Postscript Interpreter

Generated by Doxygen 1.8.10

Wed Apr 20 2016 12:40:14

Contents

Chapter 1

Hierarchical Index

1.1 Class Hierarchy

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

Chapter 2

Class Index

2.1 Class List

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

Chapter 3

File Index

3.1 File List

Here is a list of all files with brief descriptions:

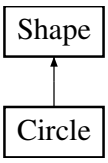
Chapter 4

Class Documentation

4.1 Circle Class Reference

`#include <circle.h>`

Inheritance diagram for Circle:



Public Member Functions

- Circle ()
- Circle (int x, int y, double radius)
- Circle (double radius)
- string draw () const

  generate ps code for shape
- string draw (int x, int y) const

  generate ps code for shape
- double radius ()

  returns radius

Protected Attributes

- double radius_

4.1.1 Detailed Description

Definition at line 14 of file circle.h.

4.1.2 Constructor & Destructor Documentation

4.1.2.1 Circle::Circle ( ) [inline]

Definition at line 17 of file circle.h.

4.1.2.2 Circle::Circle (int *x,* int *y,* double *radius*) [inline]

Definition at line 18 of file circle.h.

4.1.2.3 Circle::Circle (double *radius*) [inline]

Definition at line 19 of file circle.h.

4.1.3 Member Function Documentation

4.1.3.1 string Circle::draw ( ) const [virtual]

generate ps code for shape

generate ps code for drawing shape at default location

Returns

string containing ps code for drawing shape at default location

Reimplemented from Shape.

Definition at line 15 of file circle.cpp.

4.1.3.2 string Circle::draw (int *x,* int *y*) const [virtual]

generate ps code for shape

generate ps code for drawing shape at specified coordinates

Parameters

x	x position for center of shape's desired location
y	y position for center of shape's desired location

Returns

string containing ps code for drawing shape at specified location

Reimplemented from Shape.

Definition at line 19 of file circle.cpp.

4.1.3.3 double Circle::radius ( ) [virtual]

returns radius

returns the radius of a shape, only defined for Polygon and Circle

Returns

if Polygon or Circle, returns the radius. Otherwise returns 0.

Reimplemented from Shape.

Definition at line 11 of file circle.cpp.

4.1.4 Member Data Documentation

4.1.4.1 double Circle::radius_ [protected]

radius of circle

Definition at line 30 of file circle.h.

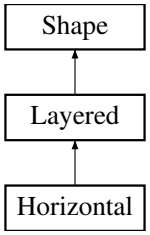
The documentation for this class was generated from the following files:

- circle.h
- circle.cpp

4.2 Horizontal Class Reference

`#include <layered.h>`

Inheritance diagram for Horizontal:



Public Member Functions

- Horizontal ()
- Horizontal (int x, int y, initializer_list< Shape * > shapes)

  Horizontal costructor.
- Horizontal (initializer_list< Shape * > shapes)
- string draw () const

  generate ps code for shape
- string draw (int x, int y) const

  generate ps code for shape

Additional Inherited Members

4.2.1 Detailed Description

Definition at line 35 of file layered.h.

4.2.2 Constructor & Destructor Documentation

4.2.2.1 Horizontal::Horizontal ( ) [inline]

Definition at line 38 of file layered.h.

4.2.2.2 Horizontal::Horizontal (int *x*, int *y*, initializer_list< Shape * > *shapes*)

Horizontal constructor.

constructs a Horizontal shape from a list of Shapes

Parameters

x	x position of center
y	y position of center
shapes	list of pointers to Shapes

Definition at line 55 of file layered.cpp.

4.2.2.3 Horizontal::Horizontal (initializer_list< Shape * > *shapes*) [inline]

Definition at line 40 of file layered.h.

4.2.3 Member Function Documentation

4.2.3.1 string Horizontal::draw ( ) const [virtual]

generate ps code for shape

generate ps code for drawing shape at default location

Returns

string containing ps code for drawing shape at default location

Reimplemented from Layered.

Definition at line 65 of file layered.cpp.

4.2.3.2 string Horizontal::draw (int *x*, int *y*) const [virtual]

generate ps code for shape

generate ps code for drawing shape at specified coordinates

Parameters

x	x position for center of shape's desired location
y	y position for center of shape's desired location

Returns

string containing ps code for drawing shape at specified location

Reimplemented from Layered.

Definition at line 69 of file layered.cpp.

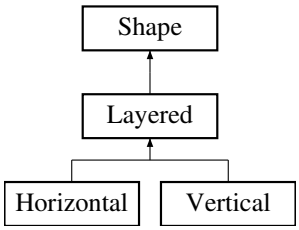
The documentation for this class was generated from the following files:

- layered.h
- layered.cpp

4.3 Layered Class Reference

`#include <layered.h>`

Inheritance diagram for Layered:



Public Member Functions

- Layered ()
- Layered (int x, int y, initializer_list< Shape * > shapes)

  Layered constructor.
- Layered (initializer_list< Shape * > shapes)
- string draw () const

  generate ps code for shape
- string draw (int x, int y) const

  generate ps code for shape

Protected Attributes

- initializer_list< Shape * > shapes_

4.3.1 Detailed Description

Definition at line 17 of file layered.h.

4.3.2 Constructor & Destructor Documentation

4.3.2.1 Layered::Layered ( ) [inline]

Definition at line 20 of file layered.h.

4.3.2.2 Layered::Layered (int *x*, int *y*, initializer_list< Shape * > *shapes*)

Layered constructor.

constructs a Layered shape from a list of shapes

Parameters

x	x position of center
y	y position of center
shapes	list of pointers to Shapes

Definition at line 19 of file layered.cpp.

4.3.2.3 Layered::Layered (initializer_list< Shape * > *shapes*) [inline]

Definition at line 22 of file layered.h.

4.3.3 Member Function Documentation

4.3.3.1 string Layered::draw ( ) const [virtual]

generate ps code for shape

generate ps code for drawing shape at default location

Returns

string containing ps code for drawing shape at default location

Reimplemented from Shape.

