

Reimplemented from [ChangeHistogram](#).

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

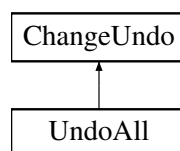
- [ShowHistogram.h](#)
- [ShowHistogram.cpp](#)

4.36 UndoAll Class Reference

responsible for getting rid of all but the last one images from the list. Implements operations while following the base strategy ([SearchFace](#)) interface.

```
#include <UndoAll.h>
```

Inheritance diagram for UndoAll:



Public Member Functions

- [UndoAll](#) ()
- [~UndoAll](#) ()
- void [change_undo](#) (std::list< std::shared_ptr< Mat >> &lista)

4.36.1 Detailed Description

responsible for getting rid of all but the last one images from the list. Implements operations while following the base strategy ([SearchFace](#)) interface.

4.36.2 Constructor & Destructor Documentation

4.36.2.1 UndoAll()

```
UndoAll::UndoAll ( )
```

Create the [UndoAll](#) object.

4.36.2.2 ~UndoAll()

```
UndoAll::~UndoAll ( )
```

Destroys objects.

4.36.3 Member Function Documentation

4.36.3.1 change_undo()

```
void UndoAll::change_undo (
    std::list< std::shared_ptr< Mat >> & lista ) [virtual]
```

Function erases all but the last one images from the list, so the image can be displayed as it was before making any change.

Parameters

<i>lista</i>	is a list containing pointers to previously loaded images in the form of Mats.
--------------	--------------------------------------------------------------------------------

Reimplemented from [ChangeUndo](#).

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

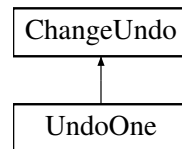
- [UndoAll.h](#)
- [UndoAll.cpp](#)

4.37 UndoOne Class Reference

responsible for getting rid of the last created image with unwanted change. Implements operations while following the base strategy ([SearchFace](#)) interface.

```
#include <UndoOne.h>
```

Inheritance diagram for UndoOne:



Public Member Functions

- [UndoOne](#) ()
- [~UndoOne](#) ()
- void [change_undo](#) (std::list< std::shared_ptr< Mat >> &lista)

4.37.1 Detailed Description

responsible for getting rid of the last created image with unwanted change. Implements operations while following the base strategy ([SearchFace](#)) interface.

4.37.2 Constructor & Destructor Documentation

4.37.2.1 UndoOne()

```
UndoOne::UndoOne ( )
```

Create the [UndoOne](#) object.

4.37.2.2 ~UndoOne()

```
UndoOne::~~UndoOne ( )
```

Destroys objects.

4.37.3 Member Function Documentation

4.37.3.1 change_undo()

```
void UndoOne::change_undo (
    std::list< std::shared_ptr< Mat >> & lista ) [virtual]
```

Function erases unwanted image from a beginning of a list, so the image before the last change will be displayed again.

Parameters

<i>lista</i>	is a list containing pointers to previously loaded images in the form of Mats.
--------------	--------------------------------------------------------------------------------

Reimplemented from [ChangeUndo](#).

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

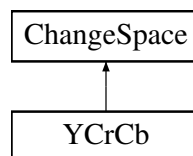
- [UndoOne.h](#)
- [UndoOne.cpp](#)

4.38 YCrCb Class Reference

responsible for changing the color space of the image to [YCrCb](#) space. Implements operations while following the base strategy ([SearchFace](#)) interface.

```
#include <YCrCb.h>
```

Inheritance diagram for YCrCb:



Public Member Functions

- [YCrCb](#) ()
- [~YCrCb](#) ()
- void [change_space](#) (std::list< std::shared_ptr< Mat >> &lista, std::vector< int > Vector)

4.38.1 Detailed Description

responsible for changing the color space of the image to [YCrCb](#) space. Implements operations while following the base strategy ([SearchFace](#)) interface.

4.38.2 Constructor & Destructor Documentation

4.38.2.1 YCrCb()

```
YCrCb::YCrCb ( )
```

Create the [YCrCb](#) object.

