ShapeParser

Generated by Doxygen 1.9.7

Chapter 1

Đ ÁN CUI KÌ MÔN LP TRÌNH HNG ĐI TNG

21120353: Vi Lý Duy Trng
21120432: Vũ Tin Đt

Chapter 2

shape

Chapter 3

Namespace Index

3.1 Namespace List

Here is a list of all namespaces with brief descriptions:

Chapter 4

Hierarchical Index

4.1 Class Hierarchy

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

Chapter 5

Class Index

5.1 Class List

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

Chapter 6

File Index

6.1 File List

Here is a list of all files with brief descriptions:

Chapter 7

Namespace Documentation

7.1 myCircle Namespace Reference

Classes

- class Circle

  Circle class, which inherits from the IShape interface and stores information about a circle shape.

7.2 myEllipse Namespace Reference

Classes

- class Ellipse

  Ellipse class, which inherits from the IShape interface and stores information about an ellipse shape.

7.3 myIsoscelesTrapezoid Namespace Reference

Classes

- class IsoscelesTrapezoid

  IsoscelesTrapezoid class, which inherits from the IShape interface and stores information about an isosceles trapezoid shape.

7.4 myParallelogram Namespace Reference

Classes

- class Parallelogram

  Parallelogram class, which inherits from the IShape interface and stores information about a parallelogram shape.

7.5 myRectangle Namespace Reference

Classes

- class Rectangle

  Rectangle class, which inherits from the IShape interface and stores information about a rectangle shape.

7.6 myRhombus Namespace Reference

Classes

- class Rhombus

  Rhombus class, which inherits from the IShape interface and stores information about a rhombus shape.

7.7 mySquare Namespace Reference

Classes

- class Square

  Square class, which inherits from the IShape interface and stores information about a square shape.

7.8 myTriangle Namespace Reference

Classes

- class Triangle

  Triangle class, which inherits from the IShape interface and stores information about a triangle shape.

Chapter 8

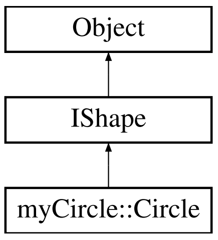
Class Documentation

8.1 myCircle::Circle Class Reference

Circle class, which inherits from the IShape interface and stores information about a circle shape.

```
#include <Circle.h>
```

Inheritance diagram for myCircle::Circle:



Public Member Functions

- Circle (double R) noexcept(false)

  Constructor for Circle class.
- double area () override

  Calculates and returns the area of the circle.
- double perimeter () override

  Calculates and returns the perimeter of the circle.
- string toString () override

  Returns a string representation of the Circle object.
- double radius ()

  Gets the length of the radius of the circle.

- virtual double area ()=0

  Get the area of an object.
- virtual double perimeter ()=0

  Get the perimeter of an object.

- virtual string toString ()=0

  Get a string representation of an object.

Private Attributes

- double _radius

    The length of the radius of the circle.

8.1.1 Detailed Description

Circle class, which inherits from the IShape interface and stores information about a circle shape.

8.1.2 Constructor & Destructor Documentation

8.1.2.1 Circle()

```
myCircle::Circle::Circle (
             double R )
```

Constructor for Circle class.

Parameters

Length	of the radius of the circle

8.1.3 Member Function Documentation

8.1.3.1 area()

```
double myCircle::Circle::area ( )  [override], [virtual]
```

Calculates and returns the area of the circle.

Returns

Area of the circle

Implements IShape.

8.1.3.2 perimeter()

```
double myCircle::Circle::perimeter ( )  [override], [virtual]
```

Calculates and returns the perimeter of the circle.

Returns

Perimeter of the circle

Implements IShape.

8.1.3.3 radius()

```
double myCircle::Circle::radius ( )
```

Gets the length of the radius of the circle.

Returns

Length of the radius of the circle

8.1.3.4 toString()

```
string myCircle::Circle::toString ( )  [override], [virtual]
```

Returns a string representation of the Circle object.

Returns

String representation of the Circle object

Implements Object.

8.1.4 Member Data Documentation

8.1.4.1 _radius

```
double myCircle::Circle::_radius  [private]
```

The length of the radius of the circle.

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

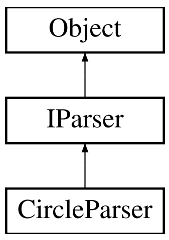
- Circle/Circle.h
- Circle/Circle.cpp

8.2 CircleParser Class Reference

CircleParser class, which inherits from the IParser interface and performs the task of parsing circle shapes.

```
#include <CircleParser.h>
```

Inheritance diagram for CircleParser:



Public Member Functions

- IShape ∗ parse (stringstream data) noexcept(false) override

    Parses the input data and returns a Circle object.
- string toString () override

    Returns a string representation of the CircleParser object.

- virtual IShape ∗ parse (stringstream data) noexcept(false)=0

    Method to parse from user input.

- virtual string toString ()=0

    Get a string representation of an object.

Static Public Member Functions

- static CircleParser ∗ getInstance ()

    Gets the singleton instance of CircleParser.

Private Member Functions

- CircleParser ()=default

    Private constructor for CircleParser class.
- ∼CircleParser ()=default

    Private destructor for CircleParser class.
- CircleParser (const CircleParser &)=delete

    Private copy constructor for CircleParser class.
- CircleParser & operator= (const CircleParser &)=delete

    Private copy assignment operator for CircleParser class.

Static Private Attributes

- static CircleParser ∗ _instance = nullptr

    Singleton instance of CircleParser.

8.2.1 Detailed Description

CircleParser class, which inherits from the IParser interface and performs the task of parsing circle shapes.

8.2.2 Constructor & Destructor Documentation

8.2.2.1 CircleParser() [1/2]

```
CircleParser::CircleParser ( )  [private], [default]
```

Private constructor for CircleParser class.

8.2.2.2 ∼CircleParser()

```
CircleParser::∼CircleParser ( )  [private], [default]
```

Private destructor for CircleParser class.

8.2.2.3 CircleParser() [2/2]

```
CircleParser::CircleParser (
             const CircleParser &  )  [private], [delete]
```

Private copy constructor for CircleParser class.

8.2.3 Member Function Documentation

8.2.3.1 getInstance()

```
CircleParser * CircleParser::getInstance ( )  [static]
```

Gets the singleton instance of CircleParser.

Returns

Singleton instance of CircleParser

8.2.3.2 operator=()

```
CircleParser & CircleParser::operator= (
             const CircleParser &  )  [private], [delete]
```

Private copy assignment operator for CircleParser class.

8.2.3.3 parse()

```
IShape * CircleParser::parse (
             stringstream data )  [override], [virtual]
```

Parses the input data and returns a Circle object.

Parameters

Input	data to parse

Returns

Circle object parsed from the input data

Exceptions

std::exception	if unable to parse the input data

Implements IParser.

8.2.3.4 toString()

```
string CircleParser::toString ( )  [override], [virtual]
```

Returns a string representation of the CircleParser object.

Returns

String representation of the CircleParser object

Implements Object.

8.2.4 Member Data Documentation

8.2.4.1 _instance

```
CircleParser* CircleParser::_instance = nullptr  [inline], [static], [private]
```

Singleton instance of CircleParser.

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

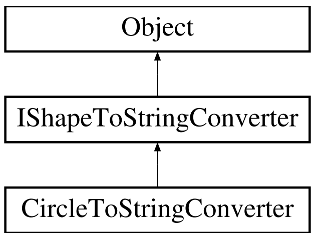
- Circle/CircleParser.h
- Circle/CircleParser.cpp

8.3 CircleToStringConverter Class Reference

CircleToStringConverter class, which inherits from the IShapeToStringConverter interface and performs the task of converting circle shape information to data set.

```
#include <CircleToStringConverter.h>
```

Inheritance diagram for CircleToStringConverter:



Public Member Functions

- SHAPE_DATA convert (IShape ∗) override

  Converts a Circle object to SHAPE_DATA format.
- string toString () override

  Returns a string representation of the CircleToStringConverter object.

- virtual SHAPE_DATA convert (IShape ∗shape)=0

  Method to convert IShape object to SHAPE_DATA data type.

- virtual string toString ()=0

  Get a string representation of an object.

8.3.1 Detailed Description

CircleToStringConverter class, which inherits from the IShapeToStringConverter interface and performs the task of converting circle shape information to data set.

8.3.2 Member Function Documentation

8.3.2.1 convert()

```
SHAPE_DATA CircleToStringConverter::convert (
            IShape ∗ shape )  [override], [virtual]
```

Converts a Circle object to SHAPE_DATA format.

Parameters

Pointer	to the Circle object to be converted

Returns

SHAPE_DATA formatted version of the Circle object

Implements IShapeToStringConverter.

8.3.2.2 toString()

```
string CircleToStringConverter::toString ( )  [override], [virtual]
```

Returns a string representation of the CircleToStringConverter object.

Returns

String representation of the CircleToStringConverter object

Implements Object.

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

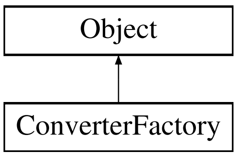
- Circle/CircleToStringConverter.h
- Circle/CircleToStringConverter.cpp

8.4 ConverterFactory Class Reference

Class to manage a list of prototypes for IShapeToStringConverter objects.

```
#include <ConverterFactory.h>
```

Inheritance diagram for ConverterFactory:



Public Member Functions

- void registerWith (string type, IShapeToStringConverter ∗parser)

  Register a new prototype with the factory.
- IShapeToStringConverter ∗ select (string type)

  Select a prototype from the factory based on its type.
- string toString () override

  Return a string representation of the list of prototypes registered with the factory.

- virtual string toString ()=0

  Get a string representation of an object.

Private Attributes

- map< string, IShapeToStringConverter ∗ > _prototypes

8.4.1 Detailed Description

Class to manage a list of prototypes for IShapeToStringConverter objects.

8.4.2 Member Function Documentation

8.4.2.1 registerWith()

```
void ConverterFactory::registerWith (
             string type,
             IShapeToStringConverter ∗ parser )
```

Register a new prototype with the factory.

Parameters

type	The name of the type of the prototype being registered.
parser	A pointer to the prototype object.

8.4.2.2 select()

```
IShapeToStringConverter * ConverterFactory::select (
              string type )
```

Select a prototype from the factory based on its type.

Parameters

type	The name of the type of the prototype being selected.

Returns

A pointer to the selected prototype object. If no prototype is found with the given type, returns null.

8.4.2.3 toString()

```
string ConverterFactory::toString ( )  [override], [virtual]
```

Return a string representation of the list of prototypes registered with the factory.

Returns

A string describing the list of prototypes registered with the factory.

Implements Object.

8.4.3 Member Data Documentation

8.4.3.1 _prototypes

```
map<string, IShapeToStringConverter*> ConverterFactory::_prototypes  [private]
```

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

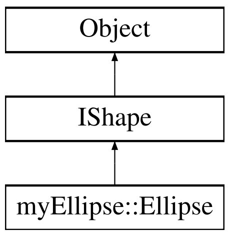
- ShapesParser/ConverterFactory.h
- ShapesParser/ConverterFactory.cpp

8.5 myEllipse::Ellipse Class Reference

Ellipse class, which inherits from the IShape interface and stores information about an ellipse shape.

```
#include <Ellipse.h>
```

Inheritance diagram for myEllipse::Ellipse:



Public Member Functions

- Ellipse (double, double) noexcept(false)
  Constructor for Ellipse class.
- double area () override
  Calculates and returns the area of the ellipse.
- double perimeter () override
  Calculates and returns the perimeter of the ellipse.
- string toString () override
  Returns a string representation of the Ellipse object.
- double semi_minor_axis ()
  Gets the semi-minor axis of the ellipse.
- double semi_major_axis ()
  Gets the semi-major axis of the ellipse.

- virtual double area ()=0
  Get the area of an object.
- virtual double perimeter ()=0
  Get the perimeter of an object.

- virtual string toString ()=0
  Get a string representation of an object.

Private Attributes

- double _semi_minor_axis
  The semi-minor axis of the ellipse.
- double _semi_major_axis
  The semi-major axis of the ellipse.

8.5.1 Detailed Description

Ellipse class, which inherits from the IShape interface and stores information about an ellipse shape.

8.5.2 Constructor & Destructor Documentation

8.5.2.1 Ellipse()

```
myEllipse::Ellipse::Ellipse (
             double semi_minor_axis,
             double semi_major_axis )
```

Constructor for Ellipse class.

Parameters

Semi-minor	axis of the ellipse
Semi-major	axis of the ellipse

8.5.3 Member Function Documentation

8.5.3.1 area()

```
double myEllipse::Ellipse::area ( )  [override], [virtual]
```

Calculates and returns the area of the ellipse.

Returns

Area of the ellipse

Implements IShape.

8.5.3.2 perimeter()

```
double myEllipse::Ellipse::perimeter ( )  [override], [virtual]
```

Calculates and returns the perimeter of the ellipse.

Returns

Perimeter of the ellipse

Implements IShape.

8.5.3.3 semi_major_axis()

```
double myEllipse::Ellipse::semi_major_axis ( )
```

Gets the semi-major axis of the ellipse.

Returns

Semi-major axis of the ellipse

8.5.3.4 semi_minor_axis()

```
double myEllipse::Ellipse::semi_minor_axis ( )
```

Gets the semi-minor axis of the ellipse.

Returns

Semi-minor axis of the ellipse

8.5.3.5 toString()

```
string myEllipse::Ellipse::toString ( )  [override], [virtual]
```

Returns a string representation of the Ellipse object.

Returns

String representation of the Ellipse object

Implements Object.

8.5.4 Member Data Documentation

8.5.4.1 _semi_major_axis

```
double myEllipse::Ellipse::_semi_major_axis  [private]
```

The semi-major axis of the ellipse.

8.5.4.2 _semi_minor_axis

```
double myEllipse::Ellipse::_semi_minor_axis  [private]
```

The semi-minor axis of the ellipse.

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

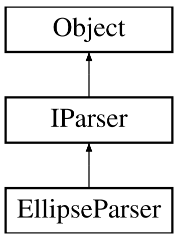
- Ellipse/Ellipse.h
- Ellipse/Ellipse.cpp

8.6 EllipseParser Class Reference

EllipseParser class, which inherits from the IParser interface and performs the task of parsing ellipse shapes.

```
#include <EllipseParser.h>
```

Inheritance diagram for EllipseParser:



Public Member Functions

- IShape ∗ parse (stringstream data) noexcept(false) override

  Parses the input data and returns an Ellipse object.
- string toString () override

  Returns a string representation of the EllipseParser object.

- virtual IShape ∗ parse (stringstream data) noexcept(false)=0

  Method to parse from user input.

- virtual string toString ()=0

  Get a string representation of an object.

Static Public Member Functions

- static EllipseParser ∗ getInstance ()

  Gets the singleton instance of EllipseParser.

Private Member Functions

- EllipseParser ()=default

  Private constructor for EllipseParser class.
- ∼EllipseParser ()=default

  Private destructor for EllipseParser class.
- EllipseParser (const EllipseParser &)=delete

  Private copy constructor for EllipseParser class.
- EllipseParser & operator= (const EllipseParser &)=delete

  Private copy assignment operator for EllipseParser class.

Static Private Attributes

- static EllipseParser ∗ _instance = nullptr

  Singleton instance of EllipseParser.

8.6.1 Detailed Description

EllipseParser class, which inherits from the IParser interface and performs the task of parsing ellipse shapes.

8.6.2 Constructor & Destructor Documentation

8.6.2.1 EllipseParser() `[1/2]`

```
EllipseParser::EllipseParser ( )  [private], [default]
```

Private constructor for EllipseParser class.

8.6.2.2 ∼EllipseParser()

```
EllipseParser::∼EllipseParser ( )  [private], [default]
```

Private destructor for EllipseParser class.

8.6.2.3 EllipseParser() [2/2]

```
EllipseParser::EllipseParser (
             const EllipseParser &  )  [private], [delete]
```

Private copy constructor for EllipseParser class.

8.6.3 Member Function Documentation

8.6.3.1 getInstance()

```
EllipseParser * EllipseParser::getInstance ( )  [static]
```

Gets the singleton instance of EllipseParser.

Returns

Singleton instance of EllipseParser

8.6.3.2 operator=()

```
EllipseParser & EllipseParser::operator= (
             const EllipseParser &  )  [private], [delete]
```

Private copy assignment operator for EllipseParser class.

8.6.3.3 parse()

```
IShape * EllipseParser::parse (
             stringstream data )  [override], [virtual]
```

Parses the input data and returns an Ellipse object.

Parameters

Input	data to parse

Returns

Ellipse object parsed from the input data

Exceptions

std::exception	if unable to parse the input data

Implements IParser.

8.6.3.4 toString()

```
string EllipseParser::toString ( )  [override], [virtual]
```

Returns a string representation of the EllipseParser object.

Returns

String representation of the EllipseParser object

Implements Object.

8.6.4 Member Data Documentation

8.6.4.1 _instance

```
EllipseParser* EllipseParser::_instance = nullptr  [inline], [static], [private]
```

Singleton instance of EllipseParser.

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

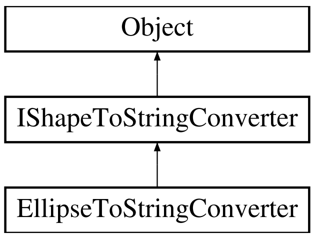
- Ellipse/EllipseParser.h
- Ellipse/EllipseParser.cpp

8.7 EllipseToStringConverter Class Reference

EllipseToStringConverter class, which inherits from the IShapeToStringConverter interface and performs the task of converting ellipse shape information to data set.

```
#include <EllipseToStringConverter.h>
```

Inheritance diagram for EllipseToStringConverter:



Public Member Functions

- SHAPE_DATA convert (IShape ∗) override

  Converts an Ellipse object to SHAPE_DATA format.
- string toString () override

  Returns a string representation of the EllipseToStringConverter object.

- virtual SHAPE_DATA convert (IShape ∗shape)=0

  Method to convert IShape object to SHAPE_DATA data type.

- virtual string toString ()=0

  Get a string representation of an object.

8.7.1 Detailed Description

EllipseToStringConverter class, which inherits from the IShapeToStringConverter interface and performs the task of converting ellipse shape information to data set.

8.7.2 Member Function Documentation

8.7.2.1 convert()

```
SHAPE_DATA EllipseToStringConverter::convert (
             IShape ∗ shape )  [override], [virtual]
```

Converts an Ellipse object to SHAPE_DATA format.

Parameters

Pointer	to the Ellipse object to be converted

Returns

SHAPE_DATA formatted version of the Ellipse object

Implements IShapeToStringConverter.