Reimplemented in Vertical, and Horizontal.

Definition at line 29 of file layered.cpp.

4.3.3.2 string Layered::draw (int *x*, int *y*) const [virtual]

generate ps code for shape

generate ps code for drawing shape at specified coordinates

Parameters

x	x position for center of shape's desired location
y	y position for center of shape's desired location

Returns

string containing ps code for drawing shape at specified location

Reimplemented from Shape.

Reimplemented in Vertical, and Horizontal.

Definition at line 33 of file layered.cpp.

4.3.4 Member Data Documentation

4.3.4.1 initializer_list<Shape*> Layered::shapes_ [protected]

vector of pointers to Shape objects

Definition at line 31 of file layered.h.

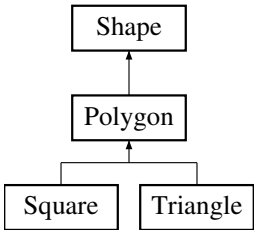
The documentation for this class was generated from the following files:

- layered.h
- layered.cpp

4.4 Polygon Class Reference

`#include <polygon.h>`

Inheritance diagram for Polygon:



Public Member Functions

- Polygon ()
- Polygon (int x, int y, int sides, double length)
- Polygon (int sides, double length)
- string draw () const

  generates ps code for drawing a polygon
- string draw (int x, int y) const

  generates ps code for drawing a polygon
- int numOfSides ()

  returns number of sides
- double sideLength ()

  returns side length
- double radius ()

  returns radius

Protected Attributes

- int numOfSides_
- double sideLength_
- double radius_

4.4.1 Detailed Description

Definition at line 14 of file polygon.h.

4.4.2 Constructor & Destructor Documentation

4.4.2.1 Polygon::Polygon ( ) [inline]

Definition at line 17 of file polygon.h.

4.4.2.2 Polygon::Polygon (int *x*, int *y*, int *sides*, double *length*) [inline]

Definition at line 24 of file polygon.h.

4.4.2.3 Polygon::Polygon (int *sides*, double *length*) [inline]

Definition at line 31 of file polygon.h.

4.4.3 Member Function Documentation

4.4.3.1 string Polygon::draw ( ) const [virtual]

generates ps code for drawing a polygon

returns a string containing the ps code for drawing any equilateral polygon

Returns

string containing ps code

Reimplemented from Shape.

Definition at line 16 of file polygon.cpp.

4.4.3.2 string Polygon::draw (int *x*, int *y*) const [virtual]

generates ps code for drawing a polygon

returns a string containing the ps code for drawing any equilateral polygon

Parameters

x	x position of center
y	y position of center

Returns

string containing ps code

Reimplemented from Shape.

Definition at line 30 of file polygon.cpp.

4.4.3.3 int Polygon::numOfSides ( ) [virtual]

returns number of sides

returns number of sides of a shape, only defined for Polygon

Returns

if Polygon, returns number of sides. Otherwise returns 0.

Reimplemented from Shape.

Definition at line 50 of file polygon.cpp.

4.4.3.4 double Polygon::radius ( ) [virtual]

returns radius

returns the radius of a shape, only defined for Polygon and Circle

Returns

if Polygon or Circle, returns the radius. Otherwise returns 0.

Reimplemented from Shape.

Definition at line 56 of file polygon.cpp.

4.4.3.5 double Polygon::sideLength ( ) [virtual]

returns side length

returns length of a side of a shape, only defined for Polygon

Returns

if Polygon, returns length of a side. Otherwise returns 0.

Reimplemented from Shape.

Definition at line 53 of file polygon.cpp.

4.4.4 Member Data Documentation

4.4.4.1 int Polygon::numOfSides_ [protected]

number of sides of the polygon

Definition at line 46 of file polygon.h.

4.4.4.2 double Polygon::radius_ [protected]

radius of polygon

Definition at line 54 of file polygon.h.

4.4.4.3 double Polygon::sideLength_ [protected]

length of polygon's sides

Definition at line 50 of file polygon.h.

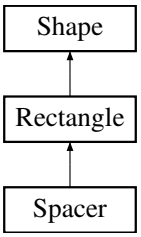
The documentation for this class was generated from the following files:

- polygon.h
- polygon.cpp

4.5 Rectangle Class Reference

`#include <rectangle.h>`

Inheritance diagram for Rectangle:



Public Member Functions

- Rectangle ()

- Rectangle (int x, int y, double w, double h)
- Rectangle (double w, double h)
- string draw () const

  generate ps code for shape
- string draw (int x, int y) const

  generate ps code for shape

Protected Attributes

- double width_
- double height_

4.5.1 Detailed Description

Definition at line 17 of file rectangle.h.

4.5.2 Constructor & Destructor Documentation

4.5.2.1 Rectangle::Rectangle ( ) [inline]

Definition at line 20 of file rectangle.h.

4.5.2.2 Rectangle::Rectangle (int *x*, int *y*, double *w*, double *h*) [inline]

Definition at line 21 of file rectangle.h.

4.5.2.3 Rectangle::Rectangle (double *w*, double *h*) [inline]

Definition at line 22 of file rectangle.h.

4.5.3 Member Function Documentation

4.5.3.1 string Rectangle::draw ( ) const [virtual]

generate ps code for shape

generate ps code for drawing shape at default location

Returns

string containing ps code for drawing shape at default location

Reimplemented from Shape.

Reimplemented in Spacer.

Definition at line 11 of file rectangle.cpp.

4.5.3.2 string Rectangle::draw (int *x*, int *y*) const [virtual]

generate ps code for shape

generate ps code for drawing shape at specified coordinates

Parameters

x	x position for center of shape's desired location
y	y position for center of shape's desired location

Returns

string containing ps code for drawing shape at specified location

Reimplemented from Shape.

Reimplemented in Spacer.

Definition at line 15 of file rectangle.cpp.

4.5.4 Member Data Documentation

4.5.4.1 double Rectangle::height_ [protected]

height of rectangle

Definition at line 35 of file rectangle.h.

