

labo\_01\_comte\_emmanuelle\_gallay\_david

Generated by Doxygen 1.8.13



# Contents

<b>1</b>	<b>Class Index</b>	<b>1</b>
1.1	Class List . . . . .	1
<b>2</b>	<b>File Index</b>	<b>3</b>
2.1	File List . . . . .	3
<b>3</b>	<b>Class Documentation</b>	<b>5</b>
3.1	Circle Class Reference . . . . .	5
3.1.1	Constructor & Destructor Documentation . . . . .	5
3.1.1.1	Circle() [1/3] . . . . .	5
3.1.1.2	Circle() [2/3] . . . . .	6
3.1.1.3	Circle() [3/3] . . . . .	6
3.1.2	Member Function Documentation . . . . .	6
3.1.2.1	display() . . . . .	6
3.1.2.2	getColor() . . . . .	7
3.1.2.3	getRadius() . . . . .	7
3.1.2.4	getSurface() . . . . .	7
3.1.2.5	setColor() [1/2] . . . . .	7
3.1.2.6	setColor() [2/2] . . . . .	8
3.1.2.7	setRadius() . . . . .	8
3.2	Color Class Reference . . . . .	9
3.2.1	Member Enumeration Documentation . . . . .	9
3.2.1.1	Code . . . . .	9
3.2.2	Constructor & Destructor Documentation . . . . .	9

3.2.2.1	Color()	10
3.2.3	Member Function Documentation	10
3.2.3.1	display()	10
3.2.3.2	getColorCode()	10
3.2.3.3	getName()	11
3.2.3.4	setColor()	11
3.3	Rectangle Class Reference	11
3.3.1	Constructor & Destructor Documentation	12
3.3.1.1	Rectangle() [1/3]	12
3.3.1.2	Rectangle() [2/3]	12
3.3.1.3	Rectangle() [3/3]	13
3.3.2	Member Function Documentation	13
3.3.2.1	display()	13
3.3.2.2	getColor()	13
3.3.2.3	getHeight()	14
3.3.2.4	getSurface()	14
3.3.2.5	getWidth()	14
3.3.2.6	setColor() [1/2]	14
3.3.2.7	setColor() [2/2]	15
3.3.2.8	setHeight()	15
3.3.2.9	setWidth()	15
3.4	Square Class Reference	16
3.4.1	Constructor & Destructor Documentation	16
3.4.1.1	Square() [1/3]	16
3.4.1.2	Square() [2/3]	17
3.4.1.3	Square() [3/3]	17
3.4.2	Member Function Documentation	17
3.4.2.1	display()	17
3.4.2.2	getColor()	18
3.4.2.3	getSide()	18

3.4.2.4	<a href="#">getSurface()</a>	18
3.4.2.5	<a href="#">setColor()</a> [1/2]	18
3.4.2.6	<a href="#">setColor()</a> [2/2]	19
3.4.2.7	<a href="#">setSide()</a>	19
3.5	<a href="#">Triangle Class Reference</a>	20
3.5.1	<a href="#">Constructor &amp; Destructor Documentation</a>	20
3.5.1.1	<a href="#">Triangle()</a> [1/3]	20
3.5.1.2	<a href="#">Triangle()</a> [2/3]	20
3.5.1.3	<a href="#">Triangle()</a> [3/3]	21
3.5.2	<a href="#">Member Function Documentation</a>	21
3.5.2.1	<a href="#">display()</a>	21
3.5.2.2	<a href="#">getBase()</a>	21
3.5.2.3	<a href="#">getColor()</a>	22
3.5.2.4	<a href="#">getHeight()</a>	22
3.5.2.5	<a href="#">getSurface()</a>	22
3.5.2.6	<a href="#">setBase()</a>	22
3.5.2.7	<a href="#">setColor()</a> [1/2]	23
3.5.2.8	<a href="#">setColor()</a> [2/2]	23
3.5.2.9	<a href="#">setHeight()</a>	23
4	<a href="#">File Documentation</a>	25
4.1	<a href="#">circle.h File Reference</a>	25
4.1.1	<a href="#">Function Documentation</a>	26
4.1.1.1	<a href="#">operator&lt;&lt;()</a>	26
4.2	<a href="#">color.h File Reference</a>	26
4.2.1	<a href="#">Function Documentation</a>	27
4.2.1.1	<a href="#">operator&lt;&lt;()</a>	27
4.2.1.2	<a href="#">toString()</a>	28
4.3	<a href="#">rectangle.h File Reference</a>	28
4.3.1	<a href="#">Function Documentation</a>	29
4.3.1.1	<a href="#">operator&lt;&lt;()</a>	29
4.4	<a href="#">square.h File Reference</a>	29
4.4.1	<a href="#">Function Documentation</a>	30
4.4.1.1	<a href="#">operator&lt;&lt;()</a>	30
4.5	<a href="#">triangle.h File Reference</a>	30
4.5.1	<a href="#">Function Documentation</a>	31
4.5.1.1	<a href="#">operator&lt;&lt;()</a>	31
	<a href="#">Index</a>	33