4.38.2.2 ~YCrCb()

```
YCrCb::~YCrCb ( )
```

Destroys objects.

4.38.3 Member Function Documentation

4.38.3.1 change_space()

```
void YCrCb::change_space (
    std::list< std::shared_ptr< Mat >> & lista,
    std::vector< int > Vector ) [virtual]
```

Function performs conversion from [BGR](#) color space to [YCrCb](#).

Parameters

<i>lista</i>	is a list containing pointers to previously loaded images in the form of Mats.
<i>wektor</i>	is a vector of integers which holds the information about the previous changes of color space made to the given image.

Reimplemented from [ChangeSpace](#).

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

- [YCrCb.h](#)
- [YCrCb.cpp](#)

Chapter 5

File Documentation

5.1 App.h File Reference

The file contains class which represents the application itself .

```
#include "MainWindow.h"
```

Classes

- class [App](#)
Responsible for application-wide settings for GUI-only apps.

5.1.1 Detailed Description

The file contains class which represents the application itself .

Author

Oliwia Mlonek

5.2 BGR.h File Reference

The file contains class which manipulates the image represented by an n-dimensional dense numerical multi-channel array (Mat) by changing the color space of the image to [BGR](#).

```
#include <opencv2/opencv.hpp>
#include <opencv2/imgproc.hpp>
#include "ChangeSpace.h"
```

Classes

- class [BGR](#)

responsible for changing the color space of the image to [BGR](#) space. Implements operations while following the base strategy ([SearchFace](#)) interface.

5.2.1 Detailed Description

The file contains class which manipulates the image represented by an n-dimensional dense numerical multi-channel array (Mat) by changing the color space of the image to [BGR](#).

Author

Oliwia Mlonek

5.3 BlackWhite.h File Reference

The file contains class which manipulates the image represented by an n-dimensional dense numerical multi-channel array (Mat) by changing the colours of the image to black and white only.

```
#include <opencv2/opencv.hpp>
#include <opencv2/imgproc.hpp>
#include "ChangeColor.h"
```

Classes

- class [BlackWhite](#)

Responsible for applying black and white colour to the image. Implements operations while following the base strategy ([ChangeColor](#)) interface.

5.3.1 Detailed Description

The file contains class which manipulates the image represented by an n-dimensional dense numerical multi-channel array (Mat) by changing the colours of the image to black and white only.

Author

Oliwia Mlonek

5.4 Brightness.h File Reference

The file contains class which manipulates the image represented by an n-dimensional dense numerical multi-channel array (Mat) by changing the lighting intensity of the image.

```
#include "ChangeShadows.h"
#include "opencv2/imgcodecs.hpp"
#include "opencv2/highgui.hpp"
```

Classes

- class [Brightness](#)

Responsible for brightening the image. Implements operations while following the base strategy ([ChangeShadows](#)) interface.

5.4.1 Detailed Description

The file contains class which manipulates the image represented by an n-dimensional dense numerical multi-channel array (Mat) by changing the lighting intensity of the image.

Author

Oliwia Mlonek

5.5 ChangeColor.h File Reference

The file contains class which manipulates the image represented by an n-dimensional dense numerical multi-channel array (Mat) by changing the colours of the image.

```
#include <memory>
#include <opencv2/opencv.hpp>
```

Classes

- class [ChangeColor](#)

5.5.1 Detailed Description

The file contains class which manipulates the image represented by an n-dimensional dense numerical multi-channel array (Mat) by changing the colours of the image.

Author

Oliwia Mlonek

5.6 ChangeEffect.h File Reference

The file contains class which manipulates the image represented by an n-dimensional dense numerical multi-channel array (Mat) by applying morphology operations.

```
#include <opencv2/opencv.hpp>
#include <unordered_map>
```

Classes

- class [ChangeEffect](#)

An abstract class responsible for applying chosen operation to the image. The class interface declares operations common to all supported versions.

5.6.1 Detailed Description

The file contains class which manipulates the image represented by an n-dimensional dense numerical multi-channel array (Mat) by applying morphology operations.