4.5.4.2 double Rectangle::width_ [protected]

width of rectangle

Definition at line 31 of file rectangle.h.

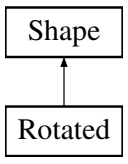
The documentation for this class was generated from the following files:

- rectangle.h
- rectangle.cpp

4.6 Rotated Class Reference

`#include <rotate.h>`

Inheritance diagram for Rotated:



Public Member Functions

- Rotated ()
- Rotated (Shape *shape, int angle)
- string draw () const

  generate ps code for shape
- string draw (int x, int y) const

  generate ps code for shape

Additional Inherited Members

4.6.1 Detailed Description

Definition at line 14 of file rotate.h.

4.6.2 Constructor & Destructor Documentation

4.6.2.1 Rotated::Rotated ( ) [inline]

Definition at line 17 of file rotate.h.

4.6.2.2 Rotated::Rotated (Shape * *shape*, int *angle*)

Definition at line 11 of file rotate.cpp.

4.6.3 Member Function Documentation

4.6.3.1 string Rotated::draw ( ) const [virtual]

generate ps code for shape

generate ps code for drawing shape at default location

Returns

string containing ps code for drawing shape at default location

Reimplemented from Shape.

Definition at line 42 of file rotate.cpp.

4.6.3.2 string Rotated::draw (int *x*, int *y*) const [virtual]

generate ps code for shape

generate ps code for drawing shape at specified coordinates

Parameters

x	x position for center of shape's desired location
y	y position for center of shape's desired location

Returns

string containing ps code for drawing shape at specified location

Reimplemented from Shape.

Definition at line 46 of file rotate.cpp.

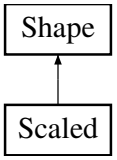
The documentation for this class was generated from the following files:

- rotate.h
- rotate.cpp

4.7 Scaled Class Reference

`#include <scaled.h>`

Inheritance diagram for Scaled:



Public Member Functions

- Scaled ()
- Scaled (Shape *shape, double sx, double sy)
- string draw () const

  generate ps code for shape
- string draw (int x, int y) const

  generate ps code for shape

Additional Inherited Members

4.7.1 Detailed Description

Definition at line 14 of file scaled.h.

4.7.2 Constructor & Destructor Documentation

4.7.2.1 Scaled::Scaled ( ) [inline]

Definition at line 17 of file scaled.h.

4.7.2.2 Scaled::Scaled (Shape * *shape*, double *sx*, double *sy*) [inline]

Definition at line 18 of file scaled.h.

4.7.3 Member Function Documentation

4.7.3.1 string Scaled::draw ( ) const [virtual]

generate ps code for shape

generate ps code for drawing shape at default location

Returns

string containing ps code for drawing shape at default location

Reimplemented from Shape.

Definition at line 11 of file scaled.cpp.

4.7.3.2 string Scaled::draw (int *x*, int *y*) const [virtual]

generate ps code for shape

generate ps code for drawing shape at specified coordinates

Parameters

x	x position for center of shape's desired location
y	y position for center of shape's desired location

Returns

string containing ps code for drawing shape at specified location

Reimplemented from Shape.

Definition at line 15 of file scaled.cpp.

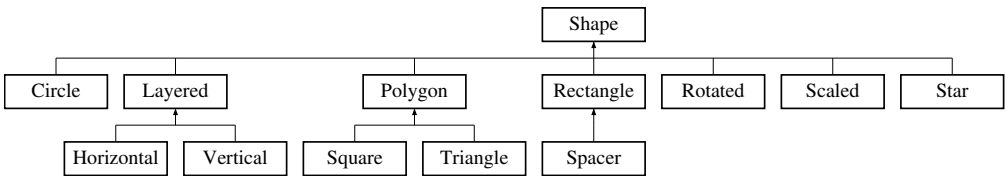
The documentation for this class was generated from the following files:

- scaled.h
- scaled.cpp

4.8 Shape Class Reference

`#include <shape.h>`

Inheritance diagram for Shape:



Public Member Functions

- Shape ()
- Shape (int x, int y, double width, double height)
- Shape (double width, double height)
- virtual ~Shape ()
- virtual string draw () const

  generate ps code for shape
- virtual string draw (int x, int y) const

  generate ps code for shape
- virtual void place (int x, int y)

  change shape position
- string bounds ()

  shape boundary
- double width ()

  get boundary width
- double height ()

  get boundary box height

- int x ()

  get x
- void x (int x)

  set x
- int y ()

  get y
- void y (int y)

  sets y
- string operator() ()

  draw shape at position
- string operator() (int x, int y)

  draw shape at position
- virtual int numOfSides ()

  returns number of sides
- virtual double sideLength ()

  returns side length
- virtual double radius ()

  returns radius

Protected Attributes

- int x_
- int y_
- double boundsWidth_
- double boundsHeight_

4.8.1 Detailed Description

Definition at line 28 of file shape.h.

4.8.2 Constructor & Destructor Documentation

4.8.2.1 Shape::Shape ( ) [inline]

Definition at line 32 of file shape.h.

4.8.2.2 Shape::Shape (int *x*, int *y*, double *width*, double *height*) [inline]

Definition at line 33 of file shape.h.

4.8.2.3 Shape::Shape (double *width*, double *height*) [inline]

Definition at line 34 of file shape.h.

4.8.2.4 virtual Shape::~Shape ( ) [inline], [virtual]

Definition at line 36 of file shape.h.

4.8.3 Member Function Documentation

4.8.3.1 string Shape::bounds ( )

shape boundary

generates string with the values of boundary box witdth and height

Returns

string with boundary

Definition at line 50 of file shape.cpp.

4.8.3.2 string Shape::draw ( ) const [virtual]

generate ps code for shape

generate ps code for drawing shape at default location

Returns

string containing ps code for drawing shape at default location

Reimplemented in Vertical, Horizontal, Polygon, Scaled, Layered, Rectangle, Circle, Spacer, Star, and Rotated.

Definition at line 16 of file shape.cpp.

