**Zygo MX Python API 完整技術文檔**

**目錄**

* [系統需求與環境](file:///Users/mac/Desktop/topgiga/projects/zygo2_erp_connector/zygo2_erp_connector/resources/ArrangedHandBook.md#%E7%B3%BB%E7%B5%B1%E9%9C%80%E6%B1%82%E8%88%87%E7%92%B0%E5%A2%83)
* [一、核心功能模塊](file:///Users/mac/Desktop/topgiga/projects/zygo2_erp_connector/zygo2_erp_connector/resources/ArrangedHandBook.md#%E4%B8%80%E6%A0%B8%E5%BF%83%E5%8A%9F%E8%83%BD%E6%A8%A1%E5%A1%8A)
  + [連接管理](file:///Users/mac/Desktop/topgiga/projects/zygo2_erp_connector/zygo2_erp_connector/resources/ArrangedHandBook.md#%E9%80%A3%E6%8E%A5%E7%AE%A1%E7%90%86)
  + [應用程序控制](file:///Users/mac/Desktop/topgiga/projects/zygo2_erp_connector/zygo2_erp_connector/resources/ArrangedHandBook.md#%E6%87%89%E7%94%A8%E7%A8%8B%E5%BA%8F%E6%8E%A7%E5%88%B6)
  + [數據操作](file:///Users/mac/Desktop/topgiga/projects/zygo2_erp_connector/zygo2_erp_connector/resources/ArrangedHandBook.md#%E6%95%B8%E6%93%9A%E6%93%8D%E4%BD%9C)
  + [運動控制](file:///Users/mac/Desktop/topgiga/projects/zygo2_erp_connector/zygo2_erp_connector/resources/ArrangedHandBook.md#%E9%81%8B%E5%8B%95%E6%8E%A7%E5%88%B6)
  + [測量與分析](file:///Users/mac/Desktop/topgiga/projects/zygo2_erp_connector/zygo2_erp_connector/resources/ArrangedHandBook.md#%E6%B8%AC%E9%87%8F%E8%88%87%E5%88%86%E6%9E%90)
* [二、基礎功能模塊](file:///Users/mac/Desktop/topgiga/projects/zygo2_erp_connector/zygo2_erp_connector/resources/ArrangedHandBook.md#%E4%BA%8C%E5%9F%BA%E7%A4%8E%E5%8A%9F%E8%83%BD%E6%A8%A1%E5%A1%8A)
  + [單位系統](file:///Users/mac/Desktop/topgiga/projects/zygo2_erp_connector/zygo2_erp_connector/resources/ArrangedHandBook.md#%E5%96%AE%E4%BD%8D%E7%B3%BB%E7%B5%B1)
  + [系統命令](file:///Users/mac/Desktop/topgiga/projects/zygo2_erp_connector/zygo2_erp_connector/resources/ArrangedHandBook.md#%E7%B3%BB%E7%B5%B1%E5%91%BD%E4%BB%A4)
  + [結果和控制](file:///Users/mac/Desktop/topgiga/projects/zygo2_erp_connector/zygo2_erp_connector/resources/ArrangedHandBook.md#%E7%B5%90%E6%9E%9C%E5%92%8C%E6%8E%A7%E5%88%B6)
  + [設置操作](file:///Users/mac/Desktop/topgiga/projects/zygo2_erp_connector/zygo2_erp_connector/resources/ArrangedHandBook.md#%E8%A8%AD%E7%BD%AE%E6%93%8D%E4%BD%9C)