8.7.2.2 toString()

```
string EllipseToStringConverter::toString ( )  [override], [virtual]
```

Returns a string representation of the EllipseToStringConverter object.

Returns

String representation of the EllipseToStringConverter object

Implements Object.

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

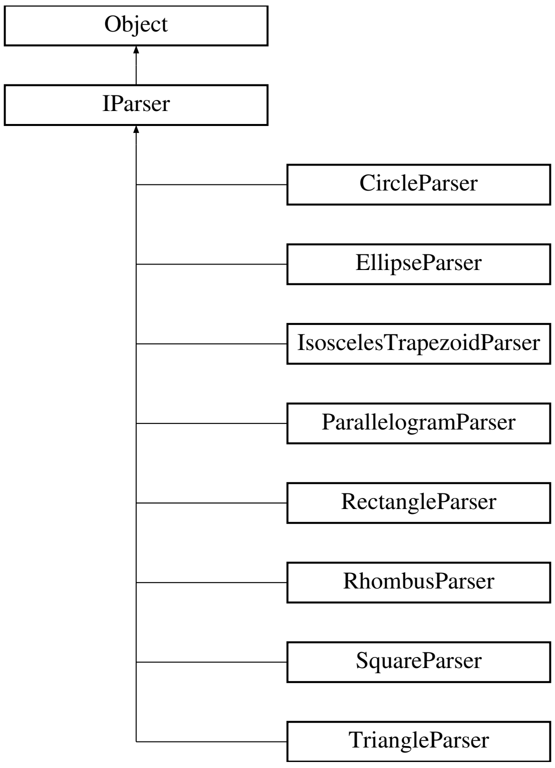
- Ellipse/EllipseToStringConverter.h
- Ellipse/EllipseToStringConverter.cpp

8.8 IParser Class Reference

IParser interface is used for declare methods for subclasses to implement.

```
#include <IParser.h>
```

Inheritance diagram for IParser:



Public Member Functions

- virtual IShape ∗ parse (stringstream data) noexcept(false)=0

  Method to parse from user input.

Public Member Functions inherited from Object

- virtual string toString ()=0

  Get a string representation of an object.

8.8.1 Detailed Description

IParser interface is used for declare methods for subclasses to implement.

8.8.2 Member Function Documentation

8.8.2.1 parse()

```
virtual IShape * IParser::parse (
            stringstream data ) [pure virtual]
```

Method to parse from user input.

Parameters

data	User input

Returns

IShape∗ object

Implemented in CircleParser, EllipseParser, IsoscelesTrapezoidParser, ParallelogramParser, RectangleParser, RhombusParser, SquareParser, and TriangleParser.

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

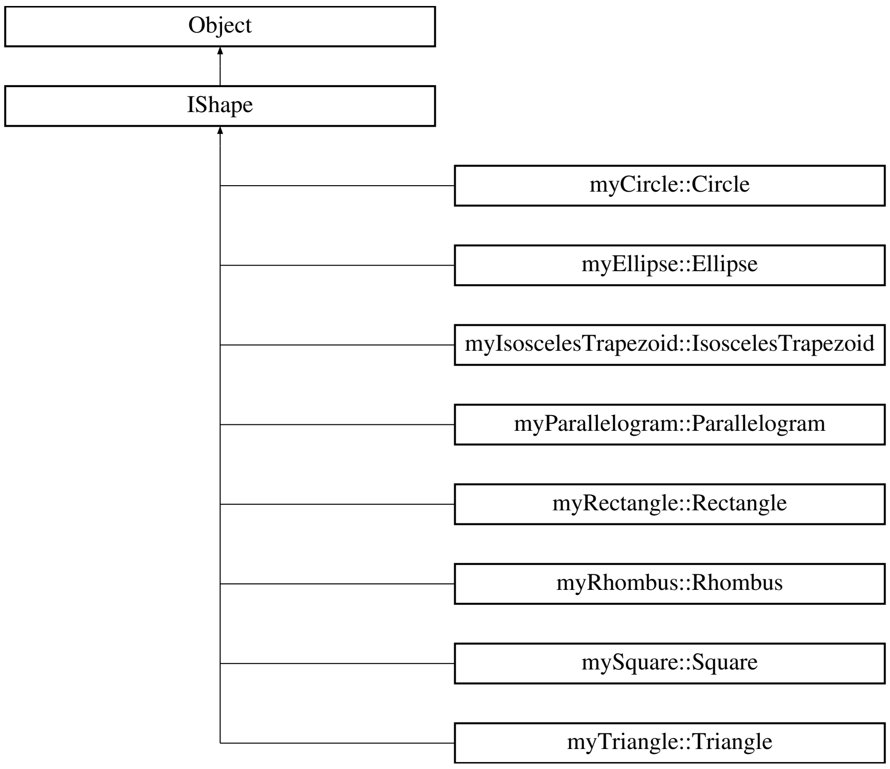
- ShapesParser/IParser.h

8.9 IShape Class Reference

IShape interface is used for declare methods for subclasses to implement.

```
#include <IShape.h>
```

Inheritance diagram for IShape:



Public Member Functions

- virtual double area ()=0

  Get the area of an object.
- virtual double perimeter ()=0

  Get the perimeter of an object.

Public Member Functions inherited from Object

- virtual string toString ()=0

  Get a string representation of an object.

8.9.1 Detailed Description

IShape interface is used for declare methods for subclasses to implement.

8.9.2 Member Function Documentation

8.9.2.1 area()

```
virtual double IShape::area ( )  [pure virtual]
```

Get the area of an object.

Returns

The area of the object

Implemented in myCircle::Circle, myEllipse::Ellipse, myIsoscelesTrapezoid::IsoscelesTrapezoid, myParallelogram::Parallelogram, myRectangle::Rectangle, myRhombus::Rhombus, mySquare::Square, and myTriangle::Triangle.

8.9.2.2 perimeter()

```
virtual double IShape::perimeter ( )  [pure virtual]
```

Get the perimeter of an object.

Returns

The perimeter of an object

Implemented in myCircle::Circle, myEllipse::Ellipse, myIsoscelesTrapezoid::IsoscelesTrapezoid, myParallelogram::Parallelogram, myRectangle::Rectangle, myRhombus::Rhombus, mySquare::Square, and myTriangle::Triangle.

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

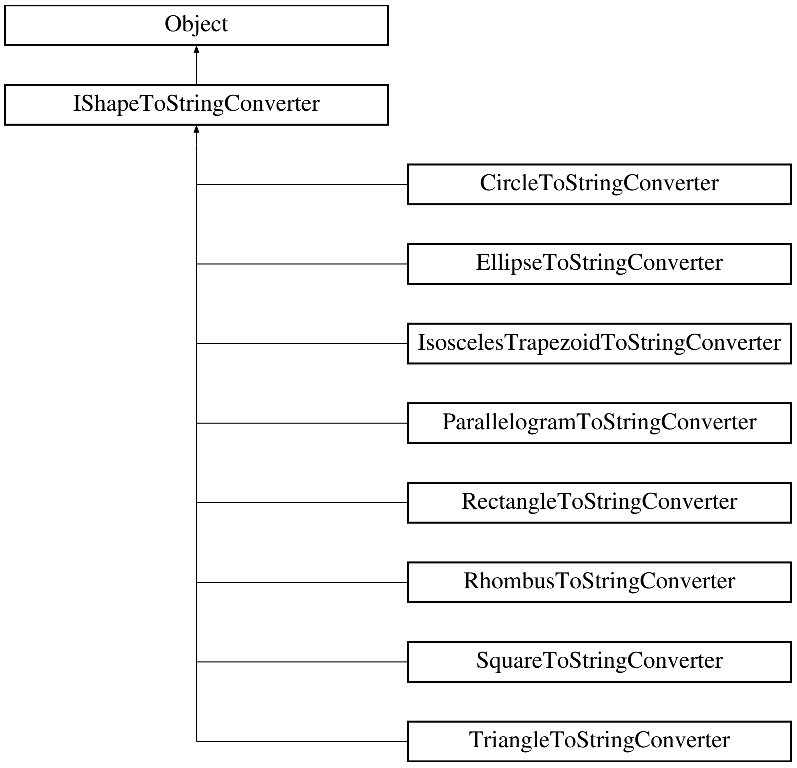
- ShapesParser/IShape.h

8.10 IShapeToStringConverter Class Reference

IShapeToStringConverter interface is used for declare methods for subclasses to implement.

```
#include <IShapeToStringConverter.h>
```

Inheritance diagram for IShapeToStringConverter:



Public Member Functions

- virtual SHAPE_DATA convert (IShape ∗shape)=0
  Method to convert IShape object to SHAPE_DATA data type.

Public Member Functions inherited from Object

- virtual string toString ()=0
  Get a string representation of an object.

8.10.1 Detailed Description

IShapeToStringConverter interface is used for declare methods for subclasses to implement.

8.10.2 Member Function Documentation

8.10.2.1 convert()

```
virtual SHAPE_DATA IShapeToStringConverter::convert (
            IShape ∗ shape )  [pure virtual]
```

Method to convert IShape object to SHAPE_DATA data type.

Parameters

shape	IShape object needs to be converted

Returns

SHAPE_DATA data type of the input object

Implemented in CircleToStringConverter, EllipseToStringConverter, IsoscelesTrapezoidToStringConverter, ParallelogramToStringConverter, RectangleToStringConverter, RhombusToStringConverter, SquareToStringConverter, and TriangleToStringConverter.

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

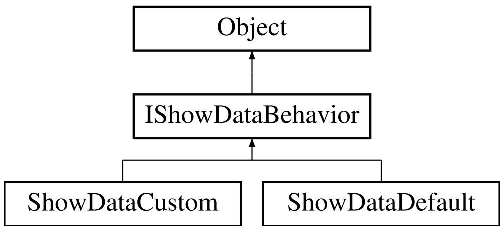
- ShapesParser/IShapeToStringConverter.h

8.11 IShowDataBehavior Class Reference

IShowDataBehavior interface is used for declare methods for subclasses to implement.

```
#include <IShowDataBehavior.h>
```

Inheritance diagram for IShowDataBehavior:



Public Member Functions

- virtual void showData (vector< SHAPE_DATA > data)=0
  Setting method for printing as data line.

Public Member Functions inherited from Object

- virtual string toString ()=0
  Get a string representation of an object.

8.11.1 Detailed Description

IShowDataBehavior interface is used for declare methods for subclasses to implement.

8.11.2 Member Function Documentation

8.11.2.1 showData()

```
virtual void IShowDataBehavior::showData (
            vector< SHAPE_DATA > data )  [pure virtual]
```

Setting method for printing as data line.

Parameters

data	Vector of SHAPE_DATA of IShape objects need to be printed

Implemented in ShowDataCustom, and ShowDataDefault.

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

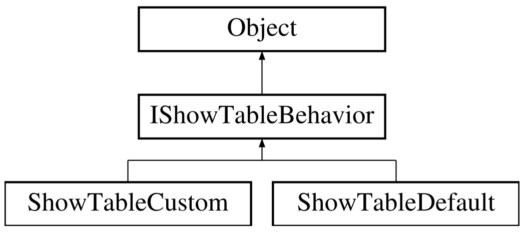
- ShapesParser/IShowDataBehavior.h

8.12 IShowTableBehavior Class Reference

IShowTableBehavior interface is used for declare methods for subclasses to implement.

```
#include <IShowTableBehavior.h>
```

Inheritance diagram for IShowTableBehavior:



Public Member Functions

- virtual void showTable (vector< SHAPE_DATA >)=0

  Setting method for printing as data sheet.

Public Member Functions inherited from Object

- virtual string toString ()=0

  Get a string representation of an object.

8.12.1 Detailed Description

IShowTableBehavior interface is used for declare methods for subclasses to implement.

8.12.2 Member Function Documentation

8.12.2.1 showTable()

```
virtual void IShowTableBehavior::showTable (
            vector< SHAPE_DATA >  )  [pure virtual]
```

Setting method for printing as data sheet.

Parameters

data	Vector of SHAPE_DATA of IShape objects need to be printed

Implemented in ShowTableCustom, and ShowTableDefault.

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

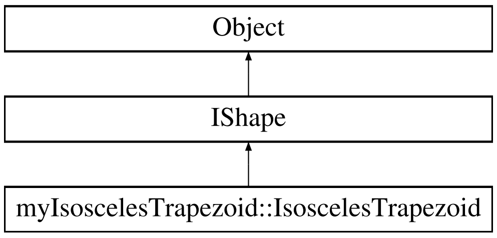
- ShapesParser/IShowTableBehavior.h

8.13 myIsoscelesTrapezoid::IsoscelesTrapezoid Class Reference

IsoscelesTrapezoid class, which inherits from the IShape interface and stores information about an isosceles trapezoid shape.

```
#include <IsoscelesTrapezoid.h>
```

Inheritance diagram for myIsoscelesTrapezoid::IsoscelesTrapezoid:



Public Member Functions

- IsoscelesTrapezoid (double, double, double) noexcept(false)

  Constructor for IsoscelesTrapezoid class.
- double area () override

  Calculates and returns the area of the isosceles trapezoid.
- double perimeter () override

  Calculates and returns the perimeter of the isosceles trapezoid.
- string toString () override

  Returns a string representation of the IsoscelesTrapezoid object.
- double top ()

  Gets the length of the top base of the isosceles trapezoid.
- double base ()

  Gets the length of the bottom base of the isosceles trapezoid.
- double height ()

  Gets the height of the isosceles trapezoid.

- virtual double area ()=0

  Get the area of an object.
- virtual double perimeter ()=0

  Get the perimeter of an object.

- virtual string toString ()=0

  Get a string representation of an object.

Private Attributes

- double _top

  The length of the top base of the isosceles trapezoid.
- double _base

  The length of the bottom base of the isosceles trapezoid.
- double _height

  The height of the isosceles trapezoid.

8.13.1 Detailed Description

IsoscelesTrapezoid class, which inherits from the IShape interface and stores information about an isosceles trapezoid shape.

8.13.2 Constructor & Destructor Documentation

8.13.2.1 IsoscelesTrapezoid()

```
myIsoscelesTrapezoid::IsoscelesTrapezoid::IsoscelesTrapezoid (
            double top,
            double base,
            double height )
```

Constructor for IsoscelesTrapezoid class.

Parameters

Length	of the top base of the isosceles trapezoid
Length	of the bottom base of the isosceles trapezoid
Height	of the isosceles trapezoid

8.13.3 Member Function Documentation

8.13.3.1 area()

```
double myIsoscelesTrapezoid::IsoscelesTrapezoid::area ( )  [override], [virtual]
```

Calculates and returns the area of the isosceles trapezoid.

Returns

Area of the isosceles trapezoid

Implements IShape.

8.13.3.2 base()

```
double myIsoscelesTrapezoid::IsoscelesTrapezoid::base ( )
```

Gets the length of the bottom base of the isosceles trapezoid.

Returns

Length of the bottom base of the isosceles trapezoid

8.13.3.3 height()

```
double myIsoscelesTrapezoid::IsoscelesTrapezoid::height ( )
```

Gets the height of the isosceles trapezoid.

Returns

Height of the isosceles trapezoid

8.13.3.4 perimeter()

```
double myIsoscelesTrapezoid::IsoscelesTrapezoid::perimeter ( )  [override], [virtual]
```

Calculates and returns the perimeter of the isosceles trapezoid.

Returns

Perimeter of the isosceles trapezoid

Implements IShape.

8.13.3.5 top()

```
double myIsoscelesTrapezoid::IsoscelesTrapezoid::top ( )
```

Gets the length of the top base of the isosceles trapezoid.

Returns

Length of the top base of the isosceles trapezoid

8.13.3.6 toString()

```
string myIsoscelesTrapezoid::IsoscelesTrapezoid::toString ( )  [override], [virtual]
```

Returns a string representation of the IsoscelesTrapezoid object.

Returns

String representation of the IsoscelesTrapezoid object

Implements Object.

8.13.4 Member Data Documentation

8.13.4.1 _base

```
double myIsoscelesTrapezoid::IsoscelesTrapezoid::_base  [private]
```

The length of the bottom base of the isosceles trapezoid.

8.13.4.2 _height

```
double myIsoscelesTrapezoid::IsoscelesTrapezoid::_height  [private]
```

The height of the isosceles trapezoid.

8.13.4.3 _top

```
double myIsoscelesTrapezoid::IsoscelesTrapezoid::_top  [private]
```

The length of the top base of the isosceles trapezoid.

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

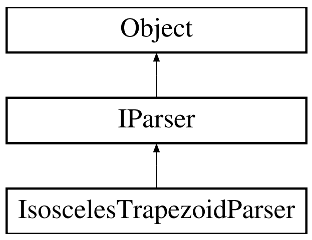
- IsoscelesTrapezoid/IsoscelesTrapezoid.h
- IsoscelesTrapezoid/IsoscelesTrapezoid.cpp

8.14 IsoscelesTrapezoidParser Class Reference

IsoscelesTrapezoidParser class, which inherits from the IParser interface and performs the task of parsing isosceles trapezoid shapes.

```
#include <IsoscelesTrapezoidParser.h>
```

Inheritance diagram for IsoscelesTrapezoidParser:



Public Member Functions

- IShape ∗ parse (stringstream data) noexcept(false) override

  Parses the input data and returns an IsoscelesTrapezoid object.
- string toString () override

  Returns a string representation of the IsoscelesTrapezoidParser object.

- virtual IShape ∗ parse (stringstream data) noexcept(false)=0

  Method to parse from user input.

- virtual string toString ()=0

  Get a string representation of an object.

Static Public Member Functions

- static IsoscelesTrapezoidParser ∗ getInstance ()

  Gets the singleton instance of IsoscelesTrapezoidParser.

Private Member Functions

- IsoscelesTrapezoidParser ()=default

  Private constructor for IsoscelesTrapezoidParser class.
- ∼IsoscelesTrapezoidParser ()=default

  Private destructor for IsoscelesTrapezoidParser class.
- IsoscelesTrapezoidParser (const IsoscelesTrapezoidParser &)=delete

  Private copy constructor for IsoscelesTrapezoidParser class.
- IsoscelesTrapezoidParser & operator= (const IsoscelesTrapezoidParser &)=delete

  Private copy assignment operator for IsoscelesTrapezoidParser class.

Static Private Attributes

- static IsoscelesTrapezoidParser ∗ _instance = nullptr

  Singleton instance of IsoscelesTrapezoidParser.

8.14.1 Detailed Description

IsoscelesTrapezoidParser class, which inherits from the IParser interface and performs the task of parsing isosceles trapezoid shapes.

8.14.2 Constructor & Destructor Documentation

8.14.2.1 IsoscelesTrapezoidParser() [1/2]

```
IsoscelesTrapezoidParser::IsoscelesTrapezoidParser ( )  [private], [default]
```

Private constructor for IsoscelesTrapezoidParser class.

