



DAE Tools Project Documentation

Release 1.3.1

by Dragan Nikolic

April 22, 2013

CONTENTS

1	Getting Started with DAE Tools	1
2	pyDAE User Guide	3
3	pyDAE API Reference	5
3.1	Module pyCore	5
3.2	Module pyActivity	24
3.3	Module pyDataReporting	27
3.4	Module pyIDAS	27
3.5	Module pyUnits	28
4	Indices and tables	29
	Python Module Index	31
	Index	33

GETTING STARTED WITH DAE TOOLS

PYDAE USER GUIDE

PYDAE API REFERENCE

3.1 Module pyCore

3.1.1 Overview

Trt mrt.

3.1.2 Autodifferentiation and equation evaluation tree support

Classes

<code>adouble</code>	Class <code>adouble</code> operates on values/derivatives of domains, parameters and variables.
<code>adouble_array</code>	Class <code>adouble_array</code> operates on arrays of values/derivatives of domains, parameters and variables.
<code>daeCondition</code>	

`class pyCore.adouble`

Bases: `Boost.Python.instance`

Class `adouble` operates on values/derivatives of domains, parameters and variables. It supports basic mathematical operators (+, -, , /, *), comparison operators (<, <=, >, >=, ==, !=), and logical operators (and, or, not). Operands can be instances of `adouble` or float values.

Derivative

Derivative

GatherInfo

Internally used by the framework.

Node

Contains the equation evaluation node.

Value

Value

`class pyCore.adouble_array`

Bases: `Boost.Python.instance`

Class `adouble_array` operates on arrays of values/derivatives of domains, parameters and variables. It supports basic mathematical operators (+, -, , /, *). Operands can be instances of `adouble_array`, `adouble` or float values.

`__len__` ((`adouble_array`)*self*) → int :

Returns the size of the `adouble_array` object.

`__getitem__` ((`adouble_array`)*self*, (*int*)*index*) → `adouble` :

Gets an `adouble` object at the specified index.

__setitem__ ((*adouble_array*)self, (*int*)index, (*adouble*)value) → None :
Sets an adouble object at the specified index.

GatherInfo

Used internally by the framework.

Node

Contains the equation evaluation node.

Resize ((*adouble_array*)self, (*int*)newSize) → None :
Resizes the adouble_array object to the new size.

items ((*object*)arg1) → object :
Returns an iterator over adouble items in adouble_array object.

class pyCore.daeCondition

Bases: Boost.Python.instance

__or__ ((*daeCondition*)self, (*daeCondition*)right) → daeCondition

Logical operator or

__and__ ((*daeCondition*)self, (*daeCondition*)right) → daeCondition

Logical operator and

EventTolerance**Expressions****RuntimeNode****SetupNode****Mathematical functions**

Exp	Exp((adouble_array)arg1)-> adouble_array
Log	Log((adouble_array)arg1)-> adouble_array
Log10	Log10((adouble_array)arg1)-> adouble_array
Sqrt	Sqrt((adouble_array)arg1)-> adouble_array
Sin	Sin((adouble_array)arg1)-> adouble_array
Cos	Cos((adouble_array)arg1)-> adouble_array
Tan	Tan((adouble_array)arg1)-> adouble_array
ASin	ASin((adouble_array)arg1)-> adouble_array
ACos	ACos((adouble_array)arg1)-> adouble_array
ATan	ATan((adouble_array)arg1)-> adouble_array
Sinh	
Cosh	
Tanh	
ASinh	
ACosh	
ATanh	
ATan2	
Ceil	Ceil((adouble_array)arg1)-> adouble_array
Floor	Floor((adouble_array)arg1)-> adouble_array
Pow	Pow((adouble)arg1, (adouble)arg2)-> adouble
Abs	Abs((adouble_array)arg1)-> adouble_array
Min	Min((float)arg1, (adouble)arg2)-> adouble
Max	Max((float)arg1, (adouble)arg2)-> adouble

pyCore.Exp ((*adouble*)arg1) → adouble
Exp((adouble_array)arg1)-> adouble_array

```
pyCore.Log ((adouble)arg1) → adouble
    Log( (adouble_array)arg1) -> adouble_array

pyCore.Log10 ((adouble)arg1) → adouble
    Log10( (adouble_array)arg1) -> adouble_array

pyCore.Sqrt ((adouble)arg1) → adouble
    Sqrt( (adouble_array)arg1) -> adouble_array

pyCore.Sin ((adouble)arg1) → adouble
    Sin( (adouble_array)arg1) -> adouble_array

pyCore.Cos ((adouble)arg1) → adouble
    Cos( (adouble_array)arg1) -> adouble_array

pyCore.Tan ((adouble)arg1) → adouble
    Tan( (adouble_array)arg1) -> adouble_array

pyCore.ASin ((adouble)arg1) → adouble
    ASin( (adouble_array)arg1) -> adouble_array

pyCore.ACos ((adouble)arg1) → adouble
    ACos( (adouble_array)arg1) -> adouble_array

pyCore.ATan ((adouble)arg1) → adouble
    ATan( (adouble_array)arg1) -> adouble_array

pyCore.Sinh ((adouble)arg1) → adouble

pyCore.Cosh ((adouble)arg1) → adouble

pyCore.Tanh ((adouble)arg1) → adouble

pyCore.ASinh ((adouble)arg1) → adouble

pyCore.ACosh ((adouble)arg1) → adouble

pyCore.ATanh ((adouble)arg1) → adouble

pyCore.ATan2 ((adouble)arg1, (adouble)arg2) → adouble

pyCore.Ceil ((adouble)arg1) → adouble
    Ceil( (adouble_array)arg1) -> adouble_array

pyCore.Floor ((adouble)arg1) → adouble
    Floor( (adouble_array)arg1) -> adouble_array

pyCore.Pow ((adouble)arg1, (float)arg2) → adouble
    Pow( (adouble)arg1, (adouble)arg2) -> adouble
    Pow( (float)arg1, (adouble)arg2) -> adouble

pyCore.Abs ((adouble)arg1) → adouble
    Abs( (adouble_array)arg1) -> adouble_array

pyCore.Min ((adouble)arg1, (adouble)arg2) → adouble
    Min( (float)arg1, (adouble)arg2) -> adouble
    Min( (adouble)arg1, (float)arg2) -> adouble
    Min( (adouble_array)adarray) -> adouble

pyCore.Max ((adouble)arg1, (adouble)arg2) → adouble
    Max( (float)arg1, (adouble)arg2) -> adouble
    Max( (adouble)arg1, (float)arg2) -> adouble
    Max( (adouble_array)adarray) -> adouble
```

3.1.3 Modelling concepts

...