4.8.3.3 string Shape::draw (int *x*, int *y*) const [virtual]

generate ps code for shape

generate ps code for drawing shape at specified coordinates

Parameters

x	x position for center of shape's desired location
y	y position for center of shape's desired location

Returns

string containing ps code for drawing shape at specified location

Reimplemented in Vertical, Horizontal, Polygon, Scaled, Layered, Rectangle, Circle, Spacer, Star, and Rotated.

Definition at line 29 of file shape.cpp.

4.8.3.4 double Shape::height ( )

get boundary box height

get the double containing the height of the shape's boundary box

Returns

double boundary height

Definition at line 70 of file shape.cpp.

4.8.3.5 int Shape::numOfSides ( ) [virtual]

returns number of sides

returns number of sides of a shape, only defined for Polygon

Returns

if Polygon, returns number of sides. Otherwise returns 0.

Reimplemented in Polygon.

Definition at line 139 of file shape.cpp.

4.8.3.6 string Shape::operator() ( )

draw shape at position

generates ps code for drawing a shape at the shape's set location

Returns

string containing ps code

Definition at line 117 of file shape.cpp.

4.8.3.7 string Shape::operator() (int *x*, int *y*)

draw shape at position

generates ps code for drawing shape at specified location

Parameters

x	x position of center of shape
y	y position of center of shape

Returns

string containing ps code

Definition at line 130 of file shape.cpp.

4.8.3.8 void Shape::place (int *x*, int *y*) [virtual]

change shape position

change shape position to passed coordinates

Parameters

x	x coordinate of new position
y	y coordinate of new position

Definition at line 40 of file shape.cpp.

4.8.3.9 double Shape::radius ( ) [virtual]

returns radius

returns the radius of a shape, only defined for Polygon and Circle

Returns

if Polygon or Circle, returns the radius. Otherwise returns 0.

Reimplemented in Polygon, and Circle.

Definition at line 157 of file shape.cpp.

4.8.3.10 double Shape::sideLength ( ) [virtual]

returns side length

returns length of a side of a shape, only defined for Polygon

Returns

if Polygon, returns length of a side. Otherwise returns 0.

Reimplemented in Polygon.

Definition at line 148 of file shape.cpp.

4.8.3.11 double Shape::width ( )

get boundary width

get the double containing the width of the shape's boundary box

Returns

double boundary width

Definition at line 61 of file shape.cpp.

4.8.3.12 int Shape::x ( )

get x

gets x value of center of shape

Returns

returns x

Definition at line 79 of file shape.cpp.

4.8.3.13 void Shape::x (int *x*)

set x

sets x value of center of shape

Parameters

x	new x value

Definition at line 89 of file shape.cpp.

4.8.3.14 int Shape::y ( )

get y

gets y value of center of shape

Returns

returns y

Definition at line 98 of file shape.cpp.

4.8.3.15 void Shape::y (int *y*)

sets y

sets y value of center of shape

Parameters

y	new y value

Definition at line 108 of file shape.cpp.

4.8.4 Member Data Documentation

4.8.4.1 double Shape::boundsHeight_ [protected]

height of boundary box

Definition at line 76 of file shape.h.

4.8.4.2 double Shape::boundsWidth_ [protected]

width of boundary box

Definition at line 71 of file shape.h.

4.8.4.3 int Shape::x_ [protected]

x coordinate of shape position

Definition at line 61 of file shape.h.

4.8.4.4 int Shape::y_ [protected]

y coordinate of shape position

Definition at line 66 of file shape.h.

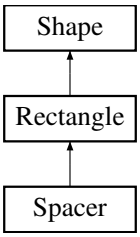
The documentation for this class was generated from the following files:

- shape.h
- shape.cpp

4.9 Spacer Class Reference

`#include <spacer.h>`

Inheritance diagram for Spacer:



Public Member Functions

- Spacer ()
- Spacer (int x, int y, double w, double h)
- Spacer (double w, double h)
- string draw () const

  generate ps code for shape
- string draw (int x, int y) const

  generate ps code for shape

Additional Inherited Members

4.9.1 Detailed Description

Definition at line 14 of file spacer.h.

4.9.2 Constructor & Destructor Documentation

4.9.2.1 Spacer::Spacer ( ) [inline]

Definition at line 17 of file spacer.h.

4.9.2.2 Spacer::Spacer (int *x*, int *y*, double *w*, double *h*) [inline]

Definition at line 18 of file spacer.h.

4.9.2.3 Spacer::Spacer (double *w*, double *h*) [inline]

Definition at line 19 of file spacer.h.

4.9.3 Member Function Documentation

4.9.3.1 string Spacer::draw ( ) const [virtual]

generate ps code for shape

generate ps code for drawing shape at default location

Returns

string containing ps code for drawing shape at default location

Reimplemented from Rectangle.

Definition at line 11 of file spacer.cpp.

4.9.3.2 string Spacer::draw (int *x*, int *y*) const [virtual]

generate ps code for shape

generate ps code for drawing shape at specified coordinates

Parameters

x	x position for center of shape's desired location
y	y position for center of shape's desired location

Returns

string containing ps code for drawing shape at specified location

Reimplemented from Rectangle.

Definition at line 15 of file spacer.cpp.

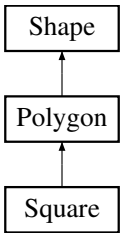
The documentation for this class was generated from the following files:

- spacer.h
- spacer.cpp

4.10 Square Class Reference

`#include <polygon.h>`

Inheritance diagram for Square:



Public Member Functions

- Square ()
- Square (int x, int y, double side)
- Square (double side)

Additional Inherited Members

4.10.1 Detailed Description

Definition at line 71 of file polygon.h.

4.10.2 Constructor & Destructor Documentation

4.10.2.1 Square::Square ( ) [inline]

Definition at line 74 of file polygon.h.

4.10.2.2 Square::Square (int *x*, int *y*, double *side*) [inline]

Definition at line 75 of file polygon.h.

4.10.2.3 Square::Square (double *side*) [inline]

Definition at line 76 of file polygon.h.