8.14.2.2 ∼IsoscelesTrapezoidParser()

```
IsoscelesTrapezoidParser::∼IsoscelesTrapezoidParser ( )  [private], [default]
```

Private destructor for IsoscelesTrapezoidParser class.

8.14.2.3 IsoscelesTrapezoidParser() [2/2]

```
IsoscelesTrapezoidParser::IsoscelesTrapezoidParser (
            const IsoscelesTrapezoidParser &  )  [private], [delete]
```

Private copy constructor for IsoscelesTrapezoidParser class.

8.14.3 Member Function Documentation

8.14.3.1 getInstance()

```
IsoscelesTrapezoidParser * IsoscelesTrapezoidParser::getInstance ( )  [static]
```

Gets the singleton instance of IsoscelesTrapezoidParser.

Returns

Singleton instance of IsoscelesTrapezoidParser

8.14.3.2 operator=()

```
IsoscelesTrapezoidParser & IsoscelesTrapezoidParser::operator= (
            const IsoscelesTrapezoidParser &  )  [private], [delete]
```

Private copy assignment operator for IsoscelesTrapezoidParser class.

8.14.3.3 parse()

```
IShape * IsoscelesTrapezoidParser::parse (
            stringstream data )  [override], [virtual]
```

Parses the input data and returns an IsoscelesTrapezoid object.

Parameters

Input	data to parse

Returns

IsoscelesTrapezoid object parsed from the input data

Exceptions

std::exception	if unable to parse the input data

Implements IParser.

8.14.3.4 toString()

```
string IsoscelesTrapezoidParser::toString ( )  [override], [virtual]
```

Returns a string representation of the IsoscelesTrapezoidParser object.

Returns

String representation of the IsoscelesTrapezoidParser object

Implements Object.

8.14.4 Member Data Documentation

8.14.4.1 _instance

```
IsoscelesTrapezoidParser* IsoscelesTrapezoidParser::_instance = nullptr  [inline], [static],
[private]
```

Singleton instance of IsoscelesTrapezoidParser.

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

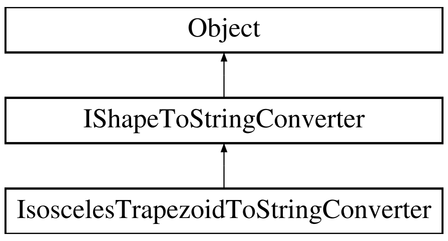
- IsoscelesTrapezoid/IsoscelesTrapezoidParser.h
- IsoscelesTrapezoid/IsoscelesTrapezoidParser.cpp

8.15 IsoscelesTrapezoidToStringConverter Class Reference

IsoscelesTrapezoidToStringConverter class, which inherits from the IShapeToStringConverter interface and performs the task of converting isosceles trapezoid shape information to data set.

```
#include <IsoscelesTrapezoidToStringConverter.h>
```

Inheritance diagram for IsoscelesTrapezoidToStringConverter:



Public Member Functions

- SHAPE_DATA convert (IShape ∗) override

    Converts an IsoscelesTrapezoid object to SHAPE_DATA format.
- string toString () override

    Returns a string representation of the IsoscelesTrapezoidToStringConverter object.

- virtual SHAPE_DATA convert (IShape ∗shape)=0

    Method to convert IShape object to SHAPE_DATA data type.

- virtual string toString ()=0

    Get a string representation of an object.

8.15.1 Detailed Description

IsoscelesTrapezoidToStringConverter class, which inherits from the IShapeToStringConverter interface and performs the task of converting isosceles trapezoid shape information to data set.

8.15.2 Member Function Documentation

8.15.2.1 convert()

```
SHAPE_DATA IsoscelesTrapezoidToStringConverter::convert (
             IShape ∗ shape )  [override], [virtual]
```

Converts an IsoscelesTrapezoid object to SHAPE_DATA format.

Parameters

Pointer	to the IsoscelesTrapezoid object to be converted

Returns

SHAPE_DATA formatted version of the IsoscelesTrapezoid object

Implements IShapeToStringConverter.

8.15.2.2 toString()

```
string IsoscelesTrapezoidToStringConverter::toString ( )  [override], [virtual]
```

Returns a string representation of the IsoscelesTrapezoidToStringConverter object.

Returns

String representation of the IsoscelesTrapezoidToStringConverter object

Implements Object.

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

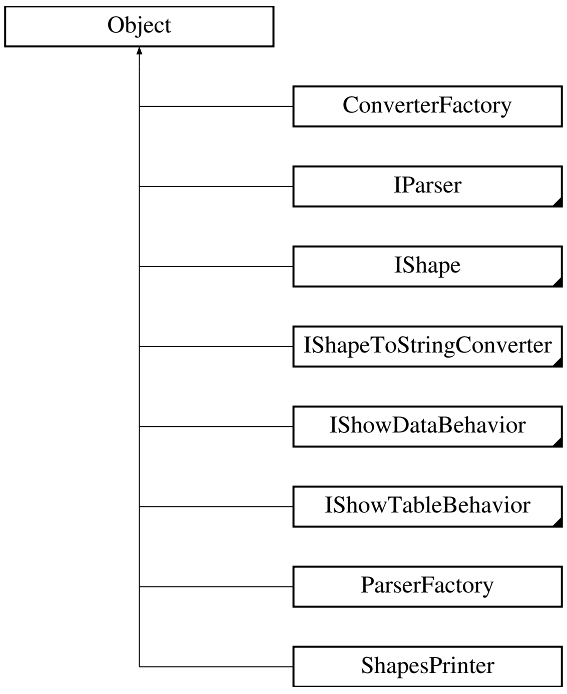
- IsoscelesTrapezoid/IsoscelesTrapezoidToStringConverter.h
- IsoscelesTrapezoid/IsoscelesTrapezoidToStringConverter.cpp

8.16 Object Class Reference

Object class is the largest superclass of all classes in the program.

```
#include <Object.h>
```

Inheritance diagram for Object:



Public Member Functions

- virtual string toString ()=0

  Get a string representation of an object.

8.16.1 Detailed Description

Object class is the largest superclass of all classes in the program.

8.16.2 Member Function Documentation

8.16.2.1 toString()

```
virtual string Object::toString ( )  [pure virtual]
```

Get a string representation of an object.

Returns

The string representation of an object

Implemented in myCircle::Circle, CircleParser, CircleToStringConverter, myEllipse::Ellipse, EllipseParser, EllipseToStringConverter, myIsoscelesTrapezoid::IsoscelesTrapezoid, IsoscelesTrapezoidParser, IsoscelesTrapezoidToStringConverte myParallelogram::Parallelogram, ParallelogramParser, ParallelogramToStringConverter, myRectangle::Rectangle, RectangleParser, RectangleToStringConverter, myRhombus::Rhombus, RhombusParser, RhombusToStringConverter, ConverterFactory, ParserFactory, ShapesPrinter, ShowDataCustom, ShowDataDefault, ShowTableCustom, ShowTableDefault, mySquare::Square, SquareParser, SquareToStringConverter, myTriangle::Triangle, TriangleParser, and TriangleToStringConverter.

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

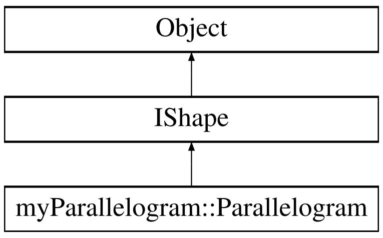
- ShapesParser/Object.h

8.17 myParallelogram::Parallelogram Class Reference

Parallelogram class, which inherits from the IShape interface and stores information about a parallelogram shape.

```
#include <Parallelogram.h>
```

Inheritance diagram for myParallelogram::Parallelogram:



Public Member Functions

- Parallelogram (double, double, double) noexcept(false)

  Constructor for Parallelogram class.
- double area () override

  Calculates and returns the area of the parallelogram.
- double perimeter () override

  Calculates and returns the perimeter of the parallelogram.
- string toString () override

  Returns a string representation of the Parallelogram object.
- double side ()

  Gets the length of one of the sides of the parallelogram.
- double base ()

  Gets the length of the base of the parallelogram.
- double height ()

  Gets the height of the parallelogram.

- virtual double area ()=0

  Get the area of an object.
- virtual double perimeter ()=0

  Get the perimeter of an object.

- virtual string toString ()=0

  Get a string representation of an object.

Private Attributes

- double _side
  The length of one of the sides of the parallelogram.
- double _base
  The length of the base of the parallelogram.
- double _height
  The height of the parallelogram.

8.17.1 Detailed Description

Parallelogram class, which inherits from the IShape interface and stores information about a parallelogram shape.

8.17.2 Constructor & Destructor Documentation

8.17.2.1 Parallelogram()

```
myParallelogram::Parallelogram::Parallelogram (
            double side,
            double base,
            double height )
```

Constructor for Parallelogram class.

Parameters

Length	of one of the sides of the parallelogram
Length	of the base of the parallelogram
Height	of the parallelogram

8.17.3 Member Function Documentation

8.17.3.1 area()

```
double myParallelogram::Parallelogram::area ( )  [override], [virtual]
```

Calculates and returns the area of the parallelogram.

Returns

Area of the parallelogram

Implements IShape.

8.17.3.2 base()

```
double myParallelogram::Parallelogram::base ( )
```

Gets the length of the base of the parallelogram.

Returns

Length of the base of the parallelogram

8.17.3.3 height()

```
double myParallelogram::Parallelogram::height ( )
```

Gets the height of the parallelogram.

Returns

Height of the parallelogram

8.17.3.4 perimeter()

```
double myParallelogram::Parallelogram::perimeter ( )  [override], [virtual]
```

Calculates and returns the perimeter of the parallelogram.

Returns

Perimeter of the parallelogram

Implements IShape.

8.17.3.5 side()

```
double myParallelogram::Parallelogram::side ( )
```

Gets the length of one of the sides of the parallelogram.

Returns

Length of one of the sides of the parallelogram

8.17.3.6 toString()

```
string myParallelogram::Parallelogram::toString ( )  [override], [virtual]
```

Returns a string representation of the Parallelogram object.

Returns

String representation of the Parallelogram object

Implements Object.

8.17.4 Member Data Documentation

8.17.4.1 _base

```
double myParallelogram::Parallelogram::_base  [private]
```

The length of the base of the parallelogram.

8.17.4.2 _height

```
double myParallelogram::Parallelogram::_height  [private]
```

The height of the parallelogram.

8.17.4.3 _side

```
double myParallelogram::Parallelogram::_side  [private]
```

The length of one of the sides of the parallelogram.

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

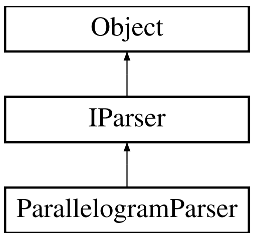
- Parallelogram/Parallelogram.h
- Parallelogram/Parallelogram.cpp

8.18 ParallelogramParser Class Reference

ParallelogramParser class, which inherits from the IParser interface and performs the task of parsing parallelogram shapes.

```
#include <ParallelogramParser.h>
```

Inheritance diagram for ParallelogramParser:



Public Member Functions

- IShape ∗ parse (stringstream data) noexcept(false) override

  Parses the input data and returns a Parallelogram object.
- string toString () override

  Returns a string representation of the ParallelogramParser object.

- virtual IShape ∗ parse (stringstream data) noexcept(false)=0

  Method to parse from user input.

- virtual string toString ()=0

  Get a string representation of an object.

Static Public Member Functions

- static ParallelogramParser ∗ getInstance ()

  Gets the singleton instance of ParallelogramParser.

Private Member Functions

- ParallelogramParser ()=default

  Private constructor for ParallelogramParser class.
- ∼ParallelogramParser ()=default

  Private destructor for ParallelogramParser class.
- ParallelogramParser (const ParallelogramParser &)=delete

  Private copy constructor for ParallelogramParser class.
- ParallelogramParser & operator= (const ParallelogramParser &)=delete

  Private copy assignment operator for ParallelogramParser class.

Static Private Attributes

- static ParallelogramParser ∗ _instance = nullptr

  Singleton instance of ParallelogramParser.

8.18.1 Detailed Description

ParallelogramParser class, which inherits from the IParser interface and performs the task of parsing parallelogram shapes.

8.18.2 Constructor & Destructor Documentation

8.18.2.1 ParallelogramParser() [1/2]

```
ParallelogramParser::ParallelogramParser ( )  [private], [default]
```

Private constructor for ParallelogramParser class.

8.18.2.2 ∼ParallelogramParser()

```
ParallelogramParser::∼ParallelogramParser ( )  [private], [default]
```

Private destructor for ParallelogramParser class.

8.18.2.3 ParallelogramParser() [2/2]

```
ParallelogramParser::ParallelogramParser (
            const ParallelogramParser &  )  [private], [delete]
```

Private copy constructor for ParallelogramParser class.

8.18.3 Member Function Documentation

8.18.3.1 getInstance()

```
ParallelogramParser * ParallelogramParser::getInstance ( )  [static]
```

Gets the singleton instance of ParallelogramParser.

Returns

Singleton instance of ParallelogramParser

8.18.3.2 operator=()

```
ParallelogramParser & ParallelogramParser::operator= (
            const ParallelogramParser &  )  [private], [delete]
```

Private copy assignment operator for ParallelogramParser class.

8.18.3.3 parse()

```
IShape * ParallelogramParser::parse (
            stringstream data )  [override], [virtual]
```

Parses the input data and returns a Parallelogram object.

Parameters

Input	data to parse

Returns

Parallelogram object parsed from the input data

Exceptions

std::exception	if unable to parse the input data

Implements IParser.

8.18.3.4 toString()

```
string ParallelogramParser::toString ( )  [override], [virtual]
```

Returns a string representation of the ParallelogramParser object.

Returns

String representation of the ParallelogramParser object

Implements Object.

8.18.4 Member Data Documentation

8.18.4.1 _instance

```
ParallelogramParser* ParallelogramParser::_instance = nullptr  [inline], [static], [private]
```

Singleton instance of ParallelogramParser.

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

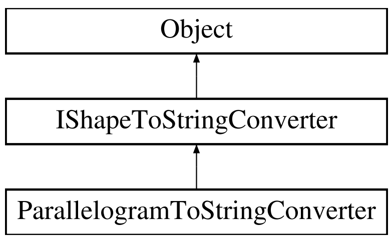
- Parallelogram/ParallelogramParser.h
- Parallelogram/ParallelogramParser.cpp

8.19 ParallelogramToStringConverter Class Reference

ParallelogramToStringConverter class, which inherits from the IShapeToStringConverter interface and performs the task of converting parallelogram shape information to data set.

```
#include <ParallelogramToStringConverter.h>
```

Inheritance diagram for ParallelogramToStringConverter:



Public Member Functions

- SHAPE_DATA convert (IShape ∗) override

  Converts a Parallelogram object to SHAPE_DATA format.
- string toString () override

  Returns a string representation of the ParallelogramToStringConverter object.

- virtual SHAPE_DATA convert (IShape ∗shape)=0

  Method to convert IShape object to SHAPE_DATA data type.

- virtual string toString ()=0

  Get a string representation of an object.

8.19.1 Detailed Description

ParallelogramToStringConverter class, which inherits from the IShapeToStringConverter interface and performs the task of converting parallelogram shape information to data set.

8.19.2 Member Function Documentation

8.19.2.1 convert()

```
SHAPE_DATA ParallelogramToStringConverter::convert (
              IShape * shape )  [override], [virtual]
```

Converts a Parallelogram object to SHAPE_DATA format.

Parameters

Pointer	to the Parallelogram object to be converted

Returns

SHAPE_DATA formatted version of the Parallelogram object

Implements IShapeToStringConverter.

8.19.2.2 toString()

```
string ParallelogramToStringConverter::toString ( )  [override], [virtual]
```

Returns a string representation of the ParallelogramToStringConverter object.

Returns

String representation of the ParallelogramToStringConverter object

Implements Object.

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

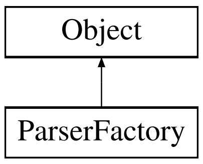
- Parallelogram/ParallelogramToStringConverter.h
- Parallelogram/ParallelogramToStringConverter.cpp

8.20 ParserFactory Class Reference

Class to manage a list of prototypes for IParser objects.

```
#include <ParserFactory.h>
```

Inheritance diagram for ParserFactory:



Public Member Functions

- void registerWith (string type, IParser ∗parser)

  Register a new prototype with the factory.
- IParser ∗ select (string type)

  Select a prototype from the factory based on its type.
- string toString () override

  Return a string representation of the list of prototypes registered with the factory.

- virtual string toString ()=0

  Get a string representation of an object.

Private Attributes

- map< string, IParser ∗ > _prototypes

8.20.1 Detailed Description

Class to manage a list of prototypes for IParser objects.

8.20.2 Member Function Documentation

8.20.2.1 registerWith()

```
void ParserFactory::registerWith (
            string type,
            IParser * parser )
```

Register a new prototype with the factory.

Parameters

type	The name of the type of the prototype being registered.
parser	A pointer to the prototype object.

8.20.2.2 select()

```
IParser * ParserFactory::select (
            string type )
```

Select a prototype from the factory based on its type.

Parameters

type	The name of the type of the prototype being selected.

Returns

A pointer to the selected prototype object. If no prototype is found with the given type, returns null.

8.20.2.3 toString()

```
string ParserFactory::toString ( )  [override], [virtual]
```

Return a string representation of the list of prototypes registered with the factory.

Returns

A string describing the list of prototypes registered with the factory.

Implements Object.

8.20.3 Member Data Documentation

8.20.3.1 _prototypes

```
map<string, IParser*> ParserFactory::_prototypes  [private]
```

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

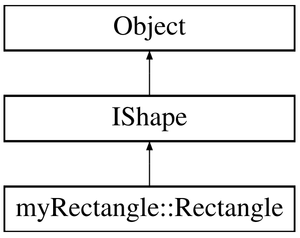
- ShapesParser/ParserFactory.h
- ShapesParser/ParserFactory.cpp

8.21 myRectangle::Rectangle Class Reference

Rectangle class, which inherits from the IShape interface and stores information about a rectangle shape.

```
#include <Rectangle.h>
```

Inheritance diagram for myRectangle::Rectangle:



Public Member Functions

- Rectangle (double, double) noexcept(false)

  Constructor for Rectangle class.
- double area () override

  Calculates and returns the area of the rectangle.
- double perimeter () override

  Calculates and returns the perimeter of the rectangle.
- string toString () override

  Returns a string representation of the Rectangle object.
- double width ()

  Gets the width of the rectangle.
- double height ()

  Gets the height of the rectangle.

- virtual double area ()=0

  Get the area of an object.
- virtual double perimeter ()=0

  Get the perimeter of an object.