* [三、進階功能模塊](file:///Users/mac/Desktop/topgiga/projects/zygo2_erp_connector/zygo2_erp_connector/resources/ArrangedHandBook.md#%E4%B8%89%E9%80%B2%E9%9A%8E%E5%8A%9F%E8%83%BD%E6%A8%A1%E5%A1%8A)
  + [切片管理](file:///Users/mac/Desktop/topgiga/projects/zygo2_erp_connector/zygo2_erp_connector/resources/ArrangedHandBook.md#%E5%88%87%E7%89%87%E7%AE%A1%E7%90%86)
  + [MST測量功能](file:///Users/mac/Desktop/topgiga/projects/zygo2_erp_connector/zygo2_erp_connector/resources/ArrangedHandBook.md#mst%E6%B8%AC%E9%87%8F%E5%8A%9F%E8%83%BD)
  + [基準點管理](file:///Users/mac/Desktop/topgiga/projects/zygo2_erp_connector/zygo2_erp_connector/resources/ArrangedHandBook.md#%E5%9F%BA%E6%BA%96%E9%BB%9E%E7%AE%A1%E7%90%86)
  + [QDAS統計分析](file:///Users/mac/Desktop/topgiga/projects/zygo2_erp_connector/zygo2_erp_connector/resources/ArrangedHandBook.md#qdas%E7%B5%B1%E8%A8%88%E5%88%86%E6%9E%90)
* [四、輔助功能模塊](file:///Users/mac/Desktop/topgiga/projects/zygo2_erp_connector/zygo2_erp_connector/resources/ArrangedHandBook.md#%E5%9B%9B%E8%BC%94%E5%8A%A9%E5%8A%9F%E8%83%BD%E6%A8%A1%E5%A1%8A)
  + [圖表控制](file:///Users/mac/Desktop/topgiga/projects/zygo2_erp_connector/zygo2_erp_connector/resources/ArrangedHandBook.md#%E5%9C%96%E8%A1%A8%E6%8E%A7%E5%88%B6)
  + [掩模管理](file:///Users/mac/Desktop/topgiga/projects/zygo2_erp_connector/zygo2_erp_connector/resources/ArrangedHandBook.md#%E6%8E%A9%E6%A8%A1%E7%AE%A1%E7%90%86)
  + [圖案管理](file:///Users/mac/Desktop/topgiga/projects/zygo2_erp_connector/zygo2_erp_connector/resources/ArrangedHandBook.md#%E5%9C%96%E6%A1%88%E7%AE%A1%E7%90%86)
  + [日誌記錄](file:///Users/mac/Desktop/topgiga/projects/zygo2_erp_connector/zygo2_erp_connector/resources/ArrangedHandBook.md#%E6%97%A5%E8%AA%8C%E8%A8%98%E9%8C%84)
  + [注釋管理](file:///Users/mac/Desktop/topgiga/projects/zygo2_erp_connector/zygo2_erp_connector/resources/ArrangedHandBook.md#%E6%B3%A8%E9%87%8B%E7%AE%A1%E7%90%86)
* [五、最佳實踐與注意事項](file:///Users/mac/Desktop/topgiga/projects/zygo2_erp_connector/zygo2_erp_connector/resources/ArrangedHandBook.md#%E4%BA%94%E6%9C%80%E4%BD%B3%E5%AF%A6%E8%B8%90%E8%88%87%E6%B3%A8%E6%84%8F%E4%BA%8B%E9%A0%85)