# Chapter 1

## Class Index

### 1.1 Class List

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

<a href="#">Circle</a>	5
<a href="#">Color</a>	9
<a href="#">Rectangle</a>	11
<a href="#">Square</a>	16
<a href="#">Triangle</a>	20





## Chapter 2

# File Index

### 2.1 File List

Here is a list of all files with brief descriptions:

<a href="#">circle.h</a>	25
<a href="#">color.h</a>	26
<a href="#">rectangle.h</a>	28
<a href="#">square.h</a>	29
<a href="#">triangle.h</a>	30



## Chapter 3

# Class Documentation

### 3.1 Circle Class Reference

```
#include <circle.h>
```

#### Public Member Functions

- **Circle** (double radius=0, const **Color** &color=**Color**())  
*Create a circle with color and radius, a radius or with nothing.*
- **Circle** (const **Color** &color)  
*Create a circle with an object color.*
- **Circle** (**Color::Code** code)  
*Create a circle with a color by a color code (enum)*
- **Circle** & **setRadius** (double radius)  
*Change the radius of the circle.*
- **Circle** & **setColor** (const **Color** &color)  
*Change the color of the circle with an object color.*
- **Circle** & **setColor** (**Color::Code** color)  
*Change the color of the circle with a color code (enum)*
- double **getRadius** () const  
*Get the radius of the circle.*
- double **getSurface** () const  
*Calculate the surface of the circle.*
- **Color** **getColor** () const  
*Get the color of the circle.*
- std::ostream & **display** (std::ostream &stream=std::cout) const  
*Display a circle.*

#### 3.1.1 Constructor & Destructor Documentation

##### 3.1.1.1 **Circle()** [1/3]

```
Circle::Circle (
    double radius = 0,
    const Color & color = Color() )
```

Create a circle with color and radius, a radius or with nothing.

**Parameters**

<i>radius</i>	default value 0
<i>color</i>	default value <a href="#">Color()</a>

**3.1.1.2 Circle()** [2/3]

```
Circle::Circle (
    const Color & color )
```

Create a circle with an object color.

**Parameters**

<i>color</i>	
--------------	--

**3.1.1.3 Circle()** [3/3]

```
Circle::Circle (
    Color::Code code )
```

Create a circle with a color by a color code (enum)

**Parameters**

<i>code</i>	
-------------	--

**3.1.2 Member Function Documentation****3.1.2.1 display()**

```
std::ostream& Circle::display (
    std::ostream & stream = std::cout ) const
```

Display a circle.

**Parameters**

<i>stream</i>	default valut cout
---------------	--------------------

**Returns**

The stream

**3.1.2.2 getColor()**

```
Color Circle::getColor ( ) const
```

Get the color of the circle.

**Returns**

The color of the circle

**3.1.2.3 getRadius()**

```
double Circle::getRadius ( ) const
```

Get the radius of the circle.

**Returns**

The radius of the circle

**3.1.2.4 getSurface()**

```
double Circle::getSurface ( ) const
```

Calculate the surface of the circle.

**Returns**

The surface of the circle

**3.1.2.5 setColor()** [1/2]

```
Circle& Circle::setColor (
    const Color & color )
```

Change the color of the circle with an object color.

**Parameters**

<i>color</i>	
--------------	--

**Returns**

The circle

**3.1.2.6 setColor()** [2/2]

```
Circle& Circle::setColor (
    Color::Code color )
```

Change the color of the circle with a color code (enum)

**Parameters**

<i>color</i>	
--------------	--

**Returns**

The circle

**3.1.2.7 setRadius()**

```
Circle& Circle::setRadius (
    double radius )
```

Change the radius of the circle.