- virtual string toString ()=0

  Get a string representation of an object.

Private Attributes

- double _width

  The width of the rectangle.
- double _height

  The height of the rectangle.

8.21.1 Detailed Description

Rectangle class, which inherits from the IShape interface and stores information about a rectangle shape.

8.21.2 Constructor & Destructor Documentation

8.21.2.1 Rectangle()

```
myRectangle::Rectangle::Rectangle (
              double width,
              double height )
```

Constructor for Rectangle class.

Parameters

Width	of the rectangle
Height	of the rectangle

8.21.3 Member Function Documentation

8.21.3.1 area()

```
double myRectangle::Rectangle::area ( )  [override], [virtual]
```

Calculates and returns the area of the rectangle.

Returns

Area of the rectangle

Implements IShape.

8.21.3.2 height()

```
double myRectangle::Rectangle::height ( )
```

Gets the height of the rectangle.

Returns

Height of the rectangle

8.21.3.3 perimeter()

```
double myRectangle::Rectangle::perimeter ( )  [override], [virtual]
```

Calculates and returns the perimeter of the rectangle.

Returns

Perimeter of the rectangle

Implements IShape.

8.21.3.4 toString()

```
string myRectangle::Rectangle::toString ( )  [override], [virtual]
```

Returns a string representation of the Rectangle object.

Returns

String representation of the Rectangle object

Implements Object.

8.21.3.5 width()

```
double myRectangle::Rectangle::width ( )
```

Gets the width of the rectangle.

Returns

Width of the rectangle

8.21.4 Member Data Documentation

8.21.4.1 _height

```
double myRectangle::Rectangle::_height  [private]
```

The height of the rectangle.

8.21.4.2 _width

```
double myRectangle::Rectangle::_width  [private]
```

The width of the rectangle.

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

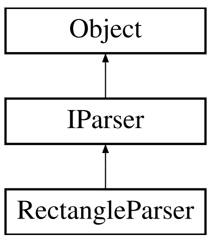
- Rectangle/Rectangle.h
- Rectangle/Rectangle.cpp

8.22 RectangleParser Class Reference

RectangleParser class, which inherits from the IParser interface and performs the task of parsing rectangle shapes.

```
#include <RectangleParser.h>
```

Inheritance diagram for RectangleParser:



Public Member Functions

- IShape ∗ parse (stringstream data) noexcept(false) override

  Parses the input data and returns a Rectangle object.
- string toString () override

  Returns a string representation of the RectangleParser object.

- virtual IShape ∗ parse (stringstream data) noexcept(false)=0

  Method to parse from user input.

- virtual string toString ()=0

  Get a string representation of an object.

Static Public Member Functions

- static RectangleParser ∗ getInstance ()

  Gets the singleton instance of RectangleParser.

Private Member Functions

- RectangleParser ()=default

  Private constructor for RectangleParser class.
- ∼RectangleParser ()=default

  Private destructor for RectangleParser class.
- RectangleParser (const RectangleParser &)=delete

  Private copy constructor for RectangleParser class.
- RectangleParser & operator= (const RectangleParser &)=delete

  Private copy assignment operator for RectangleParser class.

Static Private Attributes

- static RectangleParser ∗ _instance = nullptr

  Singleton instance of RectangleParser.

8.22.1 Detailed Description

RectangleParser class, which inherits from the IParser interface and performs the task of parsing rectangle shapes.

8.22.2 Constructor & Destructor Documentation

8.22.2.1 RectangleParser() `[1/2]`

```
RectangleParser::RectangleParser ( )  [private], [default]
```

Private constructor for RectangleParser class.

8.22.2.2 ∼RectangleParser()

```
RectangleParser::∼RectangleParser ( )  [private], [default]
```

Private destructor for RectangleParser class.

8.22.2.3 RectangleParser() [2/2]

```
RectangleParser::RectangleParser (
            const RectangleParser &  )  [private], [delete]
```

Private copy constructor for RectangleParser class.

8.22.3 Member Function Documentation

8.22.3.1 getInstance()

```
RectangleParser * RectangleParser::getInstance ( )  [static]
```

Gets the singleton instance of RectangleParser.

Returns

Singleton instance of RectangleParser

8.22.3.2 operator=()

```
RectangleParser & RectangleParser::operator= (
            const RectangleParser &  )  [private], [delete]
```

Private copy assignment operator for RectangleParser class.

8.22.3.3 parse()

```
IShape * RectangleParser::parse (
            stringstream data )  [override], [virtual]
```

Parses the input data and returns a Rectangle object.

Parameters

Input	data to parse

Returns

Rectangle object parsed from the input data

Exceptions

std::exception	if unable to parse the input data

Implements IParser.

8.22.3.4 toString()

```
string RectangleParser::toString ( )  [override], [virtual]
```

Returns a string representation of the RectangleParser object.

Returns

String representation of the RectangleParser object

Implements Object.

8.22.4 Member Data Documentation

8.22.4.1 _instance

```
RectangleParser* RectangleParser::_instance = nullptr  [inline], [static], [private]
```

Singleton instance of RectangleParser.

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

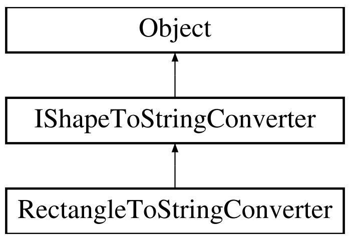
- Rectangle/RectangleParser.h
- Rectangle/RectangleParser.cpp

8.23 RectangleToStringConverter Class Reference

RectangleToStringConverter class, which inherits from the IShapeToStringConverter interface and performs the task of converting rectangle shape information to data set.

```
#include <RectangleToStringConverter.h>
```

Inheritance diagram for RectangleToStringConverter:



Public Member Functions

- SHAPE_DATA convert (IShape ∗) override

    Converts a Rectangle object to SHAPE_DATA format.
- string toString () override

    Returns a string representation of the RectangleToStringConverter object.

- virtual SHAPE_DATA convert (IShape ∗shape)=0

    Method to convert IShape object to SHAPE_DATA data type.

- virtual string toString ()=0

    Get a string representation of an object.

8.23.1 Detailed Description

RectangleToStringConverter class, which inherits from the IShapeToStringConverter interface and performs the task of converting rectangle shape information to data set.

8.23.2 Member Function Documentation

8.23.2.1 convert()

```
SHAPE_DATA RectangleToStringConverter::convert (
             IShape ∗ shape )  [override], [virtual]
```

Converts a Rectangle object to SHAPE_DATA format.

Parameters

Pointer	to the Rectangle object to be converted

Returns

SHAPE_DATA formatted version of the Rectangle object

Implements IShapeToStringConverter.

8.23.2.2 toString()

```
string RectangleToStringConverter::toString ( )  [override], [virtual]
```

Returns a string representation of the RectangleToStringConverter object.

Returns

String representation of the RectangleToStringConverter object

Implements Object.

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

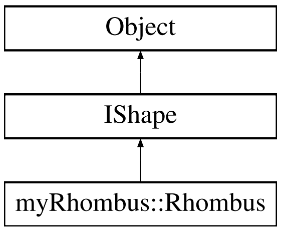
- Rectangle/RectangleToStringConverter.h
- Rectangle/RectangleToStringConverter.cpp

8.24 myRhombus::Rhombus Class Reference

Rhombus class, which inherits from the IShape interface and stores information about a rhombus shape.

```
#include <Rhombus.h>
```

Inheritance diagram for myRhombus::Rhombus:



Public Member Functions

- Rhombus (double, double) noexcept(false)

  Constructor for Rhombus class.
- double area () override

  Calculates and returns the area of the rhombus.
- double perimeter () override

  Calculates and returns the perimeter of the rhombus.
- string toString () override

  Returns a string representation of the Rhombus object.
- double short_diagonal ()

  Gets the length of the short diagonal of the rhombus.
- double long_diagonal ()

  Gets the length of the long diagonal of the rhombus.

- virtual double area ()=0

  Get the area of an object.
- virtual double perimeter ()=0

  Get the perimeter of an object.

- virtual string toString ()=0

  Get a string representation of an object.

Private Attributes

- double _short_diagonal

  The length of the short diagonal of the rhombus.
- double _long_diagonal

  The length of the long diagonal of the rhombus.

8.24.1 Detailed Description

Rhombus class, which inherits from the IShape interface and stores information about a rhombus shape.

8.24.2 Constructor & Destructor Documentation

8.24.2.1 Rhombus()

```
myRhombus::Rhombus::Rhombus (
            double short_diagonal,
            double long_diagonal )
```

Constructor for Rhombus class.

Parameters

Length	of the short diagonal of the rhombus
Length	of the long diagonal of the rhombus

8.24.3 Member Function Documentation

8.24.3.1 area()

```
double myRhombus::Rhombus::area ( ) [override], [virtual]
```

Calculates and returns the area of the rhombus.

Returns

Area of the rhombus

Implements IShape.

8.24.3.2 long_diagonal()

```
double myRhombus::Rhombus::long_diagonal ( )
```

Gets the length of the long diagonal of the rhombus.

Returns

Length of the long diagonal of the rhombus

8.24.3.3 perimeter()

```
double myRhombus::Rhombus::perimeter ( ) [override], [virtual]
```

Calculates and returns the perimeter of the rhombus.

Returns

Perimeter of the rhombus

Implements IShape.

8.24.3.4 short_diagonal()

```
double myRhombus::Rhombus::short_diagonal ( )
```

Gets the length of the short diagonal of the rhombus.

Returns

Length of the short diagonal of the rhombus

8.24.3.5 toString()

```
string myRhombus::Rhombus::toString ( )  [override], [virtual]
```

Returns a string representation of the Rhombus object.

Returns

String representation of the Rhombus object

Implements Object.

8.24.4 Member Data Documentation

8.24.4.1 _long_diagonal

```
double myRhombus::Rhombus::_long_diagonal  [private]
```

The length of the long diagonal of the rhombus.

8.24.4.2 _short_diagonal

```
double myRhombus::Rhombus::_short_diagonal  [private]
```

The length of the short diagonal of the rhombus.

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

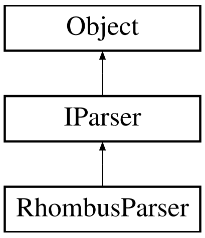
- Rhombus/Rhombus.h
- Rhombus/Rhombus.cpp

8.25 RhombusParser Class Reference

RhombusParser class, which inherits from the IParser interface and performs the task of parsing rhombus shapes.

```
#include <RhombusParser.h>
```

Inheritance diagram for RhombusParser:



Public Member Functions

- IShape ∗ parse (stringstream data) noexcept(false) override

  Parses the input data and returns a Rhombus object.
- string toString () override

  Returns a string representation of the RhombusParser object.

- virtual IShape ∗ parse (stringstream data) noexcept(false)=0

  Method to parse from user input.

- virtual string toString ()=0

  Get a string representation of an object.

Static Public Member Functions

- static RhombusParser ∗ getInstance ()

  Gets the singleton instance of RhombusParser.

Private Member Functions

- RhombusParser ()=default

  Private constructor for RhombusParser class.
- ∼RhombusParser ()=default

  Private destructor for RhombusParser class.
- RhombusParser (const RhombusParser &)=delete

  Private copy constructor for RhombusParser class.
- RhombusParser & operator= (const RhombusParser &)=delete

  Private copy assignment operator for RhombusParser class.

Static Private Attributes

- static RhombusParser ∗ _instance = nullptr

  Singleton instance of RhombusParser.

8.25.1 Detailed Description

RhombusParser class, which inherits from the IParser interface and performs the task of parsing rhombus shapes.

8.25.2 Constructor & Destructor Documentation

8.25.2.1 RhombusParser() [1/2]

```
RhombusParser::RhombusParser ( )  [private], [default]
```

Private constructor for RhombusParser class.

8.25.2.2 ∼RhombusParser()

```
RhombusParser::∼RhombusParser ( )  [private], [default]
```

Private destructor for RhombusParser class.

8.25.2.3 RhombusParser() [2/2]

```
RhombusParser::RhombusParser (
            const RhombusParser &  )  [private], [delete]
```

Private copy constructor for RhombusParser class.

8.25.3 Member Function Documentation

8.25.3.1 getInstance()

```
RhombusParser * RhombusParser::getInstance ( )  [static]
```

Gets the singleton instance of RhombusParser.

Returns

Singleton instance of RhombusParser

8.25.3.2 operator=()

```
RhombusParser & RhombusParser::operator= (
            const RhombusParser &  )  [private], [delete]
```

Private copy assignment operator for RhombusParser class.

8.25.3.3 parse()

```
IShape * RhombusParser::parse (
            stringstream data )  [override], [virtual]
```

Parses the input data and returns a Rhombus object.

Parameters

Input	data to parse

Returns

Rhombus object parsed from the input data

Exceptions

std::exception	if unable to parse the input data

Implements IParser.

8.25.3.4 toString()

```
string RhombusParser::toString ( )  [override], [virtual]
```

Returns a string representation of the RhombusParser object.

Returns

String representation of the RhombusParser object

Implements Object.

8.25.4 Member Data Documentation

8.25.4.1 _instance

```
RhombusParser* RhombusParser::_instance = nullptr  [inline], [static], [private]
```

Singleton instance of RhombusParser.

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

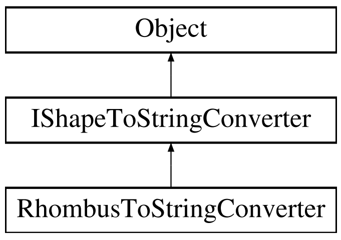
- Rhombus/RhombusParser.h
- Rhombus/RhombusParser.cpp

8.26 RhombusToStringConverter Class Reference

RhombusToStringConverter class, which inherits from the IShapeToStringConverter interface and performs the task of converting rhombus shape information to data set.

```
#include <RhombusToStringConverter.h>
```

Inheritance diagram for RhombusToStringConverter:



Public Member Functions

- SHAPE_DATA convert (IShape ∗) override

  Converts a Rhombus object to SHAPE_DATA format.
- string toString () override

  Returns a string representation of the RhombusToStringConverter object.

- virtual SHAPE_DATA convert (IShape ∗shape)=0

  Method to convert IShape object to SHAPE_DATA data type.

- virtual string toString ()=0

  Get a string representation of an object.

8.26.1 Detailed Description

RhombusToStringConverter class, which inherits from the IShapeToStringConverter interface and performs the task of converting rhombus shape information to data set.

8.26.2 Member Function Documentation

8.26.2.1 convert()

```
SHAPE_DATA RhombusToStringConverter::convert (
            IShape ∗ shape )  [override], [virtual]
```

Converts a Rhombus object to SHAPE_DATA format.

Parameters

Pointer	to the Rhombus object to be converted

Returns

SHAPE_DATA formatted version of the Rhombus object

Implements IShapeToStringConverter.

8.26.2.2 toString()

```
string RhombusToStringConverter::toString ( )  [override], [virtual]
```

Returns a string representation of the RhombusToStringConverter object.

Returns

String representation of the RhombusToStringConverter object

Implements Object.

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

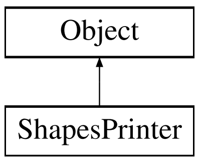
- Rhombus/RhombusToStringConverter.h
- Rhombus/RhombusToStringConverter.cpp

8.27 ShapesPrinter Class Reference

ShapesPrinter class, responsible for printing shapes to the screen.

```
#include <ShapesPrinter.h>
```

Inheritance diagram for ShapesPrinter:



Public Member Functions

- ShapesPrinter ()

  Default constructor for ShapesPrinter class.
- void setShowDataBehavior (IShowDataBehavior ∗)

  Sets the behavior for showing data.
- void performShowDataBehavior (vector< SHAPE_DATA >)

  Formats the way data is displayed.
- void setShowTableBehavior (IShowTableBehavior ∗)

  Sets the behavior for showing tables.
- void performShowTableBehavior (vector< SHAPE_DATA >)

  Formats the way tables are displayed.
- void push (SHAPE_DATA)

  Adds a shape object to the vector.
- void clear ()

  Clears all shape objects from the vector.
- vector< SHAPE_DATA > getData ()

  Gets all the shape objects that have been added.
- string toString () override

  Returns a string representation of the ShapesPrinter object.

- virtual string toString ()=0

  Get a string representation of an object.

Private Attributes

- vector< SHAPE_DATA > _data

  Vector storing information of the shapes.
- IShowTableBehavior ∗ _showTableBehavior

  Outputs in table format.
- IShowDataBehavior ∗ _showDataBehavior

  Outputs in data format.

8.27.1 Detailed Description

ShapesPrinter class, responsible for printing shapes to the screen.

8.27.2 Constructor & Destructor Documentation

8.27.2.1 ShapesPrinter()

```
ShapesPrinter::ShapesPrinter ( )
```

Default constructor for ShapesPrinter class.

8.27.3 Member Function Documentation

8.27.3.1 clear()

```
void ShapesPrinter::clear ( )
```

Clears all shape objects from the vector.

Parameters

Shape	data

8.27.3.2 getData()

```
vector< SHAPE_DATA > ShapesPrinter::getData ( )
```

Gets all the shape objects that have been added.

Returns

Vector containing all shape data

8.27.3.3 performShowDataBehavior()

```
void ShapesPrinter::performShowDataBehavior (
             vector< SHAPE_DATA > data )
```

Formats the way data is displayed.

Parameters

Vector	containing shape data

8.27.3.4 performShowTableBehavior()

```
void ShapesPrinter::performShowTableBehavior (
            vector< SHAPE_DATA > data )
```

Formats the way tables are displayed.

Parameters

Vector	containing shape data

8.27.3.5 push()

```
void ShapesPrinter::push (
            SHAPE_DATA other )
```

Adds a shape object to the vector.

Parameters

Shape	data

8.27.3.6 setShowDataBehavior()

```
void ShapesPrinter::setShowDataBehavior (
            IShowDataBehavior * showDataBehavior )
```

Sets the behavior for showing data.

Parameters

The	show data behavior

8.27.3.7 setShowTableBehavior()

```
void ShapesPrinter::setShowTableBehavior (
            IShowTableBehavior * showTableBehavior )
```

Sets the behavior for showing tables.

Parameters

The	show table behavior

8.27.3.8 toString()

```
string ShapesPrinter::toString ( )  [override], [virtual]
```

Returns a string representation of the ShapesPrinter object.

Returns

String representation of the ShapesPrinter object

Implements Object.

8.27.4 Member Data Documentation

8.27.4.1 _data

```
vector<SHAPE_DATA> ShapesPrinter::_data  [private]
```

Vector storing information of the shapes.

8.27.4.2 _showDataBehavior

```
IShowDataBehavior* ShapesPrinter::_showDataBehavior  [private]
```

Outputs in data format.