Author

Oliwia Mlonek

5.7 ChangeFlip.h File Reference

The file contains class which manipulates the image represented by an n-dimensional dense numerical multi-channel array (Mat) by flipping it in three ways.

```
#include <memory>
#include <opencv2/opencv.hpp>
```

Classes

- class [ChangeFlip](#)

An abstract class responsible for flipping the image. The class interface declares operations common to all supported versions.

5.7.1 Detailed Description

The file contains class which manipulates the image represented by an n-dimensional dense numerical multi-channel array (Mat) by flipping it in three ways.

Author

Oliwia Mlonek

5.8 ChangeHistogram.h File Reference

The file contains class which manipulates the image represented by an n-dimensional dense numerical multi-channel array (Mat) by executing some operations related to image histogram.

```
#include <memory>
#include <opencv2/opencv.hpp>
```

Classes

- class [ChangeHistogram](#)

5.8.1 Detailed Description

The file contains class which manipulates the image represented by an n-dimensional dense numerical multi-channel array (Mat) by executing some operations related to image histogram.

Author

Oliwia Mlonek

5.9 ChangeRotation.h File Reference

The file contains class which manipulates the image represented by an n-dimensional dense numerical multi-channel array (Mat) by rotating the image.

```
#include <memory>
#include <opencv2/opencv.hpp>
```

Classes

- class [ChangeRotation](#)

An abstract class responsible for rotating the given image. The class interface declares operations common to all supported versions.

5.9.1 Detailed Description

The file contains class which manipulates the image represented by an n-dimensional dense numerical multi-channel array (Mat) by rotating the image.

Author

Oliwia Mlonek

5.10 ChangeShadows.h File Reference

The file contains class which manipulates the image represented by an n-dimensional dense numerical multi-channel array (Mat) by changing the lighting intensity of the image.

```
#include <memory>
#include <opencv2/opencv.hpp>
```

Classes

- class [ChangeShadows](#)

An abstract class responsible for applying chosen colours to the image. The class interface declares operations common to all supported versions.

5.10.1 Detailed Description

The file contains class which manipulates the image represented by an n-dimensional dense numerical multi-channel array (Mat) by changing the lighting intensity of the image.

Author

Oliwia Mlonek

5.11 ChangeSpace.h File Reference

The file contains class which manipulates the image represented by an n-dimensional dense numerical multi-channel array (Mat) by changing the color space of the image.

```
#include <memory>
#include <opencv2/opencv.hpp>
#include <opencv2/core/types_c.h>
```

Classes

- class [ChangeSpace](#)

An abstract class responsible for changing the color space of the image. The class interface declares operations common to all supported versions.

5.11.1 Detailed Description

The file contains class which manipulates the image represented by an n-dimensional dense numerical multi-channel array (Mat) by changing the color space of the image.

Author

Oliwia Mlonek

5.12 ChangeUndo.h File Reference

The file contains class which manipulates the list of pointers to the images represented by an n-dimensional dense numerical multi-channel array (Mat) by deleting unwanted nodes.

```
#include <memory>
#include <opencv2/opencv.hpp>
```

Classes

- class [ChangeUndo](#)

An abstract class responsible for getting rid of images with unwanted changes. The class interface declares operations common to all supported versions.

5.12.1 Detailed Description

The file contains class which manipulates the list of pointers to the images represented by an n-dimensional dense numerical multi-channel array (Mat) by deleting unwanted nodes.

Author

Oliwia Mlonek

5.13 Dilate.h File Reference

The file contains class which manipulates the image represented by an n-dimensional dense numerical multi-channel array (Mat) by applying dilation effect on it.

```
#include <opencv2/opencv.hpp>
#include <opencv2/imgproc.hpp>
#include "ChangeEffect.h"
```

Classes

- class [Dilate](#)

Responsible for applying the dilation effect to the image. Implements operations while following the base strategy ([ChangeEffect](#)) interface.

5.13.1 Detailed Description

The file contains class which manipulates the image represented by an n-dimensional dense numerical multi-channel array (Mat) by applying dilation effect on it.