**系統需求與環境**

* Python 3.4.3
* Windows 10 Enterprise 2016 LTSB
* Zygo MX 軟件
* 硬件配置：
  + CPU: Intel64 Family 6 Model 158 Stepping 9
  + 記憶體: 32,632 MB
  + 系統類型: x64-based PC

**一、核心功能模塊**

**連接管理**

核心類別：

class WebServiceState(IntEnum):

none = 0 # 無狀態

idle = 1 # 空閒

active = 2 # 活動

核心功能：

def connect(force\_if\_active=False, host='localhost', port=8733, uid=''):

"""建立與MX的連接

Parameters:

force\_if\_active (bool): 是否強制連接

host (str): 主機名或IP

port (int): 端口號

uid (str): 連接ID

Returns:

str: 連接的唯一標識符

"""

def terminate():

"""關閉與MX的連接"""

def get\_service\_state():

"""獲取當前服務狀態"""

def get\_uid():

"""獲取當前連接ID"""

def get\_is\_remote\_access\_connected():

"""檢查遠程訪問狀態"""

HTTP請求處理：

def send\_request(service, method, params=None, \*, decode=True):

"""發送HTTP請求

Parameters:

service (str): 服務名稱

method (str): 方法名稱

params (dict): 參數字典

decode (bool): 是否解碼JSON

"""

def get\_send\_request(service, method, params=None, \*, decode=True):

"""發送HTTP請求並獲取返回值"""

**應用程序控制**

核心方法：

def is\_application\_open():

"""檢查MX應用程序是否開啟

Returns: bool"""

def get\_application\_path():

"""獲取應用程序路徑

Returns: str or None"""

def open\_application(filename):

"""打開MX應用程序

Parameters:

filename (str): 應用程序路徑"""

def close\_application():

"""關閉當前應用程序"""

def save\_application\_as(filename):

"""保存應用程序

Parameters:

filename (str): 保存路徑"""

**數據操作**

基本操作：

def analyze():

"""分析當前數據"""

def load\_data(filename):

"""加載數據文件

Parameters:

filename (str): 數據文件路徑"""

def save\_data(filename):

"""保存數據

Parameters:

filename (str): 保存路徑"""

高級數據操作：

def subtract\_data(filename, ignore\_lateral\_res=True, use\_input\_size=False,

use\_system\_size=False, use\_fiducial\_alignment=False,

alignment\_type=FiducialAlignmentType.fixed,

alignment\_tolerance=1.0):

"""數據減法操作"""

def scale\_data(scale\_value):

"""數據縮放

Parameters:

scale\_value (float): 縮放因子"""

**運動控制**

軸類型定義：

class AxisType(IntEnum):

"""軸名稱定義"""

unknown = 0 # 未知軸

x = 1 # X軸

y = 2 # Y軸

z = 3 # Z軸

rx = 4 # Pitch軸

ry = 5 # Roll軸

rz = 6 # Theta軸

x2 = 7 # 第二X軸

y2 = 8 # 第二Y軸

z2 = 9 # 第二Z軸

rx2 = 10 # 第二Pitch軸

ry2 = 11 # 第二Roll軸

rz2 = 12 # 第二Theta軸

class StageType(IntEnum):

"""平台類型定義"""

stage\_all = 0 # 所有平台

stage1 = 1 # 平台1

stage2 = 2 # 平台2

位置控制：

# 基本軸位置獲取

def get\_x\_pos(unit, stage=StageType.stage1):

"""獲取X軸位置"""

def get\_y\_pos(unit, stage=StageType.stage1):

"""獲取Y軸位置"""

def get\_z\_pos(unit, stage=StageType.stage1):

"""獲取Z軸位置"""

# 角度軸位置獲取

def get\_p\_pos(unit, stage=StageType.stage1):

"""獲取Pitch軸位置"""

def get\_r\_pos(unit, stage=StageType.stage1):

"""獲取Roll軸位置"""

def get\_t\_pos(unit, stage=StageType.stage1):

"""獲取Theta軸位置"""

運動命令：

# 單軸運動

def move\_x(x\_pos, unit, wait=True, stage=StageType.stage1):

"""移動X軸"""

def move\_y(y\_pos, unit, wait=True, stage=StageType.stage1):

"""移動Y軸"""

def move\_z(z\_pos, unit, wait=True, stage=StageType.stage1):

"""移動Z軸"""

# 多軸聯動

def move\_xy(x\_pos, y\_pos, unit, wait=True, stage=StageType.stage1):

"""同時移動XY軸"""

def move\_xyz(x\_pos, y\_pos, z\_pos, unit, wait=True, stage=StageType.stage1):

"""同時移動XYZ軸"""

回原點操作：

def home\_x(wait=True, stage=StageType.stage1):

"""X軸回原點"""

def home\_y(wait=True, stage=StageType.stage1):

"""Y軸回原點"""

def home\_z(wait=True, stage=StageType.stage1):

"""Z軸回原點"""

def home\_xyz(wait=True, stage=StageType.stage1):

"""XYZ軸同時回原點"""

def home\_all(wait=True, stage=StageType.stage\_all):

"""所有軸回原點"""

**測量與分析**

數據採集：

def acquire(wait=True):

"""獲取數據

Parameters:

wait (bool): 是否等待完成

Returns:

AcquisitionTask: 採集任務對象"""

def measure(wait=True):

"""測量數據（acquire + analyze）"""

自動優化：

def auto\_focus():

"""自動對焦"""

def auto\_tilt():

"""自動傾斜調整"""

def auto\_focus\_tilt():

"""自動對焦和傾斜調整"""

def auto\_light\_level():

"""自動光照水平調整"""

def auto\_lat\_cal(value, unit):

"""自動橫向校準"""

硬件控制：

# 轉台控制

def get\_turret():

"""獲取轉台位置"""

def move\_turret(position):

"""移動轉台"""

# 變焦控制

def get\_zoom():

"""獲取變焦值"""

def set\_zoom(zoom):

"""設置變焦"""

def lock\_zoom():

"""鎖定變焦"""

def unlock\_zoom():

"""解鎖變焦"""

# 光照控制

def get\_light\_level():

"""獲取光照水平"""

def set\_light\_level(light\_level):

"""設置光照水平"""

**二、基礎功能模塊**

**單位系統**

測量單位定義：

class Units(IntEnum):

# 線性測量單位

Angstroms = 1

MicroMeters = 2

MilliMeters = 3

Meters = 4

Inches = 5

MicroInches = 6

Mils = 7

NanoMeters = 8

# 角度單位

Degrees = 20

ArcMinutes = 21

ArcSeconds = 22

Radians = 23

MicroRadians = 24

MilliRadians = 25

# 面積單位

SquareMicroMeters = 40

SquareMilliMeters = 41

SquareMeters = 42

SquareInches = 43

# 體積單位

CubicMicroMeters = 60

CubicMilliMeters = 61

CubicMeters = 62

CubicInches = 63

# 特殊單位

Waves = 80

UserWaves = 81

Fringes = 82

FringeRadians = 83

Pixels = 84

Invalid = 0

NotSet = -1

NoUnits = -2

Scaled = -3

**系統命令**

文件類型：

class FileTypes(Enum):

All = ()

UI\_Application = ()

Script = ()

Csv = ()

Data = ()

Signal\_Data = ()

Setting = ()

Image = ()

Recipe = ()

Result = ()

Mask = ()

目錄操作：

def get\_bin\_dir():

"""獲取可執行文件目錄"""

def get\_working\_dir():

"""獲取工作目錄"""

def get\_open\_dir(file\_type):

"""獲取打開目錄"""

def get\_save\_dir(file\_type):

"""獲取保存目錄"""

def set\_open\_dir(file\_type, path):

"""設置打開目錄"""

def set\_save\_dir(file\_type, path):

"""設置保存目錄"""

文件操作：

def list\_files\_in\_open\_dir(file\_type):

"""列出打開目錄中的文件"""

def list\_files\_in\_dir(directory, extensions, recursive=False):

"""列出指定目錄中的文件"""

**結果和控制**

獲取值：

def get\_result\_number(path, unit=None):

"""獲取數值結果"""

def get\_control\_string(path):

"""獲取控制字符串"""

def get\_result\_bool(path):

"""獲取布爾結果"""

設置值：

def set\_result\_number(path, value, unit=None):

"""設置數值結果"""

def set\_control\_string(path, value):

"""設置控制字符串"""

def set\_result\_bool(path, value):

"""設置布爾結果"""

**設置操作**

def load\_settings(filename):

"""加載設置文件"""

def save\_settings(filename):

"""保存設置"""

def load\_settings\_using\_options(filename, options):

"""使用選項加載設置"""

**三、進階功能模塊**

**切片管理**

切片類型：

class LinearSliceType(IntEnum):

"""線性切片類型"""

linear = 1

class RadialSliceType(IntEnum):

"""徑向切片類型"""

radial = 1

radial\_center = 2

average\_radial = 3

average\_radial\_center = 4

class CircularSliceType(IntEnum):

"""圓形切片類型"""

circular = 1

circular\_center = 2

circular\_min\_pv = 3

基礎切片類：

class Slice:

@property

def label(self):

"""獲取切片標籤"""

def get\_dimension(self, units):

