FyTok/SpDM API 参考（alpha版）

FuYun 开发组

Nov 30, 2023

CONTENTS

1 FyTok API 参考 3

1.1 fytok.Scenario . . . 3

1.2 fytok.Tokamak . . . 3

1.3 fytok.modules . . . 4

2 SpDM API 参考 41

2.1 spdm.data . . . 41

2.2 spdm.mesh . . . 57

2.3 spdm.view . . . 61

Python Module Index 63

Index 65

Fy**Tok** 和 Sp**DM** 的 API 参考。

本文档的组织结构

- FyTok API 参考
- SpDM API 参考

版本信息

- 版本：alpha
- 日期：2023-11-28

开发人员

- 于治, Zhi Yu, yuzhi@ipp.ac.cn (ASIPP)
- 刘晓娟, Xiaojuan Liu, lxj@ipp.ac.cn (ASIPP)
- ……

致谢

本工作得到了《聚变堆主机关键系统综合研究设施（CRAFT）》项目，《总控课题：集成数值建模和数据分析系统框架开发》的支持（项目编号：2018-000052-73-01-001228.）

CHAPTER

ONE

FYTOK API 参考

1.1 fytok.Scenario

class Scenario(*args, **kwargs)

Bases: `spdm.data.Actor.Actor`

tokamak: `fytok.Tokamak.Tokamak`

pulse_schedule: `fytok.modules.PulseSchedule.PulseSchedule`

time_slice: `TimeSeriesAoS[TimeSlice]`

时间片序列，保存 Actor 历史状态。@note: TimeSeriesAoS 长度为 n(=3) 循环队列。当压入序列的 TimeSlice 数量超出 n 时，会调用 TimeSeriesAoS.__full__(first_slice)

1.2 fytok.Tokamak

class Tokamak(*args, device: str = tags.not_found, shot: int = tags.not_found, run: int = tags.not_found, time: Optional[float] = None, **kwargs)

Bases: `spdm.data.Actor.Actor`

property brief_summary: str

综述模拟内容

property title: str

标题，由初始化信息 dataset_fair.description

property tag: str

当前状态标签，由程序版本、用户名、时间戳等信息确定

dataset_fair: `fytok.modules.DatasetFAIR.DatasetFAIR`

wall: `fytok.modules.Wall.Wall`

tf: `fytok.modules.TF.TF`

pf_active: `fytok.modules.PFActive.PFActive`

magnetics: `fytok.modules.Magnetics.Magnetics`

ec_launchers: `fytok.modules.ECLaunchers.ECLaunchers`

ic_antennas: `fytok.modules.ICAntennas.ICAntennas`

lh_antennas: `fytok.modules.LHAntennas.LHAntennas`

nbi: `fytok.modules.NBI.NBI`

pellets: fytok.modules.Pellets.Pellets

interferometer: fytok.modules.Interferometer.Interferometer

equilibrium: fytok.modules.Equilibrium.Equilibrium

core_profiles: fytok.modules.CoreProfiles.CoreProfiles

core_transport: fytok.modules.CoreTransport.CoreTransport

core_sources: fytok.modules.CoreSources.CoreSources

transport_solver: fytok.modules.TransportSolverNumerics.TransportSolverNumerics

summary: fytok.modules.Summary.Summary

advance(*args, **kwargs)

refresh(*args, **kwargs) → None
更新当前 Actor 的状态。若 time 为 None 或者与当前时间一致，则更新当前状态树，并执行 self.iteration+=1 否则，向 time_slice 队列中压入新的时间片。

time_slice: TimeSeriesAoS[TimeSlice]
时间片序列，保存 Actor 历史状态。@note: TimeSeriesAoS 长度为 n(=3) 循环队列。当压入序列的 TimeSlice 数量超出 n 时，会调用 TimeSeriesAoS.__full__(first_slice)

1.3 fytok.modules

class AMNSData(*args, **kwargs)
Bases: `fytok.ontology.imas_lastest.amns_data._T_amns_data`

class AMNS(*args, **kwargs)
Bases: `spdm.data.HTree.Dict[fytok.modules.AMNSData.AMNSData]`

class CoreProfilesIon(*args, **kwargs)
Bases: `fytok.ontology.imas_lastest.utilities._T_core_profile_ions`

is_impurity: bool

has_fast_particle: bool

label: str
String identifying ion (e.g. H, D, T, He, C, D2, ⋯)

neutral_index: int
Index of the corresponding neutral species in the ../../neutral array

z: float

a: float

mass

charge

z_ion_1d: Expression
Average charge of the ion species (sum of states charge weighted by state density and divided by ion density)

z_ion_square_1d: Expression
Average square charge of the ion species (sum of states square charge weighted by state density and divided by ion density)

element: AoS[PlasmaCompositionNeutralElement]
List of elements forming the atom or molecule

temperature: Expression
Temperature (average over charge states when multiple charge states are considered)

density: Expression
Density (thermal+non-thermal) (sum over charge states when multiple charge states are considered)

density_thermal: Expression
Density (thermal) (sum over charge states when multiple charge states are considered)

density_fast: Expression
Density of fast (non-thermal) particles (sum over charge states when multiple charge states are considered)

pressure: Expression
Pressure (thermal+non-thermal) (sum over charge states when multiple charge states are considered)

pressure_thermal: Expression
Pressure (thermal) associated with random motion ~average((v-average(v))^2) (sum over charge states when multiple charge states are considered)

pressure_fast_perpendicular: Expression
Fast (non-thermal) perpendicular pressure (sum over charge states when multiple charge states are considered)

pressure_fast_parallel: Expression
Fast (non-thermal) parallel pressure (sum over charge states when multiple charge states are considered)

rotation_frequency_tor: Expression
Toroidal rotation frequency (i.e. toroidal velocity divided by the major radius at which the toroidal velocity is taken) (average over charge states when multiple charge states are considered)

multiple_states_flag: int
0-Only the ‘ion’ level is considered and the ‘state’ array of structure is empty; 1-Ion states are considered and are described in the ‘state’ array of structure

Type Multiple states calculation flag

collision_frequency

density_fit: _T_core_profiles_1D_fit
Information on the fit used to obtain the density profile

density_validity: int
valid from automated processing, 1: valid and certified by the RO; - 1 means problem identified in the data processing (request verification by the RO), -2: invalid data, should not be used

Type Indicator of the validity of the density profile. 0

state: AoS[_T_core_profiles_ions_charge_states2]
Quantities related to the different states of the species (ionisation, energy, excitation, …)

temperature_fit: _T_core_profiles_1D_fit
Information on the fit used to obtain the temperature profile

temperature_validity: int
valid from automated processing, 1: valid and certified by the RO; - 1 means problem identified in the data processing (request verification by the RO), -2: invalid data, should not be used

Type Indicator of the validity of the temperature profile. 0

velocity: _T_core_profiles_vector_components_2
Velocity (average over charge states when multiple charge states are considered) at the position of maximum major radius on every flux surface

z_ion: float
Ion charge (of the dominant ionisation state; lumped ions are allowed), volume averaged over plasma radius

class CoreProfilesNeutral(*args, **kwargs)
Bases: `fytok.ontology.imas_lastest.utilities._T_core_profile_neutral`

label: str
String identifying the species (e.g. H, D, T, He, C, D2, DT, CD4, …)

ion_index: int
Index of the corresponding ion species in the ../../ion array

element: AoS[PlasmaCompositionNeutralElement]
List of elements forming the atom or molecule

temperature: Expression
Temperature (average over charge states when multiple charge states are considered)

density: Expression
Density (thermal+non-thermal) (sum over charge states when multiple charge states are considered)

density_thermal: Expression
Density (thermal) (sum over charge states when multiple charge states are considered)

density_fast: Expression
Density of fast (non-thermal) particles (sum over charge states when multiple charge states are considered)

pressure: Expression
Pressure (thermal+non-thermal) (sum over charge states when multiple charge states are considered)

pressure_thermal: Expression
Pressure (thermal) associated with random motion ~average((v-average(v))^2) (sum over charge states when multiple charge states are considered)

pressure_fast_perpendicular: Expression
Fast (non-thermal) perpendicular pressure (sum over charge states when multiple charge states are considered)

pressure_fast_parallel: Expression
Fast (non-thermal) parallel pressure (sum over charge states when multiple charge states are considered)

multiple_states_flag: int
0-Only one state is considered; 1-Multiple states are considered and are described in the state structure

Type Multiple states calculation flag

state: AoS[_T_core_profiles_neutral_state]
Quantities related to the different states of the species (energy, excitation, …)

class CoreProfilesElectrons(*args, **kwargs)
Bases: `fytok.ontology.imas_lastest.utilities._T_core_profiles_profiles_1d_electrons`

z: float

a: float

charge: float

mass: float

temperature: Expression
Temperature

density: Expression
Density (thermal+non-thermal)

density_thermal: Expression
Density of thermal particles

density_fast: Expression
Density of fast (non-thermal) particles

pressure: Expression
Pressure (thermal+non-thermal)

pressure_thermal: Expression
Pressure (thermal) associated with random motion ~average((v-average(v))^2)

pressure_fast_perpendicular: Expression
Fast (non-thermal) perpendicular pressure

pressure_fast_parallel: Expression
Fast (non-thermal) parallel pressure

collisionality_norm: Expression
Collisionality normalised to the bounce frequency

tau

vT

density_fit: _T_core_profiles_1D_fit
Information on the fit used to obtain the density profile

density_validity: int
valid from automated processing, 1: valid and certified by the RO; - 1 means problem identified in the data processing (request verification by the RO), -2: invalid data, should not be used

Type Indicator of the validity of the density profile. 0

temperature_fit: _T_core_profiles_1D_fit
Information on the fit used to obtain the temperature profile

temperature_validity: int
valid from automated processing, 1: valid and certified by the RO; - 1 means problem identified in the data processing (request verification by the RO), -2: invalid data, should not be used

Type Indicator of the validity of the temperature profile. 0

class CoreProfiles1D(*args, **kwargs)
Bases: `fytok.ontology.imas_lastest.utilities._T_core_profiles_profiles_1d`

Ion
alias of `fytok.modules.CoreProfiles.CoreProfilesIon`

Electrons
alias of `fytok.modules.CoreProfiles.CoreProfilesElectrons`

Neutral
alias of `fytok.modules.CoreProfiles.CoreProfilesNeutral`

grid: `CoreRadialGrid`
Radial grid

electrons: `CoreProfilesElectrons`
Quantities related to the electrons

ion: `AoS[CoreProfilesIon]`
Quantities related to the different ion species, in the sense of isonuclear or isomolecular sequences. Ionisation states (or other types of states) must be differentiated at the state level below

neutral: `AoS[CoreProfilesNeutral]`
Quantities related to the different neutral species

n_i_total

pressure

t_i_average: `Expression`
Ion temperature (averaged on charge states and ion species)

n_i_total_over_n_e: `Expression`
Ratio of total ion density (sum over species and charge states) over electron density. (thermal+non-thermal)

n_i_thermal_total: `Expression`
Total ion thermal density (sum over species and charge states)

momentum_tor: `Expression`
Total plasma toroidal momentum, summed over ion species and electrons weighted by their density and major radius, i.e. sum_over_species(n*R*m*Vphi)

zeff: `Expression`
Effective charge

pressure_ion_total: `Expression`
Total (sum over ion species) thermal ion pressure

pressure_thermal: `Expression`
Thermal pressure (electrons+ions)

pressure_perpendicular: `Expression`
Total perpendicular pressure (electrons+ions, thermal+non-thermal)

pressure_parallel: `Expression`
Total parallel pressure (electrons+ions, thermal+non-thermal)

j_total: `Expression`
Total parallel current density = average(jtot.B) / B0, where B0 = Core_Profiles/ Vacuum_Toroidal_Field/ B0

current_parallel_inside: Expression
: Parallel current driven inside the flux surface. Cumulative surface integral of j_total

j_tor: Expression
: Total toroidal current density = average(J_Tor/R) / average(1/R)

j_ohmic: Expression
: Ohmic parallel current density = average(J_Ohmic.B) / B0, where B0 = Core_Profiles/Vacuum_Toroidal_Field/ B0

j_non_inductive: Expression
: Non-inductive (includes bootstrap) parallel current density = average(jni.B) / B0, where B0 = Core_Profiles/Vacuum_Toroidal_Field/ B0

j_bootstrap: Expression
: Bootstrap current density = average(J_Bootstrap.B) / B0, where B0 = Core_Profiles/Vacuum_Toroidal_Field/ B0

conductivity_parallel: Expression
: Parallel conductivity

class EFieldVectorComponents(*args, **kwargs)
: Bases: `fytok.modules.CoreProfiles.CoreProfiles1D.EFieldVectorComponents`, `spdm.data.sp_property.SpTree`

  radial: spdm.data.Expression.Expression

  diamagnetic: spdm.data.Expression.Expression

  poloidal: spdm.data.Expression.Expression

  toroidal: spdm.data.Expression.Expression

e_field: EFieldVectorComponents
: Electric field, averaged on the magnetic surface. E.g for the parallel component, average(E.B) / B0, using core_profiles/vacuum_toroidal_field/b0

phi_potential: Expression
: Electrostatic potential, averaged on the magnetic flux surface

rotation_frequency_tor_sonic: Expression
: Derivative of the flux surface averaged electrostatic potential with respect to the poloidal flux, multiplied by -1. This quantity is the toroidal angular rotation frequency due to the ExB drift, introduced in formula (43) of Hinton and Wong, Physics of Fluids 3082 (1985), also referred to as sonic flow in regimes in which the toroidal velocity is dominant over the poloidal velocity

q: Expression
: only positive when toroidal current and magnetic field are in same direction)

  Type Safety factor (IMAS uses COCOS=11

magnetic_shear: Expression
: Magnetic shear, defined as rho_tor/q . dq/drho_tor

beta_pol

coulomb_logarithm

electron_collision_time

ffprime: Expression

pprime: Expression

t_i_average_fit: _T_core_profiles_1D_fit
Information on the fit used to obtain the t_i_average profile

time: float

zeff_fit: _T_core_profiles_1D_fit
Information on the fit used to obtain the zeff profile

class CoreGlobalQuantities(*args, **kwargs)
Bases: `fytok.ontology.imas_lastest.core_profiles._T_core_profiles_global_quantities`

vacuum_toroidal_field: VacuumToroidalField

ip: float
Total plasma current (toroidal component). Positive sign means anti-clockwise when viewed from above.

current_non_inductive: float
Total non-inductive current (toroidal component). Positive sign means anti-clockwise when viewed from above.

current_bootstrap: float
Bootstrap current (toroidal component). Positive sign means anti-clockwise when viewed from above.

v_loop: float
LCFS loop voltage (positive value drives positive ohmic current that flows anti-clockwise when viewed from above)

li_3: float
Internal inductance. The li_3 definition is used, i.e. li_3 = 2/R0/mu0^2/Ip^2 * int(Bp^2 dV).

beta_tor: float
Toroidal beta, defined as the volume-averaged total perpendicular pressure divided by (B0^2/(2*mu0)), i.e. beta_toroidal = 2 mu0 int(p dV) / V / B0^2

beta_tor_norm: float
Normalised toroidal beta, defined as 100 * beta_tor * a[m] * B0 [T] / ip [MA]

beta_pol: float
Poloidal beta. Defined as betap = 4 int(p dV) / [R_0 * mu_0 * Ip^2]

energy_diamagnetic: float
Plasma energy content = 3/2 * integral over the plasma volume of the total perpendicular pressure

z_eff_resistive: float
Volume average plasma effective charge, estimated from the flux consumption in the ohmic phase

t_e_peaking: float
Electron temperature peaking factor, defined as the Te value at the magnetic axis divided by the volume averaged Te (average over the plasma volume up to the LCFS)

t_i_average_peaking: float
Ion temperature (averaged over ion species and states) peaking factor, defined as the Ti value at the magnetic axis divided by the volume averaged Ti (average over the plasma volume up to the LCFS)

resistive_psi_losses: float
Resistive part of the poloidal flux losses, defined as the volume-averaged scalar product of the electric field and the ohmic current density, normalized by the plasma current and integrated in time from the beginning of the plasma discharge: int ((int(E_field_tor.j_ohm_tor) dV) / Ip) dt)

ejima: float
resistive psi losses divided by (mu0*R*Ip). See S. Ejima et al, Nuclear Fusion, Vol.22, No.10 (1982), 1313

Type Ejima coefficient

t_e_volume_average: float
Volume averaged electron temperature (average over the plasma volume up to the LCFS)

n_e_volume_average: float
Volume averaged electron density (average over the plasma volume up to the LCFS)

class GlobalQuantitiesIon(*args, **kwargs)
Bases: `fytok.modules.CoreProfiles.CoreGlobalQuantities.GlobalQuantitiesIon`, `spdm.data.sp_property.SpTree`

t_i_volume_average: float

n_i_volume_average: float

ion: AoS[GlobalQuantitiesIon]
Quantities related to the different ion species, in the sense of isonuclear or isomolecular sequences. The set of ion species of this array must be the same as the one defined in profiles_1d/ion, at the time slice indicated in ion_time_slice

ion_time_slice: float
Time slice of the profiles_1d array used to define the ion composition of the global_quantities/ion array.

class CoreProfilesTimeSlice(*args, **kwargs)
Bases: `spdm.data.TimeSeries.TimeSlice`

Profiles1D
alias of `fytok.modules.CoreProfiles.CoreProfiles1D`

GlobalQuantities
alias of `fytok.modules.CoreProfiles.CoreGlobalQuantities`

profiles_1d: fytok.modules.CoreProfiles.CoreProfiles1D

global_quantities: fytok.modules.CoreProfiles.CoreGlobalQuantities

vacuum_toroidal_field: fytok.modules.Utilities.VacuumToroidalField

time: float

class CoreProfiles(*args, **kwargs)
Bases: `fytok.modules.Utilities.Module`

code: fytok.modules.Utilities.Code
对于 Module 的一般性说明。@note code 在 __init__ 时由初始化参数定义，同时会根据 code.name 查找相应的 plugin 。

ids_properties: fytok.modules.Utilities.IDSProperties

TimeSlice
alias of `fytok.modules.CoreProfiles.CoreProfilesTimeSlice`

time_slice: `spdm.data.TimeSeries.TimeSeriesAoS[fytok.modules.CoreProfiles.CoreProfilesTimeSlice]`
: 时间片序列，保存 Actor 历史状态。@note: TimeSeriesAoS 长度为 n(=3) 循环队列。当压入序列的 TimeSlice 数量超出 n 时，会调用 TimeSeriesAoS.__full__(first_slice)

refresh(*args, **kwargs)
: 更新当前 Actor 的状态。若 time 为 None 或者与当前时间一致，则更新当前状态树，并执行 self.iteration+=1 否则，向 time_slice 队列中压入新的时间片。

advance(*args, **kwargs)

class CoreSourcesElectrons(*args, **kwargs)
Bases: `fytok.ontology.imas_lastest.core_sources._T_core_sources_source_profiles_1d_electrons`

particles_decomposed: _T_core_sources_source_profiles_1d_particles_decomposed_3
: Decomposition of the source term for electron density equation into implicit and explicit parts

energy_decomposed: _T_core_sources_source_profiles_1d_energy_decomposed_3
: Decomposition of the source term for electron energy equation into implicit and explicit parts

particles: Expression
: Source term for electron density equation

energy: Expression
: Source term for the electron energy equation

particles_inside: Expression
: Electron source inside the flux surface. Cumulative volume integral of the source term for the electron density equation.

power_inside: Expression
: Power coupled to electrons inside the flux surface. Cumulative volume integral of the source term for the electron energy equation

class CoreSourcesIon(*args, **kwargs)
Bases: `fytok.ontology.imas_lastest.core_sources._T_core_sources_source_profiles_1d_ions`

particles_decomposed: _T_core_sources_source_profiles_1d_particles_decomposed_3
: Decomposition of the source term for ion density equation into implicit and explicit parts

energy_decomposed: _T_core_sources_source_profiles_1d_energy_decomposed_3
: Decomposition of the source term for ion energy equation into implicit and explicit parts

particles: Expression
: Source term for ion density equation

element: AoS[_T_plasma_composition_neutral_element]
: List of elements forming the atom or molecule

energy: Expression
: Source term for the ion energy transport equation.

label: str
: String identifying ion (e.g. H, D, T, He, C, D2, ···)

momentum: _T_core_sources_source_profiles_1d_components_2
: Source term for the ion momentum transport equations along various components (directions)

multiple_states_flag: int
0-Only the 'ion' level is considered and the 'state' array of structure is empty; 1-Ion states are considered and are described in the 'state' array of structure

Type Multiple states calculation flag

neutral_index: int
Index of the corresponding neutral species in the ../../neutral array

state: AoS[_T_core_sources_source_profiles_1d_ions_charge_states]
Source terms related to the different charge states of the species (ionisation, energy, excitation, …)

z_ion: float
Ion charge (of the dominant ionisation state; lumped ions are allowed)

class CoreSourcesNeutral(*args, **kwargs)
Bases: `fytok.ontology.imas_lastest.core_sources._T_core_sources_source_profiles_1d_neutral`

element: AoS[_T_plasma_composition_neutral_element]
List of elements forming the atom or molecule

energy: Expression
Source term for the neutral energy transport equation.

ion_index: int
Index of the corresponding ion species in the ../../ion array

label: str
String identifying the neutral species (e.g. H, D, T, He, C, …)

multiple_states_flag: int
0-Only one state is considered; 1-Multiple states are considered and are described in the state structure

Type Multiple states calculation flag

particles: Expression
Source term for neutral density equation

state: AoS[_T_core_sources_source_profiles_1d_neutral_state]
Source terms related to the different charge states of the species (energy, excitation, …)

class CoreSourcesProfiles1D(*args, **kwargs)
Bases: `fytok.ontology.imas_lastest.core_sources._T_core_sources_source_profiles_1d`

grid: CoreRadialGrid
Radial grid

electrons: CoreSourcesElectrons
Sources for electrons

total_ion_energy: Expression
Source term for the total (summed over ion species) energy equation

total_ion_energy_decomposed: _T_core_sources_source_profiles_1d_energy_decomposed_2
Decomposition of the source term for total ion energy equation into implicit and explicit parts

total_ion_power_inside: Expression
Total power coupled to ion species (summed over ion species) inside the flux surface. Cumulative volume integral of the source term for the total ion energy equation

momentum_tor: Expression
Source term for total toroidal momentum equation

torque_tor_inside: Expression
Toroidal torque inside the flux surface. Cumulative volume integral of the source term for the total toroidal momentum equation

momentum_tor_j_cross_b_field: Expression
Contribution to the toroidal momentum source term (already included in the momentum_tor node) corresponding to the toroidal torques onto the thermal plasma due to Lorentz force associated with radial currents. These currents appear as return-currents (enforcing quasi-neutrality, div(J)=0) balancing radial currents of non-thermal particles, e.g. radial currents of fast and trapped neutral-beam-ions.