Author

Oliwia Mlonek

5.14 Editor.h File Reference

The file contains class which maintains a reference to one of the concrete strategies and communicates with this object only via the strategy interface.

```
#include "Photo.h"
#include "ChangeColor.h"
#include "ChangeFlip.h"
#include "ChangeUndo.h"
#include "ChangeShadows.h"
#include "ChangeRotation.h"
#include "ChangeHistogram.h"
#include "ChangeEffect.h"
#include "ChangeSpace.h"
#include "SearchFace.h"
#include "BlackWhite.h"
#include "Grey.h"
#include "Sepia.h"
#include "FlipX.h"
#include "FlipY.h"
#include "FlipXY.h"
#include "UndoOne.h"
#include "UndoAll.h"
#include "Brightness.h"
#include "Gamma.h"
#include "FaceDetection.h"
#include "RotateRight.h"
#include "RotateLeft.h"
#include "ShowHistogram.h"
#include "HistogramEq.h"
#include "HSV.h"
#include "RBG.h"
#include "HLS.h"
#include "YCrCb.h"
#include "BGR.h"
#include "Erode.h"
#include "Dilate.h"
#include "Exceptions.h"
#include <memory>
```

Classes

- class [Editor](#)

Class is responsible of maintaining a reference to one of the concrete strategies and communicates with this object only via the strategy interface.

5.14.1 Detailed Description

The file contains class which maintains a reference to one of the concrete strategies and communicates with this object only via the strategy interface.

Author

Oliwia Mlonek

5.15 Erode.h File Reference

The file contains class which manipulates the image represented by an n-dimensional dense numerical multi-channel array (Mat) by applying erosion effect on it.

```
#include <opencv2/opencv.hpp>
#include <opencv2/imgproc.hpp>
#include "ChangeEffect.h"
```

Classes

- class [Erode](#)
Responsible for applying the erosion effect to the image. Implements operations while following the base strategy ([ChangeEffect](#)) interface.

5.15.1 Detailed Description

The file contains class which manipulates the image represented by an n-dimensional dense numerical multi-channel array (Mat) by applying erosion effect on it.

Author

Oliwia Mlonek

5.16 Exceptions.h File Reference

The file contains three classes which are thrown as exceptions when the need arises.

```
#include <iostream>
#include <exception>
```

Classes

- class [IsNull](#)
Responsible for throwing a message when the used list is empty.
- class [IsOneElementOnly](#)
Responsible for throwing a message when the used list has only one element.
- class [NoFaceFound](#)
Responsible for throwing a message when face detector cannot recognize any face.

5.16.1 Detailed Description

The file contains three classes which are thrown as exceptions when the need arises.

Author

Oliwia Mlonek

5.17 FaceDetection.h File Reference

The file contains class which is designed to detect all human faces that are in a given image (represented by an n-dimensional dense numerical multi-channel array- Mat).

```
#include "SearchFace.h"
#include "opencv2/imgcodecs.hpp"
#include "opencv2/highgui.hpp"
```

Classes

- class [FaceDetection](#)

Responsible for finding human faces in the given image and outlining the area in which they are located. Implements operations while following the base strategy ([SearchFace](#)) interface.

5.17.1 Detailed Description

The file contains class which is designed to detect all human faces that are in a given image (represented by an n-dimensional dense numerical multi-channel array- Mat).

Author

Oliwia Mlonek

5.18 FlipX.h File Reference

The file contains class which manipulates the image represented by an n-dimensional dense numerical multi-channel array (Mat) by flipping it horizontally.

```
#include <opencv2/opencv.hpp>
#include <opencv2/imgproc.hpp>
#include "ChangeFlip.h"
```

Classes

- class [FlipX](#)

Responsible for flipping the image horizontally. Implements operations while following the base strategy ([ChangeFlip](#)) interface.

5.18.1 Detailed Description

The file contains class which manipulates the image represented by an n-dimensional dense numerical multi-channel array (Mat) by flipping it horizontally.