"""獲取切片尺寸"""

def set\_dimension(self, value, units):

"""設置切片尺寸"""

def get\_midpoint(self, units):

"""獲取中點坐標"""

def set\_midpoint(self, position, units):

"""設置中點坐標"""

def get\_angle(self, units):

"""獲取切片角度"""

def set\_angle(self, value, units):

"""設置切片角度"""

class LinearSlice(Slice):

"""線性切片"""

def get\_endpoints(self, units):

"""獲取起點和終點坐標"""

def get\_start(self, units):

"""獲取起點坐標"""

def set\_start(self, position, units):

"""設置起點坐標"""

def get\_end(self, units):

"""獲取終點坐標"""

def set\_end(self, position, units):

"""設置終點坐標"""

def get\_length(self, units):

"""獲取切片長度"""

def set\_length(self, value, units):

"""設置切片長度"""

class RadialSlice(Slice):

"""徑向切片"""

def get\_length(self, units):

"""獲取切片長度"""

def set\_length(self, value, units):

"""設置切片長度"""

def get\_endpoints(self, units):

"""獲取切片端點"""

def get\_start(self, units):

"""獲取起點"""

def set\_start(self, position, units):

"""設置起點"""

class CircularSlice(Slice):

"""圓形切片"""

def get\_radius(self, units):

"""獲取半徑"""

def set\_radius(self, value, units):

"""設置半徑"""

def get\_center(self, units):

"""獲取圓心"""

def set\_center(self, position, units):

"""設置圓心"""

# 切片獲取函數

def get\_linear\_slices(control, linear\_slice\_type):

"""獲取線性切片"""

def get\_all\_linear\_slices(control):

"""獲取所有線性切片"""

def get\_radial\_slices(control, radial\_slice\_type):

"""獲取徑向切片"""

def get\_all\_radial\_slices(control):

"""獲取所有徑向切片"""

def get\_circular\_slices(control, circular\_slice\_type):

"""獲取圓形切片"""

def get\_all\_circular\_slices(control):

"""獲取所有圓形切片"""

### MST測量功能

峰值分析：

```python

def get\_ftpsi\_peak(peak\_num, num\_peaks, range\_min, range\_max, unit):

"""獲取指定峰值位置

Parameters:

peak\_num (int): 峰值序號

num\_peaks (int): 預期峰值數量

range\_min (float): 最小光程差

range\_max (float): 最大光程差

unit (Units): 單位

Returns:

float: 峰值光程差"""

def get\_averaged\_ftpsi\_peak(peak\_num, num\_peaks, range\_min, range\_max, unit, averages):

"""獲取多點平均峰值位置

Parameters:

averages (int): 採樣點數"""

def analyze\_ftpsi\_peak(peak\_location, unit):

"""在指定光程差位置執行MST分析"""

``` {data-source-line="564"}

掃描參數：

```python

def get\_min\_excursion(unit):

"""獲取最小掃描振幅"""

def get\_max\_excursion(unit):

"""獲取最大掃描振幅"""

def get\_min\_rate(unit):

"""獲取最小掃描速率"""

def get\_max\_rate(unit):

"""獲取最大掃描速率"""

def get\_min\_frames():

"""獲取最小幀數"""

def get\_max\_frames():

"""獲取最大幀數"""

``` {data-source-line="585"}

測試點控制：

```python

def get\_test\_x():

"""獲取測試點X坐標"""

def get\_test\_y():

"""獲取測試點Y坐標"""

def get\_reference\_x():

"""獲取參考點X坐標"""

def get\_reference\_y():

"""獲取參考點Y坐標"""

def set\_test(x, y):

"""設置測試點"""

def set\_reference(x, y):

"""設置參考點"""

def clear\_test():

"""清除測試點"""

def clear\_reference():

"""清除參考點"""

``` {data-source-line="612"}

掃描參數估算：

```python

def estimate\_ramp\_excursion(maximum\_opd\_gap, tuning\_factor, opd\_units,

minimum\_opd\_gap=0, result\_units=Units.GigaHertz):

"""估算掃描振幅"""

def estimate\_ramp\_frames(maximum\_opd\_gap, tuning\_factor, opd\_units,

minimum\_opd\_gap=0):

"""估算掃描幀數"""