j_parallel: Expression
Parallel current density source, average(J.B) / B0, where B0 = core_sources/ vacuum_toroidal_field/b0

current_parallel_inside: Expression
Parallel current driven inside the flux surface. Cumulative surface integral of j_parallel

conductivity_parallel: Expression
Parallel conductivity due to this source

ion: AoS[CoreSourcesIon]
Source terms related to the different ions species, in the sense of isonuclear or isomolecular sequences. Ionisation states (and other types of states) must be differentiated at the state level below

neutral: AoS[CoreSourcesNeutral]
Source terms related to the different neutral species

class CoreSourcesGlobalQuantities(*args, **kwargs)
Bases: `fytok.ontology.imas_lastest.core_sources._T_core_sources_source_global`

current_parallel: float
Parallel current driven

electrons: _T_core_sources_source_global_electrons
Sources for electrons

power: float
Total power coupled to the plasma

torque_tor: float
Toroidal torque

total_ion_particles: float
Total ion particle source (summed over ion species)

total_ion_power: float
Total power coupled to ion species (summed over ion species)

class CoreSourcesTimeSlice(*args, **kwargs)
Bases: `spdm.data.TimeSeries.TimeSlice`

Profiles1D
alias of `fytok.modules.CoreSources.CoreSourcesProfiles1D`

GlobalQuantities
alias of `fytok.modules.CoreSources.CoreSourcesGlobalQuantities`

profiles_1d: fytok.modules.CoreSources.CoreSourcesProfiles1D

global_quantities: fytok.modules.CoreSources.CoreSourcesGlobalQuantities

time: float

class CoreSourcesSource(*args, **kwargs)
Bases: fytok.modules.Utilities.Module

identifier: str

species: fytok.modules.Utilities.DistributionSpecies

TimeSlice
alias of fytok.modules.CoreSources.CoreSourcesTimeSlice

time_slice: spdm.data.TimeSeries.TimeSeriesAoS[fytok.modules.CoreSources.CoreSourcesTimeSlice]
时间片序列，保存 Actor 历史状态。@note: TimeSeriesAoS 长度为 n(=3) 循环队列。当压入序列的 TimeSlice 数量超出 n 时，会调用 TimeSeriesAoS.__full__(first_slice)

refresh(*args, equilibrium: fytok.modules.Equilibrium.Equilibrium, **kwargs)
更新当前 Actor 的状态。若 time 为 None 或者与当前时间一致，则更新当前状态树，并执行 self.iteration+=1 否则，向 time_slice 队列中压入新的时间片。

fetch(x: spdm.data.Expression.Expression, **vars) → fytok.modules.CoreSources.CoreSourcesTimeSlice
获取 Actor 的输出

code: fytok.modules.Utilities.Code
对于 Module 的一般性说明。@note code 在 __init__ 时由初始化参数定义，同时会根据 code.name 查找相应的 plugin 。

class CoreSources(*args, **kwargs)
Bases: fytok.modules.Utilities.IDS

Source
alias of fytok.modules.CoreSources.CoreSourcesSource

source: spdm.data.AoS.AoS[fytok.modules.CoreSources.CoreSourcesSource]

refresh(*args, equilibrium: Optional[fytok.modules.Equilibrium.Equilibrium] = None, core_profiles: Optional[fytok.modules.CoreProfiles.CoreProfiles] = None, **kwargs)
更新当前 Actor 的状态。若 time 为 None 或者与当前时间一致，则更新当前状态树，并执行 self.iteration+=1 否则，向 time_slice 队列中压入新的时间片。

advance(*args, equilibrium: Optional[fytok.modules.Equilibrium.Equilibrium] = None, core_profiles: Optional[fytok.modules.CoreProfiles.CoreProfiles] = None, **kwargs)

code: fytok.modules.Utilities.Code
对于 Module 的一般性说明。@note code 在 __init__ 时由初始化参数定义，同时会根据 code.name 查找相应的 plugin 。

ids_properties: fytok.modules.Utilities.IDSProperties
Interface Data Structure properties. This element identifies the node above as an IDS

time_slice: spdm.data.TimeSeries.TimeSeriesAoS[spdm.data.TimeSeries.TimeSlice]
时间片序列，保存 Actor 历史状态。@note: TimeSeriesAoS 长度为 n(=3) 循环队列。当压入序列的 TimeSlice 数量超出 n 时，会调用 TimeSeriesAoS.__full__(first_slice)

class CoreTransportModelParticles(*args, **kwargs)
Bases: fytok.ontology.imas_lastest.core_transport._T_core_transport_model_2_density

d: spdm.data.Expression.Expression
Effective diffusivity

v: spdm.data.Expression.Expression
Effective convection

flux: spdm.data.Expression.Expression
Flux

class CoreTransportModelEnergy(*args, **kwargs)
Bases: `fytok.ontology.imas_lastest.core_transport._T_core_transport_model_2_energy`

d: spdm.data.Expression.Expression
Effective diffusivity

v: spdm.data.Expression.Expression
Effective convection

flux: spdm.data.Expression.Expression
Flux

class CoreTransportModelMomentum(*args, **kwargs)
Bases: `fytok.ontology.imas_lastest.core_transport._T_core_transport_model_4_momentum`

d: spdm.data.Expression.Expression
Effective diffusivity

v: spdm.data.Expression.Expression
Effective convection

flux: spdm.data.Expression.Expression
Flux

flow_damping_rate: spdm.data.Expression.Expression
Damping rate for this flow component (e.g. due to collisions, calculated from a neoclassical model)

class CoreTransportElectrons(*args, **kwargs)
Bases: `fytok.ontology.imas_lastest.core_transport._T_core_transport_model_electrons`

particles: fytok.modules.CoreTransport.CoreTransportModelParticles
Transport quantities for the electron density equation

energy: fytok.modules.CoreTransport.CoreTransportModelEnergy
Transport quantities for the electron energy equation

momentum: fytok.modules.CoreTransport.CoreTransportModelMomentum

class CoreTransportIon(*args, **kwargs)
Bases: `fytok.ontology.imas_lastest.core_transport._T_core_transport_model_ions`

particles: fytok.modules.CoreTransport.CoreTransportModelParticles
Transport related to the ion density equation

energy: fytok.modules.CoreTransport.CoreTransportModelEnergy
Transport coefficients related to the ion energy equation

momentum: fytok.modules.CoreTransport.CoreTransportModelMomentum
Transport coefficients related to the ion momentum equations for various components (directions)

element: AoS[_T_plasma_composition_neutral_element]
List of elements forming the atom or molecule

label: str
String identifying ion (e.g. H, D, T, He, C, D2, …)

multiple_states_flag: int
0-Only the ‘ion’ level is considered and the ‘state’ array of structure is empty; 1-Ion states are considered and are described in the ‘state’ array of structure

Type Multiple states calculation flag

neutral_index: int
Index of the corresponding neutral species in the ../../neutral array

state: AoS[_T_core_transport_model_ions_charge_states]
Transport coefficients related to the different states of the species

z_ion: float
Ion charge (of the dominant ionisation state; lumped ions are allowed)

class CoreTransportNeutral(*args, **kwargs)
Bases: `fytok.ontology.imas_lastest.core_transport._T_core_transport_model_neutral`

particles: fytok.modules.CoreTransport.CoreTransportModelParticles
Transport related to the neutral density equation

energy: fytok.modules.CoreTransport.CoreTransportModelEnergy
Transport coefficients related to the neutral energy equation

element: AoS[_T_plasma_composition_neutral_element]
List of elements forming the atom or molecule

ion_index: int
Index of the corresponding ion species in the ../../ion array

label: str
String identifying ion (e.g. H+, D+, T+, He+2, C+, …)

multiple_states_flag: int
0-Only one state is considered; 1-Multiple states are considered and are described in the state structure

Type Multiple states calculation flag

state: AoS[_T_core_transport_model_neutral_state]
Transport coefficients related to the different states of the species

class CoreTransportProfiles1D(*args, **kwargs)
Bases: `fytok.ontology.imas_lastest.core_transport._T_core_transport_model_profiles_1d`

grid_d: fytok.modules.Utilities.CoreRadialGrid
Grid for effective diffusivities and parallel conductivity

grid_v: _T_core_radial_grid
Grid for effective convections

grid_flux: _T_core_radial_grid
Grid for fluxes

Electrons
alias of `fytok.modules.CoreTransport.CoreTransportElectrons`

Ion
alias of `fytok.modules.CoreTransport.CoreTransportIon`

Neutral
alias of fytok.modules.CoreTransport.CoreTransportNeutral

electrons: fytok.modules.CoreTransport.CoreTransportElectrons
Transport quantities related to the electrons

ion: spdm.data.AoS.AoS[fytok.modules.CoreTransport.CoreTransportIon]
Transport coefficients related to the various ion species, in the sense of isonuclear or isomolecular sequences. Ionisation states (and other types of states) must be differentiated at the state level below

neutral: spdm.data.AoS.AoS[fytok.modules.CoreTransport.CoreTransportNeutral]
Transport coefficients related to the various neutral species

conductivity_parallel: Expression
Parallel conductivity

e_field_radial: Expression
Radial component of the electric field (calculated e.g. by a neoclassical model)

momentum_tor: _T_core_transport_model_1_momentum
Transport coefficients for total toroidal momentum equation

time: float

total_ion_energy: _T_core_transport_model_1_energy
Transport coefficients for the total (summed over ion species) energy equation

class CoreTransportTimeSlice(*args, **kwargs)
Bases: spdm.data.TimeSeries.TimeSlice

Profiles1D
alias of fytok.modules.CoreTransport.CoreTransportProfiles1D

vacuum_toroidal_field: fytok.modules.Utilities.VacuumToroidalField

flux_multiplier: float

profiles_1d: fytok.modules.CoreTransport.CoreTransportProfiles1D

time: float

class CoreTransportModel(*args, **kwargs)
Bases: fytok.modules.Utilities.Module

code: fytok.modules.Utilities.Code
对于 Module 的一般性说明。@note code 在 __init__ 时由初始化参数定义，同时会根据 code.name 查找相应的 plugin 。

TimeSlice
alias of fytok.modules.CoreTransport.CoreTransportTimeSlice

identifier: str

time_slice: spdm.data.TimeSeries.TimeSeriesAoS[fytok.modules.CoreTransport.CoreTransportTimeSlice]
时间片序列，保存 Actor 历史状态。@note: TimeSeriesAoS 长度为 n(=3) 循环队列。当压入序列的 TimeSlice 数量超出 n 时，会调用 TimeSeriesAoS.__full__(first_slice)

refresh(*args, equilibrium: fytok.modules.Equilibrium.Equilibrium, **kwargs)
更新当前 Actor 的状态。若 time 为 None 或者与当前时间一致，则更新当前状态树，并执行 self.iteration+=1 否则，向 time_slice 队列中压入新的时间片。

class CoreTransport(*args, **kwargs)

Bases: `fytok.ontology.imas_lastest.core_transport._T_core_transport`

Model

alias of `fytok.modules.CoreTransport.CoreTransportModel`

model: spdm.data.AoS.AoS[fytok.modules.CoreTransport.CoreTransportModel]

Transport is described by a combination of various transport models

refresh(*args, equilibrium: Optional[fytok.modules.Equilibrium.Equilibrium] = None, core_profiles: Optional[fytok.modules.CoreProfiles.CoreProfiles] = None, **kwargs)

更新当前 Actor 的状态。若 time 为 None 或者与当前时间一致，则更新当前状态树，并执行 self.iteration+=1 否则，向 time_slice 队列中压入新的时间片。

advance(*args, equilibrium: Optional[fytok.modules.Equilibrium.Equilibrium] = None, core_profiles: Optional[fytok.modules.CoreProfiles.CoreProfiles] = None, **kwargs)

advance time_series to next slice

code: Code

对于 Module 的一般性说明。@note code 在 __init__ 时由初始化参数定义，同时会根据 code.name 查找相应的 plugin 。

ids_properties: IDSProperties

Interface Data Structure properties. This element identifies the node above as an IDS

time_slice: TimeSeriesAoS[TimeSlice]

时间片序列，保存 Actor 历史状态。@note: TimeSeriesAoS 长度为 n(=3) 循环队列。当压入序列的 TimeSlice 数量超出 n 时，会调用 TimeSeriesAoS.__full__(first_slice)

vacuum_toroidal_field: _T_b_tor_vacuum_1

Characteristics of the vacuum toroidal field (used in Rho_Tor definition and in the normalization of current densities)

class DataDescription(*args, **kwargs)

Bases: `fytok.modules.DatasetFAIR.DataDescription`, `spdm.data.sp_property.SpTree`

device: str

shot: int

run: int

summary: str

class DatasetFAIR(*args, **kwargs)

Bases: `fytok.ontology.imas_lastest.dataset_fair._T_dataset_fair`

ontology: str

description: fytok.modules.DatasetFAIR.DataDescription

creator

create_time

site

code: Code

对于 Module 的一般性说明。@note code 在 __init__ 时由初始化参数定义，同时会根据 code.name 查找相应的 plugin 。

identifier: str
Persistent identifier allowing to cite this data in a public and persistent way, should be provided as HTTP URIs

ids_properties: IDSProperties
Interface Data Structure properties. This element identifies the node above as an IDS

is_referenced_by: List[str]
List of documents (e.g. publications) or datasets making use of this data entry (e.g. PIDs of other datasets using this data entry as input)

is_replaced_by: str
Persistent identifier referencing the new version of this data (replacing the present version)

license: str
License(s) under which the data is made available (license description or, more convenient, publicly accessible URL pointing to the full license text)

replaces: str
Persistent identifier referencing the previous version of this data

rights_holder: str
The organisation owning or managing rights over this data

time_slice: TimeSeriesAoS[TimeSlice]
时间片序列，保存 Actor 历史状态。@note: TimeSeriesAoS 长度为 n(=3) 循环队列。当压入序列的 TimeSlice 数量超出 n 时，会调用 TimeSeriesAoS.__full__(first_slice)

valid: str
Date range during which the data is or was valid. Expressed as YYYY-MM-DD/YYYY-MM-DD, where the former (resp. latter) date is the data at which the data started (resp. ceased) to be valid. If the data is still valid, the slash should still be present, i.e. indicate the validity start date with YYYY-MM-DD/. If the data ceased being valid but there is no information on the validity start date, indicate /YYYY-MM-DD.

class ECLaunchers(*args, **kwargs)
Bases: `fytok.ontology.imas_lastest.ec_launchers._T_ec_launchers`

class EquilibriumCoordinateSystem(*args, **kwargs)
Bases: `fytok.ontology.imas_lastest.utilities._T_equilibrium_coordinate_system`

grid_type: fytok.modules.Utilities.Identifier
Type of coordinate system

grid: spdm.mesh.Mesh.Mesh
Definition of the 2D grid

radial_grid: fytok.modules.Utilities.CoreRadialGrid

r: spdm.data.Field.Field
Values of the major radius on the grid

z: spdm.data.Field.Field
Values of the Height on the grid

jacobian: spdm.data.Field.Field
Absolute value of the jacobian of the coordinate system

tensor_covariant: numpy.ndarray
Covariant metric tensor on every point of the grid described by grid_type

tensor_contravariant: numpy.ndarray
Contravariant metric tensor on every point of the grid described by grid_type

class EquilibriumGlobalQuantities(*args, **kwargs)
Bases: `fytok.ontology.imas_lastest.equilibrium._T_equilibrium_global_quantities`

beta_pol: float
Poloidal beta. Defined as betap = 4 int(p dV) / [R_0 * mu_0 * Ip^2]

beta_tor: float
Toroidal beta, defined as the volume-averaged total perpendicular pressure divided by (B0^2/(2*mu0)), i.e. beta_toroidal = 2 mu0 int(p dV) / V / B0^2

beta_normal: float
Normalised toroidal beta, defined as 100 * beta_tor * a[m] * B0 [T] / ip [MA]

ip: float
Plasma current (toroidal component). Positive sign means anti-clockwise when viewed from above.

li_3: float
Internal inductance

volume: float
Total plasma volume

area: float
Area of the LCFS poloidal cross section

surface: float
Surface area of the toroidal flux surface

length_pol: float
Poloidal length of the magnetic surface

psi_axis: float
Poloidal flux at the magnetic axis

psi_boundary: float
Poloidal flux at the selected plasma boundary

class MagneticAxis(*args, **kwargs)
Bases: `fytok.modules.Equilibrium.EquilibriumGlobalQuantities.MagneticAxis`, `spdm.data.sp_property.SpTree`

r: float

z: float

b_field_tor: float

magnetic_axis: MagneticAxis
Magnetic axis position and toroidal field

class CurrentCentre(*args, **kwargs)
Bases: `fytok.modules.Equilibrium.EquilibriumGlobalQuantities.CurrentCentre`, `spdm.data.sp_property.SpTree`

r: float

z: float

velocity_z: float

current_centre: CurrentCentre
Position and vertical velocity of the current centre

q_axis: float
q at the magnetic axis

q_95: float
only positive when toroidal current and magnetic field are in same direction)

Type q at the 95% poloidal flux surface (IMAS uses COCOS=11

class Qmin(*args, **kwargs)
Bases: `fytok.modules.Equilibrium.EquilibriumGlobalQuantities.Qmin`, `spdm.data.sp_property.SpTree`

value: float

rho_tor_norm: float

q_min: Qmin
Minimum q value and position

energy_mhd: float
Plasma energy content = 3/2 * int(p,dV) with p being the total pressure (thermal + fast particles) [J]. Time-dependent; Scalar

psi_external_average: float
Average (over the plasma poloidal cross section) plasma poloidal magnetic flux produced by all external circuits (CS and PF coils, eddy currents, VS in-vessel coils), given by the following formula : int(psi_external.j_tor.dS) / Ip

v_external: float
External voltage, i.e. time derivative of psi_external_average (with a minus sign : -d_psi_external_average/d_time)

plasma_inductance: float
Plasma inductance 2 E_magnetic/Ip^2, where E_magnetic = 1/2 * int(psi.j_tor.dS) (integral over the plasma poloidal cross-section)

plasma_resistance: float
Plasma resistance = int(e_field.j.dV) / Ip^2

class EquilibriumProfiles1D(*args, **kwargs)
Bases: `fytok.ontology.imas_lastest.equilibrium._T_equilibrium_profiles_1d`

1D profiles of the equilibrium quantities .. note:

```
- psi_norm is the normalized poloidal flux
- psi is the poloidal flux,
- 以psi而不是psi_norm为主坐标，原因是 profiles1d 中涉及对 psi 的求导和积分
```

grid

psi_norm: array_type

psi: Expression
Poloidal flux

dphi_dpsi: Expression

phi: Expression
Toroidal flux

pressure: Expression
Pressure

f: Expression
Diamagnetic function (F=R B_Phi)

dpressure_dpsi: Expression
Derivative of pressure w.r.t. psi

f_df_dpsi: Expression
Derivative of F w.r.t. Psi, multiplied with F

j_tor: Expression
Flux surface averaged toroidal current density = average(j_tor/R) / average(1/R)

j_parallel: Expression
Flux surface averaged parallel current density = average(j.B) / B0, where B0 = Equilibrium/Global/Toroidal_Field/B0

q: Expression
only positive when toroidal current and magnetic field are in same direction)

Type Safety factor (IMAS uses COCOS=11

magnetic_shear: Expression
Magnetic shear, defined as rho_tor/q . dq/drho_tor

r_inboard: Expression
Radial coordinate (major radius) on the inboard side of the magnetic axis

r_outboard: Expression
Radial coordinate (major radius) on the outboard side of the magnetic axis

rho_tor: Expression
Toroidal flux coordinate = sqrt(phi/(pi*b0)), where the toroidal flux, phi, corresponds to time_slice/ profiles_1d/ phi, the toroidal magnetic field, b0, corresponds to vacuum_toroidal_field/b0 and pi can be found in the IMAS constants

rho_tor_norm: Expression
Normalised toroidal flux coordinate. The normalizing value for rho_tor_norm, is the toroidal flux coordinate at the equilibrium boundary (LCFS or 99.x % of the LCFS in case of a fixed boundary equilibium calculation)

dpsi_drho_tor: Expression
Derivative of Psi with respect to Rho_Tor

geometric_axis: _T_equilibrium_profiles_1d_rz1d_dynamic_aos
RZ position of the geometric axis of the magnetic surfaces (defined as (Rmin+Rmax) / 2 and (Zmin+Zmax) / 2 of the surface)

minor_radius: Expression

major_radius: Expression

magnetic_z: Expression

elongation: Expression
Elongation

triangularity_upper: Expression
Upper triangularity w.r.t. magnetic axis

triangularity_lower: Expression
Lower triangularity w.r.t. magnetic axis

triangularity

squareness_upper_inner: Expression
Upper inner squareness (definition from T. Luce, Plasma Phys. Control. Fusion 55 (2013) 095009)

squareness_upper_outer: Expression
Upper outer squareness (definition from T. Luce, Plasma Phys. Control. Fusion 55 (2013) 095009)

squareness_lower_inner: Expression
Lower inner squareness (definition from T. Luce, Plasma Phys. Control. Fusion 55 (2013) 095009)

squareness_lower_outer: Expression
Lower outer squareness (definition from T. Luce, Plasma Phys. Control. Fusion 55 (2013) 095009)

squareness: Expression

volume: Expression
Volume enclosed in the flux surface

rho_volume_norm: Expression
Normalised square root of enclosed volume (radial coordinate). The normalizing value is the enclosed volume at the equilibrium boundary (LCFS or 99.x % of the LCFS in case of a fixed boundary equilibium calculation)

dvolume_dpsi: Expression
Radial derivative of the volume enclosed in the flux surface with respect to Psi

dvolume_drho_tor: Expression
Radial derivative of the volume enclosed in the flux surface with respect to Rho_Tor

area: Expression
Cross-sectional area of the flux surface

darea_dpsi: Expression
Radial derivative of the cross-sectional area of the flux surface with respect to psi

darea_drho_tor: Expression
Radial derivative of the cross-sectional area of the flux surface with respect to rho_tor

surface: Expression
Surface area of the toroidal flux surface

trapped_fraction: Expression
Trapped particle fraction

gm1: Expression
Flux surface averaged 1/R^2

gm2: Expression
Flux surface averaged |grad_rho_tor|^2/R^2

gm3: Expression
Flux surface averaged |grad_rho_tor|^2

gm4: Expression
Flux surface averaged 1/B^2

gm5: Expression
Flux surface averaged B^2

gm6: `Expression`
Flux surface averaged |grad_rho_tor|^2/B^2

gm7: `Expression`
Flux surface averaged |grad_rho_tor|

gm8: `Expression`
Flux surface averaged R

gm9: `Expression`
Flux surface averaged 1/R

b_field_average: `Expression`
Flux surface averaged modulus of B (always positive, irrespective of the sign convention for the B-field direction).