Author

Oliwia Mlonek

5.19 FlipXY.h File Reference

The file contains class which manipulates the image represented by an n-dimensional dense numerical multi-channel array (Mat) by flipping it horizontally and vertically at once.

```
#include <opencv2/opencv.hpp>
#include <opencv2/imgproc.hpp>
#include "ChangeFlip.h"
```

Classes

- class [FlipXY](#)

5.19.1 Detailed Description

The file contains class which manipulates the image represented by an n-dimensional dense numerical multi-channel array (Mat) by flipping it horizontally and vertically at once.

Author

Oliwia Mlonek

5.20 Gamma.h File Reference

The file contains class which manipulates the image represented by an n-dimensional dense numerical multi-channel array (Mat) by changing the lighting of the image using gamma correction.

```
#include "ChangeShadows.h"
#include "opencv2/imgcodecs.hpp"
#include "opencv2/highgui.hpp"
```

Classes

- class [Gamma](#)
Responsible for changing the gamma correction of the image. Implements operations while following the base strategy ([ChangeShadows](#)) interface.

5.20.1 Detailed Description

The file contains class which manipulates the image represented by an n-dimensional dense numerical multi-channel array (Mat) by changing the lighting of the image using gamma correction.

Author

Oliwia Mlonek

5.21 Grey.h File Reference

The file contains class which manipulates the image represented by an n-dimensional dense numerical multi-channel array (Mat) by changing the colours of the image to greyscale.

```
#include <opencv2/opencv.hpp>
#include <opencv2/imgproc.hpp>
#include "ChangeColor.h"
```

Classes

- class [Grey](#)
Responsible for applying grey colour to the image. Implements operations while following the base strategy ([ChangeColor](#)) interface.

5.21.1 Detailed Description

The file contains class which manipulates the image represented by an n-dimensional dense numerical multi-channel array (Mat) by changing the colours of the image to greyscale.

Author

Oliwia Mlonek

5.22 HistogramEq.h File Reference

The file contains class which manipulates the image represented by an n-dimensional dense numerical multi-channel array (Mat) by executing histogram equalization in the image.

```
#include <opencv2/opencv.hpp>
#include <opencv2/imgproc.hpp>
#include "ChangeHistogram.h"
```

Classes

- class [HistogramEq](#)
Responsible for histogram equalization. Implements operations while following the base strategy ([ChangeFlip](#)) interface.

5.22.1 Detailed Description

The file contains class which manipulates the image represented by an n-dimensional dense numerical multi-channel array (Mat) by executing histogram equalization in the image.

Author

Oliwia Mlonek

5.23 HLS.h File Reference

The file contains class which manipulates the image represented by an n-dimensional dense numerical multi-channel array (Mat) by changing the color space of the image to [HLS](#).

```
#include <opencv2/opencv.hpp>
#include <opencv2/imgproc.hpp>
#include "ChangeSpace.h"
```

Classes

- class [HLS](#)
responsible for changing the color space of the image to [HLS](#) space. Implements operations while following the base strategy ([SearchFace](#)) interface.

5.23.1 Detailed Description

The file contains class which manipulates the image represented by an n-dimensional dense numerical multi-channel array (Mat) by changing the color space of the image to [HLS](#).

Author

Oliwia Mlonek

5.24 HSV.h File Reference

The file contains class which manipulates the image represented by an n-dimensional dense numerical multi-channel array (Mat) by changing the color space of the image to [HSV](#).

```
#include <opencv2/opencv.hpp>
#include <opencv2/imgproc.hpp>
#include "ChangeSpace.h"
```

Classes

- class [HSV](#)
responsible for changing the color space of the image to [HSV](#) space. Implements operations while following the base strategy ([SearchFace](#)) interface.

5.24.1 Detailed Description

The file contains class which manipulates the image represented by an n-dimensional dense numerical multi-channel array (Mat) by changing the color space of the image to [HSV](#).