``` {data-source-line="623"}

### 基準點管理

基準點類：

```python

class Fiducial:

"""基準點"""

@property

def center(self):

"""獲取中心坐標"""

@property

def height(self):

"""獲取高度"""

@property

def width(self):

"""獲取寬度"""

def move\_absolute(self, x, y):

"""移動到絕對位置"""

def move\_relative(self, x, y):

"""相對移動"""

def resize(self, height, width):

"""調整尺寸"""

def rotate(self, value, unit):

"""旋轉"""

``` {data-source-line="655"}

基準點集合：

```python

class Fiducials:

"""基準點集合"""

def get\_num\_sets(self):

"""獲取工作集數量"""

def get\_num\_fiducials(self, working\_set=None):

"""獲取基準點數量"""

def save(self, filename):

"""保存基準點"""

def load(self, filename):

"""加載基準點"""

def delete(self, fiducial):

"""刪除基準點"""

def clear\_set(self, working\_set):

"""清空工作集"""

def delete\_set(self, working\_set):

"""刪除工作集"""

def add\_set(self):

"""新建工作集"""

def get\_fiducial\_closest\_to(self, x, y, working\_set=None):

"""獲取最近基準點"""

``` {data-source-line="688"}

### QDAS統計分析

研究類型：

```python

class QdasStudyType(IntEnum):

"""QDAS研究類型"""

standard = 0 # 標準生產

type\_1 = 1 # Type 1研究

type\_2 = 2 # Type 2研究

type\_3 = 3 # Type 3研究

class OperationPromptType(IntEnum):

"""操作提示類型"""

none = 0 # 無提示

info = 1 # 信息提示

serial = 2 # 序列號提示

unique\_serial = 3 # 唯一序列號提示

``` {data-source-line="707"}

測試計劃操作：

```python

def create\_qdas\_testplan(testplan\_path=None, qdas\_params\_path=None,

auto\_kfields=None):

"""創建測試計劃"""

def export\_qdas\_results(testplan\_path=None, qdas\_results\_path=None,

auto\_kfields=None, study\_type=QdasStudyType.standard,

silent=False):

"""導出結果"""

``` {data-source-line="719"}

研究執行：

```python

def do\_standard(spc\_testplan, testplan\_info, operation, auto\_kfields=None):

"""執行標準研究"""

def do\_type\_1(spc\_testplan, testplan\_info, operation, auto\_kfields=None,

prompt=OperationPromptType.info, sequence\_start=None):

"""執行Type 1研究"""

def do\_type\_2(spc\_testplan, testplan\_info, operation, auto\_kfields=None,

prompt=OperationPromptType.info, sequence\_start=None):

"""執行Type 2研究"""

``` {data-source-line="733"}

特徵值類：

```python

class QdasCharacteristic:

"""QDAS特徵值"""

@property

def uid(self):

"""獲取ID"""

@property

def name(self):

"""獲取名稱"""

@property

def groupname(self):

"""獲取分組"""

@property

def unit(self):

"""獲取單位"""

@property

def nominal(self):

"""獲取標稱值"""

@property

def ltol(self):

"""獲取下限"""

@property

def utol(self):

"""獲取上限"""

``` {data-source-line="767"}

## 四、輔助功能模塊

### 圖表控制

圖表類型：

```python

class ChartAxis(IntEnum):

"""圖表軸"""

X = 1 # X軸

Y = 2 # Y軸

All = 3 # 所有軸

class ChartLimit(IntEnum):

"""限制值類型"""

Low = 1 # 下限

High = 2 # 上限

All = 3 # 所有限制

``` {data-source-line="786"}

限制值操作：

```python

def clear\_chart\_limit(control, axis\_name=ChartAxis.All,

limit\_name=ChartLimit.All):

"""清除限制值"""

def set\_chart\_high\_limit(control, axis\_name=ChartAxis.Y,

limit\_value=0, unit=Units.MicroMeters):

"""設置上限"""

def set\_chart\_low\_limit(control, axis\_name=ChartAxis.Y,

limit\_value=0, unit=Units.MicroMeters):

"""設置下限"""

def set\_chart\_limits(control, axis\_name=ChartAxis.Y,

low\_value=0, high\_value=100, unit=Units.MicroMeters):

"""同時設置上下限"""