Enumerations

daeDomainType
daeParameterType
daePortType
daeDiscretizationMethod
daeDomainBounds
daeInitialConditionMode
daeDomainIndexType
daeRangeType
daeIndexRangeType
daeOptimizationVariableType
daeModelLanguage
daeConstraintType
daeUnaryFunctions
daeBinaryFunctions
daeSpecialUnaryFunctions
daeLogicalUnaryOperator
daeLogicalBinaryOperator
daeConditionType
daeActionType
daeEquationType
daeModelType

```
class pyCore.daeDomainType
    Bases: Boost.Python.enum

    eArray = pyCore.daeDomainType.eArray
    eDTUnknown = pyCore.daeDomainType.eDTUnknown
    eDistributed = pyCore.daeDomainType.eDistributed

class pyCore.daeParameterType
    Bases: Boost.Python.enum

    eBool = pyCore.daeParameterType.eBool
    eInteger = pyCore.daeParameterType.eInteger
    ePTUnknown = pyCore.daeParameterType.ePTUnknown
    eReal = pyCore.daeParameterType.eReal

class pyCore.daePortType
    Bases: Boost.Python.enum

    eInletPort = pyCore.daePortType.eInletPort
    eOutletPort = pyCore.daePortType.eOutletPort
    eUnknownPort = pyCore.daePortType.eUnknownPort

class pyCore.daeDiscretizationMethod
    Bases: Boost.Python.enum

    eBFDM = pyCore.daeDiscretizationMethod.eBFDM
    eCFDM = pyCore.daeDiscretizationMethod.eCFDM
```

```
eCustomDM = pyCore.daeDiscretizationMethod.eCustomDM
eDMUnknown = pyCore.daeDiscretizationMethod.eDMUnknown
eFFDM = pyCore.daeDiscretizationMethod.eFFDM

class pyCore.daeDomainBounds
    Bases: Boost.Python.enum

    eClosedClosed = pyCore.daeDomainBounds.eClosedClosed
    eClosedOpen = pyCore.daeDomainBounds.eClosedOpen
    eDBUnknown = pyCore.daeDomainBounds.eDBUnknown
    eLowerBound = pyCore.daeDomainBounds.eLowerBound
    eOpenClosed = pyCore.daeDomainBounds.eOpenClosed
    eOpenOpen = pyCore.daeDomainBounds.eOpenOpen
    eUpperBound = pyCore.daeDomainBounds.eUpperBound

class pyCore.daeInitialConditionMode
    Bases: Boost.Python.enum

    eAlgebraicValuesProvided = pyCore.daeInitialConditionMode.eAlgebraicValuesProvided
    eDifferentialValuesProvided = pyCore.daeInitialConditionMode.eDifferentialValuesProvided
    eICTUnknown = pyCore.daeInitialConditionMode.eICTUnknown
    eQuasySteadyState = pyCore.daeInitialConditionMode.eQuasySteadyState

class pyCore.daeDomainIndexType
    Bases: Boost.Python.enum

    eConstantIndex = pyCore.daeDomainIndexType.eConstantIndex
    eDITUnknown = pyCore.daeDomainIndexType.eDITUnknown
    eDomainIterator = pyCore.daeDomainIndexType.eDomainIterator
    eIncrementedDomainIterator = pyCore.daeDomainIndexType.eIncrementedDomainIterator

class pyCore.daeRangeType
    Bases: Boost.Python.enum

    eRaTUnknown = pyCore.daeRangeType.eRaTUnknown
    eRange = pyCore.daeRangeType.eRange
    eRangeDomainIndex = pyCore.daeRangeType.eRangeDomainIndex

class pyCore.daeIndexRangeType
    Bases: Boost.Python.enum

    eAllPointsInDomain = pyCore.daeIndexRangeType.eAllPointsInDomain
    eCustomRange = pyCore.daeIndexRangeType.eCustomRange
    eIRTUnknown = pyCore.daeIndexRangeType.eIRTUnknown
    eRangeOfIndexes = pyCore.daeIndexRangeType.eRangeOfIndexes

class pyCore.daeOptimizationVariableType
    Bases: Boost.Python.enum

    eBinaryVariable = pyCore.daeOptimizationVariableType.eBinaryVariable
    eContinuousVariable = pyCore.daeOptimizationVariableType.eContinuousVariable
    eIntegerVariable = pyCore.daeOptimizationVariableType.eIntegerVariable
```

```
class pyCore.daeModelLanguage
    Bases: Boost.Python.enum

    eCDAE = pyCore.daeModelLanguage.eCDAE
    eMLNone = pyCore.daeModelLanguage.eMLNone
    ePYDAE = pyCore.daeModelLanguage.ePYDAE

class pyCore.daeConstraintType
    Bases: Boost.Python.enum

    eEqualityConstraint = pyCore.daeConstraintType.eEqualityConstraint
    eInequalityConstraint = pyCore.daeConstraintType.eInequalityConstraint

class pyCore.daeUnaryFunctions
    Bases: Boost.Python.enum

    eAbs = pyCore.daeUnaryFunctions.eAbs
    eArcCos = pyCore.daeUnaryFunctions.eArcCos
    eArcSin = pyCore.daeUnaryFunctions.eArcSin
    eArcTan = pyCore.daeUnaryFunctions.eArcTan
    eCeil = pyCore.daeUnaryFunctions.eCeil
    eCos = pyCore.daeUnaryFunctions.eCos
    eExp = pyCore.daeUnaryFunctions.eExp
    eFloor = pyCore.daeUnaryFunctions.eFloor
    eLn = pyCore.daeUnaryFunctions.eLn
    eLog = pyCore.daeUnaryFunctions.eLog
    eSign = pyCore.daeUnaryFunctions.eSign
    eSin = pyCore.daeUnaryFunctions.eSin
    eSqrt = pyCore.daeUnaryFunctions.eSqrt
    eTan = pyCore.daeUnaryFunctions.eTan
    eUfUnknown = pyCore.daeUnaryFunctions.eUfUnknown

class pyCore.daeBinaryFunctions
    Bases: Boost.Python.enum

    eBfUnknown = pyCore.daeBinaryFunctions.eBfUnknown
    eDivide = pyCore.daeBinaryFunctions.eDivide
    eMax = pyCore.daeBinaryFunctions.eMax
    eMin = pyCore.daeBinaryFunctions.eMin
    eMinus = pyCore.daeBinaryFunctions.eMinus
    eMulti = pyCore.daeBinaryFunctions.eMulti
    ePlus = pyCore.daeBinaryFunctions.ePlus
    ePower = pyCore.daeBinaryFunctions.ePower

class pyCore.daeSpecialUnaryFunctions
    Bases: Boost.Python.enum

    eAverage = pyCore.daeSpecialUnaryFunctions.eAverage
    eMaxInArray = pyCore.daeSpecialUnaryFunctions.eMaxInArray
    eMinInArray = pyCore.daeSpecialUnaryFunctions.eMinInArray
```

```
eProduct = pyCore.daeSpecialUnaryFunctions.eProduct
eSUFUnknown = pyCore.daeSpecialUnaryFunctions.eSUFUnknown
eSum = pyCore.daeSpecialUnaryFunctions.eSum
class pyCore.daeLogicalUnaryOperator
    Bases: Boost.Python.enum
    eNot = pyCore.daeLogicalUnaryOperator.eNot
    eUOUnknown = pyCore.daeLogicalUnaryOperator.eUOUnknown
class pyCore.daeLogicalBinaryOperator
    Bases: Boost.Python.enum
    eAnd = pyCore.daeLogicalBinaryOperator.eAnd
    eBOUnknown = pyCore.daeLogicalBinaryOperator.eBOUnknown
    eOr = pyCore.daeLogicalBinaryOperator.eOr
class pyCore.daeConditionType
    Bases: Boost.Python.enum
    eCTUnknown = pyCore.daeConditionType.eCTUnknown
    eEQ = pyCore.daeConditionType.eEQ
    eGT = pyCore.daeConditionType.eGT
    eGTEQ = pyCore.daeConditionType.eGTEQ
    eLT = pyCore.daeConditionType.eLT
    eLTEQ = pyCore.daeConditionType.eLTEQ
    eNotEQ = pyCore.daeConditionType.eNotEQ
class pyCore.daeActionType
    Bases: Boost.Python.enum
    eChangeState = pyCore.daeActionType.eChangeState
    eReAssignOrReInitializeVariable = pyCore.daeActionType.eReAssignOrReInitializeVariable
    eSendEvent = pyCore.daeActionType.eSendEvent
    eUnknownAction = pyCore.daeActionType.eUnknownAction
    eUserDefinedAction = pyCore.daeActionType.eUserDefinedAction
class pyCore.daeEquationType
    Bases: Boost.Python.enum
    eAlgebraic = pyCore.daeEquationType.eAlgebraic
    eETUnknown = pyCore.daeEquationType.eETUnknown
    eExplicitODE = pyCore.daeEquationType.eExplicitODE
    eImplicitODE = pyCore.daeEquationType.eImplicitODE
class pyCore.daeModelType
    Bases: Boost.Python.enum
    eDAE = pyCore.daeModelType.eDAE
    eMTUnknown = pyCore.daeModelType.eMTUnknown
    eODE = pyCore.daeModelType.eODE
    eSteadyState = pyCore.daeModelType.eSteadyState
```

Classes

daeVariableType	
daeDomain	
daeParameter	
daeVariable	
daeModel	Base model class.
daeSTN	
daeIF	
daeEquation	
daeState	
daeStateTransition	
daePort	
daeEventPort	
daePortConnection	
daeScalarExternalFunction	
daeVectorExternalFunction	
daeDomainIndex	
daeIndexRange	
daeArrayRange	
daeDEDI	
daeAction	
daeOptimizationVariable	
daeObjectiveFunction	
daeOptimizationConstraint	
daeMeasuredVariable	
daeEquationExecutionInfo	

class pyCore.daeVariableType

Bases: Boost.Python.instance

__init__ ((object)arg1) → None

__init__ ((object)self, (str)name, (object)units, (float)lowerBound, (float)upperBound, (float)initialGuess, (float)absTolerance) -> None

AbsoluteTolerance

InitialGuess

LowerBound

Name

Units

UpperBound

class pyCore.daeObject

Bases: Boost.Python.instance

CanonicalName

Description

GetNameRelativeToParentModel ((daeObject)self) → str

GetStrippedName ((daeObject)self) → str

GetStrippedNameRelativeToParentModel ((daeObject)self) → str

ID

Library

Model**Name****Version****class** `pyCore.daeDomain`Bases: `pyCore.daeObject``__init__` *((object)arg1)* → None`__init__` *((object)self, (str)name, (daeModel)parentModel, (object)units [, (str)description=''])* → None`__init__` *((object)self, (str)name, (daePort)parentPort, (object)units [, (str)description=''])* → None`__getitem__` *((daeDomain)self, (int)index)* → adouble`__call__` *((daeDomain)self, (int)index)* → adouble`CreateArray` *((daeDomain)self, (int)noIntervals)* → None`CreateDistributed` *((daeDomain)self, (daeDiscretizationMethod)discretizationMethod, (int)discretizationOrder, (int)numberOfIntervals, (float)lowerBound, (float)upperBound)* → None**DiscretizationMethod****DiscretizationOrder****LowerBound****NumberOfIntervals****NumberOfPoints****Points****Type****Units****UpperBound****numpyPoints****class** `pyCore.daeParameter`Bases: `pyCore.daeObject``GetValue` *((daeParameter)self[, (int)index1[, ...[, (int)index8]]])* → floatGets the value of the parameter at the specified domain indexes. How many arguments `index1, ..., index8` are used depends on the number of domains that the parameter is distributed on.`GetQuantity` *((daeParameter)self[, (int)index1[, ...[, (int)index8]]])* → quantityGets the value of the parameter at the specified domain indexes as the `quantity` object (with value and units). How many arguments `index1, ..., index8` are used depends on the number of domains that the parameter is distributed on.`SetValue` *((daeParameter)self[, (int)index1[, ...[, (int)index8]]], (object)value)* → NoneSets the value of the parameter at the specified domain indexes (as `float` or `quantity`). How many arguments `index1, ..., index8` are used depends on the number of domains that the parameter is distributed on.`SetValues` *((daeParameter)self, (float)values)* → NoneSets all values of the parameter (as `float` or `quantity`).`array` *((daeParameter)self[, (object)index1[, ...[, (object)index8]]])* → `adouble_array`Gets the array of parameter's values at the specified domain indexes (used to build equation residuals only). How many arguments `index1, ..., index8` are used depends on the number of domains that the parameter is distributed on. Argument types can be one of the following:

- `daeIndexRange` object

- plain integer (to select a single index from a domain)
- python `list` (to select a list of indexes from a domain)
- python `slice` (to select a range of indexes from a domain: `start_index`, `end_index`, `step`)
- character `'*'` (to select all points from a domain)
- integer `-1` (to select all points from a domain)
- empty python list `[]` (to select all points from a domain)

__call__ *((daeParameter)self[, (int)index1[, ...[, (int)index8]])* → adouble

Gets the value of the parameter at the specified domain indexes (used to build equation residuals only). How many arguments `index1`, ..., `index8` are used depends on the number of domains that the parameter is distributed on.

DistributeOnDomain *((daeParameter)self, (daeDomain)domain)* → None

Domains

GetDomainsIndexesMap *((daeParameter)self, (int)indexBase)* → dict

NumberOfPoints

ReportingOn

Units

numpyValues

class `pyCore.daeVariable`

Bases: `pyCore.daeObject`

GetValue *((daeVariable)self[, (int)index1[, ...[, (int)index8]])* → float

Gets the value of the variable at the specified domain indexes. How many arguments `index1`, ..., `index8` are used depends on the number of domains that the variable is distributed on.