Author

Oliwia Mlonek

5.25 MainWindow.h File Reference

The file contains class which is responsible for creating and managing the entire graphical user interface and calling appropriate actions related to image processing after actions on the part of the user in the GUI.

```
#include "Photo.h"
#include "Editor.h"
#include "Exceptions.h"
#include <iostream>
#include <stdio.h>
#include <stdlib.h>
#include <string>
#include <wx/log.h>
#include <wx/stattext.h>
#include <wx/choice.h>
#include <wx/slider.h>
#include <wx/artprov.h>
#include <wx/xrc/xmlres.h>
#include <wx/bitmap.h>
#include <wx/image.h>
#include <wx/icon.h>
#include <wx/string.h>
#include <wx/menu.h>
#include <wx/gdicmn.h>
#include <wx/font.h>
#include <wx/toolbar.h>
#include <wx/panel.h>
#include <wx/filedlg.h>
#include <wx/dcclient.h>
#include <wx/dcmemory.h>
#include <wx/button.h>
#include <wx/frame.h>
#include <wx/msgdlg.h>
#include <wx/filename.h>
#include <wx/clipbrd.h>
#include <wx/settings.h>
#include <wx/intl.h>
#include <opencv2/highgui.hpp>
#include <sstream>
```

Classes

- class [MainWindow](#)

Creates and manages the project GUI.

5.25.1 Detailed Description

The file contains class which is responsible for creating and managing the entire graphical user interface and calling appropriate actions related to image processing after actions on the part of the user in the GUI.

Author

Oliwia Mlonek

5.26 Photo.h File Reference

The file contains class which manages the loaded image during the proces of edition.

```
#include <wx/wx.h>
#include <opencv2/opencv.hpp>
#include <wx/image.h>
#include <list>
#include <string>
#include <iostream>
#include <unordered_map>
```

Classes

- class [Photo](#)

Class is responsible for holding the image and the list with all with all versions of the image that was created, as well as stores data about the properties of the photo.

5.26.1 Detailed Description

The file contains class which manages the loaded image during the proces of edition.

Author

Oliwia Mlonek

5.27 RBG.h File Reference

The file contains class which manipulates the image represented by an n-dimensional dense numerical multi-channel array (Mat) by changing the color space of the image to [_RGB](#).

```
#include <opencv2/opencv.hpp>
#include <opencv2/imgproc.hpp>
#include "ChangeSpace.h"
```

Classes

- class [_RGB](#)

responsible for changing the color space of the image to RGB space. Implements operations while following the base strategy ([SearchFace](#)) interface.

5.27.1 Detailed Description

The file contains class which manipulates the image represented by an n-dimensional dense numerical multi-channel array (Mat) by changing the color space of the image to [_RGB](#).

Author

Oliwia Mlonek

5.28 RotateLeft.h File Reference

The file contains class which manipulates the image represented by an n-dimensional dense numerical multi-channel array (Mat) by rotating the image to the left.

```
#include <opencv2/opencv.hpp>
#include <opencv2/imgproc.hpp>
#include "ChangeRotation.h"
```

Classes

- class [RotateLeft](#)

An abstract class responsible for rotating the given image 90 degrees to the left. Implements operations while following the base strategy ([ChangeFlip](#)) interface.

5.28.1 Detailed Description

The file contains class which manipulates the image represented by an n-dimensional dense numerical multi-channel array (Mat) by rotating the image to the left.

Author

Oliwia Mlonek

5.29 RotateRight.h File Reference

The file contains class which manipulates the image represented by an n-dimensional dense numerical multi-channel array (Mat) by rotating the image to the left.

```
#include <opencv2/opencv.hpp>
#include <opencv2/imgproc.hpp>
#include "ChangeRotation.h"
```

Classes

- class [RotateRight](#)

An abstract class responsible for rotating the given image 90 degrees to the right. Implements operations while following the base strategy ([ChangeFlip](#)) interface.

5.29.1 Detailed Description

The file contains class which manipulates the image represented by an n-dimensional dense numerical multi-channel array (Mat) by rotating the image to the left.