``` {data-source-line="805"}

### 掩模管理

掩模類：

```python

class Mask:

"""掩模"""

@property

def center(self):

"""獲取中心"""

@property

def height(self):

"""獲取高度"""

@property

def width(self):

"""獲取寬度"""

@property

def type(self):

"""獲取類型"""

def move\_absolute(self, x, y):

"""移動到絕對位置"""

def move\_relative(self, x, y):

"""相對移動"""

def resize(self, height, width):

"""調整尺寸"""

def rotate(self, value, unit):

"""旋轉"""

``` {data-source-line="841"}

掩模集合：

```python

class Masks:

"""掩模集合"""

def get\_num\_masks(self, mask\_type=None):

"""獲取掩模數量"""

def save(self, filename):

"""保存掩模"""

def load(self, filename):

"""加載掩模"""

def delete(self, mask):

"""刪除掩模"""

def clear(self, mask\_type=None):

"""清除掩模"""

def get\_mask\_closest\_to(self, x, y, mask\_type=None):

"""獲取最近掩模"""

``` {data-source-line="865"}

### 圖案管理

基本操作：

```python

def save(filename):

"""保存圖案"""

def load(filename):

"""加載圖案"""

def run():

"""執行圖案"""

``` {data-source-line="879"}

拼接操作：

```python

def load\_stitch(filename):

"""加載拼接設置"""

def load\_and\_stitch(folder):

"""加載並執行拼接"""

``` {data-source-line="888"}

對齊操作：

```python

def prealign():

"""預對齊"""

def align():

"""最終對齊"""

``` {data-source-line="897"}

### 日誌記錄

```python

def log\_trace(message):

"""跟蹤日誌"""

def log\_debug(message):

"""調試日誌"""

def log\_info(message):

"""信息日誌"""

def log\_warn(message):

"""警告日誌"""

def log\_error(message):

"""錯誤日誌"""

def log\_fatal(message):

"""致命錯誤日誌"""

``` {data-source-line="919"}

### 注釋管理

```python

def create\_annotation(name, value):

"""創建注釋"""

def get\_annotation(path):

"""獲取注釋"""

def set\_annotation(path, value):

"""設置注釋"""

def delete\_annotation(path):

"""刪除注釋"""

``` {data-source-line="935"}

## 五、最佳實踐與注意事項

1. 錯誤處理

```python

try:

# 執行操作

connect()

open\_application()

except ZygoError as e:

log\_error(f"操作失敗: {str(e)}")

finally:

terminate()

``` {data-source-line="949"}

2. 資源管理

- 使用 try-finally 確保資源釋放

- 及時關閉連接

- 正確處理文件操作

3. 性能優化

- 批量處理時使用異步操作

- 合理設置等待超時

- 避免不必要的數據轉換

4. 代碼規範

- 使用恰當的異常處理

- 保持代碼結構清晰

- 添加必要的註釋和文檔

- 使用有意義的變量命名

- 避免重複代碼

5. 測量流程建議

- 測量前進行必要的校準

- 確保環境穩定性

- 正確設置測量參數

- 保存原始數據

- 定期備份重要數據

6. 具體實現範例

### 基本測量流程

```python

from zygo.mx import connect, terminate

from zygo.application import open\_application

from zygo.motion import home\_all

from zygo.units import Units

try:

# 建立連接

uid = connect(host='localhost', port=8733)

print(f"Connected with UID: {uid}")

# 打開應用

open\_application("D:/MX/Applications/standard\_measure.mx")

# 回原點

home\_all(wait=True)

# 載入設置

load\_settings("D:/MX/Settings/measure\_settings.xml")

# 自動優化

auto\_focus()

auto\_tilt()

auto\_light\_level()

# 執行測量

measure(wait=True)

# 保存數據

save\_data("D:/MX/Data/measurement\_result.dat")

except ZygoError as e:

log\_error(f"Measurement failed: {str(e)}")

finally:

terminate()