GetQuantity *((daeVariable)self[, (int)index1[, ...[, (int)index8]])* → quantity

Gets the value of the variable at the specified domain indexes as the `quantity` object (with value and units). How many arguments `index1`, ..., `index8` are used depends on the number of domains that the variable is distributed on.

SetValue *((daeVariable)self[, (int)index1[, ...[, (int)index8]]], (object)value)* → None

Sets the value of the variable at the specified domain indexes (as `float` or `quantity`). How many arguments `index1`, ..., `index8` are used depends on the number of domains that the variable is distributed on.

SetValues *((daeVariable)self, (object)values)* → None

Sets all values of the variable (as `float` or `quantity`).

AssignValue *((daeVariable)self[, (int)index1[, ...[, (int)index8]]], (object)value)* → None

AssignValues *((daeVariable)self, (object)values)* → None

ReAssignValue *((daeVariable)self[, (int)index1[, ...[, (int)index8]]], (object)value)* → None

ReAssignValues *((daeVariable)self, (object)values)* → None

SetInitialCondition *((daeVariable)self[, (int)index1[, ...[, (int)index8]]], (object)initialCondition)* → None

SetInitialConditions *((daeVariable)self, (object)initialConditions)* → None

ResetInitialCondition *((daeVariable)self[, (int)index1[, ...[, (int)index8]]], (object)initialCondition)* → None

ResetInitialConditions *((daeVariable)self, (object)initialConditions)* → None

SetInitialGuess *((daeVariable)self[, (int)index1[, ...[, (int)index8]]], (object)initialGuess)* → None

SetInitialGuesses *((daeVariable)self, (object)initialGuesses)* → None

SetAbsoluteTolerances ((*daeVariable*)self, (*object*)tolerances) → None

array ((*daeVariable*)self[, (*object*)index1[, ...[, (*object*)index8]]]) → adouble_array

Gets the array of variable's values at the specified domain indexes (used to build equation residuals only). How many arguments *index1*, ..., *index8* are used depends on the number of domains that the variable is distributed on. Argument types are the same as those described in `pyCore.daeParameter.array()`

d_array ((*daeVariable*)self[, (*object*)index1[, ...[, (*object*)index8]]]) → adouble_array

Gets the array of partial derivatives at the specified domain indexes (used to build equation residuals only). How many arguments *index1*, ..., *index8* are used depends on the number of domains that the variable is distributed on. Argument types are the same as those described in `pyCore.daeParameter.array()`.

d2_array ((*daeVariable*)self[, (*object*)index1[, ...[, (*object*)index8]]]) → adouble_array

Gets the array of partial derivatives of the second order at the specified domain indexes (used to build equation residuals only). How many arguments *index1*, ..., *index8* are used depends on the number of domains that the variable is distributed on. Argument types are the same as those described in `pyCore.daeParameter.array()`.

dt_array ((*daeVariable*)self[, (*object*)index1[, ...[, (*object*)index8]]]) → adouble_array

Gets the array of time derivatives at the specified domain indexes (used to build equation residuals only). How many arguments *index1*, ..., *index8* are used depends on the number of domains that the variable is distributed on. Argument types are the same as those described in `pyCore.daeParameter.array()`.

__call__ ((*daeVariable*)self[, (*int*)index1[, ...[, (*int*)index8]]]) → adouble

Gets the value of the variable at the specified domain indexes (used to build equation residuals only). How many arguments *index1*, ..., *index8* are used depends on the number of domains that the variable is distributed on.

d ((*daeVariable*)self, (*daeDomain*)domain[, (*int*)index1[, ...[, (*int*)index8]]]) → adouble

Gets the partial derivative of the variable at the specified domain indexes (used to build equation residuals only). How many arguments *index1*, ..., *index8* are used depends on the number of domains that the variable is distributed on.

d2 ((*daeVariable*)self, (*daeDomain*)domain[, (*int*)index1[, ...[, (*int*)index8]]]) → adouble

Gets the partial derivative of second order of the variable at the specified domain indexes (used to build equation residuals only). How many arguments *index1*, ..., *index8* are used depends on the number of domains that the variable is distributed on.

dt ((*daeVariable*)self[, (*int*)index1[, ...[, (*int*)index8]]]) → adouble

Gets the time derivative of the variable at the specified domain indexes (used to build equation residuals only). How many arguments *index1*, ..., *index8* are used depends on the number of domains that the variable is distributed on.

DistributeOnDomain ((*daeVariable*)self, (*daeDomain*)domain) → None

Domains

GetDomainsIndexesMap ((*daeVariable*)arg1, (*int*)self) → dict

NumberOfPoints

OverallIndex

ReportingOn

VariableType

numpyIDs

numpyValues

class `pyCore.daeModel`

Bases: `pyCore.daeObject`

Base model class.

__init__ ((*object*)*arg1*) → None

__init__ ((*object*)*self*, (*str*)*name* [, (*daeModel*)*parent*=0 [, (*str*)*description*='']] → None :
Constructor...

ComponentArrays

A list of arrays of components in the model.

Components

A list of components in the model.

ConnectEventPorts ((*daeModel*)*self*, (*daeEventPort*)*portFrom*, (*daeEventPort*)*portTo*) → None
:
Connects two event ports.

ConnectPorts ((*daeModel*)*self*, (*daePort*)*portFrom*, (*daePort*)*portTo*) → None :
Connects two ports.

CreateEquation ((*daeModel*)*self*, (*str*)*name* [, (*str*)*description*=' ' [, (*float*)*scaling*=1.0]]) →
daeEquation :
Creates a new equation. Used to add equations to models or states in state transition networks

DeclareEquations ((*daeModel*)*self*) → None :

User-defined function where all model equations and state transition networks are declared.
Must be always implemented in derived classes.

DeclareEquations((*daeModel*)*self*) → None

Domains

A list of domains in the model.

ELSE ((*daeModel*)*self*) → None :
Adds the last state to a reversible state transition network.

ELSE_IF ((*daeModel*)*self*, (*daeCondition*)*condition* [, (*float*)*eventTolerance*=0.0]) → None :
Adds a new state to a reversible state transition network.

END_IF ((*daeModel*)*self*) → None :
Finalises a reversible state transition network.

END_STN ((*daeModel*)*self*) → None :

.

Equations

A list of equations in the model.

EventPorts

A list of event ports in the model.

Export ((*daeModel*)*self*, (*str*)*content*, (*daeModelLanguage*)*language*, (*daeModelExportContext*)*modelExportContext*) → None :

.

ExportObjects ((*daeModel*)*self*, (*list*)*objects*, (*daeModelLanguage*)*language*) → *str* :

.

IF ((*daeModel*)*self*, (*daeCondition*)*condition* [, (*float*)*eventTolerance*=0.0]) → None :
Creates a reversible state transition network and adds the first state.

InitialConditionMode

A mode used to calculate initial conditions ...

IsModelDynamic

Boolean flag that determines whether the model is dynamic or steady-state.

ModelType

A type of the model ().

```
ON_CONDITION ((daeModel)self, (daeCondition)condition[, (str)switchTo='',
                (list)setVariableValues=[], (list)triggerEvents=[],
                (list)userDefinedActions=[], (float)eventTolerance=0.0 ])]]) → None
.
```

```
ON_EVENT ((daeModel)self, (daeEventPort)eventPort[, (list)switchToStates=[],
                (list)setVariableValues=[], (list)triggerEvents=[], (list)userDefinedActions=[]
                ])]]) → None :
```

OnEventActions

A list of OnEvent actions in the model.

Parameters

A list of parameters in the model.

PortArrays

A list of arrays of ports in the model.

PortConnections

A list of port connections in the model.

Ports

A list of ports in the model.

STATE ((daeModel)self, (str)stateName) → daeState :

.

STN ((daeModel)self, (str)stnName) → daeSTN :

.

STNs

A list of state transition networks in the model.

```
SWITCH_TO ((daeModel)self, (str)targetState, (daeCondition)condition[, (float)eventTolerance=0.0
                ])]]) → None :
```

SaveModelReport ((daeModel)self, (str)xmlFilename) → None :

.

SaveRuntimeModelReport ((daeModel)self, (str)xmlFilename) → None :

.

SetReportingOn ((daeModel)self, (bool)reportingOn) → None :

Switches the reporting of the model variables/parameters to the data reporter on or off.

Variables

A list of variables in the model.

class pyCore.daeSTN

Bases: pyCore.daeObject

ActiveState

States

class pyCore.daeIF

Bases: pyCore.daeSTN

class pyCore.daeEquation

Bases: pyCore.daeObject

DistributeOnDomain ((daeEquation)arg1, (daeDomain)arg2, (daeDomainBounds)arg3) →

daeDEDI
DistributeOnDomain((daeEquation)arg1, (daeDomain)arg2, (list)arg3) -> daeDEDI

DistributedEquationDomainInfos

EquationExecutionInfos

```
EquationType
Residual
Scaling
class pyCore.daeState
    Bases: pyCore.daeObject
    Equations
    NestedSTNs
    StateTransitions
class pyCore.daeStateTransition
    Bases: pyCore.daeObject
    Actions
    Condition
class pyCore.daePort
    Bases: pyCore.daeObject
    Domains
    Export ((daePort)arg1, (str)arg2, (daeModelLanguage)arg3, (daeModelExportContext)arg4) →
        None
    Parameters
    SetReportingOn ((daePort)arg1, (bool)arg2) → None
    Type
    Variables
class pyCore.daeEventPort
    Bases: pyCore.daeObject
    EventData
    Events
    ReceiveEvent ((daeEventPort)arg1, (float)arg2) → None
    RecordEvents
    SendEvent ((daeEventPort)arg1, (float)arg2) → None
    Type
class pyCore.daePortConnection
    Bases: pyCore.daeObject
    Equations
    PortFrom
    PortTo
class pyCore.daeScalarExternalFunction
    Bases: Boost.Python.instance
    __call__ ((daeScalarExternalFunction)arg1) → adouble
    Calculate ((daeScalarExternalFunction)arg1, (tuple)arg2, (dict)arg3) → object
        Calculate( (daeScalarExternalFunction)arg1, (tuple)arg2, (dict)arg3) -> None
    Name
class pyCore.daeVectorExternalFunction
    Bases: Boost.Python.instance
```