8.27.4.3 _showTableBehavior

```
IShowTableBehavior* ShapesPrinter::_showTableBehavior  [private]
```

Outputs in table format.

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

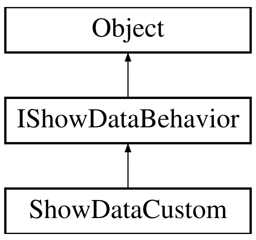
- ShapesParser/ShapesPrinter.h
- ShapesParser/ShapesPrinter.cpp

8.28 ShowDataCustom Class Reference

Custom implementation of IShowDataBehavior, responsible for displaying shape data in a customized format.

```
#include <ShowDataCustom.h>
```

Inheritance diagram for ShowDataCustom:



Public Member Functions

- void showData (vector< SHAPE_DATA >)

    Displays shape data in a customized format.
- string toString () override

    Returns a string representation of the ShowDataCustom object.

- virtual void showData (vector< SHAPE_DATA > data)=0

    Setting method for printing as data line.

- virtual string toString ()=0

    Get a string representation of an object.

8.28.1 Detailed Description

Custom implementation of IShowDataBehavior, responsible for displaying shape data in a customized format.

8.28.2 Member Function Documentation

8.28.2.1 showData()

```
void ShowDataCustom::showData (
            vector< SHAPE_DATA > data )  [virtual]
```

Displays shape data in a customized format.

Parameters

Vector	containing shape data to be displayed

Implements IShowDataBehavior.

8.28.2.2 toString()

```
string ShowDataCustom::toString ( )  [override], [virtual]
```

Returns a string representation of the ShowDataCustom object.

Returns

String representation of the ShowDataCustom object

Implements Object.

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

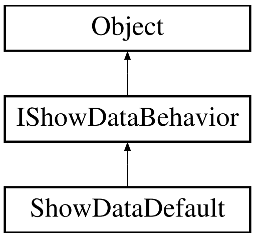
- ShapesParser/ShowDataCustom.h
- ShapesParser/ShowDataCustom.cpp

8.29 ShowDataDefault Class Reference

Default implementation of IShowDataBehavior, responsible for displaying shape data in a default format.

```
#include <ShowDataDefault.h>
```

Inheritance diagram for ShowDataDefault:



Public Member Functions

- void showData (vector< SHAPE_DATA >)

  Displays shape data in a default format.
- string toString () override

  Returns a string representation of the ShowDataDefault object.

- virtual void showData (vector< SHAPE_DATA > data)=0

  Setting method for printing as data line.

- virtual string toString ()=0

  Get a string representation of an object.

8.29.1 Detailed Description

Default implementation of IShowDataBehavior, responsible for displaying shape data in a default format.

8.29.2 Member Function Documentation

8.29.2.1 showData()

```
void ShowDataDefault::showData (
             vector< SHAPE_DATA > data )  [virtual]
```

Displays shape data in a default format.

Parameters

Vector	containing shape data to be displayed

Implements IShowDataBehavior.

8.29.2.2 toString()

```
string ShowDataDefault::toString ( )  [override], [virtual]
```

Returns a string representation of the ShowDataDefault object.

Returns

String representation of the ShowDataDefault object

Implements Object.

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

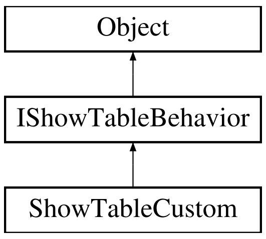
- ShapesParser/ShowDataDefault.h
- ShapesParser/ShowDataDefault.cpp

8.30 ShowTableCustom Class Reference

Custom implementation of IShowTableBehavior, responsible for displaying shape data in a customized table format.

```
#include <ShowTableCustom.h>
```

Inheritance diagram for ShowTableCustom:



Public Member Functions

- void showTable (vector< SHAPE_DATA >)

  Displays shape data in a customized table format.
- string toString () override

  Returns a string representation of the ShowTableCustom object.

- virtual void showTable (vector< SHAPE_DATA >)=0

  Setting method for printing as data sheet.

- virtual string toString ()=0

  Get a string representation of an object.

8.30.1 Detailed Description

Custom implementation of IShowTableBehavior, responsible for displaying shape data in a customized table format.

8.30.2 Member Function Documentation

8.30.2.1 showTable()

```
void ShowTableCustom::showTable (
             vector< SHAPE_DATA > data )  [virtual]
```

Displays shape data in a customized table format.

Parameters

Vector	containing shape data to be displayed

Implements IShowTableBehavior.

8.30.2.2 toString()

```
string ShowTableCustom::toString ( )  [override], [virtual]
```

Returns a string representation of the ShowTableCustom object.

Returns

String representation of the ShowTableCustom object

Implements Object.

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

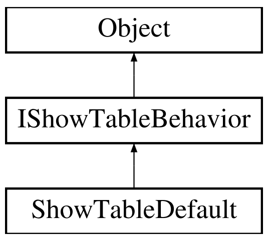
- ShapesParser/ShowTableCustom.h
- ShapesParser/ShowTableCustom.cpp

8.31 ShowTableDefault Class Reference

Default implementation of IShowTableBehavior, responsible for displaying shape data in a default table format.

```
#include <ShowTableDefault.h>
```

Inheritance diagram for ShowTableDefault:



Public Member Functions

- void showTable (vector< SHAPE_DATA >)

  Displays shape data in a default table format.
- string toString () override

  Returns a string representation of the ShowTableDefault object.

- virtual void showTable (vector< SHAPE_DATA >)=0

  Setting method for printing as data sheet.

- virtual string toString ()=0

  Get a string representation of an object.

8.31.1 Detailed Description

Default implementation of IShowTableBehavior, responsible for displaying shape data in a default table format.

8.31.2 Member Function Documentation

8.31.2.1 showTable()

```
void ShowTableDefault::showTable (
             vector< SHAPE_DATA > data )  [virtual]
```

Displays shape data in a default table format.

Parameters

Vector	containing shape data to be displayed

Implements IShowTableBehavior.

8.31.2.2 toString()

```
string ShowTableDefault::toString ( )  [override], [virtual]
```

Returns a string representation of the ShowTableDefault object.

Returns

String representation of the ShowTableDefault object

Implements Object.

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

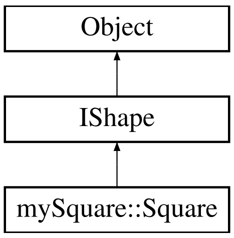
- ShapesParser/ShowTableDefault.h
- ShapesParser/ShowTableDefault.cpp

8.32 mySquare::Square Class Reference

Square class, which inherits from the IShape interface and stores information about a square shape.

```
#include <Square.h>
```

Inheritance diagram for mySquare::Square:



Public Member Functions

- Square (double) noexcept(false)

  Constructor for Square class.
- double area () override

  Calculates and returns the area of the square.
- double perimeter () override

  Calculates and returns the perimeter of the square.
- std::string toString () override

  Returns a string representation of the Square object.
- double length ()

  Gets the length of the sides of the square.

- virtual double area ()=0

  Get the area of an object.
- virtual double perimeter ()=0

  Get the perimeter of an object.

- virtual string toString ()=0

  Get a string representation of an object.

Private Attributes

- double _length

  The length of the sides of the square.

8.32.1 Detailed Description

Square class, which inherits from the IShape interface and stores information about a square shape.

8.32.2 Constructor & Destructor Documentation

8.32.2.1 Square()

```
mySquare::Square::Square (
                double len )
```

Constructor for Square class.

Parameters

Length	of the sides of the square

8.32.3 Member Function Documentation

8.32.3.1 area()

```
double mySquare::Square::area ( )  [override], [virtual]
```

Calculates and returns the area of the square.

Returns

Area of the square

Implements IShape.

8.32.3.2 length()

```
double mySquare::Square::length ( )
```

Gets the length of the sides of the square.

Returns

Length of the sides of the square

8.32.3.3 perimeter()

```
double mySquare::Square::perimeter ( )  [override], [virtual]
```

Calculates and returns the perimeter of the square.

Returns

Perimeter of the square

Implements IShape.

8.32.3.4 toString()

```
string mySquare::Square::toString ( )  [override], [virtual]
```

Returns a string representation of the Square object.

Returns

String representation of the Square object

Implements Object.

8.32.4 Member Data Documentation

8.32.4.1 _length

```
double mySquare::Square::_length  [private]
```

The length of the sides of the square.

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

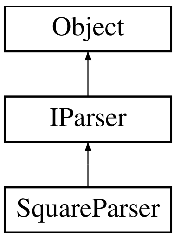
- Square/Square.h
- Square/Square.cpp

8.33 SquareParser Class Reference

SquareParser class, which inherits from the IParser interface and performs the task of parsing square shapes.

```
#include <SquareParser.h>
```

Inheritance diagram for SquareParser:



Public Member Functions

- IShape ∗ parse (stringstream data) noexcept(false) override

  Parses the input data and returns a Square object.
- string toString () override

  Returns a string representation of the SquareParser object.

- virtual IShape ∗ parse (stringstream data) noexcept(false)=0

  Method to parse from user input.

- virtual string toString ()=0

  Get a string representation of an object.

Static Public Member Functions

- static SquareParser ∗ getInstance ()

  Gets the singleton instance of SquareParser.

Private Member Functions

- SquareParser ()=default
  Private constructor for SquareParser class.
- ∼SquareParser ()=default
  Private destructor for SquareParser class.
- SquareParser (const SquareParser &)=delete
  Private copy constructor for SquareParser class.
- SquareParser & operator= (const SquareParser &)=delete
  Private copy assignment operator for SquareParser class.

Static Private Attributes

- static SquareParser ∗ _instance = nullptr
  Singleton instance of SquareParser.

8.33.1 Detailed Description

SquareParser class, which inherits from the IParser interface and performs the task of parsing square shapes.

8.33.2 Constructor & Destructor Documentation

8.33.2.1 SquareParser() [1/2]

```
SquareParser::SquareParser ( )  [private], [default]
```

Private constructor for SquareParser class.

8.33.2.2 ∼SquareParser()

```
SquareParser::∼SquareParser ( )  [private], [default]
```

Private destructor for SquareParser class.

8.33.2.3 SquareParser() [2/2]

```
SquareParser::SquareParser (
              const SquareParser &  )  [private], [delete]
```

Private copy constructor for SquareParser class.

8.33.3 Member Function Documentation

8.33.3.1 getInstance()

```
SquareParser * SquareParser::getInstance ( )  [static]
```

Gets the singleton instance of SquareParser.

Returns

Singleton instance of SquareParser

8.33.3.2 operator=()

```
SquareParser & SquareParser::operator= (
             const SquareParser &  )  [private], [delete]
```

Private copy assignment operator for SquareParser class.

8.33.3.3 parse()

```
IShape * SquareParser::parse (
             stringstream data )  [override], [virtual]
```

Parses the input data and returns a Square object.

Parameters

Input	data to parse

Returns

Square object parsed from the input data

Exceptions

std::exception	if unable to parse the input data

Implements IParser.

8.33.3.4 toString()

```
string SquareParser::toString ( )  [override], [virtual]
```

Returns a string representation of the SquareParser object.

Returns

String representation of the SquareParser object

Implements Object.

8.33.4 Member Data Documentation

8.33.4.1 _instance

```
SquareParser* SquareParser::_instance = nullptr  [inline], [static], [private]
```

Singleton instance of SquareParser.

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

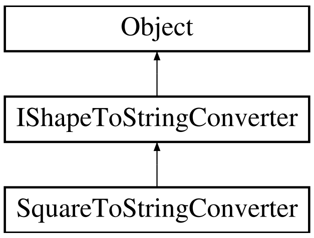
- Square/SquareParser.h
- Square/SquareParser.cpp

8.34 SquareToStringConverter Class Reference

SquareToStringConverter class, which inherits from the IShapeToStringConverter interface and performs the task of converting square shape information to data set.

```
#include <SquareToStringConverter.h>
```

Inheritance diagram for SquareToStringConverter:



Public Member Functions

- SHAPE_DATA convert (IShape ∗) override

  Converts a Square object to SHAPE_DATA format.
- string toString () override

  Returns a string representation of the SquareToStringConverter object.

- virtual SHAPE_DATA convert (IShape ∗shape)=0

  Method to convert IShape object to SHAPE_DATA data type.

- virtual string toString ()=0

  Get a string representation of an object.

8.34.1 Detailed Description

SquareToStringConverter class, which inherits from the IShapeToStringConverter interface and performs the task of converting square shape information to data set.

8.34.2 Member Function Documentation

8.34.2.1 convert()

```
SHAPE_DATA SquareToStringConverter::convert (
             IShape * shape )  [override], [virtual]
```

Converts a Square object to SHAPE_DATA format.

Parameters

Pointer	to the Square object to be converted

Returns

SHAPE_DATA formatted version of the Square object

Implements IShapeToStringConverter.

8.34.2.2 toString()

```
string SquareToStringConverter::toString ( )  [override], [virtual]
```

Returns a string representation of the SquareToStringConverter object.

Returns

String representation of the SquareToStringConverter object

Implements Object.

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

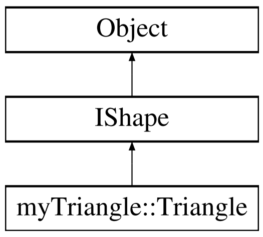
- Square/SquareToStringConverter.h
- Square/SquareToStringConverter.cpp

8.35 myTriangle::Triangle Class Reference

Triangle class, which inherits from the IShape interface and stores information about a triangle shape.

`#include <Triangle.h>`

Inheritance diagram for myTriangle::Triangle:



Public Member Functions

- Triangle (double, double, double) noexcept(false)

  Constructor for Triangle class.
- double area () override

  Calculates and returns the area of the triangle.
- double perimeter () override

  Calculates and returns the perimeter of the triangle.
- string toString () override

  Returns a string representation of the Triangle object.
- double first_edge ()

  Gets the length of the first edge of the triangle.
- double second_edge ()

  Gets the length of the second edge of the triangle.
- double third_edge ()

  Gets the length of the third edge of the triangle.

- virtual double area ()=0

  Get the area of an object.
- virtual double perimeter ()=0

  Get the perimeter of an object.

- virtual string toString ()=0

  Get a string representation of an object.

Private Attributes

- double _first_edge

  The length of the first edge of the triangle.
- double _second_edge

  The length of the second edge of the triangle.
- double _third_edge

  The length of the third edge of the triangle.

8.35.1 Detailed Description

Triangle class, which inherits from the IShape interface and stores information about a triangle shape.

8.35.2 Constructor & Destructor Documentation

8.35.2.1 Triangle()

```
myTriangle::Triangle::Triangle (
            double firstEgde,
            double secondEdge,
            double thirdEdge )
```

Constructor for Triangle class.

Parameters

Length	of the first edge of the triangle
Length	of the second edge of the triangle
Length	of the third edge of the triangle

8.35.3 Member Function Documentation

8.35.3.1 area()

```
double myTriangle::Triangle::area ( )  [override], [virtual]
```

Calculates and returns the area of the triangle.

Returns

Area of the triangle

Implements IShape.

8.35.3.2 first_edge()

```
double myTriangle::Triangle::first_edge ( )
```

Gets the length of the first edge of the triangle.

Returns

Length of the first edge of the triangle

8.35.3.3 perimeter()

```
double myTriangle::Triangle::perimeter ( )  [override], [virtual]
```

Calculates and returns the perimeter of the triangle.

Returns

Perimeter of the triangle

Implements IShape.

8.35.3.4 second_edge()

```
double myTriangle::Triangle::second_edge ( )
```

Gets the length of the second edge of the triangle.

Returns

Length of the second edge of the triangle

8.35.3.5 third_edge()

```
double myTriangle::Triangle::third_edge ( )
```

Gets the length of the third edge of the triangle.

Returns

Length of the third edge of the triangle

8.35.3.6 toString()

```
string myTriangle::Triangle::toString ( )  [override], [virtual]
```

Returns a string representation of the Triangle object.

Returns

String representation of the Triangle object

Implements Object.

8.35.4 Member Data Documentation

8.35.4.1 _first_edge

```
double myTriangle::Triangle::_first_edge  [private]
```

The length of the first edge of the triangle.

8.35.4.2 _second_edge

```
double myTriangle::Triangle::_second_edge  [private]
```

The length of the second edge of the triangle.

8.35.4.3 _third_edge

```
double myTriangle::Triangle::_third_edge  [private]
```

The length of the third edge of the triangle.

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

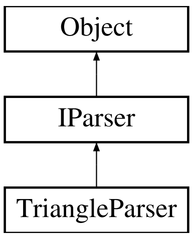
- Triangle/Triangle.h
- Triangle/Triangle.cpp

8.36 TriangleParser Class Reference

TriangleParser class, which inherits from the IParser interface and performs the task of parsing triangle shapes.

```
#include <TriangleParser.h>
```

Inheritance diagram for TriangleParser:



Public Member Functions

- IShape ∗ parse (stringstream data) noexcept(false) override

    Parses the input data and returns a Triangle object.
- string toString () override

    Returns a string representation of the TriangleParser object.

- virtual IShape ∗ parse (stringstream data) noexcept(false)=0

    Method to parse from user input.

- virtual string toString ()=0

    Get a string representation of an object.

Static Public Member Functions

- static TriangleParser ∗ getInstance ()

    Gets the singleton instance of TriangleParser.

Private Member Functions

- TriangleParser ()=default

  Private constructor for TriangleParser class.
- ∼TriangleParser ()=default

  Private destructor for TriangleParser class.
- TriangleParser (const TriangleParser &)=delete

  Private copy constructor for TriangleParser class.
- TriangleParser & operator= (const TriangleParser &)=delete

  Private copy assignment operator for TriangleParser class.

Static Private Attributes

- static TriangleParser ∗ _instance = nullptr

  Singleton instance of TriangleParser.

8.36.1 Detailed Description

TriangleParser class, which inherits from the IParser interface and performs the task of parsing triangle shapes.

8.36.2 Constructor & Destructor Documentation

8.36.2.1 TriangleParser() [1/2]

```
TriangleParser::TriangleParser ( )  [private], [default]
```

Private constructor for TriangleParser class.

8.36.2.2 ∼TriangleParser()

```
TriangleParser::∼TriangleParser ( )  [private], [default]
```

Private destructor for TriangleParser class.

8.36.2.3 TriangleParser() [2/2]

```
TriangleParser::TriangleParser (
            const TriangleParser & )  [private], [delete]
```

Private copy constructor for TriangleParser class.

8.36.3 Member Function Documentation

8.36.3.1 getInstance()

```
TriangleParser * TriangleParser::getInstance ( )  [static]
```

Gets the singleton instance of TriangleParser.