**Parameters**

<i>radius</i>	
---------------	--

**Returns**

The circle

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

- [circle.h](#)

## 3.2 Color Class Reference

```
#include <color.h>
```

### Public Types

- enum `Code` {  
    `Code::WHITE`, `Code::BLUE`, `Code::GREEN`, `Code::RED`,  
    `Code::BLACK` }

### Public Member Functions

- `Color` (`Code` code=`Code::WHITE`)  
    *Create a color with a color by code (enum) or with nothing.*
- `Code` `getColorCode` () const  
    *Get the color.*
- `Color` & `setColor` (`Code` code)  
    *Change the color.*
- std::string `getName` () const  
    *Get the name of the color.*
- std::ostream & `display` (std::ostream &stream=std::cout) const  
    *Display a color.*

### 3.2.1 Member Enumeration Documentation

#### 3.2.1.1 Code

```
enum Color::Code [strong]
```

#### Enumerator

WHITE	
BLUE	
GREEN	
RED	
BLACK	

### 3.2.2 Constructor & Destructor Documentation

### 3.2.2.1 Color()

```
Color::Color (
    Code code = Code::WHITE )
```

Create a color with a color by code (enum) or with nothing.

#### Parameters

<i>code</i>	default value WHITE
-------------	---------------------

## 3.2.3 Member Function Documentation

### 3.2.3.1 display()

```
std::ostream& Color::display (
    std::ostream & stream = std::cout ) const
```

Display a color.

#### Parameters

<i>stream</i>	
---------------	--

#### Returns

The srteam

### 3.2.3.2 getColorCode()

```
Code Color::getColorCode ( ) const
```

Get the color.

#### Returns

The code of the color (enum)



### 3.2.3.3 getName()

```
std::string Color::getName ( ) const
```

Get the name of the color.

#### Returns

The name of the color

### 3.2.3.4 setColor()

```
Color& Color::setColor (
    Code code )
```

Change the color.

#### Parameters

<i>code</i>	
-------------	--

#### Returns

The color

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

- [color.h](#)

## 3.3 Rectangle Class Reference

```
#include <rectangle.h>
```

### Public Member Functions

- [Rectangle](#) (double width=0, double height=0, const [Color](#) &color=[Color](#)())  
*Create a rectangle with a width, a height and a color, a width and a height, a width or with nothing.*
- [Rectangle](#) (const [Color](#) &color)  
*Create a rectangle with an object color.*
- [Rectangle](#) ([Color::Code](#) code)  
*Create a rectangle with a color code (enum)*
- double [getHeight](#) () const  
*Get the height of the rectangle.*
- double [getWidth](#) () const  
*Get the width of the rectangle.*

- double `getSurface` () const  
*Calculate the surface of the rectangle.*
- `Color` `getColor` () const  
*Get the color of the rectangle.*
- `Rectangle` & `setHeight` (double height)  
*Change the height of the rectangle.*
- `Rectangle` & `setWidth` (double width)  
*Change the width of the rectangle.*
- `Rectangle` & `setColor` (const `Color` &color)  
*Change the color of the rectangle with an object color.*
- `Rectangle` & `setColor` (`Color::Code` color)  
*Change the color of the rectangle with a color code (enum)*
- `std::ostream` & `display` (`std::ostream` &stream=`std::cout`) const  
*Display a rectangle.*

### 3.3.1 Constructor & Destructor Documentation

#### 3.3.1.1 `Rectangle()` [1/3]

```
Rectangle::Rectangle (
    double width = 0,
    double height = 0,
    const Color & color = Color() )
```

Create a rectangle with a width, a height and a color, a width and a height, a width or with nothing.

##### Parameters

<i>width</i>	default value 0
<i>height</i>	default value 0
<i>color</i>	default value <code>Color()</code>

#### 3.3.1.2 `Rectangle()` [2/3]

```
Rectangle::Rectangle (
    const Color & color )
```

Create a rectangle with an object color.

##### Parameters

<i>color</i>	
--------------	--

### 3.3.1.3 Rectangle() [3/3]

```
Rectangle::Rectangle (
    Color::Code code )
```

Create a rectangle with a color code (enum)

#### Parameters

<i>code</i>	
-------------	--

## 3.3.2 Member Function Documentation

### 3.3.2.1 display()

```
std::ostream& Rectangle::display (
    std::ostream & stream = std::cout ) const
```

Display a rectangle.