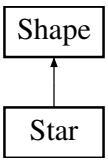
The documentation for this class was generated from the following file:

- polygon.h

4.11 Star Class Reference

`#include <star.h>`

Inheritance diagram for Star:



Public Member Functions

- Star ()
- Star (int x, int y, int n, double oRadius, double iRadius)
- Star (int n, double oRadius, double iRadius)
- string draw () const

  generate ps code for shape
- string draw (int x, int y) const

  generate ps code for shape
- double outerRadius ()
- double innerRadius ()

Additional Inherited Members

4.11.1 Detailed Description

Definition at line 14 of file star.h.

4.11.2 Constructor & Destructor Documentation

4.11.2.1 Star::Star ( ) [inline]

Definition at line 17 of file star.h.

4.11.2.2 Star::Star (int *x*, int *y*, int *n*, double *oRadius*, double *iRadius*) [inline]

Definition at line 18 of file star.h.

4.11.2.3 Star::Star (int *n*, double *oRadius*, double *iRadius*) [inline]

Definition at line 19 of file star.h.

4.11.3 Member Function Documentation

4.11.3.1 string Star::draw ( ) const [virtual]

generate ps code for shape

generate ps code for drawing shape at default location

Returns

string containing ps code for drawing shape at default location

Reimplemented from Shape.

Definition at line 11 of file star.cpp.

4.11.3.2 string Star::draw (int *x*, int *y*) const [virtual]

generate ps code for shape

generate ps code for drawing shape at specified coordinates

Parameters

x	x position for center of shape's desired location
y	y position for center of shape's desired location

Returns

string containing ps code for drawing shape at specified location

Reimplemented from Shape.

Definition at line 15 of file star.cpp.

4.11.3.3 double Star::innerRadius ( )

4.11.3.4 double Star::outerRadius ( )

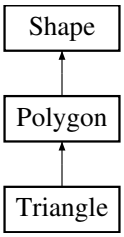
The documentation for this class was generated from the following files:

- star.h
- star.cpp

4.12 Triangle Class Reference

`#include <polygon.h>`

Inheritance diagram for Triangle:



Public Member Functions

- Triangle ()
- Triangle (int x, int y, double side)
- Triangle (double side)

Additional Inherited Members

4.12.1 Detailed Description

Definition at line 60 of file polygon.h.

4.12.2 Constructor & Destructor Documentation

4.12.2.1 Triangle::Triangle ( ) [inline]

Definition at line 63 of file polygon.h.

4.12.2.2 Triangle::Triangle (int *x*, int *y*, double *side*) [inline]

Definition at line 64 of file polygon.h.

4.12.2.3 Triangle::Triangle (double *side*) [inline]

Definition at line 65 of file polygon.h.

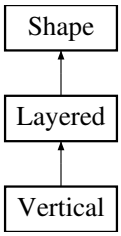
The documentation for this class was generated from the following file:

- polygon.h

4.13 Vertical Class Reference

`#include <layered.h>`

Inheritance diagram for Vertical:



Public Member Functions

- Vertical ()
- Vertical (int x, int y, initializer_list< Shape * > shapes)

  Vertical constructor.
- Vertical (initializer_list< Shape * > shapes)
- string draw () const

  generate ps code for shape
- string draw (int x, int y) const

  generate ps code for shape

Additional Inherited Members

4.13.1 Detailed Description

Definition at line 46 of file layered.h.

4.13.2 Constructor & Destructor Documentation

4.13.2.1 Vertical::Vertical ( ) [inline]

Definition at line 49 of file layered.h.

4.13.2.2 Vertical::Vertical (int *x*, int *y*, initializer_list< Shape * > *shapes*)

Vertical constructor.

constructs a Vertical shape from a list of Shapes

Parameters

x	x position of center
y	y position of center
shapes	list of pointers to Shapes

Definition at line 94 of file layered.cpp.

4.13.2.3 Vertical::Vertical (initializer_list< Shape * > *shapes*) [inline]

Definition at line 51 of file layered.h.

4.13.3 Member Function Documentation

4.13.3.1 string Vertical::draw ( ) const [virtual]

generate ps code for shape

generate ps code for drawing shape at default location

Returns

string containing ps code for drawing shape at default location

Reimplemented from Layered.

Definition at line 104 of file layered.cpp.

4.13.3.2 string Vertical::draw (int *x*, int *y*) const [virtual]

generate ps code for shape

generate ps code for drawing shape at specified coordinates

Parameters

x	x position for center of shape's desired location
y	y position for center of shape's desired location

Returns

string containing ps code for drawing shape at specified location

Reimplemented from Layered.

Definition at line 108 of file layered.cpp.

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

- layered.h
- layered.cpp

Chapter 5

File Documentation

5.1 circle.cpp File Reference

```
#include "circle.h"
```

5.2 circle.h File Reference

```
#include "shape.h"
```

Classes

- class Circle

5.3 layered.cpp File Reference

```
#include "layered.h"
```

5.4 layered.h File Reference

```
#include <initializer_list>
#include "shape.h"
```

Classes

- class Layered
- class Horizontal
- class Vertical

5.5 main.cpp File Reference

```
#include "shape.h"
#include "circle.h"
#include "polygon.h"
#include "rectangle.h"
#include "spacer.h"
#include "layered.h"
#include "rotate.h"
#include "scaled.h"
#include "utils.h"
#include "star.h"
#include <typeinfo>
#include <iostream>
```

Functions

- int main ()

5.5.1 Function Documentation

5.5.1.1 int main ( )

Definition at line 33 of file main.cpp.