__call__ ((*daeVectorExternalFunction*)arg1) → adouble_array

Calculate ((*daeVectorExternalFunction*)arg1, (*tuple*)arg2, (*dict*)arg3) → list
Calculate((*daeVectorExternalFunction*)arg1, (*tuple*)arg2, (*dict*)arg3) -> None

Name

class pyCore.daeDomainIndex

Bases: Boost.Python.instance

__init__ ((*object*)arg1) → None

__init__((*object*)self, (*int*)index) -> None

__init__((*object*)self, (*daeDEDI*)dedi) -> None

__init__((*object*)self, (*daeDEDI*)dedi, (*int*)increment) -> None

__init__((*object*)self, (*daeDomainIndex*)domainIndex) -> None

DEDI

Increment

Index

Type

class pyCore.daeIndexRange

Bases: Boost.Python.instance

__init__ ((*object*)arg1) → None

__init__((*object*)self, (*daeDomain*)domain) -> None

__init__((*object*)arg1, (*daeDomain*)arg2, (*list*)arg3) -> object

__init__((*object*)self, (*daeDomain*)domain, (*int*)startIndex, (*int*)endIndex, (*int*)step) -> None

Domain

EndIndex

NoPoints

StartIndex

Step

Type

class pyCore.daeArrayRange

Bases: Boost.Python.instance

__init__ ((*object*)arg1) → None

__init__((*object*)self, (*daeDomainIndex*)domainIndex) -> None

__init__((*object*)self, (*daeIndexRange*)indexRange) -> None

DomainIndex

NoPoints

Range

Type

class pyCore.daeDEDI

Bases: pyCore.daeObject

__call__ ((*daeDEDI*)self) → adouble

Domain

DomainBounds

DomainPoints

```
class pyCore.daeAction
    Bases: pyCore.daeObject
    Execute ((daeAction)arg1) → None
        Execute( (daeAction)arg1) -> None
    RuntimeNode
    STN
    SendEventPort
    SetupNode
    StateTo
    Type
    VariableWrapper

class pyCore.daeOptimizationVariable
    Bases: pyCore.daeOptimizationVariable_t
    LowerBound
    Name
    StartingPoint
    Type
    UpperBound
    Value

class pyCore.daeObjectiveFunction
    Bases: pyCore.daeObjectiveFunction_t
    Gradients
    Name
    Residual
    Value

class pyCore.daeOptimizationConstraint
    Bases: pyCore.daeOptimizationConstraint_t
    Gradients
    Name
    Residual
    Type
    Value

class pyCore.daeMeasuredVariable
    Bases: pyCore.daeMeasuredVariable_t
    Gradients
    Name
    Residual
    Value

class pyCore.daeEquationExecutionInfo
    Bases: Boost.Python.instance
    EquationType
```


Node**VariableIndexes**

Logging

<code>daeLog_t</code>
<code>daeBaseLog</code>
<code>daeFileLog</code>
<code>daeStdOutLog</code>
<code>daeTCPIPLog</code>
<code>daeTCPIPLogServer</code>

class `pyCore.daeLog_t`Bases: `Boost.Python.instance`**DecreaseIndent** `((daeLog_t)arg1, (int)arg2) → None`

DecreaseIndent((daeLog_t)arg1, (int)arg2) -> None

ETA**Enabled****IncreaseIndent** `((daeLog_t)arg1, (int)arg2) → None`

IncreaseIndent((daeLog_t)arg1, (int)arg2) -> None

Indent**IndentString****JoinMessages** `((daeLog_t)arg1, (str)arg2) → str`

JoinMessages((daeLog_t)arg1, (str)arg2) -> None

Message `((daeLog_t)arg1, (str)arg2, (int)arg3) → None`

Message((daeLog_t)arg1, (str)arg2, (int)arg3) -> None

PercentageDone**PrintProgress****Progress****class** `pyCore.daeBaseLog`Bases: `pyCore.daeLog_t`**DecreaseIndent** `((daeBaseLog)arg1, (int)arg2) → None`**IncreaseIndent** `((daeBaseLog)arg1, (int)arg2) → None`**Message** `((daeBaseLog)arg1, (str)arg2, (int)arg3) → None`

Message((daeBaseLog)arg1, (str)arg2, (int)arg3) -> None

SetProgress `((daeBaseLog)arg1, (float)arg2) → None`

SetProgress((daeBaseLog)arg1, (float)arg2) -> None

class `pyCore.daeFileLog`Bases: `pyCore.daeBaseLog`**Message** `((daeFileLog)arg1, (str)arg2, (int)arg3) → None`

Message((daeFileLog)arg1, (str)arg2, (int)arg3) -> None

class `pyCore.daeStdOutLog`Bases: `pyCore.daeBaseLog`**Message** `((daeStdOutLog)arg1, (str)arg2, (int)arg3) → None`

Message((daeStdOutLog)arg1, (str)arg2, (int)arg3) -> None

```

class pyCore.daeTCPIPLog
    Bases: pyCore.daeBaseLog

    Message((daeTCPIPLog)arg1, (str)arg2, (int)arg3) → None
    Message( (daeTCPIPLog)arg1, (str)arg2, (int)arg3) -> None

class pyCore.daeTCPIPLogServer
    Bases: Boost.Python.instance

    MessageReceived((daeTCPIPLogServer)arg1, (str)arg2) → None
    MessageReceived( (daeTCPIPLogServer)arg1, (str)arg2) -> None

```

Functions

d
dt
Time
Constant Constant((object)value) -> adouble
Array
Sum
Product
Integral
Average

```

pyCore.d((adouble)arg1, (daeDomain)ad) → adouble
pyCore.dt((adouble)ad) → adouble
pyCore.Time() → adouble
pyCore.Constant((float)value) → adouble
    Constant( (object)value) -> adouble
pyCore.Array((list)values) → adouble_array
pyCore.Sum((adouble_array)adarray) → adouble
pyCore.Product((adouble_array)adarray) → adouble
pyCore.Integral((adouble_array)adarray) → adouble
pyCore.Average((adouble_array)adarray) → adouble

```

Auxiliary classes

daeVariableWrapper
daeConfig

```

class pyCore.daeVariableWrapper
    Bases: Boost.Python.instance

    __init__((object)self, (daeVariable)variable[, (str)name='']) → None
    __init__( (object)self, (adouble)ad [, (str)name='']) -> None

    DomainIndexes
    Name
    OverallIndex
    Value

```

Variable**VariableType****class** `pyCore.daeConfig`Bases: `Boost.Python.instance``__contains__` `((daeConfig)self, (object)propertyPath) → object``__getitem__` `((daeConfig)self, (object)propertyPath) → object``__setitem__` `((daeConfig)self, (object)propertyPath, (object)value) → None``GetBoolean` `((daeConfig)self, (str)propertyPath[, (bool)defaultValue]) → bool``GetFloat` `((daeConfig)self, (str)propertyPath[, (float)defaultValue]) → float``GetInteger` `((daeConfig)self, (str)propertyPath[, (int)defaultValue]) → int``GetString` `((daeConfig)self, (str)propertyPath[, (str)defaultValue]) → str``Reload` `((daeConfig)self) → None``SetBoolean` `((daeConfig)self, (str)propertyPath, (bool)value) → None``SetFloat` `((daeConfig)self, (str)propertyPath, (float)value) → None``SetInteger` `((daeConfig)self, (str)propertyPath, (int)value) → None``SetString` `((daeConfig)self, (str)propertyPath, (str)value) → None``has_key` `((daeConfig)self, (object)propertyPath) → object`**Auxiliary functions**`daeGetConfig``daeVersion``daeVersionMajor``daeVersionMinor``daeVersionBuild``pyCore.daeGetConfig()` `→ object``pyCore.daeVersion` `([(bool)includeBuild=False]) → str``pyCore.daeVersionMajor()` `→ int``pyCore.daeVersionMinor()` `→ int``pyCore.daeVersionBuild()` `→ int`**Global constants**`cnAlgebraic` `int(x[, base]) -> integer``cnDifferential` `int(x[, base]) -> integer``cnAssigned` `int(x[, base]) -> integer``pyCore.cnAlgebraic = 0``int(x[, base]) -> integer`

Convert a string or number to an integer, if possible. A floating point argument will be truncated towards zero (this does not include a string representation of a floating point number!) When converting a string, use the optional base. It is an error to supply a base when converting a non-string. If base is zero, the proper base is guessed based on the string content. If the argument is outside the integer range a long object will

be returned instead.

```
pyCore.cnDifferential = 1
int(x[, base]) -> integer
```

Convert a string or number to an integer, if possible. A floating point argument will be truncated towards zero (this does not include a string representation of a floating point number!) When converting a string, use the optional base. It is an error to supply a base when converting a non-string. If base is zero, the proper base is guessed based on the string content. If the argument is outside the integer range a long object will be returned instead.

```
pyCore.cnAssigned = 2
int(x[, base]) -> integer
```

Convert a string or number to an integer, if possible. A floating point argument will be truncated towards zero (this does not include a string representation of a floating point number!) When converting a string, use the optional base. It is an error to supply a base when converting a non-string. If base is zero, the proper base is guessed based on the string content. If the argument is outside the integer range a long object will be returned instead.

3.2 Module pyActivity

3.2.1 Overview

Trt mrt.