b_field_min: `Expression`
Minimum(modulus(B)) on the flux surface (always positive, irrespective of the sign convention for the B-field direction)

b_field_max: `Expression`
Maximum(modulus(B)) on the flux surface (always positive, irrespective of the sign convention for the B-field direction)

beta_pol: `Expression`
Poloidal beta profile. Defined as betap = 4 int(p dV) / [R_0 * mu_0 * Ip^2]

mass_density: `Expression`
Mass density

class EquilibriumProfiles2D(*args, **kwargs)
Bases: `fytok.ontology.imas_lastest.equilibrium._T_equilibrium_profiles_2d`

type: `fytok.modules.Utilities.Identifier`
Type of profiles (distinguishes contribution from plasma, vaccum fields and total fields)

grid_type: `fytok.modules.Utilities.Identifier`
Selection of one of a set of grid types

grid: `spdm.mesh.Mesh.Mesh`
Definition of the 2D grid (the content of dim1 and dim2 is defined by the selected grid_type)

r: `spdm.data.Field.Field`
Values of the major radius on the grid

z: `spdm.data.Field.Field`
Values of the Height on the grid

psi: `spdm.data.Field.Field`
Values of the poloidal flux at the grid in the poloidal plane

theta: `spdm.data.Field.Field`
Values of the poloidal angle on the grid

phi: `spdm.data.Field.Field`
Toroidal flux

j_tor: `spdm.data.Field.Field`
Toroidal plasma current density

j_parallel: spdm.data.Field.Field
Defined as (j.B)/B0 where j and B are the current density and magnetic field vectors and B0 is the (signed) vacuum toroidal magnetic field strength at the geometric reference point (R0,Z0). It is formally not the component of the plasma current density parallel to the magnetic field

b_field_r: spdm.data.Field.Field
R component of the poloidal magnetic field

b_field_z: spdm.data.Field.Field
Z component of the poloidal magnetic field

b_field_tor: spdm.data.Field.Field
Toroidal component of the magnetic field

class EquilibriumBoundary(*args, **kwargs)
Bases: `fytok.ontology.imas_lastest.equilibrium._T_equilibrium_boundary`

type: int
0 (limiter) or 1 (diverted)

outline: spdm.geometry.Curve.Curve
RZ outline of the plasma boundary

psi_norm: float
Value of the normalised poloidal flux at which the boundary is taken (typically 99.x %), the flux being normalised to its value at the separatrix

psi: float
Value of the poloidal flux at which the boundary is taken

geometric_axis: spdm.geometry.Point.Point
RZ position of the geometric axis (defined as (Rmin+Rmax) / 2 and (Zmin+Zmax) / 2 of the boundary)

minor_radius: float
Minor radius of the plasma boundary (defined as (Rmax-Rmin) / 2 of the boundary)

elongation: float
Elongation of the plasma boundary

elongation_upper: float
Elongation (upper half w.r.t. geometric axis) of the plasma boundary

elongation_lower: float
Elongation (lower half w.r.t. geometric axis) of the plasma boundary

triangularity: float
Triangularity of the plasma boundary

triangularity_upper: float
Upper triangularity of the plasma boundary

triangularity_lower: float
Lower triangularity of the plasma boundary

squareness_upper_inner: float
Upper inner squareness of the plasma boundary (definition from T. Luce, Plasma Phys. Control. Fusion 55 (2013) 095009)

squareness_upper_outer: float
Upper outer squareness of the plasma boundary (definition from T. Luce, Plasma Phys. Control. Fusion 55 (2013) 095009)

squareness_lower_inner: float
Lower inner squareness of the plasma boundary (definition from T. Luce, Plasma Phys. Control. Fusion 55 (2013) 095009)

squareness_lower_outer: float
Lower outer squareness of the plasma boundary (definition from T. Luce, Plasma Phys. Control. Fusion 55 (2013) 095009)

x_point: spdm.data.AoS.AoS[spdm.geometry.Point.Point]
Array of X-points, for each of them the RZ position is given

strike_point: spdm.data.AoS.AoS[spdm.geometry.Point.Point]
Array of strike points, for each of them the RZ position is given

active_limiter_point: spdm.geometry.Point.Point
RZ position of the active limiter point (point of the plasma boundary in contact with the limiter)

class EquilibriumBoundarySeparatrix(*args, **kwargs)
Bases: `fytok.ontology.imas_lastest.equilibrium._T_equilibrium_boundary_separatrix`

type: int
0 (limiter) or 1 (diverted)

outline: CurveRZ
RZ outline of the plasma boundary

psi: float
Value of the poloidal flux at the separatrix

geometric_axis: Point
RZ position of the geometric axis (defined as (Rmin+Rmax) / 2 and (Zmin+Zmax) / 2 of the boundary)

minor_radius: float
Minor radius of the plasma boundary (defined as (Rmax-Rmin) / 2 of the boundary)

elongation: float
Elongation of the plasma boundary

elongation_upper: float
Elongation (upper half w.r.t. geometric axis) of the plasma boundary

elongation_lower: float
Elongation (lower half w.r.t. geometric axis) of the plasma boundary

triangularity: float
Triangularity of the plasma boundary

triangularity_upper: float
Upper triangularity of the plasma boundary

triangularity_lower: float
Lower triangularity of the plasma boundary

squareness_upper_inner: float
Upper inner squareness of the plasma boundary (definition from T. Luce, Plasma Phys. Control. Fusion 55 (2013) 095009)

squareness_upper_outer: float
Upper outer squareness of the plasma boundary (definition from T. Luce, Plasma Phys. Control. Fusion 55 (2013) 095009)

squareness_lower_inner: float
Lower inner squareness of the plasma boundary (definition from T. Luce, Plasma Phys. Control. Fusion 55 (2013) 095009)

squareness_lower_outer: float
Lower outer squareness of the plasma boundary (definition from T. Luce, Plasma Phys. Control. Fusion 55 (2013) 095009)

x_point: AoS[Point]
Array of X-points, for each of them the RZ position is given

strike_point: AoS[Point]
Array of strike points, for each of them the RZ position is given

active_limiter_point: Point
RZ position of the active limiter point (point of the plasma boundary in contact with the limiter)

closest_wall_point: _T_equilibrium_boundary_closest
Position and distance to the plasma boundary of the point of the first wall which is the closest to plasma boundary

dr_dz_zero_point: PointRZ
Outboard point on the separatrix on which dr/dz = 0 (local maximum of the major radius of the separatrix). In case of multiple local maxima, the closest one from z=z_magnetic_axis is chosen.

gap: AoS[_T_equilibrium_gap]
Set of gaps, defined by a reference point and a direction.

class EequilibriumConstraints(*args, **kwargs)
Bases: `fytok.ontology.imas_lastest.equilibrium._T_equilibrium_constraints`

b_field_tor_vacuum_r: _T_equilibrium_constraints_0D
Vacuum field times major radius in the toroidal field magnet. Positive sign means anti-clockwise when viewed from above

bpol_probe: AoS[_T_equilibrium_constraints_0D_one_like]
Set of poloidal field probes

diamagnetic_flux: _T_equilibrium_constraints_0D_b0_like
Diamagnetic flux

faraday_angle: AoS[_T_equilibrium_constraints_0D]
Set of faraday angles

flux_loop: AoS[_T_equilibrium_constraints_0D_psi_like]
Set of flux loops

ip: _T_equilibrium_constraints_0D_ip_like
Plasma current. Positive sign means anti-clockwise when viewed from above

iron_core_segment: AoS[_T_equilibrium_constraints_magnetisation]
Magnetisation M of a set of iron core segments

mse_polarisation_angle: AoS[_T_equilibrium_constraints_0D]
Set of MSE polarisation angles

n_e: AoS[_T_equilibrium_constraints_0D]
Set of local density measurements

n_e_line: AoS[_T_equilibrium_constraints_0D]
Set of line integrated density measurements

pf_current: AoS[_T_equilibrium_constraints_0D_ip_like]
Current in a set of poloidal field coils

pf_passive_current: AoS[_T_equilibrium_constraints_0D]
Current in a set of axisymmetric passive conductors

pressure: AoS[_T_equilibrium_constraints_0D]
Set of total pressure estimates

q: AoS[_T_equilibrium_constraints_0D_position]
Set of safety factor estimates at various positions

strike_point: AoS[_T_equilibrium_constraints_pure_position]
Array of strike points, for each of them the RZ position is given

x_point: AoS[_T_equilibrium_constraints_pure_position]
Array of X-points, for each of them the RZ position is given

class EquilibriumGGD(*args, **kwargs)
Bases: `fytok.ontology.imas_lastest.equilibrium._T_equilibrium_ggd`

b_field_r: AoS[_T_generic_grid_scalar]
R component of the poloidal magnetic field, given on various grid subsets

b_field_tor: AoS[_T_generic_grid_scalar]
Toroidal component of the magnetic field, given on various grid subsets

b_field_z: AoS[_T_generic_grid_scalar]
Z component of the poloidal magnetic field, given on various grid subsets

j_parallel: AoS[_T_generic_grid_scalar]
Parallel (to magnetic field) plasma current density, given on various grid subsets

j_tor: AoS[_T_generic_grid_scalar]
Toroidal plasma current density, given on various grid subsets

phi: AoS[_T_generic_grid_scalar]
Values of the toroidal flux, given on various grid subsets

psi: AoS[_T_generic_grid_scalar]
Values of the poloidal flux, given on various grid subsets

r: AoS[_T_generic_grid_scalar]
Values of the major radius on various grid subsets

theta: AoS[_T_generic_grid_scalar]
Values of the poloidal angle, given on various grid subsets

z: AoS[_T_generic_grid_scalar]
Values of the Height on various grid subsets

class EquilibriumTimeSlice(*args, **kwargs)
Bases: `fytok.ontology.imas_lastest.equilibrium._T_equilibrium_time_slice`

Constraints
alias of `fytok.modules.Equilibrium.EequilibriumConstraints`

BoundarySeparatrix
alias of `fytok.modules.Equilibrium.EquilibriumBoundarySeparatrix`

Boundary
alias of `fytok.modules.Equilibrium.EquilibriumBoundary`

GlobalQuantities
alias of `fytok.modules.Equilibrium.EquilibriumGlobalQuantities`

CoordinateSystem
alias of `fytok.modules.Equilibrium.EquilibriumCoordinateSystem`

Profiles1D
alias of `fytok.modules.Equilibrium.EquilibriumProfiles1D`

Profiles2D
alias of `fytok.modules.Equilibrium.EquilibriumProfiles2D`

GGD
alias of `fytok.modules.Equilibrium.EquilibriumGGD`

vacuum_toroidal_field: VacuumToroidalField

boundary: EquilibriumBoundary
Description of the plasma boundary used by fixed-boundary codes and typically chosen at psi_norm = 99.x% of the separatrix

boundary_separatrix: BoundarySeparatrix
Description of the plasma boundary at the separatrix

constraints: Constraints
In case of equilibrium reconstruction under constraints, measurements used to constrain the equilibrium, reconstructed values and accuracy of the fit. The names of the child nodes correspond to the following definition: the solver aims at minimizing a cost function defined as : J=1/2*sum_i [weight_i^2 (reconstructed_i - measured_i)^2 / sigma_i^2]. in which sigma_i is the standard deviation of the measurement error (to be found in the IDS of the measurement)

global_quantities: EquilibriumGlobalQuantities
0D parameters of the equilibrium

profiles_1d: Profiles1D
Equilibrium profiles (1D radial grid) as a function of the poloidal flux

profiles_2d: Profiles2D
Equilibrium 2D profiles in the poloidal plane. Multiple 2D representations of the equilibrium can be stored here.

coordinate_system: CoordinateSystem
Flux surface coordinate system on a square grid of flux and poloidal angle

ggd: GGD
Set of equilibrium representations using the generic grid description

boundary_secondary_separatrix: _T_equilibrium_boundary_second_separatrix
Geometry of the secondary separatrix, defined as the outer flux surface with an X-point

convergence: _T_equilibrium_convergence
Convergence details

time: float

class Equilibrium(*args, **kwargs)
Bases: `fytok.modules.Utilities.IDS`

Description of a 2D, axi-symmetric, tokamak equilibrium; result of an equilibrium code.

Reference:

- O. Sauter and S. Yu Medvedev, "Tokamak coordinate conventions: COCOS" , Computer Physics Communications 184, 2 (2013), pp. 293–302.

code: fytok.modules.Utilities.Code
对于 Module 的一般性说明。@note code 在 __init__ 时由初始化参数定义，同时会根据 code.name 查找相应的 plugin 。

ids_properties: fytok.modules.Utilities.IDSProperties
Interface Data Structure properties. This element identifies the node above as an IDS

TimeSlice
alias of `fytok.modules.Equilibrium.EquilibriumTimeSlice`

time_slice: spdm.data.TimeSeries.TimeSeriesAoS[fytok.modules.Equilibrium.EquilibriumTimeSlice]
时间片序列，保存 Actor 历史状态。@note: TimeSeriesAoS 长度为 n(=3) 循环队列。当压入序列的 TimeSlice 数量超出 n 时，会调用 TimeSeriesAoS.__full__(first_slice)

class ICAntennas(*args, **kwargs)
Bases: `fytok.ontology.imas_lastest.ic_antennas._T_ic_antennas`

class Interferometer(*args, **kwargs)
Bases: `fytok.ontology.imas_lastest.interferometer._T_interferometer`

class LHAntennas(*args, **kwargs)
Bases: `fytok.ontology.imas_lastest.lh_antennas._T_lh_antennas`

class Magnetics(*args, **kwargs)
Bases: `fytok.ontology.imas_lastest.magnetics._T_magnetics`

Magnetic diagnostics for equilibrium identification and plasma shape control.

draw_nbi_unit(unit: fytok.ontology.imas_lastest.nbi._T_nbi_unit, name: str)

class NBI(*args, **kwargs)
Bases: `fytok.ontology.imas_lastest.nbi._T_nbi`

class Pellets(*args, **kwargs)
Bases: `fytok.ontology.imas_lastest.pellets._T_pellets`

class PFActive(*args, **kwargs)
Bases: `fytok.ontology.imas_lastest.pf_active._T_pf_active`

class PulseSchedule(*args, **kwargs)
Bases: `fytok.ontology.imas_lastest.pulse_schedule._T_pulse_schedule`

class TF(*args, **kwargs)
Bases: `fytok.ontology.imas_lastest.tf._T_tf`

class IDSProperties(*args, **kwargs)
Bases: `fytok.modules.Utilities.IDSProperties`, `spdm.data.sp_property.SpTree`

comment: str

homogeneous_time: int

provider: str

creation_date: str

version_put: spdm.data.sp_property.SpTree

provenance: spdm.data.sp_property.SpTree

class Library(*args, **kwargs)
Bases: `fytok.modules.Utilities.Library`, `spdm.data.sp_property.SpTree`

name: str

commit: str

version: str

repository: str

parameters: spdm.data.sp_property.SpTree

class Code(*args, **kwargs)
Bases: `fytok.modules.Utilities.Code`, `spdm.data.sp_property.SpTree`

name: str
代码名称，也是调用 plugin 的 identifier

parameters: spdm.data.sp_property.PropertyTree
指定参数列表，代码调用时所需，但不在由 Module 定义的参数列表中的参数。

commit: str

version: str

copyright: str

repository: str

output_flag: numpy.ndarray

library: spdm.data.HTree.List[fytok.modules.Utilities.Library]

class Identifier(*args, **kwargs)
Bases: `fytok.modules.Utilities.Identifier`, `spdm.data.sp_property.SpTree`

name: str

index: int

description: str

class Module(*args, **kwargs)
Bases: `spdm.data.Actor.Actor`

code: fytok.modules.Utilities.Code
对于 Module 的一般性说明。@note code 在 __init__ 时由初始化参数定义，同时会根据 code.name 查找相应的 plugin 。

property tag: str

execute(*args, **kwargs) → Type[spdm.data.Actor.Actor]
根据 inputs 和前序 time slice 更显当前time slice

time_slice: spdm.data.TimeSeries.TimeSeriesAoS[spdm.data.TimeSeries.TimeSlice]
时间片序列，保存 Actor 历史状态。@note: TimeSeriesAoS 长度为 n(=3) 循环队列。当压入序列的 TimeSlice 数量超出 n 时，会调用 TimeSeriesAoS.__full__(first_slice)

class IDS(*args, **kwargs)
Bases: `fytok.modules.Utilities.Module`

Base class of IDS

ids_properties: fytok.modules.Utilities.IDSProperties
Interface Data Structure properties. This element identifies the node above as an IDS

class RZTuple(*args, **kwargs)
Bases: fytok.modules.Utilities.RZTuple, spdm.data.sp_property.SpTree

r: Any

z: Any

class PointRZ(*args, **kwargs)
Bases: fytok.modules.Utilities.PointRZ, spdm.data.sp_property.SpTree

r: float

z: float

class CurveRZ(*args, **kwargs)
Bases: fytok.modules.Utilities.CurveRZ, spdm.data.sp_property.SpTree

r: numpy.ndarray

z: numpy.ndarray

class VacuumToroidalField(*args, **kwargs)
Bases: fytok.modules.Utilities.VacuumToroidalField, spdm.data.sp_property.SpTree

r0: float

b0: float

class CoreRadialGrid(*args, **kwargs)
Bases: fytok.modules.Utilities.CoreRadialGrid, spdm.data.sp_property.SpTree

psi_axis: float

psi_boundary: float

psi_norm: numpy.ndarray

rho_tor_boundary: float

rho_tor_norm: numpy.ndarray

class DetectorAperture
Bases: object

class PlasmaCompositionIonState(*args, **kwargs)
Bases: fytok.modules.Utilities.PlasmaCompositionIonState, spdm.data.sp_property.SpTree

label: str

z_min: float

z_max: float

electron_configuration: str

vibrational_level: float

vibrational_mode: str

class PlasmaCompositionSpecies(*args, **kwargs)
Bases: fytok.modules.Utilities.PlasmaCompositionSpecies, spdm.data.sp_property.SpTree

label: str

a: float

z_n: float

class PlasmaCompositionNeutralElement(*args, **kwargs)
  Bases: `spdm.data.sp_property.SpTree`

  a: float

  z_n: float

  atoms_n: int

class PlasmaCompositionIons(*args, **kwargs)
  Bases: `fytok.modules.Utilities.PlasmaCompositionIons`, `spdm.data.sp_property.SpTree`

  label: str

  element: spdm.data.AoS.AoS[fytok.modules.Utilities.PlasmaCompositionNeutralElement]

  z_ion: float

  state: fytok.modules.Utilities.PlasmaCompositionIonState

class PlasmaCompositionNeutralState
  Bases: `object`

  label: str

  electron_configuration: str

  vibrational_level: float

  vibrational_mode: str

  neutral_type: str

class PlasmaCompositionNeutral
  Bases: `object`

  label: str

  element: spdm.data.AoS.AoS[fytok.modules.Utilities.PlasmaCompositionNeutralElement]

  state: fytok.modules.Utilities.PlasmaCompositionNeutralState

class DistributionSpecies(*args, **kwargs)
  Bases: `spdm.data.sp_property.SpTree`

  type: str

  ion: fytok.modules.Utilities.PlasmaCompositionIons

  neutral: fytok.modules.Utilities.PlasmaCompositionNeutral

class Summary(*args, **kwargs)
  Bases: `fytok.ontology.imas_lastest.summary._T_summary`

  boundary: _T_summary_boundary
    Description of the plasma boundary

  code: Code
    对于 Module 的一般性说明。@note code 在 __init__ 时由初始化参数定义，同时会根据 code.name 查找相应的 plugin 。

  configuration: _T_summary_static_str_0d
    Device configuration (the content may be device-specific)

  disruption: _T_summary_disruption
    Disruption characteristics, if the pulse is terminated by a disruption

elms: _T_summary_elms
Edge Localized Modes related quantities

fusion: _T_summary_fusion
Fusion reactions

gas_injection_accumulated: _T_summary_gas_injection_accumulated
Accumulated injected gas since the plasma breakdown in equivalent electrons

gas_injection_prefill: _T_summary_gas_injection_prefill
Accumulated injected gas during the prefill in equivalent electrons

gas_injection_rates: _T_summary_gas_injection
Gas injection rates in equivalent electrons.s^-1

global_quantities: _T_summary_global_quantities
Various global quantities derived from the profiles

heating_current_drive: _T_summary_h_cd
Heating and current drive parameters

ids_properties: IDSProperties
Interface Data Structure properties. This element identifies the node above as an IDS

kicks: _T_summary_kicks
Vertical kicks of the plasma position

limiter: _T_summary_limiter
Limiter characteristics

line_average: _T_summary_average_quantities
Line average plasma parameters

local: _T_summary_local
Plasma parameter values at different locations

magnetic_shear_flag: _T_summary_static_int_0d
0 for shearless stellarators (W7-A, W7-AS, W7-X); 1, otherwise. See [Stroth U. et al 1996 Nucl. Fusion 36 1063]

Type Magnetic field shear indicator for stellarators

midplane: _E_midplane_identifier
Choice of midplane definition (use the lowest index number if more than one value is relevant)

pedestal_fits: _T_summary_pedestal_fits
Quantities derived from specific fits of pedestal profiles, typically used in the Pedestal Database.