5.6 polygon.cpp File Reference

```
#include "polygon.h"
```

5.7 polygon.h File Reference

```
#include "shape.h"
```

Classes

- class Polygon
- class Triangle
- class Square

5.8 rectangle.cpp File Reference

```
#include "rectangle.h"
```

5.9 rectangle.h File Reference

```
#include "shape.h"
#include <sstream>
```

Classes

- class Rectangle

5.10 rotate.cpp File Reference

```
#include "rotate.h"
```

5.11 rotate.h File Reference

```
#include "shape.h"
```

Classes

- class Rotated

5.12 scaled.cpp File Reference

```
#include "scaled.h"
```

5.13 scaled.h File Reference

```
#include "shape.h"
```

Classes

- class Scaled

5.14 shape.cpp File Reference

```
#include "shape.h"
```

Functions

- ostream & operator<< (ostream &os, const Shape &shape)

  overloaded output operator

5.14.1 Function Documentation

5.14.1.1 ostream& operator<< (ostream & *os,* const Shape & *shape*)

overloaded output operator

prints the ps code for drawing the shape to the calling ostream

Parameters

os	target ostream, passed implicitly
shape	Shape to print, passed implicitly

Definition at line 168 of file shape.cpp.

5.15 shape.h File Reference

```
#include <utility>
#include <string>
#include <sstream>
#include <cmath>
#include <iostream>
#include <algorithm>
#include <typeinfo>
#include "utils.h"
```

Classes

- class Shape

Functions

- ostream & operator<< (ostream &os, const Shape &shape)

  overloaded output operator

5.15.1 Function Documentation

5.15.1.1 ostream& operator<< (ostream & *os,* const Shape & *shape*)

overloaded output operator

prints the ps code for drawing the shape to the calling ostream

Parameters

os	target ostream, passed implicitly
shape	Shape to print, passed implicitly

Definition at line 168 of file shape.cpp.

5.16 spacer.cpp File Reference

```
#include "spacer.h"
```

5.17 spacer.h File Reference

```
#include "rectangle.h"
```

Classes

- class Spacer

5.18 star.cpp File Reference

```
#include "star.h"
```

5.19 star.h File Reference

```
#include "shape.h"
```

Classes

- class Star

5.20 test.cpp File Reference

```
#include "catch.hpp"
#include <random>
#include "rectangle.h"
#include "polygon.h"
#include "spacer.h"
#include "circle.h"
#include "layered.h"
#include "rotate.h"
#include "star.h"
#include "scaled.h"
```

Macros

- #define CATCH_CONFIG_MAIN

Functions

- string testPsLine (int x, int y)
- string testPsMove (int x, int y)
- string testPsArc (int x, int y, double r, int startAngle, int endAngle)
- string testPsHeader (int x, int y)
- string testPsFooter ()
- double testCalcX (int k, int n, double l)

- double testCalcY (int k, int n, double l)
- double testGetwidth (int sides, double len)
- double testGetHeight (int n, double l)
- double testGetRadius (int n, double l)
- double testGetConvexX (int k, int n, double r)
- double testGetConvexY (int k, int n, double r)
- double testGetConcaveX (int k, int n, double r)
- double testGetConcaveY (int k, int n, double r)
- string testPolyDraw (int x, int y, int sides, double length)
- string testCircleDraw (int x, int y, double radius)
- TEST_CASE ("Testing utils drawing helpers","[Utils]")
- TEST_CASE ("Testing Centers","[Utils]")
- TEST_CASE ("Testing width and height calculations","[Utils]")
- TEST_CASE ("Simple Shape Default Construction","[Construction]")
- TEST_CASE ("Drawing and Constructing Simple Shapes ","Construction, Drawing")
- TEST_CASE ("Polygon Draw","[Polygon] [draw function]")
- TEST_CASE ("Shape operator <<","[Shape] [operator <<]")
- TEST_CASE ("Shape operator ()","[Shape] [operator ()]")

Variables

- const double ERROR = 0.0000001

5.20.1 Macro Definition Documentation

5.20.1.1 #define CATCH_CONFIG_MAIN

Definition at line 1 of file test.cpp.

5.20.2 Function Documentation

5.20.2.1 TEST_CASE ("Testing utils drawing helpers" , "" *[Utils]*)

Definition at line 150 of file test.cpp.

5.20.2.2 TEST_CASE ("Testing Centers" , "" *[Utils]*)

Definition at line 211 of file test.cpp.

5.20.2.3 TEST_CASE ("Testing width and height calculations" , "" *[Utils]*)

Definition at line 256 of file test.cpp.

5.20.2.4 TEST_CASE ("Simple Shape Default Construction" , "" *[Construction]*)

Definition at line 335 of file test.cpp.

5.20.2.5 TEST_CASE ("Drawing and Constructing Simple Shapes " , " *Construction,* Drawing")

Definition at line 370 of file test.cpp.

5.20.2.6 TEST_CASE ("Polygon Draw" , " " *[Polygon][draw function]*)

Definition at line 658 of file test.cpp.

5.20.2.7 TEST_CASE ("Shape operator <<" , " " *[Shape][operator<<]*)

Definition at line 679 of file test.cpp.

5.20.2.8 TEST_CASE ("Shape operator ()" , " " *[Shape][operator()]*)

Definition at line 709 of file test.cpp.

5.20.2.9 double testCalcX (int *k,* int *n,* double *l*)

Definition at line 63 of file test.cpp.

5.20.2.10 double testCalcY (int *k,* int *n,* double *l*)

Definition at line 68 of file test.cpp.

5.20.2.11 string testCircleDraw (int *x,* int *y,* double *radius*)

Definition at line 140 of file test.cpp.

5.20.2.12 double testGetConcaveX (int *k,* int *n,* double *r*)

Definition at line 113 of file test.cpp.

5.20.2.13 double testGetConcaveY (int *k,* int *n,* double *r*)

Definition at line 117 of file test.cpp.

5.20.2.14 double testGetConvexX (int *k,* int *n,* double *r*)

Definition at line 105 of file test.cpp.

5.20.2.15 double testGetConvexY (int *k,* int *n,* double *r*)

Definition at line 109 of file test.cpp.

5.20.2.16 double testGetHeight (int *n,* double *l*)

Definition at line 93 of file test.cpp.

5.20.2.17 double testGetRadius (int *n,* double *l*)

Definition at line 101 of file test.cpp.

5.20.2.18 double testGetwidth (int *sides*, double *len*)

Definition at line 73 of file test.cpp.