<code>daeSimulation</code>
<code>daeOptimization</code>

daeSimulation

```
class pyActivity.daeSimulation
  Bases: pyActivity.daeSimulation_t

  AbsoluteTolerances
  ActivityAction
  CleanupSetupData ((daeSimulation)arg1) → None
    CleanupSetupData( (daeSimulation)arg1) -> None
  Constraints
  CreateEqualityConstraint ((daeSimulation)arg1, (str)arg2) → object
  CreateInequalityConstraint ((daeSimulation)arg1, (str)arg2) → object
  CurrentTime
  DAESolver
  DataReporter
  Finalize ((daeSimulation)arg1) → None
  IndexMappings
  InitialConditionMode
  InitialDerivatives
  InitialValues
  Initialize ((daeSimulation)arg1, (object)arg2, (object)arg3, (object)arg4[,
    (bool)CalculateSensitivities=False ]) → None
```

InputVariables

Integrate ((*daeSimulation*)arg1, (*daeStopCriterion*)arg2[, (*bool*)ReportDataAroundDiscontinuities=True
]) → float

IntegrateForTimeInterval ((*daeSimulation*)arg1, (float)arg2[, (*bool*)ReportDataAroundDiscontinuities=True]) → float

IntegrateUntilTime ((*daeSimulation*)arg1, (float)arg2, (*daeStopCriterion*)arg3[, (*bool*)ReportDataAroundDiscontinuities=True]) → float

LoadInitializationValues ((*daeSimulation*)arg1, (*str*)arg2) → None

Log

MeasuredVariables

Model

ModelParameters

NextReportingTime

NumberOfEquations

NumberOfObjectiveFunctions

ObjectiveFunction

ObjectiveFunctions

OptimizationVariables

Pause ((*daeSimulation*)arg1) → None

ReRun ((*daeSimulation*)arg1) → None

RegisterData ((*daeSimulation*)arg1, (*str*)arg2) → None

Reinitialize ((*daeSimulation*)arg1) → None

RelativeTolerance

ReportData ((*daeSimulation*)arg1, (float)arg2) → None

ReportingInterval

ReportingTimes

Reset ((*daeSimulation*)arg1) → None

Resume ((*daeSimulation*)arg1) → None

Run ((*daeSimulation*)arg1) → None
Run((*daeSimulation*)arg1) -> None

SetBinaryOptimizationVariable ((*daeSimulation*)arg1, (*object*)arg2, (*bool*)arg3) → ob-
ject
SetBinaryOptimizationVariable((*daeSimulation*)arg1, (*object*)arg2, (*bool*)arg3) -> object

SetContinuousOptimizationVariable ((*daeSimulation*)arg1, (*object*)arg2, (float)arg3,
(float)arg4, (float)arg5) → object
SetContinuousOptimizationVariable((*daeSimulation*)arg1, (*object*)arg2, (float)arg3, (float)arg4,
(float)arg5) -> object

SetInputVariable ((*daeSimulation*)arg1, (*object*)arg2) → object
SetInputVariable((*daeSimulation*)arg1, (*object*)arg2) -> object

SetIntegerOptimizationVariable ((*daeSimulation*)arg1, (*object*)arg2, (*int*)arg3,
(*int*)arg4, (*int*)arg5) → object
SetIntegerOptimizationVariable((*daeSimulation*)arg1, (*object*)arg2, (*int*)arg3, (*int*)arg4, (*int*)arg5) ->
object

```
SetMeasuredVariable ((daeSimulation)arg1, (object)arg2) → object
    SetMeasuredVariable( (daeSimulation)arg1, (object)arg2) -> object

SetModelParameter ((daeSimulation)arg1, (object)arg2, (float)arg3, (float)arg4, (float)arg5) →
    object
    SetModelParameter( (daeSimulation)arg1, (object)arg2, (float)arg3, (float)arg4, (float)arg5) -> object

SetUpOptimization ((daeSimulation)arg1) → None
    SetUpOptimization( (daeSimulation)arg1) -> None

SetUpParameterEstimation ((daeSimulation)arg1) → None
    SetUpParameterEstimation( (daeSimulation)arg1) -> None

SetUpParametersAndDomains ((daeSimulation)arg1) → None
    SetUpParametersAndDomains( (daeSimulation)arg1) -> None

SetUpSensitivityAnalysis ((daeSimulation)arg1) → None
    SetUpSensitivityAnalysis( (daeSimulation)arg1) -> None

SetUpVariables ((daeSimulation)arg1) → None
    SetUpVariables( (daeSimulation)arg1) -> None

SimulationMode

SolveInitial ((daeSimulation)arg1) → None

StoreInitializationValues ((daeSimulation)arg1, (str)arg2) → None

TimeHorizon

TotalNumberOfVariables

VariableTypes

__init__ ((object)arg1) → None
__instance_size__ = 440
__module__ = 'pyActivity'
__reduce__ ()
m
model
```

daeOptimization

```
class pyActivity.daeOptimization
    Bases: pyActivity.daeOptimization_t

    Finalize ((daeOptimization)arg1) → None

    Initialize ((daeOptimization)arg1, (daeSimulation_t)arg2, (object)arg3, (object)arg4, (ob-
        ject)arg5, (object)arg6) → None

    Run ((daeOptimization)arg1) → None

    __init__ ((object)arg1) → None
    __instance_size__ = 88
    __module__ = 'pyActivity'
    __reduce__ ()
```

3.3 Module pyDataReporting

3.3.1 Overview

Trt mrt.

pyDataReporting

daeDataReporter_t

class pyDataReporting.daeDataReporter_t

Bases: Boost.Python.instance

Connect ((daeDataReporter_t)arg1, (str)arg2, (str)arg3) → bool
 Connect((daeDataReporter_t)arg1, (str)arg2, (str)arg3) -> None

Disconnect ((daeDataReporter_t)arg1) → bool
 Disconnect((daeDataReporter_t)arg1) -> None

EndOfData ((daeDataReporter_t)arg1) → bool
 EndOfData((daeDataReporter_t)arg1) -> None

EndRegistration ((daeDataReporter_t)arg1) → bool
 EndRegistration((daeDataReporter_t)arg1) -> None

IsConnected ((daeDataReporter_t)arg1) → bool
 IsConnected((daeDataReporter_t)arg1) -> None

RegisterDomain ((daeDataReporter_t)arg1, (daeDataReporterDomain)arg2) → bool
 RegisterDomain((daeDataReporter_t)arg1, (daeDataReporterDomain)arg2) -> None

RegisterVariable ((daeDataReporter_t)arg1, (daeDataReporterVariable)arg2) → bool
 RegisterVariable((daeDataReporter_t)arg1, (daeDataReporterVariable)arg2) -> None

SendVariable ((daeDataReporter_t)arg1, (daeDataReporterVariableValue)arg2) → bool
 SendVariable((daeDataReporter_t)arg1, (daeDataReporterVariableValue)arg2) -> None

StartNewResultSet ((daeDataReporter_t)arg1, (float)arg2) → bool
 StartNewResultSet((daeDataReporter_t)arg1, (float)arg2) -> None

StartRegistration ((daeDataReporter_t)arg1) → bool
 StartRegistration((daeDataReporter_t)arg1) -> None

3.4 Module pyIDAS

3.4.1 Overview

Trt mrt.

daeIDAS

daeIDAS

class pyIDAS.daeIDAS

Bases: pyIDAS.daeDAESolver_t

SaveMatrixAsXPM ((daeIDAS)arg1, (str)arg2) → None

SetLASolver ((daeIDAS)arg1, (daeIDASolverType)arg2) → None

```
SetLASolver( (daeIDAS)arg1, (object)arg2) -> None
```

3.5 Module pyUnits

3.5.1 Overview

Trt mrt.

3.5.2 Classes

<code>unit</code>
<code>quantity</code>

unit

```
class pyUnits.unit
    Bases: Boost.Python.instance
    baseUnit
    unitDictionary
```

quantity

```
class pyUnits.quantity
    Bases: Boost.Python.instance
    scaleTo ((quantity)arg1, (object)arg2) → quantity
    units
    value
    valueInSIUnits
```


INDICES AND TABLES

- *genindex*
- *modindex*
- *search*

PYTHON MODULE INDEX

p

`pyActivity`, [24](#)

`pyCore`, [5](#)

`pyDataReporting`, [27](#)

`pyIDAS`, [27](#)

`pyUnits`, [28](#)

INDEX

Symbols

`__and__()` (pyCore.daeCondition method), 6
`__call__()` (pyCore.daeDEDI method), 19
`__call__()` (pyCore.daeDomain method), 13
`__call__()` (pyCore.daeParameter method), 14
`__call__()` (pyCore.daeScalarExternalFunction method), 18
`__call__()` (pyCore.daeVariable method), 15
`__call__()` (pyCore.daeVectorExternalFunction method), 18
`__contains__()` (pyCore.daeConfig method), 23
`__getitem__()` (pyCore.adouble_array method), 5
`__getitem__()` (pyCore.daeConfig method), 23
`__getitem__()` (pyCore.daeDomain method), 13
`__init__()` (pyActivity.daeOptimization method), 26
`__init__()` (pyActivity.daeSimulation method), 26
`__init__()` (pyCore.daeArrayRange method), 19
`__init__()` (pyCore.daeDomain method), 13
`__init__()` (pyCore.daeDomainIndex method), 19
`__init__()` (pyCore.daeIndexRange method), 19
`__init__()` (pyCore.daeModel method), 15
`__init__()` (pyCore.daeVariableType method), 12
`__init__()` (pyCore.daeVariableWrapper method), 22
`__instance_size__` (pyActivity.daeOptimization attribute), 26
`__instance_size__` (pyActivity.daeSimulation attribute), 26
`__len__()` (pyCore.adouble_array method), 5
`__module__` (pyActivity.daeOptimization attribute), 26
`__module__` (pyActivity.daeSimulation attribute), 26
`__or__()` (pyCore.daeCondition method), 6
`__reduce__()` (pyActivity.daeOptimization method), 26
`__reduce__()` (pyActivity.daeSimulation method), 26
`__setitem__()` (pyCore.adouble_array method), 5
`__setitem__()` (pyCore.daeConfig method), 23

A

`Abs()` (in module pyCore), 7
`AbsoluteTolerance` (pyCore.daeVariableType attribute), 12
`AbsoluteTolerances` (pyActivity.daeSimulation attribute), 24
`ACos()` (in module pyCore), 7
`ACosh()` (in module pyCore), 7
`Actions` (pyCore.daeStateTransition attribute), 18