#### Parameters

<i>stream</i>	
---------------	--

#### Returns

The stream

### 3.3.2.2 getColor()

```
Color Rectangle::getColor ( ) const
```

Get the color of the rectangle.

#### Returns

The color of the rectangle

### 3.3.2.3 getHeight()

```
double Rectangle::getHeight ( ) const
```

Get the height of the rectangle.

#### Returns

The height of the rectangle

### 3.3.2.4 getSurface()

```
double Rectangle::getSurface ( ) const
```

Calculate the surface of the rectangle.

#### Returns

The surface f the rectangle

### 3.3.2.5 getWidth()

```
double Rectangle::getWidth ( ) const
```

Get the width of the rectangle.

#### Returns

The width of the rectangle

### 3.3.2.6 setColor() [1/2]

```
Rectangle& Rectangle::setColor (
    const Color & color )
```

Change the color of the rectangle with an object color.

#### Parameters

<i>color</i>	
--------------	--

**Returns**

The rectangle

**3.3.2.7 setColor()** [2/2]

```
Rectangle& Rectangle::setColor (
    Color::Code color )
```

Change the color of the rectangle with a color code (enum)

**Parameters**

<i>color</i>	
--------------	--

**Returns**

The rectangle

**3.3.2.8 setHeight()**

```
Rectangle& Rectangle::setHeight (
    double height )
```

Change the height of the rectangle.

**Parameters**

<i>height</i>	
---------------	--

**Returns**

The rectangle

**3.3.2.9 setWidth()**

```
Rectangle& Rectangle::setWidth (
    double width )
```

Change the width of the rectangle.

## Parameters

<i>base</i>	
-------------	--

## Returns

The rectangle

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

- [rectangle.h](#)

## 3.4 Square Class Reference

```
#include <square.h>
```

### Public Member Functions

- [Square](#) (double side=0, const [Color](#) &color=[Color](#)())  
*Create a square with a side and a color, a side or with nothing.*
- [Square](#) (const [Color](#) &color)  
*Create a square with an object color.*
- [Square](#) ([Color::Code](#) code)  
*Create a square with a color code (enum)*
- double [getSide](#) () const  
*Get the side of the square.*
- double [getSurface](#) () const  
*Calculate the surface of the square.*
- [Color](#) [getColor](#) () const  
*Get the color of the square.*
- [Square](#) & [setSide](#) (double side)  
*Change the side of the square.*
- [Square](#) & [setColor](#) (const [Color](#) &color)  
*Change the color of the square with an object color.*
- [Square](#) & [setColor](#) ([Color::Code](#) color)  
*Change the color of the square with a color code (enum)*
- std::ostream & [display](#) (std::ostream &stream=std::cout) const  
*Display a square.*

### 3.4.1 Constructor & Destructor Documentation

#### 3.4.1.1 [Square](#)() [1/3]

```
Square::Square (
    double side = 0,
    const Color & color = Color() )
```

Create a square with a side and a color, a side or with nothing.

## Parameters

<i>side</i>	defalut value 0
<i>color</i>	default value <a href="#">Color()</a>

3.4.1.2 [Square\(\)](#) [2/3]

```
Square::Square (
    const Color & color )
```

Create a square with an object color.

## Parameters

<i>color</i>	
--------------	--

3.4.1.3 [Square\(\)](#) [3/3]

```
Square::Square (
    Color::Code code )
```

Create a square with a color code (enum)

## Parameters

<i>code</i>	
-------------	--

## 3.4.2 Member Function Documentation

3.4.2.1 [display\(\)](#)

```
std::ostream& Square::display (
    std::ostream & stream = std::cout ) const
```

Display a square.

## Parameters

<i>stream</i>	
---------------	--

**Returns**

The stream

**3.4.2.2 getColor()**

```
Color Square::getColor ( ) const
```

Get the color of the square.

**Returns**

The color of the square

**3.4.2.3 getSide()**

```
double Square::getSide ( ) const
```

Get the side of the square.

**Returns**

The side of the square

**3.4.2.4 getSurface()**

```
double Square::getSurface ( ) const
```

Calculate the surface of the square.