5.20.2.19 string testPolyDraw (int *x*, int *y*, int *sides*, double *length*)

Definition at line 121 of file test.cpp.

5.20.2.20 string testPsArc (int *x*, int *y*, double *r*, int *startAngle*, int *endAngle*)

Definition at line 47 of file test.cpp.

5.20.2.21 string testPsFooter ( )

Definition at line 57 of file test.cpp.

5.20.2.22 string testPsHeader (int *x*, int *y*)

Definition at line 52 of file test.cpp.

5.20.2.23 string testPsLine (int *x*, int *y*)

Definition at line 37 of file test.cpp.

5.20.2.24 string testPsMove (int *x*, int *y*)

Definition at line 42 of file test.cpp.

5.20.3 Variable Documentation

5.20.3.1 const double ERROR = 0.0000001

Definition at line 34 of file test.cpp.

5.21 utils.cpp File Reference

```
#include "utils.h"
```

Functions

- string psBegin ()
  generates ps code for ps file header
- string psPageBreak ()
  generates ps code for printing a page
- string psLine (int x, int y)
  generates ps code for drawing a line
- string psMove (int x, int y)
  generates ps code for moving the cursor

- string psArc (int x, int y, double r, int startAngle, int endAngle)

  generates ps code for drawing an arc
- string psHeader (int x, int y)

  generates ps code for header
- string psFooter ()

  generates ps code for footer
- double calcX (int k, int n, double l)

  calculate x
- double calcY (int k, int n, double l)

  calculate y
- double getWidth (int n, double l)

  cacluate width of polygon
- double getHeight (int n, double l)

  cacluate height of polygon
- double getRadius (int n, double l)

  calculate radius
- double getConvexX (int k, int n, double r)

  get x
- double getConvexY (int k, int n, double r)

  get y
- double getConcaveX (int k, int n, double r)

  get x
- double getConcaveY (int k, int n, double r)

  get y

5.21.1 Function Documentation

5.21.1.1 double calcX (int *k*, int *n*, double *l*)

calculate x

calculate the x coordinate of a given vertex of an equilateral polygon

Parameters

k	vertex number
n	number of sides
l	length of sides

Returns

double, x coordinate of vertex

Definition at line 102 of file utils.cpp.

5.21.1.2 double calcY (int *k*, int *n*, double *l*)

calculate y

calcualte the y coordinate of a given vertex of an equilateral polygon

Parameters

k	vertex number
n	number of sides
l	length of sides

Returns

double, y coordinate of vertex

Definition at line 115 of file utils.cpp.

5.21.1.3 double getConcaveX (int *k*, int *n*, double *r*)

get x

calculate x coordinate of a concave vertex of a star

Parameters

k	vertex number, 0 - (n-1)
n	number of verticies
r	radius of inner circle

Returns

double, x coordinate

Definition at line 205 of file utils.cpp.

5.21.1.4 double getConcaveY (int *k*, int *n*, double *r*)

get y

calculate y coordinate of a concave vertex of a star

Parameters

k	vertex number, 0 - (n-1)
n	number of verticies
r	radius of inner circle

Returns

double, y coordinate

Definition at line 218 of file utils.cpp.

5.21.1.5 double getConvexX (int *k*, int *n*, double *r*)

get x

calculate x coordinate of a convex vertex of a star

Parameters

k	vertex number, 0 - (n-1)
n	number of verticies
r	radius out outer circle

Returns

double, x coordinate

Definition at line 179 of file utils.cpp.

5.21.1.6 double getConvexY (int *k*, int *n*, double *r*)

get y

calculate y coordinate of a convex vertex of a star

Parameters

k	vertex number, 0 - (n-1)
n	number of verticies
r	radius out outer circle

Returns

double, y coordinate

Definition at line 192 of file utils.cpp.

5.21.1.7 double getHeight (int *n*, double *l*)

cacluate height of polygon

calculate the height of any equlateral polygon

Parameters

n	number of sides
l	length of sides

Returns

height of polygon

Definition at line 149 of file utils.cpp.

5.21.1.8 double getRadius (int *n*, double *l*)

calculate radius

calculate radius of an equlateral polygon

Parameters

n	number of sides
l	length of sides

Returns

double, radius

Definition at line 166 of file utils.cpp.

5.21.1.9 double getWidth (int *n*, double *l*)

cacluate width of polygon

calculate the width of any equlateral polygon

Parameters

n	number of sides
l	length of sides

Returns

width of polygon

Definition at line 128 of file utils.cpp.

5.21.1.10 string psArc (int *x*, int *y*, double *r*, int *startAngle*, int *endAngle*)

generates ps code for drawing an arc

returns a string containing the ps code for drawing an arc, in the form 'x y r angle1 angle2 arc'

Parameters

x	c position of the center
y	y position of the center
r	radius of the arc
startAngle	start angle for the arc, from 0 to 360 degrees
endAngle	end angle for the arc, from 0 to 360 degrees

Returns

string containing ps code

Definition at line 67 of file utils.cpp.

5.21.1.11 string psBegin ( )

generates ps code for ps file header

returns a string for ps file headers

Returns

string "%!\n"

Definition at line 16 of file utils.cpp.

5.21.1.12 string psFooter ( )

generates ps code for footer

returns a string containing the ps code for a footer for draw functions in this library

Returns

string containing ps code

Definition at line 89 of file utils.cpp.

5.21.1.13 string psHeader (int *x*, int *y*)

generates ps code for header

returns a string containing the ps code for a header for draw functions in this library

Parameters

x	x position of the center of the shape
y	y position of the center of the shape

Returns

string containing ps code

Definition at line 80 of file utils.cpp.

5.21.1.14 string psLine (int *x*, int *y*)

generates ps code for drawing a line

returns a string containing the ps code for drawing a line, in the form "x y lineto\n"

Parameters

x	x coordinate of endpoint
y	y coordinate of endpoint

Returns

string with ps code

Definition at line 39 of file utils.cpp.