`ActiveState` (pyCore.daeSTN attribute), 17
`ActivityAction` (pyActivity.daeSimulation attribute), 24
`adouble` (class in pyCore), 5
`adouble_array` (class in pyCore), 5
`Array()` (in module pyCore), 22
`array()` (pyCore.daeParameter method), 13
`array()` (pyCore.daeVariable method), 15
`ASin()` (in module pyCore), 7
`ASinh()` (in module pyCore), 7
`AssignValue()` (pyCore.daeVariable method), 14
`AssignValues()` (pyCore.daeVariable method), 14
`ATan()` (in module pyCore), 7
`ATan2()` (in module pyCore), 7
`ATanh()` (in module pyCore), 7
`Average()` (in module pyCore), 22

B

`baseUnit` (pyUnits.unit attribute), 28

C

`Calculate()` (pyCore.daeScalarExternalFunction method), 18
`Calculate()` (pyCore.daeVectorExternalFunction method), 19
`CanonicalName` (pyCore.daeObject attribute), 12
`Ceil()` (in module pyCore), 7
`CleanUpSetupData()` (pyActivity.daeSimulation method), 24
`cnAlgebraic` (in module pyCore), 23
`cnAssigned` (in module pyCore), 24
`cnDifferential` (in module pyCore), 24
`ComponentArrays` (pyCore.daeModel attribute), 16
`Components` (pyCore.daeModel attribute), 16
`Condition` (pyCore.daeStateTransition attribute), 18
`Connect()` (pyDataReporting.daeDataReporter_t method), 27
`ConnectEventPorts()` (pyCore.daeModel method), 16
`ConnectPorts()` (pyCore.daeModel method), 16
`Constant()` (in module pyCore), 22
`Constraints` (pyActivity.daeSimulation attribute), 24
`Cos()` (in module pyCore), 7
`Cosh()` (in module pyCore), 7
`CreateArray()` (pyCore.daeDomain method), 13
`CreateDistributed()` (pyCore.daeDomain method), 13
`CreateEqualityConstraint()` (pyActivity.daeSimulation method), 24

CreateEquation() (pyCore.daeModel method), 16
CreateInequalityConstraint() (pyActivity.daeSimulation method), 24
CurrentTime (pyActivity.daeSimulation attribute), 24

D

d() (in module pyCore), 22
d() (pyCore.daeVariable method), 15
d2() (pyCore.daeVariable method), 15
d2_array() (pyCore.daeVariable method), 15
d_array() (pyCore.daeVariable method), 15
daeAction (class in pyCore), 19
daeArrayRange (class in pyCore), 19
daeBaseLog (class in pyCore), 21
daeCondition (class in pyCore), 6
daeConfig (class in pyCore), 23
daeDataReporter_t (class in pyDataReporting), 27
daeDEDI (class in pyCore), 19
daeDomain (class in pyCore), 13
daeDomainIndex (class in pyCore), 19
daeActionType (class in pyCore), 11
daeBinaryFunctions (class in pyCore), 10
daeConditionType (class in pyCore), 11
daeConstraintType (class in pyCore), 10
daeDiscretizationMethod (class in pyCore), 8
daeDomainBounds (class in pyCore), 9
daeDomainIndexType (class in pyCore), 9
daeDomainType (class in pyCore), 8
daeEquationType (class in pyCore), 11
daeInitialConditionMode (class in pyCore), 9
daeLogicalBinaryOperator (class in pyCore), 11
daeLogicalUnaryOperator (class in pyCore), 11
daeModelLanguage (class in pyCore), 9
daeModelType (class in pyCore), 11
daeOptimizationVariableType (class in pyCore), 9
daeParameterType (class in pyCore), 8
daePortType (class in pyCore), 8
daeEquation (class in pyCore), 17
daeEquationExecutionInfo (class in pyCore), 20
daeRangeType (class in pyCore), 9
daeSpecialUnaryFunctions (class in pyCore), 10
daeUnaryFunctions (class in pyCore), 10
daeEventPort (class in pyCore), 18
daeFileLog (class in pyCore), 21
daeGetConfig() (in module pyCore), 23
daeIDAS (class in pyIDAS), 27
daeIF (class in pyCore), 17
daeIndexRange (class in pyCore), 19
daeIndexRangeType (class in pyCore), 9
daeLog_t (class in pyCore), 21
daeMeasuredVariable (class in pyCore), 20
daeModel (class in pyCore), 15
daeObject (class in pyCore), 12
daeObjectiveFunction (class in pyCore), 20
daeOptimization (class in pyActivity), 26
daeOptimizationConstraint (class in pyCore), 20
daeOptimizationVariable (class in pyCore), 20
daeParameter (class in pyCore), 13

daePort (class in pyCore), 18
daePortConnection (class in pyCore), 18
daeScalarExternalFunction (class in pyCore), 18
daeSimulation (class in pyActivity), 24
DAESolver (pyActivity.daeSimulation attribute), 24
daeState (class in pyCore), 18
daeStateTransition (class in pyCore), 18
daeStdOutLog (class in pyCore), 21
daeSTN (class in pyCore), 17
daeTCPIPLog (class in pyCore), 21
daeTCPIPLogServer (class in pyCore), 22
daeVariable (class in pyCore), 14
daeVariableType (class in pyCore), 12
daeVariableWrapper (class in pyCore), 22
daeVectorExternalFunction (class in pyCore), 18
daeVersion() (in module pyCore), 23
daeVersionBuild() (in module pyCore), 23
daeVersionMajor() (in module pyCore), 23
daeVersionMinor() (in module pyCore), 23
DataReporter (pyActivity.daeSimulation attribute), 24
DeclareEquations() (pyCore.daeModel method), 16
DecreaseIndent() (pyCore.daeBaseLog method), 21
DecreaseIndent() (pyCore.daeLog_t method), 21
DEDI (pyCore.daeDomainIndex attribute), 19
Derivative (pyCore.adouble attribute), 5
Description (pyCore.daeObject attribute), 12
Disconnect() (pyDataReporting.daeDataReporter_t method), 27
DiscretizationMethod (pyCore.daeDomain attribute), 13
DiscretizationOrder (pyCore.daeDomain attribute), 13
DistributedEquationDomainInfos (pyCore.daeEquation attribute), 17
DistributeOnDomain() (pyCore.daeEquation method), 17
DistributeOnDomain() (pyCore.daeParameter method), 14
DistributeOnDomain() (pyCore.daeVariable method), 15
Domain (pyCore.daeDEDI attribute), 19
Domain (pyCore.daeIndexRange attribute), 19
DomainBounds (pyCore.daeDEDI attribute), 19
DomainIndex (pyCore.daeArrayRange attribute), 19
DomainIndexes (pyCore.daeVariableWrapper attribute), 22
DomainPoints (pyCore.daeDEDI attribute), 19
Domains (pyCore.daeModel attribute), 16
Domains (pyCore.daeParameter attribute), 14
Domains (pyCore.daePort attribute), 18
Domains (pyCore.daeVariable attribute), 15
dt() (in module pyCore), 22
dt() (pyCore.daeVariable method), 15
dt_array() (pyCore.daeVariable method), 15

E

eAbs (pyCore.daeUnaryFunctions attribute), 10
eAlgebraic (pyCore.daeEquationType attribute), 11

- eAlgebraicValuesProvided (pyCore.daeInitialConditionMode attribute), 9
- eAllPointsInDomain (pyCore.daeIndexRangeType attribute), 9
- eAnd (pyCore.daeLogicalBinaryOperator attribute), 11
- eArcCos (pyCore.daeUnaryFunctions attribute), 10
- eArcSin (pyCore.daeUnaryFunctions attribute), 10
- eArcTan (pyCore.daeUnaryFunctions attribute), 10
- eArray (pyCore.daeDomainType attribute), 8
- eAverage (pyCore.daeSpecialUnaryFunctions attribute), 10
- eBFDM (pyCore.daeDiscretizationMethod attribute), 8
- eBFUnknown (pyCore.daeBinaryFunctions attribute), 10
- eBinaryVariable (pyCore.daeOptimizationVariableType attribute), 9
- eBool (pyCore.daeParameterType attribute), 8
- eBOUnknown (pyCore.daeLogicalBinaryOperator attribute), 11
- eCDAE (pyCore.daeModelLanguage attribute), 10
- eCeil (pyCore.daeUnaryFunctions attribute), 10
- eCFDM (pyCore.daeDiscretizationMethod attribute), 8
- eChangeState (pyCore.daeActionType attribute), 11
- eClosedClosed (pyCore.daeDomainBounds attribute), 9
- eClosedOpen (pyCore.daeDomainBounds attribute), 9
- eConstantIndex (pyCore.daeDomainIndexType attribute), 9
- eContinuousVariable (pyCore.daeOptimizationVariableType attribute), 9
- eCos (pyCore.daeUnaryFunctions attribute), 10
- eCTUnknown (pyCore.daeConditionType attribute), 11
- eCustomDM (pyCore.daeDiscretizationMethod attribute), 8
- eCustomRange (pyCore.daeIndexRangeType attribute), 9
- eDAE (pyCore.daeModelType attribute), 11
- eDBUnknown (pyCore.daeDomainBounds attribute), 9
- eDifferentialValuesProvided (pyCore.daeInitialConditionMode attribute), 9
- eDistributed (pyCore.daeDomainType attribute), 8
- eDITUnknown (pyCore.daeDomainIndexType attribute), 9
- eDivide (pyCore.daeBinaryFunctions attribute), 10
- eDMUnknown (pyCore.daeDiscretizationMethod attribute), 9
- eDomainIterator (pyCore.daeDomainIndexType attribute), 9
- eDTUnknown (pyCore.daeDomainType attribute), 8
- eEQ (pyCore.daeConditionType attribute), 11
- eEqualityConstraint (pyCore.daeConstraintType attribute), 10
- eETUnknown (pyCore.daeEquationType attribute), 11
- eExp (pyCore.daeUnaryFunctions attribute), 10
- eExplicitODE (pyCore.daeEquationType attribute), 11
- eFFDM (pyCore.daeDiscretizationMethod attribute), 9
- eFloor (pyCore.daeUnaryFunctions attribute), 10
- eGT (pyCore.daeConditionType attribute), 11
- eGTEQ (pyCore.daeConditionType attribute), 11
- eICTUnknown (pyCore.daeInitialConditionMode attribute), 9
- eImplicitODE (pyCore.daeEquationType attribute), 11
- eIncrementedDomainIterator (pyCore.daeDomainIndexType attribute), 9
- eInequalityConstraint (pyCore.daeConstraintType attribute), 10
- eInletPort (pyCore.daePortType attribute), 8
- eInteger (pyCore.daeParameterType attribute), 8
- eIntegerVariable (pyCore.daeOptimizationVariableType attribute), 9
- eIRTUnknown (pyCore.daeIndexRangeType attribute), 9
- eLn (pyCore.daeUnaryFunctions attribute), 10
- eLog (pyCore.daeUnaryFunctions attribute), 10
- eLowerBound (pyCore.daeDomainBounds attribute), 9
- ELSE() (pyCore.daeModel method), 16
- ELSE_IF() (pyCore.daeModel method), 16
- eLT (pyCore.daeConditionType attribute), 11
- eLTEQ (pyCore.daeConditionType attribute), 11
- eMax (pyCore.daeBinaryFunctions attribute), 10
- eMaxInArray (pyCore.daeSpecialUnaryFunctions attribute), 10
- eMin (pyCore.daeBinaryFunctions attribute), 10
- eMinInArray (pyCore.daeSpecialUnaryFunctions attribute), 10
- eMinus (pyCore.daeBinaryFunctions attribute), 10
- eMLNone (pyCore.daeModelLanguage attribute), 10
- eMTUnknown (pyCore.daeModelType attribute), 11
- eMulti (pyCore.daeBinaryFunctions attribute), 10
- Enabled (pyCore.daeLog_t attribute), 21
- END_IF() (pyCore.daeModel method), 16
- END_STN() (pyCore.daeModel method), 16
- EndIndex (pyCore.daeIndexRange attribute), 19
- EndOfData() (pyDataReporting.daeDataReporter_t method), 27
- EndRegistration() (pyDataReporting.daeDataReporter_t method), 27
- eNot (pyCore.daeLogicalUnaryOperator attribute), 11
- eNotEQ (pyCore.daeConditionType attribute), 11
- eODE (pyCore.daeModelType attribute), 11
- eOpenClosed (pyCore.daeDomainBounds attribute), 9
- eOpenOpen (pyCore.daeDomainBounds attribute), 9
- eOr (pyCore.daeLogicalBinaryOperator attribute), 11
- eOutletPort (pyCore.daePortType attribute), 8