Author

Oliwia Mlonek

5.30 SearchFace.h File Reference

The file contains class which is designed to detect all human faces that are in a given image (represented by an n-dimensional dense numerical multi-channel array- Mat).

```
#include <memory>
#include <opencv2/opencv.hpp>
#include "opencv2/objdetect.hpp"
#include "opencv2/highgui.hpp"
#include "opencv2/imgproc.hpp"
#include <opencv2/core/core_c.h>
#include <iostream>
```

Classes

- class [SearchFace](#)

An abstract clas responsible for finding human faces in the given image. The class interface declares operations common to all supported versions.

5.30.1 Detailed Description

The file contains class which is designed to detect all human faces that are in a given image (represented by an n-dimensional dense numerical multi-channel array- Mat).

Author

Oliwia Mlonek

5.31 Sepia.h File Reference

The file contains class which manipulates the image represented by an n-dimensional dense numerical multi-channel array (Mat) by changing the colours of the image to black and white only.

```
#include <opencv2/opencv.hpp>
#include <opencv2/imgproc.hpp>
#include "ChangeColor.h"
```

Classes

- class [Sepia](#)

Responsible for applying sepia colour to the image. Implements operations while following the base strategy ([ChangeColor](#)) interface.

5.31.1 Detailed Description

The file contains class which manipulates the image represented by an n-dimensional dense numerical multi-channel array (Mat) by changing the colours of the image to black and white only.

Author

Oliwia Mlonek

5.32 ShowHistogram.h File Reference

The file contains class which manipulates the image represented by an n-dimensional dense numerical multi-channel array (Mat) by plotting and displaying image histogram.

```
#include <opencv2/opencv.hpp>
#include <opencv2/imgproc.hpp>
#include "ChangeHistogram.h"
```

Classes

- class [ShowHistogram](#)

Responsible for displaying histogram. Implements operations while following the base strategy ([ChangeFlip](#)) interface.

5.32.1 Detailed Description

The file contains class which manipulates the image represented by an n-dimensional dense numerical multi-channel array (Mat) by plotting and displaying image histogram.

Author

Oliwia Mlonek

5.33 UndoAll.h File Reference

The file contains class which manipulates the list of pointers to the images represented by an n-dimensional dense numerical multi-channel array (Mat) by deleting all but the last one elements from the list.

```
#include <opencv2/opencv.hpp>
#include <opencv2/imgproc.hpp>
#include "ChangeUndo.h"
```

Classes

- class [UndoAll](#)

responsible for getting rid of all but the last one images from the list. Implements operations while following the base strategy ([SearchFace](#)) interface.

5.33.1 Detailed Description

The file contains class which manipulates the list of pointers to the images represented by an n-dimensional dense numerical multi-channel array (Mat) by deleting all but the last one elements from the list.

Author

Oliwia Mlonek

5.34 UndoOne.h File Reference

The file contains class which manipulates the list of pointers to the images represented by an n-dimensional dense numerical multi-channel array (Mat) by deleting unwanted (first) node.

```
#include <opencv2/opencv.hpp>
#include <opencv2/imgproc.hpp>
#include "ChangeUndo.h"
```

Classes

- class [UndoOne](#)
responsible for getting rid of the last created image with unwanted change. Implements operations while following the base strategy ([SearchFace](#)) interface.

5.34.1 Detailed Description

The file contains class which manipulates the list of pointers to the images represented by an n-dimensional dense numerical multi-channel array (Mat) by deleting unwanted (first) node.

Author

Oliwia Mlonek

5.35 YCrCb.h File Reference

The file contains class which manipulates the image represented by an n-dimensional dense numerical multi-channel array (Mat) by changing the color space of the image to [YCrCb](#).

```
#include <opencv2/opencv.hpp>
#include <opencv2/imgproc.hpp>
#include "ChangeSpace.h"
```

Classes

- class [YCrCb](#)
responsible for changing the color space of the image to [YCrCb](#) space. Implements operations while following the base strategy ([SearchFace](#)) interface.

5.35.1 Detailed Description

The file contains class which manipulates the image represented by an n-dimensional dense numerical multi-channel array (Mat) by changing the color space of the image to [YCrCb](#).

Author

Oliwia Mlonek