pellets: _T_summary_pellets
Pellet related quantities

plasma_duration: _T_summary_constant_flt_0d
Duration of existence of a confined plasma during the pulse

rmps: _T_summary_rmp
Resonant magnetic perturbations related quantities

runaways: _T_summary_runaways
Runaway electrons

scrape_off_layer: _T_summary_sol
Scrape-Off-Layer (SOL) characteristics

stationary_phase_flag: _T_summary_dynamic_int_1d_root
This flag is set to one if the pulse is in a stationary phase from the point of the of the energy content (if the time derivative of the energy dW/dt can be neglected when calculating tau_E as W/(P_abs-dW/dt).)

tag: _T_entry_tag
Tag qualifying this data entry (or a list of data entries)

time_breakdown: _T_summary_constant_flt_0d
Time of the plasma breakdown

time_slice: TimeSeriesAoS[TimeSlice]
时间片序列，保存 Actor 历史状态。@note: TimeSeriesAoS 长度为 n(=3) 循环队列。当压入序列的 TimeSlice 数量超出 n 时，会调用 TimeSeriesAoS.__full__(first_slice)

time_width: Expression
In case the time-dependent quantities of this IDS are averaged over a time interval, this node is the width of this time interval (empty otherwise). By convention, the time interval starts at time-time_width and ends at time.

volume_average: _T_summary_average_quantities
Volume average plasma parameters

wall: _T_summary_wall
Wall characteristics

class TransportSolverNumericsEquationPrimary(*args, **kwargs)
Bases: `fytok.modules.TransportSolverNumerics.TransportSolverNumericsEquationPrimary`, `spdm.data.sp_property.SpTree`

identifier: str
Identifier of the primary quantity of the transport equation. The description node contains the path to the quantity in the physics IDS (example: core_profiles/profiles_1d/ion/D/density)

profile: spdm.data.Expression.Variable | numpy.ndarray
Profile of the primary quantity

flux: spdm.data.Expression.Variable | numpy.ndarray
Flux of the primary quantity

d_dr: spdm.data.Expression.Expression | numpy.ndarray
Radial derivative with respect to the primary coordinate

dflux_dr: spdm.data.Expression.Expression | numpy.ndarray
Radial derivative of Flux of the primary quantity

d2_dr2: spdm.data.Expression.Expression | numpy.ndarray
Second order radial derivative with respect to the primary coordinate

d_dt: spdm.data.Expression.Expression | numpy.ndarray
Time derivative

d_dt_cphi: spdm.data.Expression.Expression | numpy.ndarray
Derivative with respect to time, at constant toroidal flux (for current diffusion equation)

d_dt_cr: spdm.data.Expression.Expression | numpy.ndarray
Derivative with respect to time, at constant primary coordinate coordinate (for current diffusion equation)

class TransportSolverNumericsEquation(*args, **kwargs)
Bases: fytok.modules.TransportSolverNumerics.TransportSolverNumericsEquation, spdm.data.sp_property.SpTree

primary_quantity: TransportSolverNumericsEquationPrimary
Profile and derivatives of the primary quantity of the transport equation

boundary_condition: AoS[EquationBC]

coefficient: AoS
Set of numerical coefficients involved in the transport equation

convergence: PropertyTree
Convergence details

class TransportSolverNumericsTimeSlice(*args, **kwargs)
Bases: spdm.data.TimeSeries.TimeSlice

Numerics related to 1D radial solver for a given time slice

Equation
alias of fytok.modules.TransportSolverNumerics.TransportSolverNumericsEquation

grid: fytok.modules.Utilities.CoreRadialGrid
Radial grid

equation: spdm.data.AoS.AoS[fytok.modules.TransportSolverNumerics.TransportSolverNumericsEquation]
Set of transport equations

control_parameters: spdm.data.sp_property.PropertyTree
Solver-specific input or output quantities

drho_tor_dt: spdm.data.Expression.Expression
Partial derivative of the toroidal flux coordinate profile with respect to time

d_dvolume_drho_tor_dt: spdm.data.Expression.Expression
Partial derivative with respect to time of the derivative of the volume with respect to the toroidal flux coordinate

time: float

class TransportSolverNumericsBC(*args, **kwargs)
Bases: fytok.modules.TransportSolverNumerics.TransportSolverNumericsBC, spdm.data.sp_property.SpTree

rho_tor_norm: float
Position, in normalised toroidal flux, at which the boundary condition is imposed. Outside this position, the value of the data are considered to be prescribed.

identifier: Identifier
ip; 3: loop voltage; 4: undefined; 5: generic boundary condition y expressed as a1y' +a2y=a3. 6: equation not solved;

Type Identifier of the boundary condition type. ID = 1

Type poloidal flux; 2

current: float
Boundary condition for the current diffusion equation.

electrons: ParticleBC
Quantities related to the electrons

ion: AoS[ParticleBC]
Quantities related to the different ion species

energy_ion_total: ParticleBC
Boundary condition for the ion total (sum over ion species) energy equation (temperature if ID = 1)

momentum_tor: ParticleBC
Boundary condition for the total plasma toroidal momentum equation (summed over ion species and electrons) (momentum if ID = 1)

class TransportSolverNumerics(*args, **kwargs)
Bases: `fytok.modules.Utilities.IDS`

Solve transport equations $rho=sqrt{ Phi/pi B_{0}}$

ids_properties: fytok.modules.Utilities.IDSProperties
Interface Data Structure properties. This element identifies the node above as an IDS

code: fytok.modules.Utilities.Code
对于 Module 的一般性说明。@note code 在 __init__ 时由初始化参数定义，同时会根据 code.name 查找相应的 plugin 。

solver: fytok.modules.Utilities.Identifier

primary_coordinate: spdm.data.Expression.Variable
$rho_{tor}=sqrt{ Phi/pi B_{0}}$

class TransEquatuion(*args, **kwargs)
Bases: `fytok.modules.TransportSolverNumerics.TransportSolverNumerics.TransEquatuion`, `spdm.data.sp_property.SpTree`

identifier: str

boundary_condition: spdm.data.HTree.List[int]

profile: spdm.data.Expression.Variable

flux: spdm.data.Expression.Variable

equations: spdm.data.AoS.AoS[fytok.modules.TransportSolverNumerics.TransEquatuion]

TimeSlice
alias of `fytok.modules.TransportSolverNumerics.TransportSolverNumericsTimeSlice`

time_slice: spdm.data.TimeSeries.TimeSeriesAoS[fytok.modules.TransportSolverNumerics.TransportSolverNumericsTimeSlice]
时间片序列，保存 Actor 历史状态。@note: TimeSeriesAoS 长度为 n(=3) 循环队列。当压入序列的 TimeSlice 数量超出 n 时，会调用 TimeSeriesAoS.__full__(first_slice)

preprocess(*args, boundary_value=None, control_parameters=None, **kwargs)

refresh(*args, equilibrium: Optional[fytok.modules.Equilibrium.Equilibrium] = None, core_transport: Optional[fytok.modules.CoreTransport.CoreTransport] = None, core_sources: Optional[fytok.modules.CoreSources.CoreSources] = None, **kwargs)
更新当前 Actor 的状态。若 time 为 None 或者与当前时间一致，则更新当前状态树，并执行 self.iteration+=1 否则，向 time_slice 队列中压入新的时间片。

class Wall(*args, **kwargs)
Bases: `fytok.ontology.imas_lastest.wall._T_wall`

Description of the torus wall and its interaction with the plasma

Description2D
alias of `fytok.ontology.imas_lastest.wall._T_wall_2d`

class Waves(*args, **kwargs)
Bases: `fytok.ontology.imas_lastest.waves._T_waves`

CHAPTER

TWO

SPDM API 参考

2.1 spdm.data

class Actor(*args, **kwargs)

Bases: `spdm.data.Actor.Actor`, `spdm.data.sp_property.SpTree`

time_slice: spdm.data.TimeSeries.TimeSeriesAoS[spdm.data.TimeSeries.TimeSlice]

时间片序列，保存 Actor 历史状态。@note: TimeSeriesAoS 长度为 n(=3) 循环队列。当压入序列的 TimeSlice 数量超出 n 时，会调用 TimeSeriesAoS.__full__(first_slice)

class QueryResult(query: str | int | slice | dict | list | spdm.data.Path.OpTags | None, *args, **kwargs)

Bases: `spdm.data.HTree.HTree`

Handle the result of query

children() → Generator[Union[spdm.data.AoS._T, spdm.data.HTree.HTree], None, None]

遍历 children

class AoS(*args, identifier: Optional[str] = None, **kwargs)

Bases: `spdm.data.HTree.List[spdm.data.AoS._T]`

Array of structure

FIXME: 需要优化!!

- 数据结构应为 named list or ordered dict
- 可以自动转换 list 类型 cache 和 entry

dump(entry: spdm.data.Entry.Entry, **kwargs) → None

将数据写入 entry

class InsertOneResult(inserted_id, success)

Bases: `tuple`

inserted_id

Alias for field number 0

success

Alias for field number 1

class InsertManyResult(inserted_ids, success)

Bases: `tuple`

inserted_ids

Alias for field number 0

success
Alias for field number 1

class UpdateResult(inserted_id, success)
Bases: `tuple`

inserted_id
Alias for field number 0

success
Alias for field number 1

class DeleteResult(deleted_id, success)
Bases: `tuple`

deleted_id
Alias for field number 0

success
Alias for field number 1

class Collection(*args, **kwargs)
Bases: `spdm.data.Document.Document`

Collection of documents

property mapper

guess_id(predicate, *args, fragment: Optional[int] = None, **kwargs) → int

property next_id

create_one(*args, **kwargs)

create_many(docs: List[Any], *args, **kwargs)

create_doc(docs, *args, **kwargs)

insert_one(doc, *args, **kwargs) → spdm.data.Collection.InsertOneResult

insert_many(docs: List[Any], *args, **kwargs) → spdm.data.Collection.InsertManyResult

insert(docs, *args, **kwargs)

find_one(*args, **kwargs) → spdm.data.Entry.Entry

find_many(*args, **kwargs) → List[spdm.data.Entry.Entry]

find(predicate, projection=None, only_one=False, **kwargs) → Union[spdm.data.Entry.Entry, List[spdm.data.Entry.Entry]]

replace_one(predicate, replacement, *args, **kwargs) → spdm.data.Collection.UpdateResult

update_one(predicate, update, *args, **kwargs) → spdm.data.Collection.UpdateResult

update_many(predicate, updates: list, *args, **kwargs) → spdm.data.Collection.UpdateResult

delete_one(predicate, *args, **kwargs) → spdm.data.Collection.DeleteResult

delete_many(predicate, *args, **kwargs) → spdm.data.Collection.DeleteResult

count(predicate=None, *args, **kwargs) → int

create_indexes(indexes: List[str], session=None, **kwargs)

create_index(keys: List[str], session=None, **kwargs)

ensure_index(key_or_list, cache_for=300, **kwargs)

drop_indexes(session=None, **kwargs)

drop_index(index_or_name, session=None, **kwargs)

reindex(session=None, **kwargs)

list_indexes(session=None)

open_collection(uri: Union[str, spdm.utils.uri_utils.URITuple], *args, schema=None, **kwargs) → spdm.data.Collection.Collection

open_db(uri: Union[str, spdm.utils.uri_utils.URITuple], *args, schema=None, **kwargs) → spdm.data.Collection.Collection

class Directory(*args, mask=511, createparents=False, **kwargs)
Bases: `spdm.data.Document.Document`

Default entry for Directory

property path: pathlib.Path

property cwd: pathlib.Path

cd(path) → spdm.data.Directory.Directory

class LocalFileDB(*args, **kwargs)
Bases: `spdm.data.Collection.Collection`

property glob: str

guess_id(d, auto_inc=True)

guess_filepath(**kwargs) → pathlib.Path

open_document(fid, mode=None) → spdm.data.Entry.Entry

insert_one(predicate, *args, **kwargs) → spdm.data.Collection.InsertOneResult

find_one(predicate, projection=None, **kwargs) → spdm.data.Entry.Entry

update_one(predicate, update, *args, **kwargs)

delete_one(predicate, *args, **kwargs)

count(predicate=None, *args, **kwargs) → int

class CollectionLocalFile(*args, **kwargs)
Bases: `spdm.data.Collection.Collection`

Collection of local files.

property next_id

guess_path(*args, fid=None, **kwargs)

find_one(*args, projection=None, **kwargs)

insert_one(*args, projection=None, **kwargs)

update_one(predicate, update, *args, **kwargs)

delete_one(predicate, *args, **kwargs)

class Document(uri, *args, mode: Any = Mode.read, **kwargs)
Bases: `spdm.utils.plugin.Pluggable`

Connection like object

class Mode(value)

Bases: enum.Flag

An enumeration.

read = 1

write = 2

create = 4

append = 7

temporary = 8

MOD_MAP = {<Mode.read: 1>: 'r', <Mode.write|read: 3>: 'rw', <Mode.write: 2>: 'x', <Mode.create|write: 6>: 'w', <Mode.append: 7>: 'a'}

INV_MOD_MAP = {'a': Mode.append, 'r': Mode.read, 'rw': Mode.None, 'w': Mode.None, 'x': Mode.write}

class Status(value)

Bases: enum.Flag

An enumeration.

opened = 1

closed = 2

property url: spdm.utils.uri_utils.URITuple

property path: Any

property mode: Mode

property is_readable: bool

property is_writable: bool

property is_creatable: bool

property is_temporary: bool

property is_open: bool

open() → spdm.data.Document.Document

close() → None

read(lazy=False) → spdm.data.Entry.Entry

write(data=None, lazy=False, **kwargs) → spdm.data.Entry.Entry

property entry: spdm.data.Entry.Entry

class Edge(source=None, target=None, source_type_hint=None, target_type_hint=None, graph=None, **kwargs)

Bases: object

Edge defines a connection between two ‘Port ‘s

Attribute

- source : the start of edge which must be OUTPUT Port
- target : the start of edge which must be INPUT Port
- dtype : defines what ‘Port ‘s it can be connected, (default: string)

- label : short string
- description : long string

class Endpoint(node, type_hint=None)
Bases: `object`

update(node=None, type_hint=None) → spdm.data.Edge.Edge.Endpoint

unlink()

property is_changed: bool

property metadata: spdm.data.sp_property.PropertyTree

property source: Endpoint

property target: Endpoint

property is_linked

split(*args, **kwargs)
using Slot Node split edge into chain, add In(Out)Slot not to graph

return list of splitted edges

class Ports(holder)
Bases: `Dict[str, spdm.data.Edge.Edge]`

abstract link(id, node, type_hint=None) → spdm.data.Edge.Edge

fetch() → Dict[int | str, Any]

refresh()

get_source(key, default_value=tags.not_found)

get_target(key, default_value=tags.not_found)

class InPorts(holder)
Bases: `spdm.data.Edge.Ports`

link(id, source, type_hint=None)

update([E], **F) → None. Update D from dict/iterable E and F.
If E is present and has a .keys() method, then does: for k in E: D[k] = E[k] If E is present and lacks a .keys() method, then does: for k, v in E: D[k] = v In either case, this is followed by: for k in F: D[k] = F[k]

class OutPorts(holder)
Bases: `spdm.data.Edge.Ports`

link(id, target, type_hint=None) → spdm.data.Edge.Edge

update([E], **F) → None. Update D from dict/iterable E and F.
If E is present and has a .keys() method, then does: for k in E: D[k] = E[k] If E is present and lacks a .keys() method, then does: for k, v in E: D[k] = v In either case, this is followed by: for k in F: D[k] = F[k]

set(key, value)

class Entry(data: Optional[Any] = None, path: Optional[Union[spdm.data.Path.Path, str, int, slice, dict, list, spdm.data.Path.OpTags]] = None, *args, scheme=None, **kwargs)
Bases: `spdm.utils.plugin.Pluggable`

reset(value=None, path=None) → spdm.data.Entry.Entry

property is_writable: bool

property path: spdm.data.Path.Path

property is_leaf: bool

property is_list: bool

property is_dict: bool

property is_root: bool

property is_generator: bool

get(query=None, default_value: Any = Ellipsis, **kwargs) → Any

put(pth, value, *args, **kwrags) → spdm.data.Entry.Entry

dump() → Any

equal(other) → bool

property count: int

property exists: bool

check_type(tp: Type) → bool

property root: spdm.data.Entry.Entry

property parent: spdm.data.Entry.Entry

child(path=None, *args, **kwargs) → spdm.data.Entry.Entry

next(inc: int = 1) → spdm.data.Entry.Entry

insert(value, **kwargs) → spdm.data.Entry.Entry

update(value, **kwargs) → spdm.data.Entry.Entry

remove(**kwargs) → int

fetch(op=None, *args, **kwargs) → Any
Query the Entry. Same function as find, but put result into a contianer. Could be overridden by subclasses.

keys() → Generator[str, None, None]

for_each(*args, **kwargs) → Generator[Tuple[int, Any], None, None]
Return a generator of the results.

find_next(*, __fun__=<function Entry.find_next>, **kwargs)

find(*args, **kwargs) → spdm.data.Entry.Entry

class ChainEntry(*args, **kwargs)
Bases: `spdm.data.Entry.Entry`

property is_writable: bool

fetch(*args, default_value=tags.not_found, **kwargs)
Query the Entry. Same function as find, but put result into a contianer. Could be overridden by subclasses.

for_each(*args, **kwargs) → Generator[Tuple[int, Any], None, None]
Return a generator of the results.

find(*args, **kwargs)

property exists: bool

open_entry(entry, **kwargs) → spdm.data.Entry.Entry

asentry(obj, *args, **kwargs) → spdm.data.Entry.Entry

as_dataclass(dclass, obj, default_value=None)

deep_reduce(first=None, *others, level=- 1)

convert_fromentry(cls, obj, *args, **kwargs)

class EntryProxy(*args, **kwargs)
Bases: `spdm.data.Entry.Entry`

classmethod load(url: Optional[str] = None, local_schema: Optional[str] = None, global_schema: Optional[str] = None, mapping_files=None, **kwargs)
检索并导入 mapping files

child(*args, **kwargs) → spdm.data.Entry.Entry

insert(value, **kwargs) → spdm.data.Entry.Entry

update(value, **kwargs) → spdm.data.Entry.Entry

remove(**kwargs) → int

fetch(*args, default_value=tags.not_found, **kwargs) → Any
Query the Entry. Same function as find, but put result into a contianer. Could be overridden by subclasses.

for_each(*args, **kwargs) → Generator[Tuple[int, Any], None, None]
Return a generator of the results.

find(*args, **kwargs)

class DomainBase(*args, **kwargs)
Bases: `object`

函数定义域

property is_simple: bool

property dims: Tuple[numpy.ndarray[Any, numpy.dtype[numpy.floating | numpy.complexfloating]]]
函数的网格，即定义域的网格

property ndims: int

property shape: Tuple[int]

property points: Tuple[numpy.ndarray[Any, numpy.dtype[numpy.floating | numpy.complexfloating]]]

property bbox: Tuple[List[float], List[float]]
函数的定义域

property periods

mask(*args) → bool | numpy.ndarray[Any, numpy.dtype[numpy.bool_]]

check(*x) → bool | numpy.ndarray[Any, numpy.dtype[numpy.bool_]]
当坐标在定义域内时返回 True，否则返回 False

eval(func, *xargs, **kwargs)
根据 __domain__ 函数的返回值，对输入坐标进行筛选

guess_coords(holder, prefix='　coordinate', **kwargs)

class Expression(expr: Callable[[...], bool | int | float | complex | numpy.float64 | numpy.complex64 | numpy.complex128 | numpy.integer | numpy.floating | numpy.bool_ | None | numpy.ndarray[Any, numpy.dtype[numpy.floating | numpy.complexfloating]]], *children, domain=tags.not_found, **kwargs)

Bases: `spdm.data.HTree.HTreeNode`

表达式是由多个操作数和运算符按照约定的规则构成的一个序列。其中运算符表示对操作数进行何种操作，而操作数可以是变量、常量、数组或者表达式。表达式可以理解为树状结构，每个节点都是一个操作数或运算符，每个节点都可以有多个子节点。表达式的值可以通过对树状结构进行遍历计算得到。没有子节点的节点称为叶子节点，叶子节点可以是常量、数组，也可以是变量和函数。

变量是一种特殊的函数，它的值由上下文决定。

例如：

```
>>> import spdm
>>> x = spdm.data.Expression(op=np.sin)
>>> y = spdm.data.Expression(op=np.cos)
>>> z = x + y
>>> z
<Expression   op="add" />
>>> z(0.0)
3.0
```

Domain

alias of `spdm.data.Expression.DomainBase`

property domain: Domain

返回表达式的定义域

property has_children: bool

判断是否有子节点

property empty: bool

property callable

property name: str

property dtype

integral(**kwargs) → float

derivative(d, *args, **kwargs) → Type[spdm.data.Expression.Derivative]

pd(*d) → spdm.data.Expression.Expression

property d: spdm.data.Expression.Expression

1st derivative 一阶导数

property d2: spdm.data.Expression.Expression

2nd derivative 二阶导数

property I: spdm.data.Expression.Expression

antiderivative 原函数

property dln: spdm.data.Expression.Expression

logarithmic derivative 对数求导

find_roots(*args, **kwargs) → Generator[float, None, None]

class Variable(idx: int | str, name: Optional[str] = None, **kwargs)
Bases: `spdm.data.Expression.Expression`

变量是一种特殊的函数，它的值由上下文决定。例如：»> import spdm »> x = spdm.data.Variable(0,” x”) »> y = spdm.data.Variable(1,” y”) »> z = x + y »> z <Expression op=” add” /> »> z(0.0, 1.0) 1.0

property index

class Scalar(value, *args, **kwargs)
Bases: `spdm.data.Expression.Expression`

derivative(*args, **kwargs)

class ConstantZero(*args, **kwargs)
Bases: `spdm.data.Expression.Scalar`

class ConstantOne(*args, **kwargs)
Bases: `spdm.data.Expression.Scalar`

class Derivative(order, expr, **kwargs)
Bases: `spdm.data.Expression.Expression`

算符: 用于表示一个运算符，可以是函数，也可以是类的成员函数受 np.ufunc 启发而来。可以通过 ExprOp(op, method=method) 的方式构建一个 ExprOp 对象。

property order: int | None

class LogDerivative(order, expr, **kwargs)
Bases: `spdm.data.Expression.Derivative`

class PartialDerivative(order, expr, **kwargs)
Bases: `spdm.data.Expression.Derivative`

class Antiderivative(order, expr, **kwargs)
Bases: `spdm.data.Expression.Derivative`

class Piecewise(func: List[spdm.data.Expression.Expression | float | int], cond: List[Callable], **kwargs)
Bases: `spdm.data.Expression.Expression`

PiecewiseFunction A piecewise function. 一维或多维，分段函数

guess_mesh(holder, prefix=’ mesh’ , **kwargs)

class Field(*xy, **kwargs)
Bases: `spdm.data.Expression.Expression`

Field 是 Function 在流形（manifold/Mesh）上的推广，用于描述流形上的标量场，矢量场，张量场等。

Field 所在的流形记为 mesh ，可以是任意维度的，可以是任意形状的，可以是任意拓扑的，可以是任意坐标系的。

Mesh 网格描述流形的几何结构，比如网格的拓扑结构，网格的几何结构，网格的坐标系等。

Field 与 Function的区别:

- Function 的 mesh 是一维数组表示dimensions/axis
- Field 的 mesh 是 Mesh，可以表示复杂流形上的场等。

Domain
alias of `spdm.mesh.Mesh.Mesh`

property mesh: spdm.mesh.Mesh.Mesh

property domain: spdm.mesh.Mesh.Mesh
返回表达式的定义域

ppoly()

grad(n=1) → spdm.data.Field.Field

derivative(d, *args, **kwargs) → spdm.data.Field.Field

class File(url: str | pathlib.Path | spdm.utils.uri_utils.URITuple, *args, format=None, default_format=None, **kwargs)
Bases: `spdm.data.Document.Document`

File like object

property mode_str: str

property is_writable: bool

read(lazy=False) → spdm.data.Entry.Entry

write(data=None, *args, lazy=False, **kwargs)

class FileEntry(*args, file, **kwargs)
Bases: `spdm.data.Entry.Entry`

flush()

class Function(*xy, domain=None, **kwargs)
Bases: `spdm.data.Expression.Expression`

A function is a mapping between two sets, the _domain_ and the _value_. The _value_ is the set of all possible outputs of the function. The _domain_ is the set of all possible inputs to the function.