ePlus (pyCore.daeBinaryFunctions attribute), 10
 ePower (pyCore.daeBinaryFunctions attribute), 10
 eProduct (pyCore.daeSpecialUnaryFunctions attribute), 11
 ePTUnknown (pyCore.daeParameterType attribute), 8
 ePYDAE (pyCore.daeModelLanguage attribute), 10
 eQuasySteadyState (pyCore.daeInitialConditionMode attribute), 9
 EquationExecutionInfos (pyCore.daeEquation attribute), 17
 Equations (pyCore.daeModel attribute), 16
 Equations (pyCore.daePortConnection attribute), 18
 Equations (pyCore.daeState attribute), 18
 EquationType (pyCore.daeEquation attribute), 17
 EquationType (pyCore.daeEquationExecutionInfo attribute), 20
 eRange (pyCore.daeRangeType attribute), 9
 eRangeDomainIndex (pyCore.daeRangeType attribute), 9
 eRangeOfIndexes (pyCore.daeIndexRangeType attribute), 9
 eRaTUnknown (pyCore.daeRangeType attribute), 9
 eReal (pyCore.daeParameterType attribute), 8
 eReAssignOrReInitializeVariable (pyCore.daeActionType attribute), 11
 eSendEvent (pyCore.daeActionType attribute), 11
 eSign (pyCore.daeUnaryFunctions attribute), 10
 eSin (pyCore.daeUnaryFunctions attribute), 10
 eSqrt (pyCore.daeUnaryFunctions attribute), 10
 eSteadyState (pyCore.daeModelType attribute), 11
 eSUFUnknown (pyCore.daeSpecialUnaryFunctions attribute), 11
 eSum (pyCore.daeSpecialUnaryFunctions attribute), 11
 ETA (pyCore.daeLog_t attribute), 21
 eTan (pyCore.daeUnaryFunctions attribute), 10
 eUFUnknown (pyCore.daeUnaryFunctions attribute), 10
 eUnknownAction (pyCore.daeActionType attribute), 11
 eUnknownPort (pyCore.daePortType attribute), 8
 eUOUnknown (pyCore.daeLogicalUnaryOperator attribute), 11
 eUpperBound (pyCore.daeDomainBounds attribute), 9
 eUserDefinedAction (pyCore.daeActionType attribute), 11
 EventData (pyCore.daeEventPort attribute), 18
 EventPorts (pyCore.daeModel attribute), 16
 Events (pyCore.daeEventPort attribute), 18
 EventTolerance (pyCore.daeCondition attribute), 6
 Execute() (pyCore.daeAction method), 20
 Exp() (in module pyCore), 6
 Export() (pyCore.daeModel method), 16
 Export() (pyCore.daePort method), 18
 ExportObjects() (pyCore.daeModel method), 16
 Expressions (pyCore.daeCondition attribute), 6

F

Finalize() (pyActivity.daeOptimization method), 26
 Finalize() (pyActivity.daeSimulation method), 24
 Floor() (in module pyCore), 7

G

GatherInfo (pyCore.adouble attribute), 5
 GatherInfo (pyCore.adouble_array attribute), 6
 GetBoolean() (pyCore.daeConfig method), 23
 GetDomainsIndexesMap() (pyCore.daeParameter method), 14
 GetDomainsIndexesMap() (pyCore.daeVariable method), 15
 GetFloat() (pyCore.daeConfig method), 23
 GetInteger() (pyCore.daeConfig method), 23
 GetNameRelativeToParentModel() (pyCore.daeObject method), 12
 GetQuantity() (pyCore.daeParameter method), 13
 GetQuantity() (pyCore.daeVariable method), 14
 GetString() (pyCore.daeConfig method), 23
 GetStrippedName() (pyCore.daeObject method), 12
 GetStrippedNameRelativeToParentModel() (pyCore.daeObject method), 12
 GetValue() (pyCore.daeParameter method), 13
 GetValue() (pyCore.daeVariable method), 14
 Gradients (pyCore.daeMeasuredVariable attribute), 20
 Gradients (pyCore.daeObjectiveFunction attribute), 20
 Gradients (pyCore.daeOptimizationConstraint attribute), 20

H

has_key() (pyCore.daeConfig method), 23

I

ID (pyCore.daeObject attribute), 12
 IF() (pyCore.daeModel method), 16
 IncreaseIndent() (pyCore.daeBaseLog method), 21
 IncreaseIndent() (pyCore.daeLog_t method), 21
 Increment (pyCore.daeDomainIndex attribute), 19
 Indent (pyCore.daeLog_t attribute), 21
 IndentString (pyCore.daeLog_t attribute), 21
 Index (pyCore.daeDomainIndex attribute), 19
 IndexMappings (pyActivity.daeSimulation attribute), 24
 InitialConditionMode (pyActivity.daeSimulation attribute), 24
 InitialConditionMode (pyCore.daeModel attribute), 16
 InitialDerivatives (pyActivity.daeSimulation attribute), 24
 InitialGuess (pyCore.daeVariableType attribute), 12
 Initialize() (pyActivity.daeOptimization method), 26
 Initialize() (pyActivity.daeSimulation method), 24
 InitialValues (pyActivity.daeSimulation attribute), 24
 InputVariables (pyActivity.daeSimulation attribute), 24
 Integral() (in module pyCore), 22
 Integrate() (pyActivity.daeSimulation method), 25
 IntegrateForTimeInterval() (pyActivity.daeSimulation method), 25

IntegrateUntilTime() (pyActivity.daeSimulation method), 25
 IsConnected() (pyDataReporting.daeDataReporter_t method), 27
 IsModelDynamic (pyCore.daeModel attribute), 16
 items() (pyCore.adouble_array method), 6

J

JoinMessages() (pyCore.daeLog_t method), 21

L

Library (pyCore.daeObject attribute), 12
 LoadInitializationValues() (pyActivity.daeSimulation method), 25
 Log (pyActivity.daeSimulation attribute), 25
 Log() (in module pyCore), 6
 Log10() (in module pyCore), 7
 LowerBound (pyCore.daeDomain attribute), 13
 LowerBound (pyCore.daeOptimizationVariable attribute), 20
 LowerBound (pyCore.daeVariableType attribute), 12

M

m (pyActivity.daeSimulation attribute), 26
 Max() (in module pyCore), 7
 MeasuredVariables (pyActivity.daeSimulation attribute), 25
 Message() (pyCore.daeBaseLog method), 21
 Message() (pyCore.daeFileLog method), 21
 Message() (pyCore.daeLog_t method), 21
 Message() (pyCore.daeStdOutLog method), 21
 Message() (pyCore.daeTCPIPLog method), 22
 MessageReceived() (pyCore.daeTCPIPLogServer method), 22
 Min() (in module pyCore), 7
 Model (pyActivity.daeSimulation attribute), 25
 model (pyActivity.daeSimulation attribute), 26
 Model (pyCore.daeObject attribute), 12
 ModelParameters (pyActivity.daeSimulation attribute), 25
 ModelType (pyCore.daeModel attribute), 16

N

Name (pyCore.daeMeasuredVariable attribute), 20
 Name (pyCore.daeObject attribute), 13
 Name (pyCore.daeObjectiveFunction attribute), 20
 Name (pyCore.daeOptimizationConstraint attribute), 20
 Name (pyCore.daeOptimizationVariable attribute), 20
 Name (pyCore.daeScalarExternalFunction attribute), 18
 Name (pyCore.daeVariableType attribute), 12
 Name (pyCore.daeVariableWrapper attribute), 22
 Name (pyCore.daeVectorExternalFunction attribute), 19
 NestedSTNs (pyCore.daeState attribute), 18
 NextReportingTime (pyActivity.daeSimulation attribute), 25