**Returns**

The surface of the square

**3.4.2.5 setColor()** [1/2]

```
Square& Square::setColor (
    const Color & color )
```

Change the color of the square with an object color.



## Parameters

<i>color</i>	
--------------	--

## Returns

The square

**3.4.2.6 setColor()** [2/2]

```
Square& Square::setColor (
    Color::Code color )
```

Change the color of the square with a color code (enum)

## Parameters

<i>color</i>	
--------------	--

## Returns

The square

**3.4.2.7 setSide()**

```
Square& Square::setSide (
    double side )
```

Change the side of the square.

## Parameters

<i>side</i>	
-------------	--

## Returns

The square

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

- [square.h](#)

## 3.5 Triangle Class Reference

```
#include <triangle.h>
```

### Public Member Functions

- [Triangle](#) (double base=0, double height=0, const [Color](#) &color=[Color](#)())
- [Triangle](#) (const [Color](#) &color)
- [Triangle](#) ([Color::Code](#) code)
- double [getHeight](#) () const
- double [getBase](#) () const
- double [getSurface](#) () const
- [Color](#) [getColor](#) () const  
*Get the color of the triangle.*
- [Triangle](#) & [setHeight](#) (double height)
- [Triangle](#) & [setBase](#) (double base)
- [Triangle](#) & [setColor](#) (const [Color](#) &color)
- [Triangle](#) & [setColor](#) ([Color::Code](#) color)
- std::ostream & [display](#) (std::ostream &stream=std::cout) const

### 3.5.1 Constructor & Destructor Documentation

#### 3.5.1.1 [Triangle\(\)](#) [1/3]

```
Triangle::Triangle (
    double base = 0,
    double height = 0,
    const Color & color = Color() )
```

Create a triangle with a base, a height and a color, a base and a height, a base or with nothing

#### Parameters

<i>base</i>	defalut value 0
<i>height</i>	default value 0
<i>color</i>	default value <a href="#">Color</a> ()

#### 3.5.1.2 [Triangle\(\)](#) [2/3]

```
Triangle::Triangle (
    const Color & color )
```

Create a triangle with an object color

## Parameters

<i>color</i>	
--------------	--

## 3.5.1.3 Triangle() [3/3]

```
Triangle::Triangle (  
    Color::Code code )
```

Create a triangle with a color code (enum)

## Parameters

<i>code</i>	
-------------	--

## 3.5.2 Member Function Documentation

## 3.5.2.1 display()

```
std::ostream& Triangle::display (  
    std::ostream & stream = std::cout ) const
```

Display a triangle

## Parameters

<i>stream</i>	
---------------	--

## Returns

The stream

## 3.5.2.2 getBase()

```
double Triangle::getBase ( ) const
```

Get the base of the triangle

## Returns

The base of the triangle

### 3.5.2.3 getColor()

```
Color Triangle::getColor ( ) const
```

Get the color of the triangle.

#### Returns

The color of the triangle

### 3.5.2.4 getHeight()

```
double Triangle::getHeight ( ) const
```

Get the height of the triangle

#### Returns

The height of the triangle

### 3.5.2.5 getSurface()

```
double Triangle::getSurface ( ) const
```

Calculat the surface of the triangle

#### Returns

The surface of the triangle

### 3.5.2.6 setBase()

```
Triangle& Triangle::setBase (
    double base )
```

Change the base of the triangle

#### Parameters

<i>base</i>	
-------------	--

**Returns**

The triangle

**3.5.2.7 setColor()** [1/2]

```
Triangle& Triangle::setColor (
    const Color & color )
```

Change the color of the triangle with an object color

**Parameters**

<i>color</i>	
--------------	--

**Returns**

The triangle

**3.5.2.8 setColor()** [2/2]

```
Triangle& Triangle::setColor (
    Color::Code color )
```

Change the color of the triangle with a color code (enum)

**Parameters**

<i>color</i>	
--------------	--

**Returns**

The triangle

**3.5.2.9 setHeight()**

```
Triangle& Triangle::setHeight (
    double height )
```

Change the height of the triangle

**Parameters**

<i>height</i>	
---------------	--

**Returns**

The triangle

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

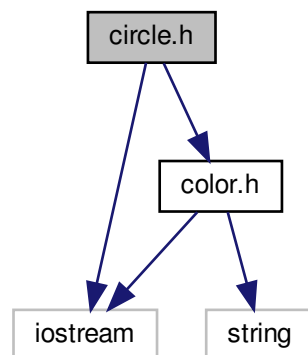
- [triangle.h](#)