函数定义域为多维空间时，网格采用rectlinear mesh，即每个维度网格表示为一个数组 _dims_ 。

property x_label: str

property dims: Tuple[numpy.ndarray[Any, numpy.dtype[numpy.floating | numpy.complexfloating]]]
函数的网格，即定义域的网格

property ndim: int
函数的维度，函数所能接收参数的个数。

property rank: int
函数的秩，rank=1 标量函数，rank=3 矢量函数 None 待定

derivative(*d, **kwargs) → spdm.data.Function.Function

integral(*args, **kwargs) → float

validate(value=None, strict=False) → bool
检查函数的定义域和值是否匹配

function_like(y: bool | int | float | complex | numpy.float64 | numpy.complex64 | numpy.complex128 | numpy.integer | numpy.floating | numpy.bool_ | None | numpy.ndarray[Any, numpy.dtype[numpy.floating | numpy.complexfloating]], *args: bool | int | float | complex | numpy.float64 | numpy.complex64 | numpy.complex128 | numpy.integer | numpy.floating | numpy.bool_ | None | numpy.ndarray[Any, numpy.dtype[numpy.floating | numpy.complexfloating]], **kwargs) → spdm.data.Function.Function

class Functor(func: Optional[Callable], /, method: Optional[str] = None, label: Optional[str] = None, **kwargs)
Bases: `object`

算符: 用于表示一个运算符，可以是函数，也可以是类的成员函数受 np.ufunc 启发而来。可以通过 ExprOp(op, method=method) 的方式构建一个 ExprOp 对象。

class ConstantsFunc(value: bool | int | float | complex | numpy.float64 | numpy.complex64 | numpy.complex128 | numpy.integer | numpy.floating | numpy.bool_ | None | numpy.ndarray[Any, numpy.dtype[numpy.floating | numpy.complexfloating]], **kwargs)
Bases: `spdm.data.Functor.Functor`

class SetpFun(y: bool | int | float | complex | numpy.float64 | numpy.complex64 | numpy.complex128 | numpy.integer | numpy.floating | numpy.bool_ | None | numpy.ndarray[Any, numpy.dtype[numpy.floating | numpy.complexfloating]], *xargs, y0: bool | int | float | complex | numpy.float64 | numpy.complex64 | numpy.complex128 | numpy.integer | numpy.floating | numpy.bool_ | None | numpy.ndarray[Any, numpy.dtype[numpy.floating | numpy.complexfloating]] = 0.0, **kwargs)
Bases: `spdm.data.Functor.Functor`

class DiracDeltaFun(y: bool | int | float | complex | numpy.float64 | numpy.complex64 | numpy.complex128 | numpy.integer | numpy.floating | numpy.bool_ | None | numpy.ndarray[Any, numpy.dtype[numpy.floating | numpy.complexfloating]], *xargs, y0: bool | int | float | complex | numpy.float64 | numpy.complex64 | numpy.complex128 | numpy.integer | numpy.floating | numpy.bool_ | None | numpy.ndarray[Any, numpy.dtype[numpy.floating | numpy.complexfloating]] = 0.0, **kwargs)
Bases: `spdm.data.Functor.Functor`

derivative(n=1) → spdm.data.Functor.SetpFun

as_functor(expr, *args, **kwargs) → spdm.data.Functor.Functor | None

class Graph(value=None, *args, **kwargs)
Bases: `spdm.data.HTree.Dict[spdm.data.Graph._T]`

Represents ‘’ Graph’’’ . * defines namespace for the ‘’ Node’’’ s * Graph is a Node

TODO (salmon 2019.7.25): add subgraph

property edges: List[spdm.data.Edge.Edge]

link(source: str | int | slice | dict | list | spdm.data.Path.OpTags | None, target: str | int | slice | dict | list | spdm.data.Path.OpTags | None, *args, **kwargs) → spdm.data.Edge.Edge

class HTreeNode(*args, **kwargs)
Bases: `object`

dump(entry: Optional[spdm.data.Entry.Entry] = None, **kwargs) → None
将数据写入 _entry

property path: List[Union[int, str]]

property root: spdm.data.HTree.HTree | None

class HTree(*args, **kwargs)
Bases: `spdm.data.HTree.HTreeNode`

Hierarchical Tree:

一种层次化的数据结构，它具有以下特性：- 树节点也可以是列表 list，也可以是字典 dict - 叶节点可以是标量或数组 array_type，或其他 type_hint 类型 - 节点可以有缓存（cache) - 节点可以有父节点（_parent) - 节点可以有元数据（metadata), 包含：唯一标识（id), 名称（name), 单位（units), 描述（description), 标签（tags), 注释（comment) - 任意节点都可以通过路径访问 - get 返回的类型由 type_hint 决定，默认为 Node

insert(*args, **kwargs)

update(*args, **kwargs)

remove(*args, **kwargs)

cache_get(pth, default_value=tags.not_found)

get(path: spdm.data.Path.Path | str | int | slice | dict | list | spdm.data.Path.OpTags | None, default_value: Any = tags.not_found, *args, force=False, **kwargs) → spdm.data.HTree._T

children() → Generator[spdm.data.HTree.HTree, None, None]

as_htree(obj, *args, **kwargs)

Node

alias of `spdm.data.HTree.HTree`

class Container(*args, **kwargs)

Bases: `spdm.data.HTree.HTree`, `Generic[spdm.data.HTree._T]`

带有type hint的容器，其成员类型为 _T，用于存储一组数据或对象，如列表，字典等

class Dict(*args, **kwargs)

Bases: `spdm.data.HTree.Container[spdm.data.HTree._T]`

items()

class List(*args, **kwargs)

Bases: `spdm.data.HTree.Container[spdm.data.HTree._T]`

property empty: bool

dump(_entry: spdm.data.Entry.Entry, **kwargs) → None

将数据写入 _entry

class OpTags(value)

Bases: `enum.Flag`

An enumeration.

root = 1

parent = 2

children = 4

ancestors = 8

descendants = 16

current = 32

next = 64

fetch = 128

update = 256

insert = 512

remove = 1024

call = 2048

exists = 4096

is_leaf = 8192

is_list = 16384

is_dict = 32768

check_type = 65536

search = 131072

dump = 262144

reduce = 524288

sort = 1048576

check = 2097152

count = 4194304

equal = 8388608

le = 16777216

ge = 33554432

less = 67108864

greater = 134217728

class Query(query: Optional[Union[dict, spdm.data.Path.OpTags]] = None, only_first=True, **kwargs)

Bases: `object`

check(target) → bool

find_next(target, start: int | None, **kwargs) → Tuple[Any, int | None]

as_query(query: Optional[Union[dict, spdm.data.Path.OpTags]] = None, **kwargs) → spdm.data.Path.Query | slice

exception PathError(path: List[str | int | slice | dict | list | spdm.data.Path.OpTags | None], message: Optional[str] = None)

Bases: `Exception`

class Path(path=[], **kwargs)

Bases: `list`

Path用于描述数据的路径, 在 HTree (Hierarchical Tree) 中定位Element, 其语法是 JSONPath 和 XPath的变体，并扩展谓词（predicate）语法/查询选择器。

HTree: Hierarchical Tree 半结构化树状数据，树节点具有 list或dict类型，叶节点为 list和dict 之外的primary数据类型，

包括 int，float,string 和 ndarray。

基本原则是用python 原生数据类型（例如，list, dict,set,tuple）等

DELIMITER= '/ ' or .

Python 算符 | 字符形式 | 描述

—- |—| —
N/A | $ | 根对象（TODO：Not Implemented ）
None | @ | 空选择符，当前对象。当以Path以None为最后一个item时，表示所指元素为leaf节点。
__truediv__，'__getattr___ ' | DELIMITER (/ or .) | 子元素选择符, DELIMITER 可选
__getitem__ | '[index|slice|selector] '| 数组元素选择符，index为整数,slice，或selector选择器（predicate谓词）

predicate 谓词, 过滤表达式，用于过滤数组元素.

set | [{a,b,1}] | 返回dict, named并集运算符，用于组合多个子元素选择器，并将element作为返回的key，{ 'a' :@[a], 'b' :@['b'], 1:@[1] }
list | ["a" ,b,1] | 返回list, 并集运算符，用于组合多个子元素选择器，[@[a], @['b'], @[1]]
slice | [start:end:step]，| 数组切片运算符, 当前元素为 ndarray 时返回数组切片 @[<slice>]，当前元素为 dict,list 以slice选取返回 list （generator），
slice(None) ' | '* | 通配符，匹配任意字段或数组元素，代表所有子节点（children）
| .. | 递归下降运算符 (Not Implemented)
dict {$eq:4, } | [?(expression)] | 谓词（predicate）或过滤表达式，用于过滤数组元素.
| ==、!=、<、<=、>、>= | 比较运算符

Examples

Path | Description
—- | —
a/b/c | 选择a节点的b节点的c节点
a/b/c/1 | 选择a节点的b节点的c节点的第二个元素
a/b/c[1:3] | 选择a节点的b节点的c节点的第二个和第三个元素
a/b/c[1:3:2] | 选择a节点的b节点的c节点的第二个和第三个元素
a/b/c[1:3:-1] | 选择a节点的b节点的c节点的第三个和第二个元素
a/b/c[d,e,f] |
'a/b/c[{d,e,f}] |
'a/b/c[{value:{$le:10}}]/value |
'a/b/c.$next/ |

delimiter = '/'

tags
alias of `spdm.data.Path.OpTags`

as_url() → str

property is_leaf: bool

property is_root: bool

property is_regular: bool

property is_generator: bool

property parent: spdm.data.Path.Path

property children: spdm.data.Path.Path

property slibings

property next: spdm.data.Path.Path

prepend(d) → spdm.data.Path.Path

append(d) → spdm.data.Path.Path
Append object to the end of the list.

extend(d: list) → spdm.data.Path.Path
Extend list by appending elements from the iterable.

collapse(idx=None) → spdm.data.Path.Path

- 从路径中删除非字符元素，例如 slice, dict, set, tuple，int。用于从 default_value 中提取数据
- 从路径中删除指定位置idx: 的元素

static reduce(path: list) → list

static normalize(p: Any, raw=False) → Any

PATH_PATTERN = re.compile('(?P<key>[^\\[\\]\\/\\,\\.]+)(\\[(?P<selector>[^\\[\\]]+)\\])?')

PATH_REGEX_DICT = re.compile('\\{(?P<selector>[^\\{\\}]+)\\}')

get(target: Any, default_value=tags.not_found)

insert(target: Any, *args, **kwargs) → Tuple[Any, spdm.data.Path.Path]
根据路径（self）向 target 添加元素。当路径指向位置为空时，创建（create）元素当路径指向位置为 list 时，追加（insert ）元素当路径指向位置为非 list 时，合并为 [old,new] 当路径指向位置为 dict, 添加值亦为 dict 时，根据 key 递归执行 insert

返回新添加元素的路径

对应 RESTful 中的 post，非幂等操作

remove(target: Any, *args, **kwargs) → Tuple[Any, int]
根据路径（self）删除 target 中的元素。

if quiet is False then raise KeyError if the path is not found

对应 RESTful 中的 delete，幂等操作

返回修改后的target和删除的元素的个数

update(target: Any, *args, **kwargs) → Any
根据路径（self）更新 target 中的元素。当路径指向位置为空时，创建（create）元素当路径指向位置为 dict, 添加值亦为 dict 时，根据 key 递归执行 update 当路径指向位置为空时，用新的值替代（replace）元素

对应 RESTful 中的 put，幂等操作

返回修改后的target

fetch(target: Any, op: Optional[Union[spdm.data.Path.OpTags, str]] = None, *args, default_value=tags.not_found, **kwargs) → Any
根据路径（self）查询元素。只读，不会修改 target

对应 RESTful 中的 read，幂等操作

find_next(*, __fun__=<function Path.find_next>, **kwargs)

has_slice() → bool

has_query() → bool

for_each(target, *args, **kwargs) → Generator[Tuple[int, Any], None, None]

keys(target, **kwargs) → Generator[str, None, None]

traversal(*, __fun__=<function Path.traversal>, **kwargs)

MAX_SLICE_STOP = 1024

update_tree(target: spdm.data.Path._T, *args, **kwargs) → spdm.data.Path._T

merge_tree(*args, **kwargs) → spdm.data.Path._T

as_path(path)

class Signal(*args, **kwargs)
Bases: `spdm.data.Signal.Signal`, `spdm.data.sp_property.SpTree`

data: numpy.ndarray

time: numpy.ndarray

class SignalND(*args, **kwargs)
Bases: `spdm.data.Signal.Signal`

class TimeSlice(*args, **kwargs)
Bases: `spdm.data.sp_property.SpTree`

time: float

property iteration: int

refresh(*args, **kwargs)

class TimeSeriesAoS(*args, **kwargs)
Bases: `spdm.data.HTree.List[spdm.data.TimeSeries._TSlice]`

A series of time slices .

用以管理随时间变化（time series）的一组状态（TimeSlice）。

current: 指向当前时间片，即为序列最后一个时间片吗。

TODO

1. 缓存时间片，避免重复创建，减少内存占用
2. 缓存应循环使用
3. cache 数据自动写入 entry 落盘

dump(entry: spdm.data.Entry.Entry, **kwargs) → None
将数据写入 entry

property time: float

property iteration: iteration

property dt: float

property current: spdm.data.TimeSeries._TSlice

property previous: spdm.data.TimeSeries._TSlice

property is_initialized: bool

initialize(*args, **kwargs)

refresh(*args, **kwargs) → Type[spdm.data.TimeSeries.TimeSeriesAoS]

advance(*args, **kwargs) → spdm.data.TimeSeries._TSlice

2.2 spdm.mesh

class Mesh(*args, **kwargs)

Bases: `spdm.data.Expression.DomainBase`, `spdm.utils.plugin.Pluggable`

Mesh 网格

@NOTE: In general, a mesh provides more flexibility in representing complex geometries and can adapt to the local features of the solution, while a grid is simpler to generate and can be more efficient for certain types of problems.

property axis_label: Tuple[str]

property name: str

property type: str

property units: Tuple[str, ...]

property geometry: spdm.geometry.GeoObject.GeoObject

Geometry of the Mesh 网格的几何形状

property ndim: int

property rank: int

property shape: Tuple[int, ...]

存储网格点数组的形状 TODO: support multiblock Mesh 结构化网格 shape 如 [n,m] n,m 为网格的长度dimension 非结构化网格 shape 如 [<number of vertices>]

parametric_coordinates(*xyz) → numpy.ndarray[Any, numpy.dtype[numpy.floating | numpy.complexfloating]]

parametric coordinates

网格点的 _参数坐标 _ Parametric coordinates, also known as computational coordinates or intrinsic coordinates, are a way to represent the position of a point within an element of a mesh. 一般记作 u,v,w in [0,1] ,其中 0 表示"起点"或"原点" origin，1 表示终点end mesh的参数坐标(u,v,w)，(…,0)和(…,1)表示边界

@return: 数组形状为 [geometry.rank, <shape of xyz …>] 的数组

coordinates(*uvw) → numpy.ndarray[Any, numpy.dtype[numpy.floating | numpy.complexfloating]]

网格点的 _空间坐标 _ @return: _数组 _ 形状为 [<shape of uvw …>,geometry.ndim]

uvw() → numpy.ndarray[Any, numpy.dtype[numpy.floating | numpy.complexfloating]]

property vertices: numpy.ndarray[Any, numpy.dtype[numpy.floating | numpy.complexfloating]]

coordinates of vertice of mesh [<shape…>, geometry.ndim]

property points: List[numpy.ndarray[Any, numpy.dtype[numpy.floating | numpy.complexfloating]]]

alias of vertices, change the shape to tuple

property xyz: List[numpy.ndarray[Any, numpy.dtype[numpy.floating | numpy.complexfloating]]]

property cells: Any

interpolator(y: bool | int | float | complex | numpy.float64 | numpy.complex64 | numpy.complex128 | numpy.integer | numpy.floating | numpy.bool_ | None | numpy.ndarray[Any, numpy.dtype[numpy.floating | numpy.complexfloating]], *args, **kwargs) → Callable[[...], bool | int | float | complex | numpy.float64 | numpy.complex64 | numpy.complex128 | numpy.integer | numpy.floating | numpy.bool_ | None | numpy.ndarray[Any, numpy.dtype[numpy.floating | numpy.complexfloating]]]

partial_derivative(order, y: bool | int | float | complex | numpy.float64 | numpy.complex64 | numpy.complex128 | numpy.integer | numpy.floating | numpy.bool_ | None | numpy.ndarray[Any, numpy.dtype[numpy.floating | numpy.complexfloating]], *args, **kwargs) → Callable[[...], bool | int | float | complex | numpy.float64 | numpy.complex64 | numpy.complex128 | numpy.integer | numpy.floating | numpy.bool_ | None | numpy.ndarray[Any, numpy.dtype[numpy.floating | numpy.complexfloating]]]

antiderivative(y: bool | int | float | complex | numpy.float64 | numpy.complex64 | numpy.complex128 | numpy.integer | numpy.floating | numpy.bool_ | None | numpy.ndarray[Any, numpy.dtype[numpy.floating | numpy.complexfloating]], *args, **kwargs) → Callable[[...], bool | int | float | complex | numpy.float64 | numpy.complex64 | numpy.complex128 | numpy.integer | numpy.floating | numpy.bool_ | None | numpy.ndarray[Any, numpy.dtype[numpy.floating | numpy.complexfloating]]]

integrate(y: bool | int | float | complex | numpy.float64 | numpy.complex64 | numpy.complex128 | numpy.integer | numpy.floating | numpy.bool_ | None | numpy.ndarray[Any, numpy.dtype[numpy.floating | numpy.complexfloating]], *args, **kwargs) → bool | int | float | complex | numpy.float64 | numpy.complex64 | numpy.complex128 | numpy.integer | numpy.floating | numpy.bool_ | None

eval(func, *args, **kwargs) → numpy.ndarray[Any, numpy.dtype[numpy.floating | numpy.complexfloating]]

根据 __domain__ 函数的返回值，对输入坐标进行筛选

class NullMesh(*args, **kwargs)

Bases: `spdm.mesh.Mesh.Mesh`

class RegularMesh(*args, **kwargs)

Bases: `spdm.mesh.Mesh.Mesh`

as_mesh(*args, **kwargs) → spdm.mesh.Mesh.Mesh

class CurvilinearMesh(*args, **kwargs)

Bases: `spdm.mesh.mesh_rectilinear.RectilinearMesh`

A curvilinear Mesh or structured Mesh is a Mesh with the same combinatorial structure as a regular Mesh, in which the cells are quadrilaterals or [general] cuboids, rather than rectangles or rectangular cuboids. –[https://en.wikipedia.org/wiki/Regular_Mesh]

TOLERANCE = 1e-05

axis(idx, axis=0)

property uv: numpy.ndarray[Any, numpy.dtype[numpy.floating | numpy.complexfloating]]

property points: List[numpy.ndarray[Any, numpy.dtype[numpy.floating | numpy.complexfloating]]]

网格点的 _空间坐标_

property volume_element: numpy.ndarray[Any, numpy.dtype[numpy.floating | numpy.complexfloating]]

property xyz

interpolator(value, **kwargs)
生成插值器 method: “linear”, “nearest”, “slinear”, “cubic”, “quintic” and “pchip”

property boundary

property geo_object

class RectangularMesh(*args: numpy.ndarray[Any, numpy.dtype[numpy.floating | numpy.complexfloating]], geometry=None, periods=None, dims=None, **kwargs)

Bases: `spdm.mesh.mesh_rectilinear.RectilinearMesh`

Rectangular Mesh, which is alias of RectilinearMesh 矩形网格

property dim1: numpy.ndarray[Any, numpy.dtype[numpy.floating | numpy.complexfloating]]

property dim2: numpy.ndarray[Any, numpy.dtype[numpy.floating | numpy.complexfloating]]

class RectilinearMesh(*args: numpy.ndarray[Any, numpy.dtype[numpy.floating | numpy.complexfloating]], geometry=None, periods=None, dims=None, **kwargs)

Bases: `spdm.mesh.mesh_structured.StructuredMesh`

A rectilinear Mesh is a tessellation by rectangles or rectangular cuboids (also known as rectangular parallelepipeds) that are not, in general, all congruent to each other. The cells may still be indexed by integers as above, but the mapping from indexes to vertex coordinates is less uniform than in a regular Mesh. An example of a rectilinear Mesh that is not regular appears on logarithmic scale graph paper. –[https://en.wikipedia.org/wiki/Regular_Mesh]