5.21.1.15 string psMove (int *x*, int *y*)

generates ps code for moving the cursor

returns a string containing the ps code for moving the ps cursor, in the form "x y moveto\n"

Parameters

x	x coordinate of target point
y	y coordinate of target point

Returns

string with ps code

Definition at line 52 of file utils.cpp.

5.21.1.16 string psPageBreak ( )

generates ps code for printing a page

returns a string containing the ps code for printing a page

Returns

string "showpage"

Definition at line 26 of file utils.cpp.

5.22 utils.h File Reference

```
#include <string>
#include <sstream>
#include <cmath>
```

Functions

- string psBegin ()
  generates ps code for ps file header
- string psPageBreak ()
  generates ps code for printing a page
- string psLine (int, int)
  generates ps code for drawing a line
- string psMove (int, int)
  generates ps code for moving the cursor
- string psArc (int, int, double, int, int)
  generates ps code for drawing an arc
- string psHeader (int x, int y)
  generates ps code for header
- string psFooter ()
  generates ps code for footer
- double calcX (int, int, double)
  calculate x
- double calcY (int, int, double)
  calculate y
- double getWidth (int, double)
  cacluate width of polygon
- double getHeight (int, double)
  cacluate height of polygon
- double getRadius (int, double)
  calculate radius
- double getConvexX (int k, int n, double r)
  get x
- double getConvexY (int k, int n, double r)
  get y
- double getConcaveX (int k, int n, double r)
  get x
- double getConcaveY (int k, int n, double r)
  get y

5.22.1 Function Documentation

5.22.1.1 double calcX (int *k*, int *n*, double *l*)

calculate x

calculate the x coordinate of a given vertex of an equilateral polygon

Parameters

k	vertex number
n	number of sides
l	length of sides

Returns

double, x coordinate of vertex

Definition at line 102 of file utils.cpp.

5.22.1.2 double calcY (int *k*, int *n*, double *l*)

calculate y

calcualte the y coordinate of a given vertex of an equilateral polygon

Parameters

k	vertex number
n	number of sides
l	length of sides

Returns

double, y coordinate of vertex

Definition at line 115 of file utils.cpp.

5.22.1.3 double getConcaveX (int *k*, int *n*, double *r*)

get x

calculate x coordinate of a concave vertex of a star

Parameters

k	vertex number, 0 - (n-1)
n	number of verticies
r	radius of inner circle

Returns

double, x coordinate

Definition at line 205 of file utils.cpp.

5.22.1.4 double getConcaveY (int *k*, int *n*, double *r*)

get y

calculate y coordinate of a concave vertex of a star

Parameters

k	vertex number, 0 - (n-1)
n	number of verticies
r	radius of inner circle

Returns

double, y coordinate

Definition at line 218 of file utils.cpp.

5.22.1.5 double getConvexX (int *k*, int *n*, double *r*)

get x

calculate x coordinate of a convex vertex of a star

Parameters

k	vertex number, 0 - (n-1)
n	number of verticies
r	radius out outer circle

Returns

double, x coordinate

Definition at line 179 of file utils.cpp.

5.22.1.6 double getConvexY (int *k*, int *n*, double *r*)

get y

calculate y coordinate of a convex vertex of a star

Parameters

k	vertex number, 0 - (n-1)
n	number of verticies
r	radius out outer circle

Returns

double, y coordinate

Definition at line 192 of file utils.cpp.

5.22.1.7 double getHeight (int *n*, double *l*)

cacluate height of polygon

calculate the height of any equlateral polygon

Parameters

n	number of sides

l	length of sides

Returns

height of polygon

Definition at line 149 of file utils.cpp.

5.22.1.8 double getRadius (int *n,* double *l*)

calculate radius

calculate radius of an equlateral polygon

Parameters

n	number of sides
l	length of sides

Returns

double, radius

Definition at line 166 of file utils.cpp.

5.22.1.9 double getWidth (int *n,* double *l*)

cacluate width of polygon

calculate the width of any equlateral polygon

Parameters

n	number of sides
l	length of sides

Returns

width of polygon

Definition at line 128 of file utils.cpp.

5.22.1.10 string psArc (int *x,* int *y,* double *r,* int *startAngle,* int *endAngle*)

generates ps code for drawing an arc

returns a string containing the ps code for drawing an arc, in the form 'x y r angle1 angle2 arc'

Parameters

x	c position of the center
y	y position of the center
r	radius of the arc
startAngle	start angle for the arc, from 0 to 360 degrees

endAngle	end angle for the arc, from 0 to 360 degrees

Returns

string containing ps code

Definition at line 67 of file utils.cpp.

5.22.1.11 string psBegin ( )

generates ps code for ps file header

returns a string for ps file headers

Returns

string "%!\n"

Definition at line 16 of file utils.cpp.

5.22.1.12 string psFooter ( )

generates ps code for footer

returns a string containing the ps code for a footer for draw functions in this library

Returns

string containing ps code

Definition at line 89 of file utils.cpp.

5.22.1.13 string psHeader (int *x*, int *y*)

generates ps code for header

returns a string containing the ps code for a header for draw functions in this library

Parameters

x	x position of the center of the shape
y	y position of the center of the shape

Returns

string containing ps code

Definition at line 80 of file utils.cpp.

5.22.1.14 string psLine (int *x*, int *y*)

generates ps code for drawing a line

returns a string containing the ps code for drawing a line, in the form "x y lineto\n"

Parameters

x	x coordinate of endpoint
y	y coordinate of endpoint

Returns

string with ps code

Definition at line 39 of file utils.cpp.

5.22.1.15 string psMove (int *x*, int *y*)

generates ps code for moving the cursor

returns a string containing the ps code for moving the ps cursor, in the form "x y moveto\n"

Parameters

x	x coordinate of target point
y	y coordinate of target point

Returns

string with ps code

Definition at line 52 of file utils.cpp.

5.22.1.16 string psPageBreak ( )

generates ps code for printing a page

returns a string containing the ps code for printing a page

Returns

string "showpage"

Definition at line 26 of file utils.cpp.

Index