## Chapter 4

# File Documentation

### 4.1 circle.h File Reference

```
#include <iostream>
#include "color.h"
Include dependency graph for circle.h:
```



#### Classes

- class [Circle](#)

#### Functions

- `std::ostream & operator<< (std::ostream &stream, const Circle &circle)`  
*Overload of the output stream to display a circle.*

## 4.1.1 Function Documentation

### 4.1.1.1 `operator<<()`

```
std::ostream& operator<< (  
    std::ostream & stream,  
    const Circle & circle )
```

Overload of the output stream to display a circle.

#### Parameters

<i>stream</i>	
<i>circle</i>	

#### Returns

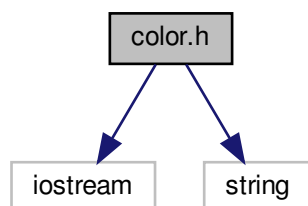
The stream

## 4.2 color.h File Reference

```
#include <iostream>
```

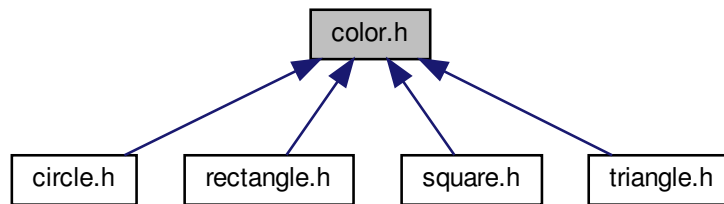
```
#include <string>
```

Include dependency graph for color.h:





This graph shows which files directly or indirectly include this file:



## Classes

- class [Color](#)

## Functions

- `std::string toString (const Color::Code &code)`  
*Convert the code color in a string.*
- `std::ostream & operator<< (std::ostream &stream, const Color &color)`  
*Overload of the output stream to display a color.*

### 4.2.1 Function Documentation

#### 4.2.1.1 operator<<()

```
std::ostream& operator<< (
    std::ostream & stream,
    const Color & color )
```

Overload of the output stream to display a color.

#### Parameters

<i>stream</i>	
<i>color</i>	

#### Returns

The stream

#### 4.2.1.2 toString()

```
std::string toString (
    const Color::Code & code )
```

Convert the code color in a string.

##### Parameters

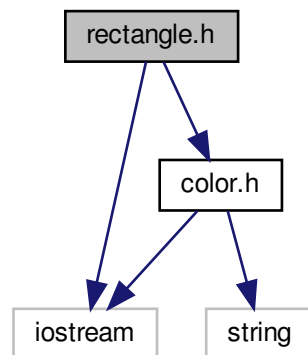
<i>code</i>	
-------------	--

##### Returns

The color in string

## 4.3 rectangle.h File Reference

```
#include <iostream>
#include "color.h"
Include dependency graph for rectangle.h:
```



## Classes

- class [Rectangle](#)

## Functions

- `std::ostream & operator<< (std::ostream &stream, const Rectangle &rectangle)`  
*Overload of the output stream to display a rectangle.*

### 4.3.1 Function Documentation

#### 4.3.1.1 operator<<()

```
std::ostream& operator<< (
    std::ostream & stream,
    const Rectangle & rectangle )
```

Overload of the output stream to display a rectangle.

##### Parameters

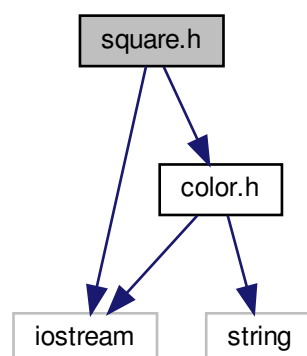
<i>stream</i>	
<i>rectangle</i>	

##### Returns

The stream

## 4.4 square.h File Reference

```
#include <iostream>
#include "color.h"
Include dependency graph for square.h:
```



### Classes

- class [Square](#)

## Functions

- `std::ostream & operator<< (std::ostream &stream, const Square &square)`  
*Overload of the output stream to display a square.*

### 4.4.1 Function Documentation

#### 4.4.1.1 operator<<()

```
std::ostream& operator<< (
    std::ostream & stream,
    const Square & square )
```

Overload of the output stream to display a square.