RectlinearMesh

可以视为由 n=rank 条称为axis的曲线 curve 平移张成的空间。

xyz= sum([axis[i](uvw[i]) for i in range(rank)])

property dim1: numpy.ndarray[Any, numpy.dtype[numpy.floating | numpy.complexfloating]]

property dim2: numpy.ndarray[Any, numpy.dtype[numpy.floating | numpy.complexfloating]]

property dims: List[numpy.ndarray[Any, numpy.dtype[numpy.floating | numpy.complexfloating]]]
函数的网格，即定义域的网格

property dimensions: List[numpy.ndarray[Any, numpy.dtype[numpy.floating | numpy.complexfloating]]]

property rank: int

property dx: numpy.ndarray[Any, numpy.dtype[numpy.floating | numpy.complexfloating]]

coordinates(*uvw) → numpy.ndarray[Any, numpy.dtype[numpy.floating | numpy.complexfloating]]
网格点的 _空间坐标_ @return: _数组_ 形状为 [geometry.dimension,<shape of uvw …>]

property vertices: numpy.ndarray[Any, numpy.dtype[numpy.floating | numpy.complexfloating]]
网格点的 _空间坐标_

property points: List[numpy.ndarray[Any, numpy.dtype[numpy.floating | numpy.complexfloating]]]
网格点的 _空间坐标 _

interpolator(value: numpy.ndarray[Any, numpy.dtype[numpy.floating | numpy.complexfloating]], **kwargs)
生成插值器 method: “linear”, “nearest”, “slinear”, “cubic”, “quintic” and “pchip”

class StructuredMesh(shape: Optional[Union[numpy._typing._array_like._SupportsArray[numpy.dtype[Any]], numpy._typing._nested_sequence._NestedSequence[numpy._typing._array_like._Supports bool, int, float, complex, str, bytes, numpy._typing._nested_sequence._NestedSequence[Union[bool, int, float, complex, str, bytes]]]], *args, cycles=None, **kwargs)
Bases: `spdm.mesh.Mesh.Mesh`

StructureMesh

结构化网格上的点可以表示为长度为n=rank的归一化ntuple，记作 uv，uv_r in [0,1]

property cycles: List[float]

property origin: numpy.ndarray[Any, numpy.dtype[numpy.floating | numpy.complexfloating]]

property dx: numpy.ndarray[Any, numpy.dtype[numpy.floating | numpy.complexfloating]]

coordinates(*uvw) → numpy.ndarray[Any, numpy.dtype[numpy.floating | numpy.complexfloating]]
网格点的 _空间坐标 _ @return: _数组 _ 形状为 [<shape of uvw …>,geometry.ndim]

parametric_coordinates(*xyz) → numpy.ndarray[Any, numpy.dtype[numpy.floating | numpy.complexfloating]]
parametric coordinates

网格点的 _参数坐标 _ Parametric coordinates, also known as computational coordinates or intrinsic coordinates, are a way to represent the position of a point within an element of a mesh. 一般记作 u,v,w in [0,1] ,其中 0 表示“起点”或“原点” origin，1 表示终点end mesh的参数坐标(u,v,w)，(…,0)和(…,1)表示边界

@return: 数组形状为 [geometry.rank, <shape of xyz …>] 的数组

interpolator(*args, **kwargs) → Callable
Interpolator of the Mesh 网格的插值器, 用于网格上的插值返回一个函数，该函数的输入是一个坐标，输出是一个值输入坐标若为标量，则返回标量值输入坐标若为数组，则返回数组

class UniformMesh(*args, **kwargs)
Bases: `spdm.mesh.mesh_structured.StructuredMesh`

property origin: Tuple[float]

property dx: Tuple[float]

vertices(*args) → numpy.ndarray[Any, numpy.dtype[numpy.floating | numpy.complexfloating]]
coordinates of vertice of mesh [<shape…>, geometry.ndim]

2.3 spdm.view

class View(*args, **kwargs)

Bases: `spdm.utils.plugin.Pluggable`

Abstract class for all views

backend = None

property signature: str

render(*args, **kwargs)

plot(*args, **kwargs)

viewer(backend=None)

Get a viewer instance

display(*args, backend=None, **kwargs)

Show an object

plot(*args, backend=None, **kwargs)

Show an object

class MatplotlibView(*args, **kwargs)

Bases: `spdm.view.View.View`

backend = 'matplotlib'

render(obj, *styles, view_point='rz', title=None, **kwargs) → Any

plot(obj, x_value: Optional[Union[spdm.data.Expression.Expression, numpy.ndarray]] = None, x_label: Optional[str] = None, x_axis: Optional[numpy.ndarray] = None, stop_if_fail=False, **kwargs) → Any

PYTHON MODULE INDEX

f

fytok.modules.AMNSData, 4
fytok.modules.CoreProfiles, 4
fytok.modules.CoreSources, 12
fytok.modules.CoreTransport, 15
fytok.modules.DatasetFAIR, 19
fytok.modules.ECLaunchers, 20
fytok.modules.Equilibrium, 20
fytok.modules.ICAntennas, 31
fytok.modules.Interferometer, 31
fytok.modules.LHAntennas, 31
fytok.modules.Magnetics, 31
fytok.modules.NBI, 31
fytok.modules.Pellets, 31
fytok.modules.PFActive, 31
fytok.modules.PulseSchedule, 31
fytok.modules.Summary, 34
fytok.modules.TF, 31
fytok.modules.TransportSolverNumerics, 36
fytok.modules.Utilities, 31
fytok.modules.Wall, 38
fytok.modules.Waves, 39
fytok.Scenario, 3
fytok.Tokamak, 3

s

spdm.data.Actor, 41
spdm.data.AoS, 41
spdm.data.Collection, 41
spdm.data.Directory, 43
spdm.data.Document, 43
spdm.data.Edge, 44
spdm.data.Entry, 45
spdm.data.Expression, 47
spdm.data.Field, 49
spdm.data.File, 50
spdm.data.Function, 50
spdm.data.Functor, 50
spdm.data.Graph, 51
spdm.data.HTree, 51
spdm.data.Path, 52
spdm.data.Signal, 56
spdm.data.TimeSeries, 56
spdm.mesh.Mesh, 57
spdm.mesh.mesh_curvilinear, 58
spdm.mesh.mesh_rectangular, 59
spdm.mesh.mesh_rectilinear, 59
spdm.mesh.mesh_structured, 60
spdm.mesh.mesh_uniform, 60
spdm.view.View, 61
spdm.view.view_matplotlib, 61

INDEX

A

a (CoreProfilesElectrons attribute), 7
a (CoreProfilesIon attribute), 4
a (PlasmaCompositionNeutralElement attribute), 34
a (PlasmaCompositionSpecies attribute), 33
active_limiter_point (EquilibriumBoundary attribute), 27
active_limiter_point (EquilibriumBoundarySeparatrix attribute), 28
Actor (class in spdm.data.Actor), 41
advance() (CoreProfiles method), 12
advance() (CoreSources method), 15
advance() (CoreTransport method), 19
advance() (TimeSeriesAoS method), 57
advance() (Tokamak method), 4
AMNS (class in fytok.modules.AMNSData), 4
AMNSData (class in fytok.modules.AMNSData), 4
ancestors (OpTags attribute), 52
Antiderivative (class in spdm.data.Expression), 49
antiderivative() (Mesh method), 58
AoS (class in spdm.data.AoS), 41
append (Document.Mode attribute), 44
append() (Path method), 55
area (EquilibriumGlobalQuantities attribute), 21
area (EquilibriumProfiles1D attribute), 24
as_dataclass() (in module spdm.data.Entry), 47
as_functor() (in module spdm.data.Functor), 51
as_htree() (in module spdm.data.HTree), 52
as_mesh() (in module spdm.mesh.Mesh), 58
as_path() (in module spdm.data.Path), 56
as_query() (in module spdm.data.Path), 53
as_url() (Path method), 54
asentry() (in module spdm.data.Entry), 47
atoms_n (PlasmaCompositionNeutralElement attribute), 34
axis() (CurvilinearMesh method), 58
axis_label (Mesh property), 57

B

b0 (VacuumToroidalField attribute), 33
b_field_average (EquilibriumProfiles1D attribute), 25
b_field_max (EquilibriumProfiles1D attribute), 25
b_field_min (EquilibriumProfiles1D attribute), 25
b_field_r (EquilibriumGGD attribute), 29
b_field_r (EquilibriumProfiles2D attribute), 26
b_field_tor (EquilibriumGGD attribute), 29
b_field_tor (EquilibriumGlobalQuantities.MagneticAxis attribute), 21
b_field_tor (EquilibriumProfiles2D attribute), 26
b_field_tor_vacuum_r (EequilibriumConstraints attribute), 28
b_field_z (EquilibriumGGD attribute), 29
b_field_z (EquilibriumProfiles2D attribute), 26
backend (MatplotlibView attribute), 61
backend (View attribute), 61
bbox (DomainBase property), 47
beta_normal (EquilibriumGlobalQuantities attribute), 21
beta_pol (CoreGlobalQuantities attribute), 10
beta_pol (CoreProfiles1D attribute), 9
beta_pol (EquilibriumGlobalQuantities attribute), 21
beta_pol (EquilibriumProfiles1D attribute), 25
beta_tor (CoreGlobalQuantities attribute), 10
beta_tor (EquilibriumGlobalQuantities attribute), 21
beta_tor_norm (CoreGlobalQuantities attribute), 10
boundary (CurvilinearMesh property), 59
Boundary (EquilibriumTimeSlice attribute), 29
boundary (EquilibriumTimeSlice attribute), 30
boundary (Summary attribute), 34
boundary_condition (TransportSolverNumericsEquation attribute), 37
boundary_condition (TransportSolverNumerics.TransEquatuion attribute), 38

boundary_secondary_separatrix (EquilibriumTimeSlice attribute), 30
boundary_separatrix (EquilibriumTimeSlice attribute), 30
BoundarySeparatrix (EquilibriumTimeSlice attribute), 29
bpol_probe (EequilibriumConstraints attribute), 28
brief_summary (Tokamak property), 3

C

cache_get() (HTree method), 52
call (OpTags attribute), 53
callable (Expression property), 48
cd() (Directory method), 43
cells (Mesh property), 57
ChainEntry (class in spdm.data.Entry), 46
charge (CoreProfilesElectrons attribute), 7
charge (CoreProfilesIon attribute), 4
check (OpTags attribute), 53
check() (DomainBase method), 47
check() (Query method), 53
check_type (OpTags attribute), 53
check_type() (Entry method), 46
child() (Entry method), 46
child() (EntryProxy method), 47
children (OpTags attribute), 52
children (Path property), 54
children() (HTree method), 52
children() (QueryResult method), 41
close() (Document method), 44
closed (Document.Status attribute), 44
closest_wall_point (EquilibriumBoundarySeparatrix attribute), 28
Code (class in fytok.modules.Utilities), 32
code (CoreProfiles attribute), 11
code (CoreSources attribute), 15
code (CoreSourcesSource attribute), 15
code (CoreTransport attribute), 19
code (CoreTransportModel attribute), 18
code (DatasetFAIR attribute), 19
code (Equilibrium attribute), 31
code (Module attribute), 32
code (Summary attribute), 34
code (TransportSolverNumerics attribute), 38
coefficient (TransportSolverNumericsEquation attribute), 37
collapse() (Path method), 55
Collection (class in spdm.data.Collection), 42
CollectionLocalFile (class in spdm.data.Directory), 43
collision_frequency (CoreProfilesIon attribute), 5
collisionality_norm (CoreProfilesElectrons attribute), 7
comment (IDSProperties attribute), 31
commit (Code attribute), 32
commit (Library attribute), 32
conductivity_parallel (CoreProfiles1D attribute), 9
conductivity_parallel (CoreSourcesProfiles1D attribute), 14
conductivity_parallel (CoreTransportProfiles1D attribute), 18
configuration (Summary attribute), 34
ConstantOne (class in spdm.data.Expression), 49
ConstantsFunc (class in spdm.data.Functor), 51
ConstantZero (class in spdm.data.Expression), 49
Constraints (EquilibriumTimeSlice attribute), 29
constraints (EquilibriumTimeSlice attribute), 30
Container (class in spdm.data.HTree), 52
control_parameters (TransportSolverNumericsTimeSlice attribute), 37
convergence (EquilibriumTimeSlice attribute), 30
convergence (TransportSolverNumericsEquation attribute), 37
convert_fromentry() (in module spdm.data.Entry), 47
coordinate_system (EquilibriumTimeSlice attribute), 30
coordinates() (Mesh method), 57
coordinates() (RectilinearMesh method), 59
coordinates() (StructuredMesh method), 60
CoordinateSystem (EquilibriumTimeSlice attribute), 30
copyright (Code attribute), 32
core_profiles (Tokamak attribute), 4
core_sources (Tokamak attribute), 4
core_transport (Tokamak attribute), 4
CoreGlobalQuantities (class in fytok.modules.CoreProfiles), 10
CoreGlobalQuantities.GlobalQuantitiesIon (class in fytok.modules.CoreProfiles), 11
CoreProfiles (class in fytok.modules.CoreProfiles), 11
CoreProfiles1D (class in fytok.modules.CoreProfiles), 7
CoreProfiles1D.EFieldVectorComponents (class in fytok.modules.CoreProfiles), 9
CoreProfilesElectrons (class in fytok.modules.CoreProfiles), 7
CoreProfilesIon (class in fytok.modules.CoreProfiles), 4
CoreProfilesNeutral (class in fy-

tok.modules.CoreProfiles), 6
CoreProfilesTimeSlice (class in fytok.modules.CoreProfiles), 11
CoreRadialGrid (class in fytok.modules.Utilities), 33
CoreSources (class in fytok.modules.CoreSources), 15
CoreSourcesElectrons (class in fytok.modules.CoreSources), 12
CoreSourcesGlobalQuantities (class in fytok.modules.CoreSources), 14
CoreSourcesIon (class in fytok.modules.CoreSources), 12
CoreSourcesNeutral (class in fytok.modules.CoreSources), 13
CoreSourcesProfiles1D (class in fytok.modules.CoreSources), 13
CoreSourcesSource (class in fytok.modules.CoreSources), 15
CoreSourcesTimeSlice (class in fytok.modules.CoreSources), 14
CoreTransport (class in fytok.modules.CoreTransport), 18
CoreTransportElectrons (class in fytok.modules.CoreTransport), 16
CoreTransportIon (class in fytok.modules.CoreTransport), 16
CoreTransportModel (class in fytok.modules.CoreTransport), 18
CoreTransportModelEnergy (class in fytok.modules.CoreTransport), 16
CoreTransportModelMomentum (class in fytok.modules.CoreTransport), 16
CoreTransportModelParticles (class in fytok.modules.CoreTransport), 15
CoreTransportNeutral (class in fytok.modules.CoreTransport), 17
CoreTransportProfiles1D (class in fytok.modules.CoreTransport), 17
CoreTransportTimeSlice (class in fytok.modules.CoreTransport), 18
coulomb_logarithm (CoreProfiles1D attribute), 9
count (Entry property), 46
count (OpTags attribute), 53
count() (Collection method), 42
count() (LocalFileDB method), 43
create (Document.Mode attribute), 44
create_doc() (Collection method), 42
create_index() (Collection method), 42
create_indexes() (Collection method), 42
create_many() (Collection method), 42
create_one() (Collection method), 42
create_time (DatasetFAIR attribute), 19
creation_date (IDSProperties attribute), 31
creator (DatasetFAIR attribute), 19
current (OpTags attribute), 52
current (TimeSeriesAoS property), 56
current (TransportSolverNumericsBC attribute), 37
current_bootstrap (CoreGlobalQuantities attribute), 10
current_centre (EquilibriumGlobalQuantities attribute), 21
current_non_inductive (CoreGlobalQuantities attribute), 10
current_parallel (CoreSourcesGlobalQuantities attribute), 14
current_parallel_inside (CoreProfiles1D attribute), 8
current_parallel_inside (CoreSourcesProfiles1D attribute), 14
CurveRZ (class in fytok.modules.Utilities), 33
CurvilinearMesh (class in spdm.mesh.mesh_curvilinear), 58
cwd (Directory property), 43
cycles (StructuredMesh property), 60

D

d (CoreTransportModelEnergy attribute), 16
d (CoreTransportModelMomentum attribute), 16
d (CoreTransportModelParticles attribute), 15
d (Expression property), 48
d2 (Expression property), 48
d2_dr2 (TransportSolverNumericsEquationPrimary attribute), 36
d_dr (TransportSolverNumericsEquationPrimary attribute), 36
d_dt (TransportSolverNumericsEquationPrimary attribute), 36
d_dt_cphi (TransportSolverNumericsEquationPrimary attribute), 36
d_dt_cr (TransportSolverNumericsEquationPrimary attribute), 36
d_dvolume_drho_tor_dt (TransportSolverNumericsTimeSlice attribute), 37
darea_dpsi (EquilibriumProfiles1D attribute), 24
darea_drho_tor (EquilibriumProfiles1D attribute), 24
data (Signal attribute), 56
DataDescription (class in fytok.modules.DatasetFAIR), 19
dataset_fair (Tokamak attribute), 3
DatasetFAIR (class in fytok.modules.DatasetFAIR), 19
deep_reduce() (in module spdm.data.Entry), 47

delete_many() (Collection method), 42
delete_one() (Collection method), 42
delete_one() (CollectionLocalFile method), 43
delete_one() (LocalFileDB method), 43
deleted_id (DeleteResult attribute), 42
DeleteResult (class in spdm.data.Collection), 42
delimiter (Path attribute), 54
density (CoreProfilesElectrons attribute), 7
density (CoreProfilesIon attribute), 5
density (CoreProfilesNeutral attribute), 6
density_fast (CoreProfilesElectrons attribute), 7
density_fast (CoreProfilesIon attribute), 5
density_fast (CoreProfilesNeutral attribute), 6
density_fit (CoreProfilesElectrons attribute), 7
density_fit (CoreProfilesIon attribute), 5
density_thermal (CoreProfilesElectrons attribute), 7
density_thermal (CoreProfilesIon attribute), 5
density_thermal (CoreProfilesNeutral attribute), 6
density_validity (CoreProfilesElectrons attribute), 7
density_validity (CoreProfilesIon attribute), 5
Derivative (class in spdm.data.Expression), 49
derivative() (DiracDeltaFun method), 51
derivative() (Expression method), 48
derivative() (Field method), 50
derivative() (Function method), 50
derivative() (Scalar method), 49
descendants (OpTags attribute), 52
description (DatasetFAIR attribute), 19
description (Identifier attribute), 32
Description2D (Wall attribute), 38
DetectorAperture (class in fytok.modules.Utilities), 33
device (DataDescription attribute), 19
dflux_dr (TransportSolverNumericsEquationPrimary attribute), 36
diamagnetic (CoreProfiles1D.EFieldVectorComponents attribute), 9
diamagnetic_flux (EequilibriumConstraints attribute), 28
Dict (class in spdm.data.HTree), 52
dim1 (RectangularMesh property), 59
dim1 (RectilinearMesh property), 59
dim2 (RectangularMesh property), 59
dim2 (RectilinearMesh property), 59
dimensions (RectilinearMesh property), 59
dims (DomainBase property), 47
dims (Function property), 50
dims (RectilinearMesh property), 59
DiracDeltaFun (class in spdm.data.Functor), 51
Directory (class in spdm.data.Directory), 43
display() (in module spdm.view.View), 61
disruption (Summary attribute), 34
DistributionSpecies (class in fytok.modules.Utilities), 34
dln (Expression property), 48
Document (class in spdm.data.Document), 43
Document.Mode (class in spdm.data.Document), 43
Document.Status (class in spdm.data.Document), 44
Domain (Expression attribute), 48
domain (Expression property), 48
Domain (Field attribute), 49
domain (Field property), 49
DomainBase (class in spdm.data.Expression), 47
dphi_dpsi (EquilibriumProfiles1D attribute), 22
dpressure_dpsi (EquilibriumProfiles1D attribute), 23
dpsi_drho_tor (EquilibriumProfiles1D attribute), 23
dr_dz_zero_point (EquilibriumBoundarySeparatrix attribute), 28
draw_nbi_unit() (in module fytok.modules.NBI), 31
drho_tor_dt (TransportSolverNumericsTimeSlice attribute), 37
drop_index() (Collection method), 43
drop_indexes() (Collection method), 42
dt (TimeSeriesAoS property), 56
dtype (Expression property), 48
dump (OpTags attribute), 53
dump() (AoS method), 41
dump() (Entry method), 46
dump() (HTreeNode method), 51
dump() (List method), 52
dump() (TimeSeriesAoS method), 56
dvolume_dpsi (EquilibriumProfiles1D attribute), 24
dvolume_drho_tor (EquilibriumProfiles1D attribute), 24
dx (RectilinearMesh property), 59
dx (StructuredMesh property), 60
dx (UniformMesh property), 60

E

e_field (CoreProfiles1D attribute), 9
e_field_radial (CoreTransportProfiles1D attribute), 18
ec_launchers (Tokamak attribute), 3
ECLaunchers (class in fytok.modules.ECLaunchers), 20
Edge (class in spdm.data.Edge), 44
Edge.Endpoint (class in spdm.data.Edge), 45
edges (Graph property), 51