Returns

Singleton instance of TriangleParser

8.36.3.2 operator=()

```
TriangleParser & TriangleParser::operator= (
                const TriangleParser &  )  [private], [delete]
```

Private copy assignment operator for TriangleParser class.

8.36.3.3 parse()

```
IShape * TriangleParser::parse (
                stringstream data )  [override], [virtual]
```

Parses the input data and returns a Triangle object.

Parameters

Input	data to parse

Returns

Triangle object parsed from the input data

Exceptions

std::exception	if unable to parse the input data

Implements IParser.

8.36.3.4 toString()

```
string TriangleParser::toString ( )  [override], [virtual]
```

Returns a string representation of the TriangleParser object.

Returns

String representation of the TriangleParser object

Implements Object.

8.36.4 Member Data Documentation

8.36.4.1 _instance

```
TriangleParser* TriangleParser::_instance = nullptr  [inline], [static], [private]
```

Singleton instance of TriangleParser.

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

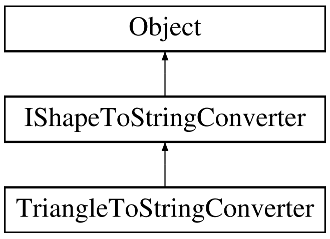
- Triangle/TriangleParser.h
- Triangle/TriangleParser.cpp

8.37 TriangleToStringConverter Class Reference

TriangleToStringConverter class, which inherits from the IShapeToStringConverter interface and performs the task of converting triangle shape information to data set.

```
#include <TriangleToStringConverter.h>
```

Inheritance diagram for TriangleToStringConverter:



Public Member Functions

- SHAPE_DATA convert (IShape ∗) override

  Converts a Triangle object to SHAPE_DATA format.
- string toString () override

  Returns a string representation of the TriangleToStringConverter object.

- virtual SHAPE_DATA convert (IShape ∗shape)=0

  Method to convert IShape object to SHAPE_DATA data type.

- virtual string toString ()=0

  Get a string representation of an object.

8.37.1 Detailed Description

TriangleToStringConverter class, which inherits from the IShapeToStringConverter interface and performs the task of converting triangle shape information to data set.

8.37.2 Member Function Documentation

8.37.2.1 convert()

```
SHAPE_DATA TriangleToStringConverter::convert (
             IShape * shape )  [override], [virtual]
```

Converts a Triangle object to SHAPE_DATA format.

Parameters

Pointer	to the Triangle object to be converted

Returns

SHAPE_DATA formatted version of the Triangle object

Implements IShapeToStringConverter.

8.37.2.2 toString()

```
string TriangleToStringConverter::toString ( )  [override], [virtual]
```

Returns a string representation of the TriangleToStringConverter object.

Returns

String representation of the TriangleToStringConverter object

Implements Object.

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

- Triangle/TriangleToStringConverter.h
- Triangle/TriangleToStringConverter.cpp

Chapter 9

File Documentation

9.1 Circle/Circle.cpp File Reference

```
#include "pch.h"
#include "Circle.h"
```

9.2 Circle/Circle.h File Reference

```
#include "pch.h"
```

Classes

- class myCircle::Circle

  Circle class, which inherits from the IShape interface and stores information about a circle shape.

Namespaces

- namespace myCircle

9.3 Circle.h

Go to the documentation of this file.

```
#pragma once
#include "pch.h"

extern "C" {
    namespace myCircle {
        class Circle :
            public IShape
        {
            double _radius;
        public:
            Circle(double R) noexcept(false);

            double area() override;

            double perimeter() override;

            string toString() override;

            double radius();
        };
    }
}

```

9.4 Circle/CircleParser.cpp File Reference

```
#include ¨pch.h¨
#include ¨CircleParser.h¨
```

9.5 Circle/CircleParser.h File Reference

```
#include ¨pch.h¨
#include ¨Circle.h¨
```

Classes

- class CircleParser

  CircleParser class, which inherits from the IParser interface and performs the task of parsing circle shapes.

9.6 CircleParser.h

Go to the documentation of this file.

```
#pragma once

#include "pch.h"
#include "Circle.h"

extern "C" {
    class CircleParser :
        public IParser
    {
    private:
        inline static CircleParser* _instance = nullptr;

        CircleParser() = default;

        ~CircleParser() = default;

        CircleParser(const CircleParser&) = delete;

        CircleParser& operator=(const CircleParser&) = delete;
    public:
        static CircleParser* getInstance();

        IShape* parse(stringstream data) noexcept(false) override;

        string toString() override;
    };
}
```

9.7 Circle/CircleToStringConverter.cpp File Reference

```
#include ¨pch.h¨
#include ¨CircleToStringConverter.h¨
```

9.8 Circle/CircleToStringConverter.h File Reference

```
#include "pch.h"
#include "Circle.h"
```

Classes

- class CircleToStringConverter

  CircleToStringConverter class, which inherits from the IShapeToStringConverter interface and performs the task of converting circle shape information to data set.

9.9 CircleToStringConverter.h

Go to the documentation of this file.

```
#pragma once

#include "pch.h"
#include "Circle.h"

extern "C" {
    class CircleToStringConverter :
        public IShapeToStringConverter
    {
    public:
        SHAPE_DATA convert(IShape*) override;

        string toString() override;
    };
}
```

9.10 Circle/dllmain.cpp File Reference

```
#include "pch.h"
#include <windows.h>
#include <objbase.h>
#include "Circle.h"
#include "CircleToStringConverter.h"
#include "CircleParser.h"
```

Functions

- __declspec (dllexport) IParser *__stdcall getParserInstance()

9.10.1 Function Documentation

9.10.1.1 __declspec()

```
__declspec (
            dllexport  )
```

9.11 Ellipse/dllmain.cpp File Reference

```
#include "pch.h"
#include <windows.h>
#include <objbase.h>
#include "Ellipse.h"
#include "EllipseToStringConverter.h"
#include "EllipseParser.h"
```

Functions

- __declspec (dllexport) IParser *__stdcall getParserInstance()

9.11.1 Function Documentation

9.11.1.1 __declspec()

```
__declspec (
             dllexport  )
```

9.12 IsoscelesTrapezoid/dllmain.cpp File Reference

```
#include "pch.h"
#include <windows.h>
#include <objbase.h>
#include "IsoscelesTrapezoid.h"
#include "IsoscelesTrapezoidToStringConverter.h"
#include "IsoscelesTrapezoidParser.h"
```

Functions

- __declspec (dllexport) IParser *__stdcall getParserInstance()

9.12.1 Function Documentation

9.12.1.1 __declspec()

```
__declspec (
             dllexport  )
```

9.13 Parallelogram/dllmain.cpp File Reference

```
#include "pch.h"
#include <windows.h>
#include <objbase.h>
#include "Parallelogram.h"
#include "ParallelogramToStringConverter.h"
#include "ParallelogramParser.h"
```

Functions

- __declspec (dllexport) IParser *__stdcall getParserInstance()

9.13.1 Function Documentation

9.13.1.1 __declspec()

```
__declspec (
            dllexport  )
```

9.14 Rectangle/dllmain.cpp File Reference

```
#include "pch.h"
#include <windows.h>
#include <objbase.h>
#include "Rectangle.h"
#include "RectangleToStringConverter.h"
#include "RectangleParser.h"
```

Functions

- __declspec (dllexport) IParser *__stdcall getParserInstance()

9.14.1 Function Documentation

9.14.1.1 __declspec()

```
__declspec (
            dllexport  )
```

9.15 Rhombus/dllmain.cpp File Reference

```
#include "pch.h"
#include <windows.h>
#include <objbase.h>
#include "Rhombus.h"
#include "RhombusToStringConverter.h"
#include "RhombusParser.h"
```

Functions

- __declspec (dllexport) IParser ∗__stdcall getParserInstance()

9.15.1 Function Documentation

9.15.1.1 __declspec()

```
__declspec (
            dllexport )
```

9.16 Square/dllmain.cpp File Reference

```
#include "pch.h"
#include <windows.h>
#include <objbase.h>
#include "Square.h"
#include "SquareToStringConverter.h"
#include "SquareParser.h"
```

Functions

- __declspec (dllexport) IParser ∗__stdcall getParserInstance()

9.16.1 Function Documentation

9.16.1.1 __declspec()

```
__declspec (
            dllexport )
```

9.17 Triangle/dllmain.cpp File Reference

```
#include "pch.h"
#include <windows.h>
#include <objbase.h>
#include "Triangle.h"
#include "TriangleToStringConverter.h"
#include "TriangleParser.h"
```

Functions

- __declspec (dllexport) IParser *__stdcall getParserInstance()

9.17.1 Function Documentation

9.17.1.1 __declspec()

```
__declspec (
            dllexport  )
```

9.18 Circle/framework.h File Reference

```
#include <windows.h>
```

Macros

- #define WIN32_LEAN_AND_MEAN

9.18.1 Macro Definition Documentation

9.18.1.1 WIN32_LEAN_AND_MEAN

```
#define WIN32_LEAN_AND_MEAN
```

9.19 framework.h

Go to the documentation of this file.

```
#pragma once

#define WIN32_LEAN_AND_MEAN             // Exclude rarely-used stuff from Windows headers
// Windows Header Files
#include <windows.h>
```

9.20 Ellipse/framework.h File Reference

```
#include <windows.h>
```

Macros

- #define WIN32_LEAN_AND_MEAN

9.20.1 Macro Definition Documentation

9.20.1.1 WIN32_LEAN_AND_MEAN

```
#define WIN32_LEAN_AND_MEAN
```

9.21 framework.h

Go to the documentation of this file.

```
#pragma once

#define WIN32_LEAN_AND_MEAN             // Exclude rarely-used stuff from Windows headers
// Windows Header Files
#include <windows.h>
```

9.22 IsoscelesTrapezoid/framework.h File Reference

```
#include <windows.h>
```

Macros

- #define WIN32_LEAN_AND_MEAN

9.22.1 Macro Definition Documentation

9.22.1.1 WIN32_LEAN_AND_MEAN

```
#define WIN32_LEAN_AND_MEAN
```

9.23 framework.h

Go to the documentation of this file.

```
#pragma once

#define WIN32_LEAN_AND_MEAN             // Exclude rarely-used stuff from Windows headers
// Windows Header Files
#include <windows.h>
```

9.24 Parallelogram/framework.h File Reference

```
#include <windows.h>
```

Macros

- #define WIN32_LEAN_AND_MEAN

9.24.1 Macro Definition Documentation

9.24.1.1 WIN32_LEAN_AND_MEAN

```
#define WIN32_LEAN_AND_MEAN
```

9.25 framework.h

Go to the documentation of this file.

```
#pragma once

#define WIN32_LEAN_AND_MEAN             // Exclude rarely-used stuff from Windows headers
// Windows Header Files
#include <windows.h>
```

9.26 Rectangle/framework.h File Reference

```
#include <windows.h>
```

Macros

- #define WIN32_LEAN_AND_MEAN

9.26.1 Macro Definition Documentation

9.26.1.1 WIN32_LEAN_AND_MEAN

```
#define WIN32_LEAN_AND_MEAN
```

9.27 framework.h

Go to the documentation of this file.

```
#pragma once

#define WIN32_LEAN_AND_MEAN             // Exclude rarely-used stuff from Windows headers
// Windows Header Files
#include <windows.h>
```

9.28 Rhombus/framework.h File Reference

```
#include <windows.h>
```

Macros

- #define WIN32_LEAN_AND_MEAN

9.28.1 Macro Definition Documentation

9.28.1.1 WIN32_LEAN_AND_MEAN

```
#define WIN32_LEAN_AND_MEAN
```

9.29 framework.h

Go to the documentation of this file.

```
#pragma once

#define WIN32_LEAN_AND_MEAN             // Exclude rarely-used stuff from Windows headers
// Windows Header Files
#include <windows.h>
```

9.30 Square/framework.h File Reference

```
#include <windows.h>
```

Macros

- #define WIN32_LEAN_AND_MEAN

9.30.1 Macro Definition Documentation

9.30.1.1 WIN32_LEAN_AND_MEAN

```
#define WIN32_LEAN_AND_MEAN
```

9.31 framework.h

Go to the documentation of this file.

```
#pragma once

#define WIN32_LEAN_AND_MEAN             // Exclude rarely-used stuff from Windows headers
// Windows Header Files
#include <windows.h>
```

9.32 Triangle/framework.h File Reference

```
#include <windows.h>
```

Macros

- #define WIN32_LEAN_AND_MEAN

9.32.1 Macro Definition Documentation

9.32.1.1 WIN32_LEAN_AND_MEAN

```
#define WIN32_LEAN_AND_MEAN
```

9.33 framework.h

Go to the documentation of this file.

```
#pragma once

#define WIN32_LEAN_AND_MEAN             // Exclude rarely-used stuff from Windows headers
// Windows Header Files
#include <windows.h>
```

9.34 utils/framework.h File Reference

Macros

- #define WIN32_LEAN_AND_MEAN

9.34.1 Macro Definition Documentation

9.34.1.1 WIN32_LEAN_AND_MEAN

```
#define WIN32_LEAN_AND_MEAN
```

9.35 framework.h

Go to the documentation of this file.

```
#pragma once

#define WIN32_LEAN_AND_MEAN             // Exclude rarely-used stuff from Windows headers
```

9.36 Circle/pch.cpp File Reference

```
#include "pch.h"
```

9.37 Ellipse/pch.cpp File Reference

```
#include "pch.h"
```

9.38 IsoscelesTrapezoid/pch.cpp File Reference

```
#include "pch.h"
```

9.39 Parallelogram/pch.cpp File Reference

```
#include "pch.h"
```

9.40 Rectangle/pch.cpp File Reference

```
#include "pch.h"
```

9.41 Rhombus/pch.cpp File Reference

```
#include "pch.h"
```

9.42 Square/pch.cpp File Reference

```
#include "pch.h"
```

9.43 Triangle/pch.cpp File Reference

```
#include "pch.h"
```

9.44 utils/pch.cpp File Reference

```
#include "pch.h"
```

9.45 Circle/pch.h File Reference

```
#include "framework.h"
#include "../utils/utils.h"
#include "../ShapesParser/IShape.h"
#include "../ShapesParser/IParser.h"
#include "../ShapesParser/IShapeToStringConverter.h"
```

9.46 pch.h

Go to the documentation of this file.

```
// pch.h: This is a precompiled header file.
// Files listed below are compiled only once, improving build performance for future builds.
// This also affects IntelliSense performance, including code completion and many code browsing
   features.
// However, files listed here are ALL re-compiled if any one of them is updated between builds.
// Do not add files here that you will be updating frequently as this negates the performance
   advantage.

#ifndef PCH_H
#define PCH_H

// add headers that you want to pre-compile here
#include "framework.h"
#include "../utils/utils.h"
#include "../ShapesParser/IShape.h"
#include "../ShapesParser/IParser.h"
#include "../ShapesParser/IShapeToStringConverter.h"

#endif //PCH_H
```

9.47 Ellipse/pch.h File Reference

```
#include "framework.h"
#include "../utils/utils.h"
#include "../ShapesParser/IShape.h"
#include "../ShapesParser/IParser.h"
#include "../ShapesParser/IShapeToStringConverter.h"
```

9.48 pch.h

Go to the documentation of this file.

```
// pch.h: This is a precompiled header file.
// Files listed below are compiled only once, improving build performance for future builds.
// This also affects IntelliSense performance, including code completion and many code browsing
   features.
// However, files listed here are ALL re-compiled if any one of them is updated between builds.
// Do not add files here that you will be updating frequently as this negates the performance
advantage.

#ifndef PCH_H
#define PCH_H

// add headers that you want to pre-compile here
#include "framework.h"
#include "../utils/utils.h"
#include "../ShapesParser/IShape.h"
#include "../ShapesParser/IParser.h"
#include "../ShapesParser/IShapeToStringConverter.h"

#endif //PCH_H
```

9.49 IsoscelesTrapezoid/pch.h File Reference

```
#include ¨framework.h¨
#include ¨../utils/utils.h¨
#include ¨../ShapesParser/IShape.h¨
#include ¨../ShapesParser/IParser.h¨
#include ¨../ShapesParser/IShapeToStringConverter.h¨
```

9.50 pch.h

Go to the documentation of this file.

```
// pch.h: This is a precompiled header file.
// Files listed below are compiled only once, improving build performance for future builds.
// This also affects IntelliSense performance, including code completion and many code browsing
features.
// However, files listed here are ALL re-compiled if any one of them is updated between builds.
// Do not add files here that you will be updating frequently as this negates the performance
advantage.

#ifndef PCH_H
#define PCH_H

// add headers that you want to pre-compile here
#include "framework.h"
#include "../utils/utils.h"
#include "../ShapesParser/IShape.h"
#include "../ShapesParser/IParser.h"
#include "../ShapesParser/IShapeToStringConverter.h"

#endif //PCH_H
```

9.51 Parallelogram/pch.h File Reference

```
#include ¨framework.h¨
#include ¨../utils/utils.h¨
#include ¨../ShapesParser/IShape.h¨
#include ¨../ShapesParser/IParser.h¨
#include ¨../ShapesParser/IShapeToStringConverter.h¨
```

9.52 pch.h

Go to the documentation of this file.

```
// pch.h: This is a precompiled header file.
// Files listed below are compiled only once, improving build performance for future builds.
// This also affects IntelliSense performance, including code completion and many code browsing
features.
// However, files listed here are ALL re-compiled if any one of them is updated between builds.
// Do not add files here that you will be updating frequently as this negates the performance
advantage.

#ifndef PCH_H
#define PCH_H

// add headers that you want to pre-compile here
#include "framework.h"
#include "../utils/utils.h"
#include "../ShapesParser/IShape.h"
#include "../ShapesParser/IParser.h"
#include "../ShapesParser/IShapeToStringConverter.h"

#endif //PCH_H
```

9.53 Rectangle/pch.h File Reference

```
#include "framework.h"
#include "../utils/utils.h"
#include "../ShapesParser/IShape.h"
#include "../ShapesParser/IParser.h"
#include "../ShapesParser/IShapeToStringConverter.h"
```

9.54 pch.h

Go to the documentation of this file.

```
// pch.h: This is a precompiled header file.
// Files listed below are compiled only once, improving build performance for future builds.
// This also affects IntelliSense performance, including code completion and many code browsing
features.
// However, files listed here are ALL re-compiled if any one of them is updated between builds.
// Do not add files here that you will be updating frequently as this negates the performance
advantage.

#ifndef PCH_H
#define PCH_H

// add headers that you want to pre-compile here
#include "framework.h"
#include "../utils/utils.h"
#include "../ShapesParser/IShape.h"
#include "../ShapesParser/IParser.h"
#include "../ShapesParser/IShapeToStringConverter.h"

#endif //PCH_H
```