#### Parameters

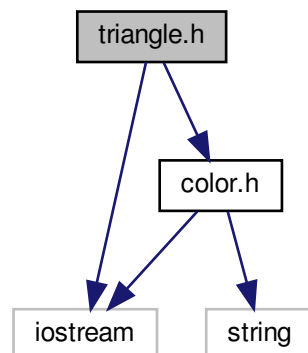
<i>stream</i>	
<i>square</i>	

#### Returns

The stream

## 4.5 triangle.h File Reference

```
#include <iostream>
#include "color.h"
Include dependency graph for triangle.h:
```



## Classes

- class [Triangle](#)

## Functions

- `std::ostream & operator<< (std::ostream &stream, const Triangle &triangle)`

### 4.5.1 Function Documentation

#### 4.5.1.1 `operator<<()`

```
std::ostream& operator<< (  
    std::ostream & stream,  
    const Triangle & triangle )
```

Overload of the output stream to display a triangle

#### Parameters

<i>stream</i>	
<i>triangle</i>	

#### Returns

The stream



# Index

- Circle, [5](#)
  - Circle, [5](#), [6](#)
  - display, [6](#)
  - getColor, [7](#)
  - getRadius, [7](#)
  - getSurface, [7](#)
  - setColor, [7](#), [8](#)
  - setRadius, [8](#)
- circle.h, [25](#)
  - operator<<, [26](#)
- Code
  - Color, [9](#)
- Color, [9](#)
  - Code, [9](#)
  - Color, [9](#)
  - display, [10](#)
  - getColorCode, [10](#)
  - getName, [10](#)
  - setColor, [11](#)
- color.h, [26](#)
  - operator<<, [27](#)
  - toString, [27](#)
- display
  - Circle, [6](#)
  - Color, [10](#)
  - Rectangle, [13](#)
  - Square, [17](#)
  - Triangle, [21](#)
- getBase
  - Triangle, [21](#)
- getColor
  - Circle, [7](#)
  - Rectangle, [13](#)
  - Square, [18](#)
  - Triangle, [21](#)
- getColorCode
  - Color, [10](#)
- getHeight
  - Rectangle, [13](#)
  - Triangle, [22](#)
- getName
  - Color, [10](#)
- getRadius
  - Circle, [7](#)
- getSide
  - Square, [18](#)
- getSurface
  - Circle, [7](#)
- Rectangle, [14](#)
- Square, [18](#)
- Triangle, [22](#)
- getWidth
  - Rectangle, [14](#)
- operator<<
  - circle.h, [26](#)
  - color.h, [27](#)
  - rectangle.h, [29](#)
  - square.h, [30](#)
  - triangle.h, [31](#)
- Rectangle, [11](#)
  - display, [13](#)
  - getColor, [13](#)
  - getHeight, [13](#)
  - getSurface, [14](#)
  - getWidth, [14](#)
  - Rectangle, [12](#)
  - setColor, [14](#), [15](#)
  - setHeight, [15](#)
  - setWidth, [15](#)
- rectangle.h, [28](#)
  - operator<<, [29](#)
- setBase
  - Triangle, [22](#)
- setColor
  - Circle, [7](#), [8](#)
  - Color, [11](#)
  - Rectangle, [14](#), [15](#)
  - Square, [18](#), [19](#)
  - Triangle, [23](#)
- setHeight
  - Rectangle, [15](#)
  - Triangle, [23](#)
- setRadius
  - Circle, [8](#)
- setSide
  - Square, [19](#)
- setWidth
  - Rectangle, [15](#)
- Square, [16](#)
  - display, [17](#)
  - getColor, [18](#)
  - getSide, [18](#)
  - getSurface, [18](#)
  - setColor, [18](#), [19](#)
  - setSide, [19](#)

- Square, [16](#), [17](#)
- square.h, [29](#)
  - operator<<, [30](#)
- toString
  - color.h, [27](#)
- Triangle, [20](#)
  - display, [21](#)
  - getBase, [21](#)
  - getColor, [21](#)
  - getHeight, [22](#)
  - getSurface, [22](#)
  - setBase, [22](#)
  - setColor, [23](#)
  - setHeight, [23](#)
  - Triangle, [20](#), [21](#)
- triangle.h, [30](#)
  - operator<<, [31](#)