KOALA

Imke Krauhausen, Charles-Théophile Coen, Simone Spolaor

Mar 06, 2023

CONTENTS:

1 Installation **3**

2 Additional information **5**
2.1 Basic classes 5

Python Module Index **9**

Index **11**

KOALA is a user-friendly Python library to stream-line (photo)mask design. You can find detailed documentation on the classes and methods implemented in it in the pages below.

CHAPTER

ONE

INSTALLATION

You can install KOALA with

```
pip install koala
```

CHAPTER

TWO

ADDITIONAL INFORMATION

Additional information and examples are available in our GitLab repository.

2.1 Basic classes

2.1.1 koala.classes

class koala.classes.**AbstractPolygon**(*name: str*, *polygon_db*, *centered=True*, *dx=0*, *dy=0*, *rotation=0*, *magnification=1*, *mirrorx=False*)

Bases: object

AbstractPolygon is the parent class of different types of polygons (rectangle, circles). It is an abstract class and never directly instantiated.

Parameters

- **name** (*str*) – Name of the AbstractPolygon
- **polygon_db** (*KLayout polygon*) – The created polygon object
- **centered** (*bool, optional*) – Specifies if the object is drawn from its geometrical center (True) or from the bottom left corner (False)
- **dx** (*int, optional*) – Movement in x direction in um
- **dy** (*int, optional*) – Movement in y direction in um
- **rotation** (*int, optional*) – Rotation in degrees
- **magnification** (*int, optional*) – Magnifying factor
- **mirrorx** (*bool, optional*) – Mirror in x direction

transformation(*dx*, *dy*, *rotation=0*, *magnification=1*, *mirrorx=False*)

Transformation allows to move, rotate, magnify and mirror a polygon or text

Parameters

- **dx** (*float*) – Movement in x direction in um
- **dy** (*float*) – Movement in y direction in um
- **rotation** (*int, optional*) – Rotation in degrees
- **magnification** (*int, optional*) – Magnifying factor
- **mirrorx** (*bool, optional*) – Mirror in x direction

class koala.classes.**Cell**(*name: str*, *gds_path=''*)
Bases: object

A cell is one of the building blocks of the layout. It can contain any type of object (polygon, region, text, etc...).

Parameters

- **name** (*str*) – Name of the cell
- **gds_path** (*str, optional*) – GDS path when importing external .gds file, such as alignment mark.

draw_path(*path_object*, *target_layer*)
Draw a path on the cell in the specified layer.

Parameters

- **path_object** (*TYPE*) – Path object to draw
- **target_layer** (*TYPE*) – Layer to draw the object in

draw_polygon(*polygon_object*, *target_layer*)
Draw a polygon on the cell in the specified layer.

Parameters

- **polygon_object** (*TYPE*) – Polygon object (rectangle, circle) to draw
- **target_layer** (*TYPE*) – Layer to draw the object in

draw_region(*region*, *target_layer*)
Draw a region on the cell in the specified layer.

Parameters

- **region** (*TYPE*) – Region to draw
- **target_layer** (*TYPE*) – Layer to draw the object in

draw_text(*text_region*, *target_layer*)
Draw a text on the cell in the specified layer.

Parameters

- **text_region** (*TYPE*) – Text to draw
- **target_layer** (*TYPE*) – Layer to draw the object in

flatten()
Flatten the layout squishing every children cell on the current cell.

insert_cell(*cell_to_insert*, *origin_x=0*, *origin_y=0*, *rotation=0*, *magnitude=1*, *mirrorx=False*)
Insert a cell in the current cell. The inserted cell can be placed, rotated, magnified and mirrored.

Parameters

- **cell_to_insert** (*TYPE*) – Cell to insert in the current cell
- **origin_x** (*int, optional*) – x coordinate in the current cell of the center of the inserted cell in um
- **origin_y** (*int, optional*) – y coordinate in the current cell of the center of the inserted cell in um
- **rotation** (*int, optional*) – Rotation in degree
- **magnitude** (*int, optional*) – Magnification of the cell

- **mirrorx** (*bool, optional*) – Mirror in x direction

insert_cell_array (*cell_to_insert*, *x_row*, *y_row*, *x_column*, *y_column*, *n_row: int*, *n_column: int*, *origin_x=0*, *origin_y=0*, *rotation=0*, *magnitude=1*, *mirrorx=False*)

Insert an array of cell in the current cell. The inserted cell can be placed, rotated, magnified and mirrored.