9.55 Rhombus/pch.h File Reference

```
#include "framework.h"
#include "../utils/utils.h"
#include "../ShapesParser/IShape.h"
#include "../ShapesParser/IParser.h"
#include "../ShapesParser/IShapeToStringConverter.h"
```

9.56 pch.h

Go to the documentation of this file.

```
// pch.h: This is a precompiled header file.
// Files listed below are compiled only once, improving build performance for future builds.
// This also affects IntelliSense performance, including code completion and many code browsing
features.
// However, files listed here are ALL re-compiled if any one of them is updated between builds.
// Do not add files here that you will be updating frequently as this negates the performance
advantage.

#ifndef PCH_H
#define PCH_H

// add headers that you want to pre-compile here
#include "framework.h"
#include "../utils/utils.h"
#include "../ShapesParser/IShape.h"
#include "../ShapesParser/IParser.h"
#include "../ShapesParser/IShapeToStringConverter.h"

#endif //PCH_H
```

9.57 Square/pch.h File Reference

```
#include ¨framework.h¨
#include ¨../utils/utils.h¨
#include ¨../ShapesParser/IShape.h¨
#include ¨../ShapesParser/IParser.h¨
#include ¨../ShapesParser/IShapeToStringConverter.h¨
```

9.58 pch.h

Go to the documentation of this file.

```
// pch.h: This is a precompiled header file.
// Files listed below are compiled only once, improving build performance for future builds.
// This also affects IntelliSense performance, including code completion and many code browsing
features.
// However, files listed here are ALL re-compiled if any one of them is updated between builds.
// Do not add files here that you will be updating frequently as this negates the performance
advantage.
// Nhng header trong file này s đc b tin x lí biên dch trc

#ifndef PCH_H
#define PCH_H

// add headers that you want to pre-compile here
#include "framework.h"
#include "../utils/utils.h"
#include "../ShapesParser/IShape.h"
#include "../ShapesParser/IParser.h"
#include "../ShapesParser/IShapeToStringConverter.h"

#endif //PCH_H
```

9.59 Triangle/pch.h File Reference

```
#include ¨framework.h¨
#include ¨../utils/utils.h¨
#include ¨../ShapesParser/IShape.h¨
#include ¨../ShapesParser/IParser.h¨
#include ¨../ShapesParser/IShapeToStringConverter.h¨
```

9.60 pch.h

Go to the documentation of this file.

```
// pch.h: This is a precompiled header file.
// Files listed below are compiled only once, improving build performance for future builds.
// This also affects IntelliSense performance, including code completion and many code browsing
features.
// However, files listed here are ALL re-compiled if any one of them is updated between builds.
// Do not add files here that you will be updating frequently as this negates the performance
advantage.

#ifndef PCH_H
#define PCH_H

// add headers that you want to pre-compile here
#include "framework.h"
#include "../utils/utils.h"
#include "../ShapesParser/IShape.h"
#include "../ShapesParser/IParser.h"
#include "../ShapesParser/IShapeToStringConverter.h"

#endif //PCH_H
```

9.61 utils/pch.h File Reference

`#include "framework.h"`

9.62 pch.h

Go to the documentation of this file.

```
// pch.h: This is a precompiled header file.
// Files listed below are compiled only once, improving build performance for future builds.
// This also affects IntelliSense performance, including code completion and many code browsing
features.
// However, files listed here are ALL re-compiled if any one of them is updated between builds.
// Do not add files here that you will be updating frequently as this negates the performance
advantage.

#ifndef PCH_H
#define PCH_H

// add headers that you want to pre-compile here
#include "framework.h"

#endif //PCH_H
```

9.63 Ellipse/Ellipse.cpp File Reference

```
#include "pch.h"
#include "Ellipse.h"
```

9.64 Ellipse/Ellipse.h File Reference

`#include "pch.h"`

Classes

- class myEllipse::Ellipse

  Ellipse class, which inherits from the IShape interface and stores information about an ellipse shape.

Namespaces

- namespace myEllipse

9.65 Ellipse.h

Go to the documentation of this file.
```
#pragma once
#include "pch.h"

extern "C" {
    namespace myEllipse {
        class Ellipse :
            public IShape
        {
            double _semi_minor_axis;

            double _semi_major_axis;
        public:
            Ellipse(double, double) noexcept(false);

            double area() override;

            double perimeter() override;

            string toString() override;

            double semi_minor_axis();

            double semi_major_axis();
        };
    };
}

```

9.66 Ellipse/EllipseParser.cpp File Reference

```
#include "pch.h"
#include "EllipseParser.h"
```

9.67 Ellipse/EllipseParser.h File Reference

```
#include "pch.h"
#include "Ellipse.h"
```

Classes

- class EllipseParser

  EllipseParser class, which inherits from the IParser interface and performs the task of parsing ellipse shapes.

9.68 EllipseParser.h

Go to the documentation of this file.

```
#pragma once
#include "pch.h"
#include "Ellipse.h"

extern "C" {
    class EllipseParser :
        public IParser
    {
    private:
        inline static EllipseParser* _instance = nullptr;

        EllipseParser() = default;

        ~EllipseParser() = default;

        EllipseParser(const EllipseParser&) = delete;

        EllipseParser& operator=(const EllipseParser&) = delete;
    public:
        static EllipseParser* getInstance();

        IShape* parse(stringstream data) noexcept(false) override;

        string toString() override;
    };
}


```

9.69 Ellipse/EllipseToStringConverter.cpp File Reference

```
#include "pch.h"
#include "EllipseToStringConverter.h"
```

9.70 Ellipse/EllipseToStringConverter.h File Reference

```
#include "pch.h"
#include "Ellipse.h"
```

Classes

- class EllipseToStringConverter

  EllipseToStringConverter class, which inherits from the IShapeToStringConverter interface and performs the task of converting ellipse shape information to data set.

9.71 EllipseToStringConverter.h

Go to the documentation of this file.

```
#pragma once
#include "pch.h"
#include "Ellipse.h"

extern "C" {
    class EllipseToStringConverter :
        public IShapeToStringConverter
    {
    public:
        SHAPE_DATA convert(IShape*) override;

        string toString() override;
    };
}

```

9.72 IsoscelesTrapezoid/IsoscelesTrapezoid.cpp File Reference

```
#include ¨pch.h¨
#include ¨IsoscelesTrapezoid.h¨
```

9.73 IsoscelesTrapezoid/IsoscelesTrapezoid.h File Reference

```
#include ¨pch.h¨
```

Classes

- class myIsoscelesTrapezoid::IsoscelesTrapezoid

  IsoscelesTrapezoid class, which inherits from the IShape interface and stores information about an isosceles trapezoid shape.

Namespaces

- namespace myIsoscelesTrapezoid

9.74 IsoscelesTrapezoid.h

Go to the documentation of this file.

```
#pragma once

#include "pch.h"

extern "C" {
    namespace myIsoscelesTrapezoid {
        class IsoscelesTrapezoid :
            public IShape
        {
            double _top;

            double _base;

            double _height;

        public:
            IsoscelesTrapezoid(double, double, double) noexcept(false);

            double area() override;

            double perimeter() override;

            string toString() override;

            double top();

            double base();

            double height();
        };
    };
}
```

9.75 IsoscelesTrapezoid/IsoscelesTrapezoidParser.cpp File Reference

```
#include ¨pch.h¨
#include ¨IsoscelesTrapezoidParser.h¨
```

9.76 IsoscelesTrapezoid/IsoscelesTrapezoidParser.h File Reference

```
#include ¨pch.h¨
#include ¨IsoscelesTrapezoid.h¨
```

Classes

- class IsoscelesTrapezoidParser

  IsoscelesTrapezoidParser class, which inherits from the IParser interface and performs the task of parsing isosceles trapezoid shapes.

9.77 IsoscelesTrapezoidParser.h

Go to the documentation of this file.

```
#pragma once

#include "pch.h"
#include "IsoscelesTrapezoid.h"

extern "C" {
    class IsoscelesTrapezoidParser :
        public IParser
    {
    private:
        inline static IsoscelesTrapezoidParser* _instance = nullptr;

        IsoscelesTrapezoidParser() = default;

        ~IsoscelesTrapezoidParser() = default;

        IsoscelesTrapezoidParser(const IsoscelesTrapezoidParser&) = delete;

        IsoscelesTrapezoidParser& operator=(const IsoscelesTrapezoidParser&) = delete;
    public:
        static IsoscelesTrapezoidParser* getInstance();

        IShape* parse(stringstream data) noexcept(false) override;

        string toString() override;
    };
}
```

9.78 IsoscelesTrapezoid/IsoscelesTrapezoidToStringConverter.cpp File Reference

```
#include ¨pch.h¨
#include ¨IsoscelesTrapezoidToStringConverter.h¨
```

9.79 IsoscelesTrapezoid/IsoscelesTrapezoidToStringConverter.h File Reference

```
#include "pch.h"
#include "IsoscelesTrapezoid.h"
```

Classes

- class IsoscelesTrapezoidToStringConverter

  IsoscelesTrapezoidToStringConverter class, which inherits from the IShapeToStringConverter interface and performs the task of converting isosceles trapezoid shape information to data set.

9.80 IsoscelesTrapezoidToStringConverter.h

Go to the documentation of this file.

```
#pragma once

#include "pch.h"
#include "IsoscelesTrapezoid.h"

extern "C" {
    class IsoscelesTrapezoidToStringConverter :
        public IShapeToStringConverter
    {
    public:
        SHAPE_DATA convert(IShape*) override;

        string toString() override;
    };
}
```

9.81 Parallelogram/Parallelogram.cpp File Reference

```
#include "pch.h"
#include "Parallelogram.h"
```

9.82 Parallelogram/Parallelogram.h File Reference

```
#include "pch.h"
```

Classes

- class myParallelogram::Parallelogram

  Parallelogram class, which inherits from the IShape interface and stores information about a parallelogram shape.

Namespaces

- namespace myParallelogram

9.83 Parallelogram.h

Go to the documentation of this file.

```
#pragma once

#include "pch.h"

extern "C" {
    namespace myParallelogram {
        class Parallelogram :
            public IShape
        {
            double _side;

            double _base;

            double _height;

        public:
            Parallelogram(double, double, double) noexcept(false);

            double area() override;

            double perimeter() override;

            string toString() override;

            double side();

            double base();

            double height();
        };
    };
}
```

9.84 Parallelogram/ParallelogramParser.cpp File Reference

```
#include "pch.h"
#include "ParallelogramParser.h"
```

9.85 Parallelogram/ParallelogramParser.h File Reference

```
#include "pch.h"
#include "Parallelogram.h"
```

Classes

- class ParallelogramParser

  ParallelogramParser class, which inherits from the IParser interface and performs the task of parsing parallelogram shapes.

9.86 ParallelogramParser.h

Go to the documentation of this file.

```
#pragma once

#include "pch.h"
#include "Parallelogram.h"

extern "C" {
    class ParallelogramParser :
        public IParser
    {
    private:
        inline static ParallelogramParser* _instance = nullptr;

        ParallelogramParser() = default;

        ~ParallelogramParser() = default;

        ParallelogramParser(const ParallelogramParser&) = delete;

        ParallelogramParser& operator=(const ParallelogramParser&) = delete;
    public:
        static ParallelogramParser* getInstance();

        IShape* parse(stringstream data) noexcept(false) override;

        string toString() override;
    };
}
```

9.87 Parallelogram/ParallelogramToStringConverter.cpp File Reference

```
#include "pch.h"
#include "ParallelogramToStringConverter.h"
```

9.88 Parallelogram/ParallelogramToStringConverter.h File Reference

```
#include "pch.h"
#include "Parallelogram.h"
```

Classes

- class ParallelogramToStringConverter

  ParallelogramToStringConverter class, which inherits from the IShapeToStringConverter interface and performs the task of converting parallelogram shape information to data set.

9.89 ParallelogramToStringConverter.h

Go to the documentation of this file.

```
#pragma once

#include "pch.h"
#include "Parallelogram.h"

extern "C" {
    class ParallelogramToStringConverter :
        public IShapeToStringConverter
    {
    public:
        SHAPE_DATA convert(IShape*) override;

        string toString() override;
    };
}
```

9.90 README.md File Reference

9.91 Rectangle/Rectangle.cpp File Reference

```
#include "pch.h"
#include "Rectangle.h"
```

9.92 Rectangle/Rectangle.h File Reference

```
#include "pch.h"
```

Classes

- class myRectangle::Rectangle

  Rectangle class, which inherits from the IShape interface and stores information about a rectangle shape.

Namespaces

- namespace myRectangle

9.93 Rectangle.h

Go to the documentation of this file.

```
#pragma once

#include "pch.h"

extern "C" {
    namespace myRectangle {
        class Rectangle :
            public IShape
        {
            double _width;

            double _height;

        public:
            Rectangle(double, double) noexcept(false);

            double area() override;

            double perimeter() override;

            string toString() override;

            double width();

            double height();
        };
    };
}
```

9.94 Rectangle/RectangleParser.cpp File Reference

```
#include "pch.h"
#include "RectangleParser.h"
```

9.95 Rectangle/RectangleParser.h File Reference

```
#include "pch.h"
#include "Rectangle.h"
```

Classes

- class RectangleParser

  RectangleParser class, which inherits from the IParser interface and performs the task of parsing rectangle shapes.

9.96 RectangleParser.h

Go to the documentation of this file.

```
#pragma once

#include "pch.h"
#include "Rectangle.h"

extern "C" {
    class RectangleParser :
        public IParser
    {
    private:
        inline static RectangleParser* _instance = nullptr;

        RectangleParser() = default;

        ~RectangleParser() = default;

        RectangleParser(const RectangleParser&) = delete;

        RectangleParser& operator=(const RectangleParser&) = delete;
    public:
        static RectangleParser* getInstance();

        IShape* parse(stringstream data) noexcept(false) override;

        string toString() override;
    };
}
```

9.97 Rectangle/RectangleToStringConverter.cpp File Reference

```
#include "pch.h"
#include "RectangleToStringConverter.h"
```

9.98 Rectangle/RectangleToStringConverter.h File Reference

```
#include "pch.h"
#include "Rectangle.h"
```

Classes

- class RectangleToStringConverter

  RectangleToStringConverter class, which inherits from the IShapeToStringConverter interface and performs the task of converting rectangle shape information to data set.

9.99 RectangleToStringConverter.h

Go to the documentation of this file.

```
#pragma once

#include "pch.h"
#include "Rectangle.h"

extern "C" {
    class RectangleToStringConverter :
        public IShapeToStringConverter
    {
    public:
        SHAPE_DATA convert(IShape*) override;

        string toString() override;
    };
}
```

9.100 Rhombus/Rhombus.cpp File Reference

```
#include "pch.h"
#include "Rhombus.h"
```

9.101 Rhombus/Rhombus.h File Reference

```
#include "pch.h"
```

Classes

- class myRhombus::Rhombus

  Rhombus class, which inherits from the IShape interface and stores information about a rhombus shape.

Namespaces

- namespace myRhombus

9.102 Rhombus.h

Go to the documentation of this file.

```
#pragma once

#include "pch.h"

extern "C" {
    namespace myRhombus {
        class Rhombus :
            public IShape
        {
            double _short_diagonal;

            double _long_diagonal;

        public:
            Rhombus(double, double) noexcept(false);

            double area() override;

            double perimeter() override;

            string toString() override;

            double short_diagonal();

            double long_diagonal();
        };
    };
}
```

9.103 Rhombus/RhombusParser.cpp File Reference

```
#include ¨pch.h¨
#include ¨RhombusParser.h¨
```

9.104 Rhombus/RhombusParser.h File Reference

```
#include ¨pch.h¨
#include ¨Rhombus.h¨
```

Classes

- class RhombusParser

  RhombusParser class, which inherits from the IParser interface and performs the task of parsing rhombus shapes.

9.105 RhombusParser.h

Go to the documentation of this file.

```
#pragma once

#include "pch.h"
#include "Rhombus.h"

extern "C" {
    class RhombusParser :
        public IParser
    {
    private:
        inline static RhombusParser* _instance = nullptr;

        RhombusParser() = default;

        ~RhombusParser() = default;

        RhombusParser(const RhombusParser&) = delete;

        RhombusParser& operator=(const RhombusParser&) = delete;
    public:
        static RhombusParser* getInstance();

        IShape* parse(stringstream data) noexcept(false) override;

        string toString() override;
    };
}
```

9.106 Rhombus/RhombusToStringConverter.cpp File Reference

```
#include "pch.h"
#include "RhombusToStringConverter.h"
```

9.107 Rhombus/RhombusToStringConverter.h File Reference

```
#include "pch.h"
#include "Rhombus.h"
```

Classes

- class RhombusToStringConverter

  RhombusToStringConverter class, which inherits from the IShapeToStringConverter interface and performs the task of converting rhombus shape information to data set.

9.108 RhombusToStringConverter.h

Go to the documentation of this file.

```
#pragma once

#include "pch.h"
#include "Rhombus.h"

extern "C" {
    class RhombusToStringConverter :
        public IShapeToStringConverter
    {
    public:
        SHAPE_DATA convert(IShape*) override;

        string toString() override;
    };
}
```

9.109 ShapesParser/ConverterFactory.cpp File Reference

```
#include "ConverterFactory.h"
```

9.110 ShapesParser/ConverterFactory.h File Reference

```
#include "IShape.h"
#include "IShapeToStringConverter.h"
#include "../utils/utils.h"
#include "Object.h"
```

Classes

- class ConverterFactory

  Class to manage a list of prototypes for IShapeToStringConverter objects.

9.111 ConverterFactory.h

Go to the documentation of this file.

```
#pragma once
#include "IShape.h"
#include "IShapeToStringConverter.h"
#include "../utils/utils.h"
#include "Object.h"

class ConverterFactory : public Object
{
    map<string, IShapeToStringConverter*> _prototypes;
public:
    void registerWith(string type, IShapeToStringConverter* parser);

    IShapeToStringConverter* select(string type);

    string toString() override;
};

```

9.112 ShapesParser/IParser.cpp File Reference

```
#include "IParser.h"
```

9.113 ShapesParser/IParser.h File Reference

```
#include "IShape.h"
#include "Object.h"
#include "../utils/utils.h"
```

Classes

- class IParser

  IParser interface is used for declare methods for subclasses to implement.

9.114 IParser.h

Go to the documentation of this file.

```
#pragma once
#include "IShape.h"
#include "Object.h"
#include "../utils/utils.h"

class IParser : public Object
{
public:
    virtual IShape* parse(stringstream data) noexcept(false) = 0;
};

```

9.115 ShapesParser/IShape.cpp File Reference

```
#include ¨IShape.h¨
```

9.116 ShapesParser/IShape.h File Reference

```
#include ¨Object.h¨
#include ¨../utils/utils.h¨
```

Classes

- class IShape

  IShape interface is used for declare methods for subclasses to implement.