Node (pyCore.adouble attribute), 5
 Node (pyCore.adouble_array attribute), 6
 Node (pyCore.daeEquationExecutionInfo attribute), 20
 NoPoints (pyCore.daeArrayRange attribute), 19
 NoPoints (pyCore.daeIndexRange attribute), 19
 npyIDs (pyCore.daeVariable attribute), 15
 npyPoints (pyCore.daeDomain attribute), 13
 npyValues (pyCore.daeParameter attribute), 14
 npyValues (pyCore.daeVariable attribute), 15
 NumberOfEquations (pyActivity.daeSimulation attribute), 25
 NumberOfIntervals (pyCore.daeDomain attribute), 13
 NumberOfObjectiveFunctions (pyActivity.daeSimulation attribute), 25
 NumberOfPoints (pyCore.daeDomain attribute), 13
 NumberOfPoints (pyCore.daeParameter attribute), 14
 NumberOfPoints (pyCore.daeVariable attribute), 15

O

ObjectiveFunction (pyActivity.daeSimulation attribute), 25
 ObjectiveFunctions (pyActivity.daeSimulation attribute), 25
 ON_CONDITION() (pyCore.daeModel method), 16
 ON_EVENT() (pyCore.daeModel method), 17
 OnEventActions (pyCore.daeModel attribute), 17
 OptimizationVariables (pyActivity.daeSimulation attribute), 25
 OverallIndex (pyCore.daeVariable attribute), 15
 OverallIndex (pyCore.daeVariableWrapper attribute), 22

P

Parameters (pyCore.daeModel attribute), 17
 Parameters (pyCore.daePort attribute), 18
 Pause() (pyActivity.daeSimulation method), 25
 PercentageDone (pyCore.daeLog_t attribute), 21
 Points (pyCore.daeDomain attribute), 13
 PortArrays (pyCore.daeModel attribute), 17
 PortConnections (pyCore.daeModel attribute), 17
 PortFrom (pyCore.daePortConnection attribute), 18
 Ports (pyCore.daeModel attribute), 17
 PortTo (pyCore.daePortConnection attribute), 18
 Pow() (in module pyCore), 7
 PrintProgress (pyCore.daeLog_t attribute), 21
 Product() (in module pyCore), 22
 Progress (pyCore.daeLog_t attribute), 21
 pyActivity (module), 24
 pyCore (module), 5
 pyDataReporting (module), 27
 pyIDAS (module), 27
 pyUnits (module), 28

Q

quantity (class in pyUnits), 28

R

Range (pyCore.daeArrayRange attribute), 19

ReAssignValue() (pyCore.daeVariable method), 14
ReAssignValues() (pyCore.daeVariable method), 14
ReceiveEvent() (pyCore.daeEventPort method), 18
RecordEvents (pyCore.daeEventPort attribute), 18
RegisterData() (pyActivity.daeSimulation method), 25
RegisterDomain() (pyDataReporting.daeDataReporter_t method), 27
RegisterVariable() (pyDataReporting.daeDataReporter_t method), 27
Reinitialize() (pyActivity.daeSimulation method), 25
RelativeTolerance (pyActivity.daeSimulation attribute), 25
Reload() (pyCore.daeConfig method), 23
ReportData() (pyActivity.daeSimulation method), 25
ReportingInterval (pyActivity.daeSimulation attribute), 25
ReportingOn (pyCore.daeParameter attribute), 14
ReportingOn (pyCore.daeVariable attribute), 15
ReportingTimes (pyActivity.daeSimulation attribute), 25
ReRun() (pyActivity.daeSimulation method), 25
Reset() (pyActivity.daeSimulation method), 25
ReSetInitialCondition() (pyCore.daeVariable method), 14
ReSetInitialConditions() (pyCore.daeVariable method), 14
Residual (pyCore.daeEquation attribute), 18
Residual (pyCore.daeMeasuredVariable attribute), 20
Residual (pyCore.daeObjectiveFunction attribute), 20
Residual (pyCore.daeOptimizationConstraint attribute), 20
Resize() (pyCore.adouble_array method), 6
Resume() (pyActivity.daeSimulation method), 25
Run() (pyActivity.daeOptimization method), 26
Run() (pyActivity.daeSimulation method), 25
RuntimeNode (pyCore.daeAction attribute), 20
RuntimeNode (pyCore.daeCondition attribute), 6

S

SaveMatrixAsXPM() (pyIDAS.daeIDAS method), 27
SaveModelReport() (pyCore.daeModel method), 17
SaveRuntimeModelReport() (pyCore.daeModel method), 17
scaleTo() (pyUnits.quantity method), 28
Scaling (pyCore.daeEquation attribute), 18
SendEvent() (pyCore.daeEventPort method), 18
SendEventPort (pyCore.daeAction attribute), 20
SendVariable() (pyDataReporting.daeDataReporter_t method), 27
SetAbsoluteTolerances() (pyCore.daeVariable method), 14
SetBinaryOptimizationVariable() (pyActivity.daeSimulation method), 25
SetBoolean() (pyCore.daeConfig method), 23
SetContinuousOptimizationVariable() (pyActivity.daeSimulation method), 25
SetFloat() (pyCore.daeConfig method), 23
SetInitialCondition() (pyCore.daeVariable method), 14
SetInitialConditions() (pyCore.daeVariable method), 14
SetInitialGuess() (pyCore.daeVariable method), 14
SetInitialGuesses() (pyCore.daeVariable method), 14
SetInputVariable() (pyActivity.daeSimulation method), 25
SetInteger() (pyCore.daeConfig method), 23
SetIntegerOptimizationVariable() (pyActivity.daeSimulation method), 25
SetLASolver() (pyIDAS.daeIDAS method), 27
SetMeasuredVariable() (pyActivity.daeSimulation method), 25
SetModelParameter() (pyActivity.daeSimulation method), 26
SetProgress() (pyCore.daeBaseLog method), 21
SetReportingOn() (pyCore.daeModel method), 17
SetReportingOn() (pyCore.daePort method), 18
SetString() (pyCore.daeConfig method), 23
SetupNode (pyCore.daeAction attribute), 20
SetupNode (pyCore.daeCondition attribute), 6
SetUpOptimization() (pyActivity.daeSimulation method), 26
SetUpParameterEstimation() (pyActivity.daeSimulation method), 26
SetUpParametersAndDomains() (pyActivity.daeSimulation method), 26
SetUpSensitivityAnalysis() (pyActivity.daeSimulation method), 26
SetUpVariables() (pyActivity.daeSimulation method), 26
SetValue() (pyCore.daeParameter method), 13
SetValue() (pyCore.daeVariable method), 14
SetValues() (pyCore.daeParameter method), 13
SetValues() (pyCore.daeVariable method), 14
SimulationMode (pyActivity.daeSimulation attribute), 26
Sin() (in module pyCore), 7
Sinh() (in module pyCore), 7
SolveInitial() (pyActivity.daeSimulation method), 26
Sqrt() (in module pyCore), 7
StartIndex (pyCore.daeIndexRange attribute), 19
StartingPoint (pyCore.daeOptimizationVariable attribute), 20
StartNewResultSet() (pyDataReporting.daeDataReporter_t method), 27
StartRegistration() (pyDataReporting.daeDataReporter_t method), 27
STATE() (pyCore.daeModel method), 17
States (pyCore.daeSTN attribute), 17
StateTo (pyCore.daeAction attribute), 20
StateTransitions (pyCore.daeState attribute), 18
Step (pyCore.daeIndexRange attribute), 19
STN (pyCore.daeAction attribute), 20
STN() (pyCore.daeModel method), 17
STNs (pyCore.daeModel attribute), 17
StoreInitializationValues() (pyActivity.daeSimulation method), 26
Sum() (in module pyCore), 22
SWITCH_TO() (pyCore.daeModel method), 17

T

Tan() (in module pyCore), 7
Tanh() (in module pyCore), 7
Time() (in module pyCore), 22
TimeHorizon (pyActivity.daeSimulation attribute), 26
TotalNumberOfVariables (pyActivity.daeSimulation attribute), 26
Type (pyCore.daeAction attribute), 20
Type (pyCore.daeArrayRange attribute), 19
Type (pyCore.daeDomain attribute), 13
Type (pyCore.daeDomainIndex attribute), 19
Type (pyCore.daeEventPort attribute), 18
Type (pyCore.daeIndexRange attribute), 19
Type (pyCore.daeOptimizationConstraint attribute), 20
Type (pyCore.daeOptimizationVariable attribute), 20
Type (pyCore.daePort attribute), 18

U

unit (class in pyUnits), 28
unitDictionary (pyUnits.unit attribute), 28
Units (pyCore.daeDomain attribute), 13
Units (pyCore.daeParameter attribute), 14
Units (pyCore.daeVariableType attribute), 12
units (pyUnits.quantity attribute), 28
UpperBound (pyCore.daeDomain attribute), 13
UpperBound (pyCore.daeOptimizationVariable attribute), 20
UpperBound (pyCore.daeVariableType attribute), 12

V

Value (pyCore.adouble attribute), 5
Value (pyCore.daeMeasuredVariable attribute), 20
Value (pyCore.daeObjectiveFunction attribute), 20
Value (pyCore.daeOptimizationConstraint attribute), 20
Value (pyCore.daeOptimizationVariable attribute), 20
Value (pyCore.daeVariableWrapper attribute), 22
value (pyUnits.quantity attribute), 28
valueInSIUnits (pyUnits.quantity attribute), 28
Variable (pyCore.daeVariableWrapper attribute), 22
VariableIndexes (pyCore.daeEquationExecutionInfo attribute), 21
Variables (pyCore.daeModel attribute), 17
Variables (pyCore.daePort attribute), 18
VariableType (pyCore.daeVariable attribute), 15
VariableType (pyCore.daeVariableWrapper attribute), 23
VariableTypes (pyActivity.daeSimulation attribute), 26
VariableWrapper (pyCore.daeAction attribute), 20
Version (pyCore.daeObject attribute), 13