EequilibriumConstraints (class in fytok.modules.Equilibrium), 28
ejima (CoreGlobalQuantities attribute), 11
electron_collision_time (CoreProfiles1D attribute), 9
electron_configuration (PlasmaCompositionIonState attribute), 33
electron_configuration (PlasmaCompositionNeutralState attribute), 34
Electrons (CoreProfiles1D attribute), 8
electrons (CoreProfiles1D attribute), 8
electrons (CoreSourcesGlobalQuantities attribute), 14
electrons (CoreSourcesProfiles1D attribute), 13
Electrons (CoreTransportProfiles1D attribute), 17
electrons (CoreTransportProfiles1D attribute), 18
electrons (TransportSolverNumericsBC attribute), 37
element (CoreProfilesIon attribute), 4
element (CoreProfilesNeutral attribute), 6
element (CoreSourcesIon attribute), 12
element (CoreSourcesNeutral attribute), 13
element (CoreTransportIon attribute), 16
element (CoreTransportNeutral attribute), 17
element (PlasmaCompositionIons attribute), 34
element (PlasmaCompositionNeutral attribute), 34
elms (Summary attribute), 34
elongation (EquilibriumBoundary attribute), 26
elongation (EquilibriumBoundarySeparatrix attribute), 27
elongation (EquilibriumProfiles1D attribute), 23
elongation_lower (EquilibriumBoundary attribute), 26
elongation_lower (EquilibriumBoundarySeparatrix attribute), 27
elongation_upper (EquilibriumBoundary attribute), 26
elongation_upper (EquilibriumBoundarySeparatrix attribute), 27
empty (Expression property), 48
empty (List property), 52
energy (CoreSourcesElectrons attribute), 12
energy (CoreSourcesIon attribute), 12
energy (CoreSourcesNeutral attribute), 13
energy (CoreTransportElectrons attribute), 16
energy (CoreTransportIon attribute), 16
energy (CoreTransportNeutral attribute), 17
energy_decomposed (CoreSourcesElectrons attribute), 12
energy_decomposed (CoreSourcesIon attribute), 12
energy_diamagnetic (CoreGlobalQuantities attribute), 10
energy_ion_total (TransportSolverNumericsBC attribute), 38
energy_mhd (EquilibriumGlobalQuantities attribute), 22
ensure_index() (Collection method), 42
Entry (class in spdm.data.Entry), 45
entry (Document property), 44
EntryProxy (class in spdm.data.Entry), 47
equal (OpTags attribute), 53
equal() (Entry method), 46
Equation (TransportSolverNumericsTimeSlice attribute), 37
equation (TransportSolverNumericsTimeSlice attribute), 37
equations (TransportSolverNumerics attribute), 38
Equilibrium (class in fytok.modules.Equilibrium), 30
equilibrium (Tokamak attribute), 4
EquilibriumBoundary (class in fytok.modules.Equilibrium), 26
EquilibriumBoundarySeparatrix (class in fytok.modules.Equilibrium), 27
EquilibriumCoordinateSystem (class in fytok.modules.Equilibrium), 20
EquilibriumGGD (class in fytok.modules.Equilibrium), 29
EquilibriumGlobalQuantities (class in fytok.modules.Equilibrium), 21
EquilibriumGlobalQuantities.CurrentCentre (class in fytok.modules.Equilibrium), 21
EquilibriumGlobalQuantities.MagneticAxis (class in fytok.modules.Equilibrium), 21
EquilibriumGlobalQuantities.Qmin (class in fytok.modules.Equilibrium), 22
EquilibriumProfiles1D (class in fytok.modules.Equilibrium), 22
EquilibriumProfiles2D (class in fytok.modules.Equilibrium), 25
EquilibriumTimeSlice (class in fytok.modules.Equilibrium), 29
eval() (DomainBase method), 47
eval() (Mesh method), 58
execute() (Module method), 32
exists (ChainEntry property), 46
exists (Entry property), 46
exists (OpTags attribute), 53
Expression (class in spdm.data.Expression), 48
extend() (Path method), 55

F

f (EquilibriumProfiles1D attribute), 23
f_df_dpsi (EquilibriumProfiles1D attribute), 23
faraday_angle (EequilibriumConstraints attribute), 28
fetch (OpTags attribute), 52
fetch() (ChainEntry method), 46
fetch() (CoreSourcesSource method), 15
fetch() (Entry method), 46
fetch() (EntryProxy method), 47
fetch() (Path method), 55
fetch() (Ports method), 45
ffprime (CoreProfiles1D attribute), 9
Field (class in spdm.data.Field), 49
File (class in spdm.data.File), 50
FileEntry (class in spdm.data.File), 50
find() (ChainEntry method), 46
find() (Collection method), 42
find() (Entry method), 46
find() (EntryProxy method), 47
find_many() (Collection method), 42
find_next() (Entry method), 46
find_next() (Path method), 55
find_next() (Query method), 53
find_one() (Collection method), 42
find_one() (CollectionLocalFile method), 43
find_one() (LocalFileDB method), 43
find_roots() (Expression method), 48
flow_damping_rate (CoreTransportModelMomentum attribute), 16
flush() (FileEntry method), 50
flux (CoreTransportModelEnergy attribute), 16
flux (CoreTransportModelMomentum attribute), 16
flux (CoreTransportModelParticles attribute), 16
flux (TransportSolverNumericsEquationPrimary attribute), 36
flux (TransportSolverNumerics.TransEquatuion attribute), 38
flux_loop (EequilibriumConstraints attribute), 28
flux_multiplier (CoreTransportTimeSlice attribute), 18
for_each() (ChainEntry method), 46
for_each() (Entry method), 46
for_each() (EntryProxy method), 47
for_each() (Path method), 56
Function (class in spdm.data.Function), 50
function_like() (in module spdm.data.Function), 50
Functor (class in spdm.data.Functor), 50
fusion (Summary attribute), 35
fytok.modules.AMNSData
    module, 4
fytok.modules.CoreProfiles
    module, 4
fytok.modules.CoreSources
    module, 12
fytok.modules.CoreTransport
    module, 15
fytok.modules.DatasetFAIR
    module, 19
fytok.modules.ECLaunchers
    module, 20
fytok.modules.Equilibrium
    module, 20
fytok.modules.ICAntennas
    module, 31
fytok.modules.Interferometer
    module, 31
fytok.modules.LHAntennas
    module, 31
fytok.modules.Magnetics
    module, 31
fytok.modules.NBI
    module, 31
fytok.modules.Pellets
    module, 31
fytok.modules.PFActive
    module, 31
fytok.modules.PulseSchedule
    module, 31
fytok.modules.Summary
    module, 34
fytok.modules.TF
    module, 31
fytok.modules.TransportSolverNumerics
    module, 36
fytok.modules.Utilities
    module, 31
fytok.modules.Wall
    module, 38
fytok.modules.Waves
    module, 39
fytok.Scenario
    module, 3
fytok.Tokamak
    module, 3

G

gap (EquilibriumBoundarySeparatrix attribute), 28
gas_injection_accumulated (Summary attribute), 35
gas_injection_prefill (Summary attribute), 35
gas_injection_rates (Summary attribute), 35
ge (OpTags attribute), 53

geo_object (CurvilinearMesh property), 59
geometric_axis (EquilibriumBoundary attribute), 26
geometric_axis (EquilibriumBoundarySeparatrix attribute), 27
geometric_axis (EquilibriumProfiles1D attribute), 23
geometry (Mesh property), 57
get() (Entry method), 46
get() (HTree method), 52
get() (Path method), 55
get_source() (Ports method), 45
get_target() (Ports method), 45
GGD (EquilibriumTimeSlice attribute), 30
ggd (EquilibriumTimeSlice attribute), 30
glob (LocalFileDB property), 43
global_quantities (CoreProfilesTimeSlice attribute), 11
global_quantities (CoreSourcesTimeSlice attribute), 14
global_quantities (EquilibriumTimeSlice attribute), 30
global_quantities (Summary attribute), 35
GlobalQuantities (CoreProfilesTimeSlice attribute), 11
GlobalQuantities (CoreSourcesTimeSlice attribute), 14
GlobalQuantities (EquilibriumTimeSlice attribute), 29
gm1 (EquilibriumProfiles1D attribute), 24
gm2 (EquilibriumProfiles1D attribute), 24
gm3 (EquilibriumProfiles1D attribute), 24
gm4 (EquilibriumProfiles1D attribute), 24
gm5 (EquilibriumProfiles1D attribute), 24
gm6 (EquilibriumProfiles1D attribute), 24
gm7 (EquilibriumProfiles1D attribute), 25
gm8 (EquilibriumProfiles1D attribute), 25
gm9 (EquilibriumProfiles1D attribute), 25
grad() (Field method), 50
Graph (class in spdm.data.Graph), 51
greater (OpTags attribute), 53
grid (CoreProfiles1D attribute), 8
grid (CoreSourcesProfiles1D attribute), 13
grid (EquilibriumCoordinateSystem attribute), 20
grid (EquilibriumProfiles1D attribute), 22
grid (EquilibriumProfiles2D attribute), 25
grid (TransportSolverNumericsTimeSlice attribute), 37
grid_d (CoreTransportProfiles1D attribute), 17
grid_flux (CoreTransportProfiles1D attribute), 17
grid_type (EquilibriumCoordinateSystem attribute), 20
grid_type (EquilibriumProfiles2D attribute), 25
grid_v (CoreTransportProfiles1D attribute), 17
guess_coords() (in module spdm.data.Expression), 47
guess_filepath() (LocalFileDB method), 43
guess_id() (Collection method), 42
guess_id() (LocalFileDB method), 43
guess_mesh() (in module spdm.data.Field), 49
guess_path() (CollectionLocalFile method), 43

H

has_children (Expression property), 48
has_fast_particle (CoreProfilesIon attribute), 4
has_query() (Path method), 56
has_slice() (Path method), 55
heating_current_drive (Summary attribute), 35
homogeneous_time (IDSProperties attribute), 31
HTree (class in spdm.data.HTree), 51
HTreeNode (class in spdm.data.HTree), 51

I

I (Expression property), 48
ic_antennas (Tokamak attribute), 3
ICAntennas (class in fytok.modules.ICAntennas), 31
Identifier (class in fytok.modules.Utilities), 32
identifier (CoreSourcesSource attribute), 15
identifier (CoreTransportModel attribute), 18
identifier (DatasetFAIR attribute), 19
identifier (TransportSolverNumericsBC attribute), 37
identifier (TransportSolverNumericsEquationPrimary attribute), 36
identifier (TransportSolverNumerics.TransEquatuion attribute), 38
IDS (class in fytok.modules.Utilities), 32
ids_properties (CoreProfiles attribute), 11
ids_properties (CoreSources attribute), 15
ids_properties (CoreTransport attribute), 19
ids_properties (DatasetFAIR attribute), 20
ids_properties (Equilibrium attribute), 31
ids_properties (IDS attribute), 32
ids_properties (Summary attribute), 35
ids_properties (TransportSolverNumerics attribute), 38
IDSProperties (class in fytok.modules.Utilities), 31
index (Identifier attribute), 32
index (Variable property), 49
initialize() (TimeSeriesAoS method), 56
InPorts (class in spdm.data.Edge), 45
insert (OpTags attribute), 52
insert() (Collection method), 42
insert() (Entry method), 46

insert() (EntryProxy method), 47
insert() (HTree method), 52
insert() (Path method), 55
insert_many() (Collection method), 42
insert_one() (Collection method), 42
insert_one() (CollectionLocalFile method), 43
insert_one() (LocalFileDB method), 43
inserted_id (InsertOneResult attribute), 41
inserted_id (UpdateResult attribute), 42
inserted_ids (InsertManyResult attribute), 41
InsertManyResult (class in spdm.data.Collection), 41
InsertOneResult (class in spdm.data.Collection), 41
integral() (Expression method), 48
integral() (Function method), 50
integrate() (Mesh method), 58
Interferometer (class in fytok.modules.Interferometer), 31
interferometer (Tokamak attribute), 4
interpolator() (CurvilinearMesh method), 59
interpolator() (Mesh method), 58
interpolator() (RectilinearMesh method), 60
interpolator() (StructuredMesh method), 60
INV_MOD_MAP (Document attribute), 44
ion (CoreGlobalQuantities attribute), 11
Ion (CoreProfiles1D attribute), 8
ion (CoreProfiles1D attribute), 8
ion (CoreSourcesProfiles1D attribute), 14
Ion (CoreTransportProfiles1D attribute), 17
ion (CoreTransportProfiles1D attribute), 18
ion (DistributionSpecies attribute), 34
ion (TransportSolverNumericsBC attribute), 37
ion_index (CoreProfilesNeutral attribute), 6
ion_index (CoreSourcesNeutral attribute), 13
ion_index (CoreTransportNeutral attribute), 17
ion_time_slice (CoreGlobalQuantities attribute), 11
ip (CoreGlobalQuantities attribute), 10
ip (EequilibriumConstraints attribute), 28
ip (EquilibriumGlobalQuantities attribute), 21
iron_core_segment (EequilibriumConstraints attribute), 28
is_changed (Edge.Endpoint property), 45
is_creatable (Document property), 44
is_dict (Entry property), 46
is_dict (OpTags attribute), 53
is_generator (Entry property), 46
is_generator (Path property), 54
is_impurity (CoreProfilesIon attribute), 4
is_initializied (TimeSeriesAoS property), 56
is_leaf (Entry property), 46
is_leaf (OpTags attribute), 53
is_leaf (Path property), 54
is_linked (Edge property), 45
is_list (Entry property), 46
is_list (OpTags attribute), 53
is_open (Document property), 44
is_readable (Document property), 44
is_referenced_by (DatasetFAIR attribute), 20
is_regular (Path property), 54
is_replaced_by (DatasetFAIR attribute), 20
is_root (Entry property), 46
is_root (Path property), 54
is_simple (DomainBase property), 47
is_temporary (Document property), 44
is_writable (ChainEntry property), 46
is_writable (Document property), 44
is_writable (Entry property), 45
is_writable (File property), 50
items() (Dict method), 52
iteration (TimeSeriesAoS property), 56
iteration (TimeSlice property), 56

J

j_bootstrap (CoreProfiles1D attribute), 9
j_non_inductive (CoreProfiles1D attribute), 9
j_ohmic (CoreProfiles1D attribute), 9
j_parallel (CoreSourcesProfiles1D attribute), 14
j_parallel (EquilibriumGGD attribute), 29
j_parallel (EquilibriumProfiles1D attribute), 23
j_parallel (EquilibriumProfiles2D attribute), 25
j_tor (CoreProfiles1D attribute), 9
j_tor (EquilibriumGGD attribute), 29
j_tor (EquilibriumProfiles1D attribute), 23
j_tor (EquilibriumProfiles2D attribute), 25
j_total (CoreProfiles1D attribute), 8
jacobian (EquilibriumCoordinateSystem attribute), 20

K

keys() (Entry method), 46
keys() (Path method), 56
kicks (Summary attribute), 35

L

label (CoreProfilesIon attribute), 4
label (CoreProfilesNeutral attribute), 6
label (CoreSourcesIon attribute), 12
label (CoreSourcesNeutral attribute), 13
label (CoreTransportIon attribute), 16
label (CoreTransportNeutral attribute), 17
label (PlasmaCompositionIons attribute), 34
label (PlasmaCompositionIonState attribute), 33
label (PlasmaCompositionNeutral attribute), 34

label (PlasmaCompositionNeutralState attribute), 34
label (PlasmaCompositionSpecies attribute), 33
le (OpTags attribute), 53
length_pol (EquilibriumGlobalQuantities attribute), 21
less (OpTags attribute), 53
lh_antennas (Tokamak attribute), 3
LHAntennas (class in fytok.modules.LHAntennas), 31
li_3 (CoreGlobalQuantities attribute), 10
li_3 (EquilibriumGlobalQuantities attribute), 21
Library (class in fytok.modules.Utilities), 31
library (Code attribute), 32
license (DatasetFAIR attribute), 20
limiter (Summary attribute), 35
line_average (Summary attribute), 35
link() (Graph method), 51
link() (InPorts method), 45
link() (OutPorts method), 45
link() (Ports method), 45
List (class in spdm.data.HTree), 52
list_indexes() (Collection method), 43
load() (EntryProxy class method), 47
local (Summary attribute), 35
LocalFileDB (class in spdm.data.Directory), 43
LogDerivative (class in spdm.data.Expression), 49

M

magnetic_axis (EquilibriumGlobalQuantities attribute), 21
magnetic_shear (CoreProfiles1D attribute), 9
magnetic_shear (EquilibriumProfiles1D attribute), 23
magnetic_shear_flag (Summary attribute), 35
magnetic_z (EquilibriumProfiles1D attribute), 23
Magnetics (class in fytok.modules.Magnetics), 31
magnetics (Tokamak attribute), 3
major_radius (EquilibriumProfiles1D attribute), 23
mapper (Collection property), 42
mask() (DomainBase method), 47
mass (CoreProfilesElectrons attribute), 7
mass (CoreProfilesIon attribute), 4
mass_density (EquilibriumProfiles1D attribute), 25
MatplotlibView (class in spdm.view.view_matplotlib), 61
MAX_SLICE_STOP (Path attribute), 56
merge_tree() (in module spdm.data.Path), 56
Mesh (class in spdm.mesh.Mesh), 57
mesh (Field property), 49
metadata (Edge property), 45
midplane (Summary attribute), 35
minor_radius (EquilibriumBoundary attribute), 26
minor_radius (EquilibriumBoundarySeparatrix attribute), 27
minor_radius (EquilibriumProfiles1D attribute), 23
MOD_MAP (Document attribute), 44
mode (Document property), 44
mode_str (File property), 50
Model (CoreTransport attribute), 19
model (CoreTransport attribute), 19
module
    fytok.modules.AMNSData, 4
    fytok.modules.CoreProfiles, 4
    fytok.modules.CoreSources, 12
    fytok.modules.CoreTransport, 15
    fytok.modules.DatasetFAIR, 19
    fytok.modules.ECLaunchers, 20
    fytok.modules.Equilibrium, 20
    fytok.modules.ICAntennas, 31
    fytok.modules.Interferometer, 31
    fytok.modules.LHAntennas, 31
    fytok.modules.Magnetics, 31
    fytok.modules.NBI, 31
    fytok.modules.Pellets, 31
    fytok.modules.PFActive, 31
    fytok.modules.PulseSchedule, 31
    fytok.modules.Summary, 34
    fytok.modules.TF, 31
    fytok.modules.TransportSolverNumerics, 36
    fytok.modules.Utilities, 31
    fytok.modules.Wall, 38
    fytok.modules.Waves, 39
    fytok.Scenario, 3
    fytok.Tokamak, 3
    spdm.data.Actor, 41
    spdm.data.AoS, 41
    spdm.data.Collection, 41
    spdm.data.Directory, 43
    spdm.data.Document, 43
    spdm.data.Edge, 44
    spdm.data.Entry, 45
    spdm.data.Expression, 47
    spdm.data.Field, 49
    spdm.data.File, 50
    spdm.data.Function, 50
    spdm.data.Functor, 50
    spdm.data.Graph, 51
    spdm.data.HTree, 51
    spdm.data.Path, 52
    spdm.data.Signal, 56
    spdm.data.TimeSeries, 56

spdm.mesh.Mesh, 57
spdm.mesh.mesh_curvilinear, 58
spdm.mesh.mesh_rectangular, 59
spdm.mesh.mesh_rectilinear, 59
spdm.mesh.mesh_structured, 60
spdm.mesh.mesh_uniform, 60
spdm.view.View, 61
spdm.view.view_matplotlib, 61
Module (class in fytok.modules.Utilities), 32
momentum (CoreSourcesIon attribute), 12
momentum (CoreTransportElectrons attribute), 16
momentum (CoreTransportIon attribute), 16
momentum_tor (CoreProfiles1D attribute), 8
momentum_tor (CoreSourcesProfiles1D attribute), 13
momentum_tor (CoreTransportProfiles1D attribute), 18
momentum_tor (TransportSolverNumericsBC attribute), 38
momentum_tor_j_cross_b_field (CoreSourcesProfiles1D attribute), 14
mse_polarisation_angle (EequilibriumConstraints attribute), 28
multiple_states_flag (CoreProfilesIon attribute), 5
multiple_states_flag (CoreProfilesNeutral attribute), 6
multiple_states_flag (CoreSourcesIon attribute), 12
multiple_states_flag (CoreSourcesNeutral attribute), 13
multiple_states_flag (CoreTransportIon attribute), 17
multiple_states_flag (CoreTransportNeutral attribute), 17

N

n_e (EequilibriumConstraints attribute), 28
n_e_line (EequilibriumConstraints attribute), 28
n_e_volume_average (CoreGlobalQuantities attribute), 11
n_i_thermal_total (CoreProfiles1D attribute), 8
n_i_total (CoreProfiles1D attribute), 8
n_i_total_over_n_e (CoreProfiles1D attribute), 8
n_i_volume_average (CoreGlobalQuantities.GlobalQuantitiesIon attribute), 11
name (Code attribute), 32
name (Expression property), 48
name (Identifier attribute), 32
name (Library attribute), 32
name (Mesh property), 57
NBI (class in fytok.modules.NBI), 31
nbi (Tokamak attribute), 3
ndim (Function property), 50
ndim (Mesh property), 57
ndims (DomainBase property), 47
Neutral (CoreProfiles1D attribute), 8
neutral (CoreProfiles1D attribute), 8
neutral (CoreSourcesProfiles1D attribute), 14
Neutral (CoreTransportProfiles1D attribute), 17
neutral (CoreTransportProfiles1D attribute), 18
neutral (DistributionSpecies attribute), 34
neutral_index (CoreProfilesIon attribute), 4
neutral_index (CoreSourcesIon attribute), 13
neutral_index (CoreTransportIon attribute), 17
neutral_type (PlasmaCompositionNeutralState attribute), 34
next (OpTags attribute), 52
next (Path property), 55
next() (Entry method), 46
next_id (Collection property), 42
next_id (CollectionLocalFile property), 43
Node (in module spdm.data.HTree), 52
normalize() (Path static method), 55
NullMesh (class in spdm.mesh.Mesh), 58

O

ontology (DatasetFAIR attribute), 19
open() (Document method), 44
open_collection() (in module spdm.data.Collection), 43
open_db() (in module spdm.data.Collection), 43
open_document() (LocalFileDB method), 43
open_entry() (in module spdm.data.Entry), 47
opened (Document.Status attribute), 44
OpTags (class in spdm.data.Path), 52
order (Derivative property), 49
origin (StructuredMesh property), 60
origin (UniformMesh property), 60
outline (EquilibriumBoundary attribute), 26
outline (EquilibriumBoundarySeparatrix attribute), 27
OutPorts (class in spdm.data.Edge), 45
output_flag (Code attribute), 32

P

parameters (Code attribute), 32
parameters (Library attribute), 32
parametric_coordinates() (Mesh method), 57
parametric_coordinates() (StructuredMesh method), 60
parent (Entry property), 46
parent (OpTags attribute), 52
parent (Path property), 54
partial_derivative() (Mesh method), 58