``` {data-source-line="1013"}

### QDAS數據收集

```python

from zygo.qdas import create\_qdas\_testplan, do\_standard

from zygo.mx import measure

def measurement\_operation():

"""單次測量操作"""

# 執行測量

measure(wait=True)

# 獲取結果

result = get\_result\_number(("Results", "PV"), unit=Units.MicroMeters)

return result

try:

# 創建測試計劃

testplan = create\_qdas\_testplan(

testplan\_path="D:/QDAS/testplan.TestPlan",

qdas\_params\_path="D:/QDAS/params.csv",

auto\_kfields=["K1001", "K2001"]

)

# 執行標準研究

do\_standard(

spc\_testplan=testplan,

testplan\_info={"part\_number": "TEST001"},

operation=measurement\_operation,

auto\_kfields={"K1001": "BATCH001"}

)

except Exception as e:

log\_error(f"QDAS collection failed: {str(e)}")

``` {data-source-line="1047"}

### 自動化數據處理

```python

from zygo.mx import load\_data, analyze, save\_data

import os

def process\_data\_files(input\_dir, output\_dir):

"""批量處理數據文件"""

try:

# 獲取所有.dat文件

data\_files = [f for f in os.listdir(input\_dir)

if f.endswith('.dat')]

for file in data\_files:

input\_path = os.path.join(input\_dir, file)

output\_path = os.path.join(output\_dir, f"processed\_{file}")

# 載入數據

load\_data(input\_path)

# 處理數據

analyze()

scale\_data(1.5)

# 保存結果

save\_data(output\_path)

log\_info(f"Processed {file}")

except Exception as e:

log\_error(f"Data processing failed: {str(e)}")

``` {data-source-line="1079"}

### 切片分析

```python

from zygo.slices import (get\_all\_linear\_slices, LinearSliceType,

get\_circular\_slices, CircularSliceType)

from zygo.units import Units

def analyze\_slices(plot\_control):

"""分析切片數據"""

results = {}

# 分析線性切片

linear\_slices = get\_all\_linear\_slices(plot\_control)

for label, slice in linear\_slices.items():

length = slice.get\_length(Units.MicroMeters)

results[f"Linear\_{label}"] = length

# 分析圓形切片

circular\_slices = get\_circular\_slices(

plot\_control,

CircularSliceType.circular

)

for slice in circular\_slices:

radius = slice.get\_radius(Units.MicroMeters)

center = slice.get\_center(Units.MicroMeters)

results[f"Circular\_{slice.label}"] = {

"radius": radius,

"center\_x": center.x,

"center\_y": center.y

}

return results

``` {data-source-line="1112"}

7. 環境配置建議

Windows系統配置：

- 啟用長路徑支持

- 設置適當的系統環境變量

- 確保Python 3.4.3版本

Python環境配置：

- 創建獨立的虛擬環境

- 安裝必要的依賴包

- 配置正確的PYTHONPATH

網絡配置：

- 確保防火牆允許需要的端口

- 配置正確的IP地址

- 測試網絡連通性

8. 故障排除指南

連接問題：

- 檢查網絡設置

- 驗證主機名和端口

- 確認服務狀態

- 檢查防火牆設置

測量問題：

- 校準儀器

- 檢查環境條件

- 驗證參數設置

- 檢查硬件狀態

數據問題：

- 驗證文件格式

- 檢查存儲空間

- 備份重要數據

- 驗證數據完整性

9. 安全性考慮

數據安全：

- 定期備份數據

- 使用安全的存儲位置

- 實施訪問控制

- 加密敏感數據

操作安全：

- 使用try-except處理異常

- 實施錯誤恢復機制

- 記錄重要操作

- 定期維護系統

網絡安全：

- 使用安全的網絡協議

- 限制網絡訪問

- 監控網絡活動

- 更新安全補丁

<p data-source-line="1170" class="empty-line final-line end-of-document" style="margin:0;"></p>