# AUTO AREA

Area Annotation Software For CAD Drawings

NOVEMBER 18, 2025

WORKING MANUAL

V1

# Table of Contents

## Introduction

This document houses the working and knowledge requirements of the project, ‘AutoArea’. The document contains a detailed description of the proposed tools, and how to use them. It merely serves as a reference for designing the associated software in the event of necessity.

## Proposed Tools

1. FreeCAD python API
2. Python pyautocad library

## FreeCad python API

FreeCAD consists of the following core modules:

1. FreeCAD: To access documents, objects and parameters.
2. FreeCADGui: To provide GUI automation.

### Workbenches

1. Draft: For 2D geometric shapes.
2. Sketcher: For sketches and constraints.

```
import FreeCAD as App

import FreeCADGui as Gui

import Part

import Draft
```

### Create a new document

```
doc = App.newDocument("TestDoc")
```

### Create a simple shape

```
box = doc.addObject("Part::Box", "MyBox")

doc.recompute()
```

## Create a polygon

```
pts = [App.Vector(0,0,0),
       App.Vector(10,0,0),
       App.Vector(5,5,0)]
poly = Draft.makePolygon(pts, closed=True)
doc.recompute()
```

## Reading External Files

```
import importDXF
importDXF.insert("/path/to/file.dxf",
App.ActiveDocument.Name)
```

## Accessing Geometry Data

Every FreeCAD object contains a **Shape** attribute with geometry information.

To get the vertices:

```
obj = App.ActiveDocument.getObject("MyObject")
for v in obj.Shape.Vertexes:
    print(v.Point)
To get the edges:
for e in obj.Shape.Edges:
    print("Length:", e.Length)
```

## To get the whole polygons (Draft Wires)

```
import Draft
if Draft.isWire(obj):
    print(obj.Points)
```

## Saving and Exporting

```
doc.saveAs("/path/myfile.FCStd")
```

## hasattr(object, "attribute_name")

It returns:

- **True** : if the object *has* that attribute
- **False** : if the object *does NOT* have that attribute

It is used to verify if an object can have a '**Shape'** attribute; several freeCAD objects like 'groups' and 'views' and metadata do not have a 'shape' attribute.

## To change the color of objects

Use the `obj.ViewObject.ShapeColor=(r, g, b)` *# r, g, b belong between 0 and 1 here.*

For the outline color: Use the `LineColor` attribute.

## To find the area

Use `obj.Area`

## Creating Text Annotations

```
import Draft

import FreeCAD as App

 pos = App.Vector(obj.Shape.BoundBox.XMax + 50,
obj.Shape.BoundBox.YMin, 0)

 txt = Draft.makeText(

    [

        "Area Information",

        f"Area = {area:.2f} sq units"
```

```
                ],
                point=pos
            )
```

## Setting up triggers with document observer

```
App.addDocumentObserver(Observer_Class())
```

**Point to NOTE:**

The drawing must be saved as a .dxf file ONLY. FreeCAD API cannot read .dwg files directly.

## Pyautocad

Pyautocad: A lightweight wrapper around AutoCAD's COM API. It allows you to open/attach to AutoCAD files, read items from an active document, create and modify geometry, add text, and automate repetitive tasks.

It talks to AutoCAD via pywin32, so AutoCAD must be installed on windows.

Pyautocad exposes a COM API -> Python accesses it through pywin32.

The Autocad class must be imported from pyautocad, at times along with APoint, a helper class for 3D point coordinnates.

### Major objects in pyautocad

Autocad object & properties:

i. App: AutoCAD application COM object.
ii. Doc: Active document.
iii. Model: modelSpace
iv. Iter_objects(item): iterates through 'item' objects
v. From .doc, you can access blocks, layers, ModelSpace, and Layouts.

## Add line or polyline

Sample code:

```
P1, p2 = APoint(0,0), APoint(100, 100)
Acad.model.AddLine(p1, p2)

Pts = [#list of points]
Pl= acad.model.AddPolyline(Pts)

Pl.Closed=True
```

## Add multiline text

Sample code:

```
    Acad.model.AddMText("Text", #coordinates of insertion
of text, height of box)
```

**Point to NOTE:**

Polygons are 'closed lines' in pyautocad. A 'polgon' is not an object in Pyautocad. A polygon can be accessed by 'polylines' or 'closed LWPolylines'.

Create a list of polygons, iterate through closed polyline objects, and append the polygons.

## To calculate the area

use `polygon.Area`

## Standard AutoCAD color chart:

| Colour | ACI Code |
|---|---|
| Red | 1 |
| Yellow | 2 |
| green | 3 |
| Cyan | 4 |
| Blue | 5 |
| Magenta | 6 |
| White | 7 |