PartialDerivative (class in spdm.data.Expression), 49
particles (CoreSourcesElectrons attribute), 12
particles (CoreSourcesIon attribute), 12
particles (CoreSourcesNeutral attribute), 13
particles (CoreTransportElectrons attribute), 16
particles (CoreTransportIon attribute), 16
particles (CoreTransportNeutral attribute), 17
particles_decomposed (CoreSourcesElectrons attribute), 12
particles_decomposed (CoreSourcesIon attribute), 12
particles_inside (CoreSourcesElectrons attribute), 12
Path (class in spdm.data.Path), 53
path (Directory property), 43
path (Document property), 44
path (Entry property), 46
path (HTreeNode property), 51
PATH_PATTERN (Path attribute), 55
PATH_REGEX_DICT (Path attribute), 55
PathError, 53
pd() (Expression method), 48
pedestal_fits (Summary attribute), 35
Pellets (class in fytok.modules.Pellets), 31
pellets (Summary attribute), 35
pellets (Tokamak attribute), 3
periods (DomainBase property), 47
pf_active (Tokamak attribute), 3
pf_current (EequilibriumConstraints attribute), 28
pf_passive_current (EequilibriumConstraints attribute), 29
PFActive (class in fytok.modules.PFActive), 31
phi (EquilibriumGGD attribute), 29
phi (EquilibriumProfiles1D attribute), 22
phi (EquilibriumProfiles2D attribute), 25
phi_potential (CoreProfiles1D attribute), 9
Piecewise (class in spdm.data.Expression), 49
plasma_duration (Summary attribute), 35
plasma_inductance (EquilibriumGlobalQuantities attribute), 22
plasma_resistance (EquilibriumGlobalQuantities attribute), 22
PlasmaCompositionIons (class in fytok.modules.Utilities), 34
PlasmaCompositionIonState (class in fytok.modules.Utilities), 33
PlasmaCompositionNeutral (class in fytok.modules.Utilities), 34
PlasmaCompositionNeutralElement (class in fytok.modules.Utilities), 33
PlasmaCompositionNeutralState (class in fytok.modules.Utilities), 34
PlasmaCompositionSpecies (class in fytok.modules.Utilities), 33
plot() (in module spdm.view.View), 61
plot() (MatplotlibView method), 61
plot() (View method), 61
PointRZ (class in fytok.modules.Utilities), 33
points (CurvilinearMesh property), 58
points (DomainBase property), 47
points (Mesh property), 57
points (RectilinearMesh property), 59
poloidal (CoreProfiles1D.EFieldVectorComponents attribute), 9
Ports (class in spdm.data.Edge), 45
power (CoreSourcesGlobalQuantities attribute), 14
power_inside (CoreSourcesElectrons attribute), 12
ppoly() (Field method), 50
pprime (CoreProfiles1D attribute), 9
prepend() (Path method), 55
preprocess() (TransportSolverNumerics method), 38
pressure (CoreProfiles1D attribute), 8
pressure (CoreProfilesElectrons attribute), 7
pressure (CoreProfilesIon attribute), 5
pressure (CoreProfilesNeutral attribute), 6
pressure (EequilibriumConstraints attribute), 29
pressure (EquilibriumProfiles1D attribute), 22
pressure_fast_parallel (CoreProfilesElectrons attribute), 7
pressure_fast_parallel (CoreProfilesIon attribute), 5
pressure_fast_parallel (CoreProfilesNeutral attribute), 6
pressure_fast_perpendicular (CoreProfilesElectrons attribute), 7
pressure_fast_perpendicular (CoreProfilesIon attribute), 5
pressure_fast_perpendicular (CoreProfilesNeutral attribute), 6
pressure_ion_total (CoreProfiles1D attribute), 8
pressure_parallel (CoreProfiles1D attribute), 8
pressure_perpendicular (CoreProfiles1D attribute), 8
pressure_thermal (CoreProfiles1D attribute), 8
pressure_thermal (CoreProfilesElectrons attribute), 7
pressure_thermal (CoreProfilesIon attribute), 5
pressure_thermal (CoreProfilesNeutral attribute), 6

previous (TimeSeriesAoS property), 56
primary_coordinate (TransportSolverNumerics attribute), 38
primary_quantity (TransportSolverNumericsEquation attribute), 37
profile (TransportSolverNumericsEquationPrimary attribute), 36
profile (TransportSolverNumerics.TransEquatuion attribute), 38
Profiles1D (CoreProfilesTimeSlice attribute), 11
Profiles1D (CoreSourcesTimeSlice attribute), 14
Profiles1D (CoreTransportTimeSlice attribute), 18
Profiles1D (EquilibriumTimeSlice attribute), 30
Profiles2D (EquilibriumTimeSlice attribute), 30
profiles_1d (CoreProfilesTimeSlice attribute), 11
profiles_1d (CoreSourcesTimeSlice attribute), 14
profiles_1d (CoreTransportTimeSlice attribute), 18
profiles_1d (EquilibriumTimeSlice attribute), 30
profiles_2d (EquilibriumTimeSlice attribute), 30
provenance (IDSProperties attribute), 31
provider (IDSProperties attribute), 31
psi (EquilibriumBoundary attribute), 26
psi (EquilibriumBoundarySeparatrix attribute), 27
psi (EquilibriumGGD attribute), 29
psi (EquilibriumProfiles1D attribute), 22
psi (EquilibriumProfiles2D attribute), 25
psi_axis (CoreRadialGrid attribute), 33
psi_axis (EquilibriumGlobalQuantities attribute), 21
psi_boundary (CoreRadialGrid attribute), 33
psi_boundary (EquilibriumGlobalQuantities attribute), 21
psi_external_average (EquilibriumGlobalQuantities attribute), 22
psi_norm (CoreRadialGrid attribute), 33
psi_norm (EquilibriumBoundary attribute), 26
psi_norm (EquilibriumProfiles1D attribute), 22
pulse_schedule (Scenario attribute), 3
PulseSchedule (class in fytok.modules.PulseSchedule), 31
put() (Entry method), 46

Q

q (CoreProfiles1D attribute), 9
q (EequilibriumConstraints attribute), 29
q (EquilibriumProfiles1D attribute), 23
q_95 (EquilibriumGlobalQuantities attribute), 22
q_axis (EquilibriumGlobalQuantities attribute), 22
q_min (EquilibriumGlobalQuantities attribute), 22
Query (class in spdm.data.Path), 53
QueryResult (class in spdm.data.AoS), 41

R

r (CurveRZ attribute), 33
r (EquilibriumCoordinateSystem attribute), 20
r (EquilibriumGGD attribute), 29
r (EquilibriumGlobalQuantities.CurrentCentre attribute), 21
r (EquilibriumGlobalQuantities.MagneticAxis attribute), 21
r (EquilibriumProfiles2D attribute), 25
r (PointRZ attribute), 33
r (RZTuple attribute), 33
r0 (VacuumToroidalField attribute), 33
r_inboard (EquilibriumProfiles1D attribute), 23
r_outboard (EquilibriumProfiles1D attribute), 23
radial (CoreProfiles1D.EFieldVectorComponents attribute), 9
radial_grid (EquilibriumCoordinateSystem attribute), 20
rank (Function property), 50
rank (Mesh property), 57
rank (RectilinearMesh property), 59
read (Document.Mode attribute), 44
read() (Document method), 44
read() (File method), 50
RectangularMesh (class in spdm.mesh.mesh_rectangular), 59
RectilinearMesh (class in spdm.mesh.mesh_rectilinear), 59
reduce (OpTags attribute), 53
reduce() (Path static method), 55
refresh() (CoreProfiles method), 12
refresh() (CoreSources method), 15
refresh() (CoreSourcesSource method), 15
refresh() (CoreTransport method), 19
refresh() (CoreTransportModel method), 18
refresh() (Ports method), 45
refresh() (TimeSeriesAoS method), 56
refresh() (TimeSlice method), 56
refresh() (Tokamak method), 4
refresh() (TransportSolverNumerics method), 38
RegularMesh (class in spdm.mesh.Mesh), 58
reindex() (Collection method), 43
remove (OpTags attribute), 52
remove() (Entry method), 46

remove() (EntryProxy method), 47
remove() (HTree method), 52
remove() (Path method), 55
render() (MatplotlibView method), 61
render() (View method), 61
replace_one() (Collection method), 42
replaces (DatasetFAIR attribute), 20
repository (Code attribute), 32
repository (Library attribute), 32
reset() (Entry method), 45
resistive_psi_losses (CoreGlobalQuantities attribute), 10
rho_tor (EquilibriumProfiles1D attribute), 23
rho_tor_boundary (CoreRadialGrid attribute), 33
rho_tor_norm (CoreRadialGrid attribute), 33
rho_tor_norm (EquilibriumGlobalQuantities.Qmin attribute), 22
rho_tor_norm (EquilibriumProfiles1D attribute), 23
rho_tor_norm (TransportSolverNumericsBC attribute), 37
rho_volume_norm (EquilibriumProfiles1D attribute), 24
rights_holder (DatasetFAIR attribute), 20
rmps (Summary attribute), 35
root (Entry property), 46
root (HTreeNode property), 51
root (OpTags attribute), 52
rotation_frequency_tor (CoreProfilesIon attribute), 5
rotation_frequency_tor_sonic (CoreProfiles1D attribute), 9
run (DataDescription attribute), 19
runaways (Summary attribute), 35
RZTuple (class in fytok.modules.Utilities), 32

S

Scalar (class in spdm.data.Expression), 49
Scenario (class in fytok.Scenario), 3
scrape_off_layer (Summary attribute), 35
search (OpTags attribute), 53
set() (OutPorts method), 45
SetpFun (class in spdm.data.Functor), 51
shape (DomainBase property), 47
shape (Mesh property), 57
shot (DataDescription attribute), 19
Signal (class in spdm.data.Signal), 56
SignalND (class in spdm.data.Signal), 56
signature (View property), 61
site (DatasetFAIR attribute), 19
slibings (Path property), 55
solver (TransportSolverNumerics attribute), 38
sort (OpTags attribute), 53
Source (CoreSources attribute), 15
source (CoreSources attribute), 15
source (Edge property), 45
spdm.data.Actor
    module, 41
spdm.data.AoS
    module, 41
spdm.data.Collection
    module, 41
spdm.data.Directory
    module, 43
spdm.data.Document
    module, 43
spdm.data.Edge
    module, 44
spdm.data.Entry
    module, 45
spdm.data.Expression
    module, 47
spdm.data.Field
    module, 49
spdm.data.File
    module, 50
spdm.data.Function
    module, 50
spdm.data.Functor
    module, 50
spdm.data.Graph
    module, 51
spdm.data.HTree
    module, 51
spdm.data.Path
    module, 52
spdm.data.Signal
    module, 56
spdm.data.TimeSeries
    module, 56
spdm.mesh.Mesh
    module, 57
spdm.mesh.mesh_curvilinear
    module, 58
spdm.mesh.mesh_rectangular
    module, 59
spdm.mesh.mesh_rectilinear
    module, 59
spdm.mesh.mesh_structured
    module, 60
spdm.mesh.mesh_uniform
    module, 60
spdm.view.View
    module, 61
spdm.view.view_matplotlib
    module, 61
species (CoreSourcesSource attribute), 15
split() (Edge method), 45

squareness (EquilibriumProfiles1D attribute), 24
squareness_lower_inner (EquilibriumBoundary attribute), 26
squareness_lower_inner (EquilibriumBoundarySeparatrix attribute), 27
squareness_lower_inner (EquilibriumProfiles1D attribute), 24
squareness_lower_outer (EquilibriumBoundary attribute), 27
squareness_lower_outer (EquilibriumBoundarySeparatrix attribute), 28
squareness_lower_outer (EquilibriumProfiles1D attribute), 24
squareness_upper_inner (EquilibriumBoundary attribute), 26
squareness_upper_inner (EquilibriumBoundarySeparatrix attribute), 27
squareness_upper_inner (EquilibriumProfiles1D attribute), 24
squareness_upper_outer (EquilibriumBoundary attribute), 26
squareness_upper_outer (EquilibriumBoundarySeparatrix attribute), 27
squareness_upper_outer (EquilibriumProfiles1D attribute), 24
state (CoreProfilesIon attribute), 5
state (CoreProfilesNeutral attribute), 6
state (CoreSourcesIon attribute), 13
state (CoreSourcesNeutral attribute), 13
state (CoreTransportIon attribute), 17
state (CoreTransportNeutral attribute), 17
state (PlasmaCompositionIons attribute), 34
state (PlasmaCompositionNeutral attribute), 34
stationary_phase_flag (Summary attribute), 35
strike_point (EequilibriumConstraints attribute), 29
strike_point (EquilibriumBoundary attribute), 27
strike_point (EquilibriumBoundarySeparatrix attribute), 28
StructuredMesh (class in spdm.mesh.mesh_structured), 60
success (DeleteResult attribute), 42
success (InsertManyResult attribute), 41
success (InsertOneResult attribute), 41
success (UpdateResult attribute), 42
Summary (class in fytok.modules.Summary), 34
summary (DataDescription attribute), 19
summary (Tokamak attribute), 4
surface (EquilibriumGlobalQuantities attribute), 21
surface (EquilibriumProfiles1D attribute), 24

T

t_e_peaking (CoreGlobalQuantities attribute), 10
t_e_volume_average (CoreGlobalQuantities attribute), 11
t_i_average (CoreProfiles1D attribute), 8
t_i_average_fit (CoreProfiles1D attribute), 9
t_i_average_peaking (CoreGlobalQuantities attribute), 10
t_i_volume_average (CoreGlobalQuantities.GlobalQuantitiesIon attribute), 11
tag (Module property), 32
tag (Summary attribute), 36
tag (Tokamak property), 3
tags (Path attribute), 54
target (Edge property), 45
tau (CoreProfilesElectrons attribute), 7
temperature (CoreProfilesElectrons attribute), 7
temperature (CoreProfilesIon attribute), 5
temperature (CoreProfilesNeutral attribute), 6
temperature_fit (CoreProfilesElectrons attribute), 7
temperature_fit (CoreProfilesIon attribute), 5
temperature_validity (CoreProfilesElectrons attribute), 7
temperature_validity (CoreProfilesIon attribute), 5
temporary (Document.Mode attribute), 44
tensor_contravariant (EquilibriumCoordinateSystem attribute), 20
tensor_covariant (EquilibriumCoordinateSystem attribute), 20
TF (class in fytok.modules.TF), 31
tf (Tokamak attribute), 3
theta (EquilibriumGGD attribute), 29
theta (EquilibriumProfiles2D attribute), 25
time (CoreProfiles1D attribute), 10
time (CoreProfilesTimeSlice attribute), 11
time (CoreSourcesTimeSlice attribute), 15
time (CoreTransportProfiles1D attribute), 18
time (CoreTransportTimeSlice attribute), 18
time (EquilibriumTimeSlice attribute), 30
time (Signal attribute), 56
time (TimeSeriesAoS property), 56
time (TimeSlice attribute), 56
time (TransportSolverNumericsTimeSlice attribute), 37
time_breakdown (Summary attribute), 36
time_slice (Actor attribute), 41
time_slice (CoreProfiles attribute), 11
time_slice (CoreSources attribute), 15

time_slice (CoreSourcesSource attribute), 15
time_slice (CoreTransport attribute), 19
time_slice (CoreTransportModel attribute), 18
time_slice (DatasetFAIR attribute), 20
time_slice (Equilibrium attribute), 31
time_slice (Module attribute), 32
time_slice (Scenario attribute), 3
time_slice (Summary attribute), 36
time_slice (Tokamak attribute), 4
time_slice (TransportSolverNumerics attribute), 38
time_width (Summary attribute), 36
TimeSeriesAoS (class in spdm.data.TimeSeries), 56
TimeSlice (class in spdm.data.TimeSeries), 56
TimeSlice (CoreProfiles attribute), 11
TimeSlice (CoreSourcesSource attribute), 15
TimeSlice (CoreTransportModel attribute), 18
TimeSlice (Equilibrium attribute), 31
TimeSlice (TransportSolverNumerics attribute), 38
title (Tokamak property), 3
Tokamak (class in fytok.Tokamak), 3
tokamak (Scenario attribute), 3
TOLERANCE (CurvilinearMesh attribute), 58
toroidal (CoreProfiles1D.EFieldVectorComponents attribute), 9
torque_tor (CoreSourcesGlobalQuantities attribute), 14
torque_tor_inside (CoreSourcesProfiles1D attribute), 14
total_ion_energy (CoreSourcesProfiles1D attribute), 13
total_ion_energy (CoreTransportProfiles1D attribute), 18
total_ion_energy_decomposed (CoreSourcesProfiles1D attribute), 13
total_ion_particles (CoreSourcesGlobalQuantities attribute), 14
total_ion_power (CoreSourcesGlobalQuantities attribute), 14
total_ion_power_inside (CoreSourcesProfiles1D attribute), 13
transport_solver (Tokamak attribute), 4
TransportSolverNumerics (class in fytok.modules.TransportSolverNumerics), 38
TransportSolverNumericsBC (class in fytok.modules.TransportSolverNumerics), 37
TransportSolverNumericsEquation (class in fytok.modules.TransportSolverNumerics), 36
TransportSolverNumericsEquationPrimary (class in fytok.modules.TransportSolverNumerics), 36
TransportSolverNumericsTimeSlice (class in fytok.modules.TransportSolverNumerics), 37
TransportSolverNumerics.TransEquatuion (class in fytok.modules.TransportSolverNumerics), 38
trapped_fraction (EquilibriumProfiles1D attribute), 24
traversal() (Path method), 56
triangularity (EquilibriumBoundary attribute), 26
triangularity (EquilibriumBoundarySeparatrix attribute), 27
triangularity (EquilibriumProfiles1D attribute), 23
triangularity_lower (EquilibriumBoundary attribute), 26
triangularity_lower (EquilibriumBoundarySeparatrix attribute), 27
triangularity_lower (EquilibriumProfiles1D attribute), 23
triangularity_upper (EquilibriumBoundary attribute), 26
triangularity_upper (EquilibriumBoundarySeparatrix attribute), 27
triangularity_upper (EquilibriumProfiles1D attribute), 23
type (DistributionSpecies attribute), 34
type (EquilibriumBoundary attribute), 26
type (EquilibriumBoundarySeparatrix attribute), 27
type (EquilibriumProfiles2D attribute), 25
type (Mesh property), 57

U

UniformMesh (class in spdm.mesh.mesh_uniform), 60
units (Mesh property), 57
unlink() (Edge.Endpoint method), 45
update (OpTags attribute), 52
update() (Edge.Endpoint method), 45
update() (Entry method), 46
update() (EntryProxy method), 47
update() (HTree method), 52
update() (InPorts method), 45
update() (OutPorts method), 45
update() (Path method), 55
update_many() (Collection method), 42
update_one() (Collection method), 42

update_one() (CollectionLocalFile method), 43
update_one() (LocalFileDB method), 43
update_tree() (in module spdm.data.Path), 56
UpdateResult (class in spdm.data.Collection), 42
url (Document property), 44
uv (CurvilinearMesh property), 58
uvw() (Mesh method), 57

V

v (CoreTransportModelEnergy attribute), 16
v (CoreTransportModelMomentum attribute), 16
v (CoreTransportModelParticles attribute), 16
v_external (EquilibriumGlobalQuantities attribute), 22
v_loop (CoreGlobalQuantities attribute), 10
vacuum_toroidal_field (CoreGlobalQuantities attribute), 10
vacuum_toroidal_field (CoreProfilesTimeSlice attribute), 11
vacuum_toroidal_field (CoreTransport attribute), 19
vacuum_toroidal_field (CoreTransportTimeSlice attribute), 18
vacuum_toroidal_field (EquilibriumTimeSlice attribute), 30
VacuumToroidalField (class in fytok.modules.Utilities), 33
valid (DatasetFAIR attribute), 20
validate() (Function method), 50
value (EquilibriumGlobalQuantities.Qmin attribute), 22
Variable (class in spdm.data.Expression), 48
velocity (CoreProfilesIon attribute), 6
velocity_z (EquilibriumGlobalQuantities.CurrentCentre attribute), 21
version (Code attribute), 32
version (Library attribute), 32
version_put (IDSProperties attribute), 31
vertices (Mesh property), 57
vertices (RectilinearMesh property), 59
vertices() (UniformMesh method), 60
vibrational_level (PlasmaCompositionIonState attribute), 33
vibrational_level (PlasmaCompositionNeutralState attribute), 34
vibrational_mode (PlasmaCompositionIonState attribute), 33
vibrational_mode (PlasmaCompositionNeutralState attribute), 34
View (class in spdm.view.View), 61
viewer() (in module spdm.view.View), 61
volume (EquilibriumGlobalQuantities attribute), 21
volume (EquilibriumProfiles1D attribute), 24
volume_average (Summary attribute), 36
volume_element (CurvilinearMesh property), 58
vT (CoreProfilesElectrons attribute), 7

W

Wall (class in fytok.modules.Wall), 38
wall (Summary attribute), 36
wall (Tokamak attribute), 3
Waves (class in fytok.modules.Waves), 39
write (Document.Mode attribute), 44
write() (Document method), 44
write() (File method), 50

X

x_label (Function property), 50
x_point (EequilibriumConstraints attribute), 29
x_point (EquilibriumBoundary attribute), 27
x_point (EquilibriumBoundarySeparatrix attribute), 28
xyz (CurvilinearMesh property), 59
xyz (Mesh property), 57

Z

z (CoreProfilesElectrons attribute), 7
z (CoreProfilesIon attribute), 4
z (CurveRZ attribute), 33
z (EquilibriumCoordinateSystem attribute), 20
z (EquilibriumGGD attribute), 29
z (EquilibriumGlobalQuantities.CurrentCentre attribute), 21
z (EquilibriumGlobalQuantities.MagneticAxis attribute), 21
z (EquilibriumProfiles2D attribute), 25
z (PointRZ attribute), 33
z (RZTuple attribute), 33
z_eff_resistive (CoreGlobalQuantities attribute), 10
z_ion (CoreProfilesIon attribute), 6
z_ion (CoreSourcesIon attribute), 13
z_ion (CoreTransportIon attribute), 17
z_ion (PlasmaCompositionIons attribute), 34
z_ion_1d (CoreProfilesIon attribute), 4
z_ion_square_1d (CoreProfilesIon attribute), 4
z_max (PlasmaCompositionIonState attribute), 33
z_min (PlasmaCompositionIonState attribute), 33
z_n (PlasmaCompositionNeutralElement attribute), 34
z_n (PlasmaCompositionSpecies attribute), 33
zeff (CoreProfiles1D attribute), 8
zeff_fit (CoreProfiles1D attribute), 10