9.117 IShape.h

Go to the documentation of this file.

```
#pragma once
#include "Object.h"
#include "../utils/utils.h"

class IShape : public Object
{
public:
    virtual double area() = 0;

    virtual double perimeter() = 0;
};

```

9.118 ShapesParser/IShapeToStringConverter.cpp File Reference

```
#include "IShapeToStringConverter.h"
```

9.119 ShapesParser/IShapeToStringConverter.h File Reference

```
#include "IShape.h"
#include "Object.h"
#include "../utils/utils.h"
```

Classes

- class IShapeToStringConverter

  IShapeToStringConverter interface is used for declare methods for subclasses to implement.

9.120 IShapeToStringConverter.h

Go to the documentation of this file.

```
#pragma once
#include "IShape.h"
#include "Object.h"
#include "../utils/utils.h"

class IShapeToStringConverter : public Object
{
public:
    virtual SHAPE_DATA convert(IShape* shape) = 0;
};

```

9.121 ShapesParser/IShowDataBehavior.cpp File Reference

```
#include "IShowDataBehavior.h"
```

9.122 ShapesParser/IShowDataBehavior.h File Reference

```
#include "../utils/utils.h"
#include "Object.h"
```

Classes

- class IShowDataBehavior

  IShowDataBehavior interface is used for declare methods for subclasses to implement.

9.123 IShowDataBehavior.h

Go to the documentation of this file.

```
#pragma once
#include "../utils/utils.h"
#include "Object.h"

class IShowDataBehavior : public Object
{
public:
    virtual void showData(vector<SHAPE_DATA> data) = 0;
};

```

9.124 ShapesParser/IShowTableBehavior.cpp File Reference

```
#include "IShowTableBehavior.h"
```

9.125 ShapesParser/IShowTableBehavior.h File Reference

```
#include "../utils/utils.h"
#include "Object.h"
```

Classes

- class IShowTableBehavior

  IShowTableBehavior interface is used for declare methods for subclasses to implement.

9.126 IShowTableBehavior.h

Go to the documentation of this file.

```
#pragma once
#include "../utils/utils.h"
#include "Object.h"

class IShowTableBehavior : public Object
{
public:
    virtual void showTable(vector<SHAPE_DATA>) = 0;
};

```

9.127 ShapesParser/main.cpp File Reference

```
#include "IShape.h"
#include "IParser.h"
#include "IShapeToStringConverter.h"
#include "IShowTableBehavior.h"
#include "IShowDataBehavior.h"
#include "ParserFactory.h"
#include "ConverterFactory.h"
#include "ShapesPrinter.h"
#include "ShowTableCustom.h"
#include "ShowDataCustom.h"
```

Functions

- void setMode ()

    Function to set Vietnamese character mode for console output.
- void readFile (wstring textFile, int &count, vector< shared_ptr< IShape > > &shapes, ParserFactory &parser_factory)

    Function to read text file and store shape objects in a vector.
- void sortWithLambdaExpression (vector< shared_ptr< IShape > > &shapes)

    Function to sort shape objects in ascending order of area.
- void loadShapesToPrinter (ShapesPrinter &printer, vector< shared_ptr< IShape > > &shapes, ConverterFactory &converter_factory)

    Function to load IShape objects into a ShapesPrinter for printing.
- void setCustomPrinter (ShapesPrinter &printer, IShowDataBehavior ∗&showDataBehavior, IShowTableBehavior ∗&showTableBehavior)

    Function to set custom printing behavior for the ShapesPrinter object.
- void printToScreen (ShapesPrinter &printer, vector< shared_ptr< IShape > > &shapes, int count)

    Function to print shape objects to the console.
- int main ()

9.127.1 Function Documentation

9.127.1.1 loadShapesToPrinter()

```
void loadShapesToPrinter (
            ShapesPrinter & printer,
            vector< shared_ptr< IShape > > & shapes,
            ConverterFactory & converter_factory )
```

Function to load IShape objects into a ShapesPrinter for printing.

Parameters

printer	ShapesPrinter object that handles printing
shapes	Vector of shape objects to be printed
converter_factory	ConverterFactory to select object instantiation method through conversion

9.127.1.2 main()

```
int main ( )
```

9.127.1.3 printToScreen()

```
void printToScreen (
            ShapesPrinter & printer,
            vector< shared_ptr< IShape > > & shapes,
            int count )
```

Function to print shape objects to the console.

Parameters

printer	ShapesPrinter object that handles printing
shapes	Vector of shape objects to be printed
count	Number of shape objects declared in the file

9.127.1.4 readFile()

```
void readFile (
            wstring textFile,
            int & count,
            vector< shared_ptr< IShape > > & shapes,
            ParserFactory & parser_factory )
```

Function to read text file and store shape objects in a vector.

Parameters

textFile	Name of the text file
count	Number of shape objects declared in the file
shapes	Vector to store shape objects
parser_factory	Parser factory to select object instantiation method through parsing

9.127.1.5 setCustomPrinter()

```
void setCustomPrinter (
            ShapesPrinter & printer,
            IShowDataBehavior *& showDataBehavior,
            IShowTableBehavior *& showTableBehavior )
```

Function to set custom printing behavior for the ShapesPrinter object.

Parameters

printer	ShapesPrinter object that handles printing
showDataBehavior	Behavior for printing data
showTableBehavior	Behavior for printing table

9.127.1.6 setMode()

```
void setMode ( )
```

Function to set Vietnamese character mode for console output.

9.127.1.7 sortWithLambdaExpression()

```
void sortWithLambdaExpression (
            vector< shared_ptr< IShape > > & shapes )
```

Function to sort shape objects in ascending order of area.

Parameters

shapes	Vector of shape objects to be sorted

9.128 ShapesParser/Object.cpp File Reference

```
#include "Object.h"
```

9.129 ShapesParser/Object.h File Reference

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

Classes

- class Object

  Object class is the largest superclass of all classes in the program.

9.130 Object.h

Go to the documentation of this file.

```
#pragma once
#include "../utils/utils.h"

class Object
{
public:
    virtual string toString() = 0;
};
```

9.131 ShapesParser/ParserFactory.cpp File Reference

```
#include "ParserFactory.h"
```

9.132 ShapesParser/ParserFactory.h File Reference

```
#include ¨IShape.h¨
#include ¨IParser.h¨
#include ¨../utils/utils.h¨
#include ¨Object.h¨
```

Classes

- class ParserFactory

  Class to manage a list of prototypes for IParser objects.

9.133 ParserFactory.h

Go to the documentation of this file.

```
#pragma once
#include "IShape.h"
#include "IParser.h"
#include "../utils/utils.h"
#include "Object.h"

class ParserFactory : public Object
{
    map<string, IParser*> _prototypes;
public:
    void registerWith(string type, IParser* parser);

    IParser* select(string type);

    string toString() override;
};


```

9.134 ShapesParser/ShapesPrinter.cpp File Reference

```
#include ¨ShapesPrinter.h¨
```

9.135 ShapesParser/ShapesPrinter.h File Reference

```
#include ¨../utils/utils.h¨
#include ¨../ShapesParser/IShape.h¨
#include ¨IShowTableBehavior.h¨
#include ¨IShowDataBehavior.h¨
#include ¨ShowTableDefault.h¨
#include ¨ShowDataDefault.h¨
#include ¨Object.h¨
```

Classes

- class ShapesPrinter

  ShapesPrinter class, responsible for printing shapes to the screen.

9.136 ShapesPrinter.h

Go to the documentation of this file.

```
#pragma once

#include "../utils/utils.h"
#include "../ShapesParser/IShape.h"
#include "IShowTableBehavior.h"
#include "IShowDataBehavior.h"
#include "ShowTableDefault.h"
#include "ShowDataDefault.h"
#include "Object.h"

class ShapesPrinter : public Object
{
private:
    vector<SHAPE_DATA> _data;

    IShowTableBehavior* _showTableBehavior;

    IShowDataBehavior* _showDataBehavior;
public:
    ShapesPrinter();

    void setShowDataBehavior(IShowDataBehavior*);

    void performShowDataBehavior(vector<SHAPE_DATA>);

    void setShowTableBehavior(IShowTableBehavior*);

    void performShowTableBehavior(vector<SHAPE_DATA>);

    void push(SHAPE_DATA);

    void clear();

    vector<SHAPE_DATA> getData();

    string toString() override;
};
```

9.137 ShapesParser/ShowDataCustom.cpp File Reference

```
#include "ShowDataCustom.h"
```

9.138 ShapesParser/ShowDataCustom.h File Reference

```
#include "IShowDataBehavior.h"
#include "../utils/utils.h"
```

Classes

- class ShowDataCustom

  Custom implementation of IShowDataBehavior, responsible for displaying shape data in a customized format.

9.139 ShowDataCustom.h

Go to the documentation of this file.

```
#pragma once

#include "IShowDataBehavior.h"
#include "../utils/utils.h"

class ShowDataCustom : public IShowDataBehavior
{
public:
    void showData(vector<SHAPE_DATA>);

    string toString() override;
};

```

9.140 ShapesParser/ShowDataDefault.cpp File Reference

```
#include "ShowDataDefault.h"
```

9.141 ShapesParser/ShowDataDefault.h File Reference

```
#include "IShowDataBehavior.h"
#include "../utils/utils.h"
```

Classes

- class ShowDataDefault

  Default implementation of IShowDataBehavior, responsible for displaying shape data in a default format.

9.142 ShowDataDefault.h

Go to the documentation of this file.

```
#pragma once

#include "IShowDataBehavior.h"
#include "../utils/utils.h"

class ShowDataDefault : public IShowDataBehavior
{
public:
    void showData(vector<SHAPE_DATA>);

    string toString() override;
};

```

9.143 ShapesParser/ShowTableCustom.cpp File Reference

```
#include "ShowTableCustom.h"
```

9.144 ShapesParser/ShowTableCustom.h File Reference

```
#include ¨IShowTableBehavior.h¨
#include ¨../utils/utils.h¨
```

Classes

- class ShowTableCustom

    Custom implementation of IShowTableBehavior, responsible for displaying shape data in a customized table format.

9.145 ShowTableCustom.h

Go to the documentation of this file.

```
#pragma once

#include "IShowTableBehavior.h"
#include "../utils/utils.h"

class ShowTableCustom : public IShowTableBehavior
{
public:
    void showTable(vector<SHAPE_DATA>);

    string toString() override;
};

```

9.146 ShapesParser/ShowTableDefault.cpp File Reference

```
#include ¨ShowTableDefault.h¨
```

9.147 ShapesParser/ShowTableDefault.h File Reference

```
#include ¨IShowTableBehavior.h¨
#include ¨../utils/utils.h¨
```

Classes

- class ShowTableDefault

    Default implementation of IShowTableBehavior, responsible for displaying shape data in a default table format.

9.148 ShowTableDefault.h

Go to the documentation of this file.

```
#pragma once
#include "IShowTableBehavior.h"
#include "../utils/utils.h"


#include "IShowTableBehavior.h"

class ShowTableDefault : public IShowTableBehavior
{
public:
    void showTable(vector<SHAPE_DATA>);

    string toString() override;
};

```

9.149 ShapesParser/Strategy.cpp File Reference

`#include "Strategy.h"`

9.150 Square/Square.cpp File Reference

`#include "pch.h"`
`#include "Square.h"`

9.151 Square/Square.h File Reference

`#include "pch.h"`

Classes

- class mySquare::Square

  Square class, which inherits from the IShape interface and stores information about a square shape.

Namespaces

- namespace mySquare

9.152 Square.h

Go to the documentation of this file.

```
#pragma once

#include "pch.h"

extern "C" {
    namespace mySquare {
        class Square :
            public IShape
        {
            double _length;

        public:
            Square(double) noexcept(false);

            double area() override;

            double perimeter() override;

            std::string toString() override;

            double length();
        };
    };
}
```

9.153 Square/SquareParser.cpp File Reference

```
#include "pch.h"
#include "SquareParser.h"
```

9.154 Square/SquareParser.h File Reference

```
#include "pch.h"
#include "Square.h"
```

Classes

- class SquareParser

  SquareParser class, which inherits from the IParser interface and performs the task of parsing square shapes.

9.155 SquareParser.h

Go to the documentation of this file.

```
#pragma once

#include "pch.h"
#include "Square.h"

extern "C" {
    class SquareParser :
        public IParser
    {
    private:
        inline static SquareParser* _instance = nullptr;

        SquareParser() = default;

        ~SquareParser() = default;

        SquareParser(const SquareParser&) = delete;

        SquareParser& operator=(const SquareParser&) = delete;
    public:
        static SquareParser* getInstance();

        IShape* parse(stringstream data) noexcept(false) override;

        string toString() override;
    };
}
```

9.156 Square/SquareToStringConverter.cpp File Reference

```
#include "pch.h"
#include "SquareToStringConverter.h"
```

9.157 Square/SquareToStringConverter.h File Reference

```
#include "pch.h"
#include "Square.h"
```

Classes

- class SquareToStringConverter

  SquareToStringConverter class, which inherits from the IShapeToStringConverter interface and performs the task of converting square shape information to data set.

9.158 SquareToStringConverter.h

Go to the documentation of this file.

```
#pragma once

#include "pch.h"
#include "Square.h"

extern "C" {
    class SquareToStringConverter :
        public IShapeToStringConverter
    {
    public:
        SHAPE_DATA convert(IShape*) override;

        string toString() override;
    };
}
```

9.159 Triangle/Triangle.cpp File Reference

```
#include "pch.h"
#include "Triangle.h"
```

9.160 Triangle/Triangle.h File Reference

```
#include "pch.h"
```

Classes

- class myTriangle::Triangle

    Triangle class, which inherits from the IShape interface and stores information about a triangle shape.

Namespaces

- namespace myTriangle

9.161 Triangle.h

Go to the documentation of this file.

```
#pragma once

#include "pch.h"

extern "C" {
    namespace myTriangle {
        class Triangle :
            public IShape
        {
            double _first_edge;

            double _second_edge;

            double _third_edge;

        public:
            Triangle(double, double, double) noexcept(false);

            double area() override;

            double perimeter() override;

            string toString() override;

            double first_edge();

            double second_edge();

            double third_edge();
        };
    };
}
```

9.162 Triangle/TriangleParser.cpp File Reference

```
#include "pch.h"
#include "TriangleParser.h"
```

9.163 Triangle/TriangleParser.h File Reference

```
#include "pch.h"
#include "Triangle.h"
```

Classes

- class TriangleParser

  TriangleParser class, which inherits from the IParser interface and performs the task of parsing triangle shapes.

9.164 TriangleParser.h

Go to the documentation of this file.

```
#pragma once

#include "pch.h"
#include "Triangle.h"

extern "C" {
    class TriangleParser :
        public IParser
    {
    private:
        inline static TriangleParser* _instance = nullptr;

        TriangleParser() = default;

        ~TriangleParser() = default;

        TriangleParser(const TriangleParser&) = delete;

        TriangleParser& operator=(const TriangleParser&) = delete;
    public:
        static TriangleParser* getInstance();

        IShape* parse(stringstream data) noexcept(false) override;

        string toString() override;
    };
}
```

9.165 Triangle/TriangleToStringConverter.cpp File Reference

```
#include "pch.h"
#include "TriangleToStringConverter.h"
```

9.166 Triangle/TriangleToStringConverter.h File Reference

```
#include "pch.h"
#include "Triangle.h"
```

Classes

- class TriangleToStringConverter

  TriangleToStringConverter class, which inherits from the IShapeToStringConverter interface and performs the task of converting triangle shape information to data set.

9.167 TriangleToStringConverter.h

Go to the documentation of this file.

```
#pragma once

#include "pch.h"
#include "Triangle.h"

extern "C" {
    class TriangleToStringConverter :
        public IShapeToStringConverter
    {
    public:
        SHAPE_DATA convert(IShape*) override;

        string toString() override;
    };
}
```

9.168 utils/utils.cpp File Reference

```
#include "pch.h"
#include "framework.h"
#include "utils.h"
```

9.169 utils/utils.h File Reference

```
#include "pch.h"
#include "framework.h"
#include <iostream>
#include <string>
#include <sstream>
#include <regex>
#include <vector>
#include <fstream>
#include <windows.h>
#include <iomanip>
#include <map>
#include <tuple>
#include <io.h>
#include <filesystem>
#include <fcntl.h>
#include <exception>
#include <memory>
#include <algorithm>
#include <cmath>
```

Typedefs

- typedef tuple< wstring, wstring, wstring, wstring > SHAPE_DATA

  SHAPE_DATA is the data type defined to store the information of the IShape object:

Functions

- const regex DOUBLE_PATTERN (¨[+-]?([0-9]+([.][0-9]∗)?|[.][0-9]+)¨)

  DOUBLE_PATTERN is a regular expression that determines whether a string is a double value or not.

Variables

- const double PI = 3.1415

9.169.1 Typedef Documentation

9.169.1.1 SHAPE_DATA

```
typedef tuple<wstring, wstring, wstring, wstring> SHAPE_DATA
```

SHAPE_DATA is the data type defined to store the information of the IShape object:

- name /n
- attributes /n
- perimeter /n
- area

9.169.2 Function Documentation

9.169.2.1 DOUBLE_PATTERN()

```
const regex DOUBLE_PATTERN (
             ¨?([0-9]+([.][0-9]∗)?|[.][0-9]+)¨ [+-] )
```

DOUBLE_PATTERN is a regular expression that determines whether a string is a double value or not.

9.169.3 Variable Documentation

9.169.3.1 PI

```
const double PI = 3.1415
```

9.170 utils.h

Go to the documentation of this file.

```
#pragma once

#include "pch.h"
#include "framework.h"

#include <iostream>
#include <string>
#include <sstream>
#include <regex>
#include <vector>
#include <fstream>
#include <windows.h>
#include <iomanip>
#include <map>
#include <tuple>
#include <io.h>
#include <filesystem>
#include <fcntl.h>
#include <exception>
#include <memory>
#include <algorithm>
#include <cmath>

using std::cout, std::cin, std::endl;
using std::string, std::stringstream, std::wstring, std::wstringstream;
using std::wcout;
using std::vector;
using std::fstream;
using std::map, std::tuple;
using std::ifstream, std::ofstream;
using std::setw, std::fixed, std::setprecision, std::left;
using std::regex, std::regex_match;
using std::exception;
using std::to_wstring;
using std::sort;
using std::unique_ptr, std::shared_ptr, std::make_unique, std::make_shared;

typedef tuple<wstring, wstring, wstring, wstring> SHAPE_DATA;

const double PI = 3.1415;

const regex DOUBLE_PATTERN("[+-]?([0-9]+([.][0-9]*)?|[.][0-9]+)");
```

Index