Parameters

- **cell_to_insert** (*TYPE*) – Cell to insert in the current cell
- **x_row** (*TYPE*) – x coordinate of row vector in um
- **y_row** (*TYPE*) – y coordinate of row vector in um
- **x_column** (*TYPE*) – x coordinate of column vector in um
- **y_column** (*TYPE*) – y coordinate of column vector in um
- **n_row** (*int*) – Number of row
- **n_column** (*int*) – Number of column
- **origin_x** (*int, optional*) – x coordinate in the current cell of the center of the inserted cell in um
- **origin_y** (*int, optional*) – y coordinate in the current cell of the center of the inserted cell in um
- **rotation** (*int, optional*) – Rotation in degree
- **magnitude** (*int, optional*) – Magnification of the cell
- **mirrorx** (*bool, optional*) – Mirror in x direction

class koala.classes.**Circle** (*name: str*, *radius: float*, *centered=True*, *nr_points=64*, *dx=0*, *dy=0*, *rotation=0*, *magnification=1*, *mirrorx=False*)

Bases: *koala.classes.AbstractPolygon*

Circle class allows to create a circular polygon object and inherits from AbstractPolygon class.

Parameters

- **name** (*str*) – Name of the circular polygon object
- **radius** (*float*) – Radius of circle
- **centered** (*bool, optional*) – Specifies if the object is drawn from its geometrical center (True), always True for circle
- **nr_points** (*int, optional*) – Number of points used to draw the circular polygon
- **dx** (*int, optional*) – Movement in x direction in um
- **dy** (*int, optional*) – Movement in y direction in um
- **rotation** (*int, optional*) – Rotation in degrees
- **magnification** (*int, optional*) – Magnifying factor
- **mirrorx** (*bool, optional*) – Mirror in x direction

class koala.classes.**Path** (*points: list*, *width: float*)

Bases: object

Path is a class used to create paths connecting different objects.

Parameters

- **points** (*list*) – List of points that the path should follow and connect

- **width** (*float*) – Width of the path

class koala.classes.**Rectangle**(*name: str*, *x: float*, *y: float*, *centered=True*, *dx=0*, *dy=0*, *magnification=1*, *rotation=0*, *mirrorx=False*)

Bases: *koala.classes.AbstractPolygon*

Rectangle class inherits from AbstractPolygon class and allows to create a rectangular polygon.

Parameters

- **name** (*str*) – Name of the rectangular polygon object
- **x** (*float*) – Width of the rectangle
- **y** (*float*) – Height of the rectangle
- **centered** (*bool, optional*) – Specifies if the object is drawn from its geometrical center (True) or from the bottom left corner (False)
- **dx** (*int, optional*) – Movement in x direction in um
- **dy** (*int, optional*) – Movement in y direction in um
- **rotation** (*int, optional*) – Rotation in degrees
- **magnification** (*int, optional*) – Magnifying factor
- **mirrorx** (*bool, optional*) – Mirror in x direction

class koala.classes.**Region**(*polygon_object_list: list*)

Bases: object

Region class allows to create regions from a list of polygons (such as rectangle or circle). Regions can be used for boolean operations.

Parameters **polygon_object_list** (*list*) – List of Polygon

add(*region_to_add*)

Add a region from another one. The boolean result is stored in the original region.

Parameters **region_to_add** (*TYPE*) – Region to add from the original region

subtract(*region_to_subtract*)

Subtract a region from another one. The boolean result is stored in the original region.

Parameters **region_to_subtract** (*TYPE*) – Region to subtract from the original region

class koala.classes.**Text**(*text: str*, *magnification=1000*, *dx=0*, *dy=0*, *rotation=0*, *mirrorx=False*)

Bases: *koala.classes.AbstractPolygon*

Text class inherits from AbstractPolygon class and allows to generate text for labelling layouts.

Parameters

- **text** (*str*) – Text that needs to be generated
- **magnification** (*int, optional*) – Magnifying factor
- **dx** (*int, optional*) – Movement in x direction in um
- **dy** (*int, optional*) – Movement in y direction in um
- **rotation** (*int, optional*) – Rotation in degrees
- **mirrorx** (*bool, optional*) – Mirror in x direction

PYTHON MODULE INDEX

k

koala.classes, 5

INDEX

A

AbstractPolygon (*class in koala.classes*), 5
add() (*koala.classes.Region method*), 8

C

Cell (*class in koala.classes*), 5
Circle (*class in koala.classes*), 7

D

draw_path() (*koala.classes.Cell method*), 6
draw_polygon() (*koala.classes.Cell method*), 6
draw_region() (*koala.classes.Cell method*), 6
draw_text() (*koala.classes.Cell method*), 6

F

flatten() (*koala.classes.Cell method*), 6

I

insert_cell() (*koala.classes.Cell method*), 6
insert_cell_array() (*koala.classes.Cell method*), 7

K

koala.classes
    module, 5

M

module
    koala.classes, 5

P

Path (*class in koala.classes*), 7

R

Rectangle (*class in koala.classes*), 8
Region (*class in koala.classes*), 8

S

subtract() (*koala.classes.Region method*), 8

T

Text (*class in koala.classes*), 8
transformation() (*koala.classes.AbstractPolygon method*), 5