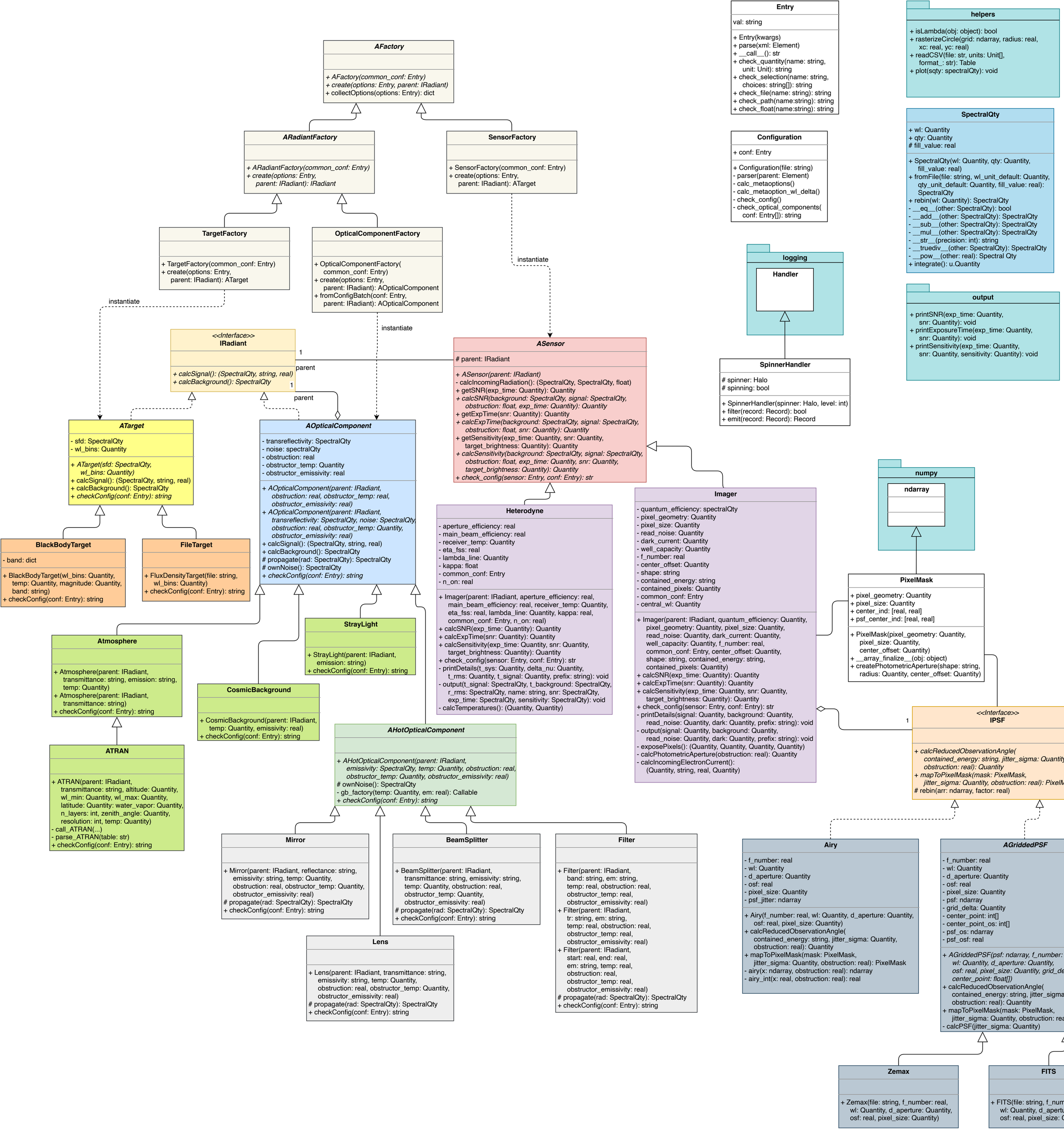
AFactory
+ AFactory(common_conf: Entry)
+ create(options: Entry, parent: IRadiant)
+ collectOptions(options: Entry): dict
ARadiantFactory
+ ARadiantFactory(common_conf: Entry)
+ create(options: Entry, parent: IRadiant): IRadiant
SensorFactory
+ SensorFactory(common_conf: Entry)
+ create(options: Entry, parent: IRadiant): ATarget
TargetFactory
+ TargetFactory(common_conf: Entry)
+ create(options: Entry, parent: IRadiant): ATarget
OpticalComponentFactory
+ OpticalComponentFactory(common_conf: Entry)
+ create(options: Entry, parent: IRadiant): AOpticalComponent
+ fromConfigBatch(conf: Entry, parent: IRadiant): AOpticalComponent
instantiate
instantiate
instantiate
<<Interface>>
IRadiant
+ calcSignal(): (SpectralQty, string, real)
+ calcBackground(): SpectralQty
1
parent
1
parent
ASensor
parent: IRadiant
+ ASensor(parent: IRadiant)
- calcIncomingRadiation(): (SpectralQty, SpectralQty, float)
+ getSNR(exp_time: Quantity): Quantity
+ calcSNR(background: SpectralQty, signal: SpectralQty, obstruction: float, exp_time: Quantity): Quantity
+ getExpTime(snr: Quantity): Quantity
+ calcExpTime(background: SpectralQty, signal: SpectralQty, obstruction: float, snr: Quantity): Quantity
+ getSensitivity(exp_time: Quantity, snr: Quantity, target_brightness: Quantity): Quantity
+ calcSensitivity(background: SpectralQty, signal: SpectralQty, obstruction: float, exp_time: Quantity, snr: Quantity, target_brightness: Quantity): Quantity
+ check_config(sensor: Entry, conf: Entry): str
ATarget
- sfd: SpectralQty
- wl_bins: Quantity
+ ATarget(sfd: SpectralQty, wl_bins: Quantity)
+ calcSignal(): (SpectralQty, string, real)
+ calcBackground(): SpectralQty
+ checkConfig(conf: Entry): string
AOpticalComponent
- transreflectivity: SpectralQty
- noise: spectralQty
- obstruction: real
- obstructor_temp: Quantity
- obstructor_emissivity: real
+ AOpticalComponent(parent: IRadiant, obstruction: real, obstructor_temp: real, obstructor_emissivity: real)
+ AOpticalComponent(parent: IRadiant, transreflectivity: SpectralQty, noise: SpectralQty, obstruction: real, obstructor_temp: Quantity, obstructor_emissivity: real)
+ calcSignal(): (SpectralQty, string, real)
+ calcBackground(): SpectralQty
# propagate(rad: SpectralQty): SpectralQty
# ownNoise(): SpectralQty
+ checkConfig(conf: Entry): string
BlackBodyTarget
- band: dict
+ BlackBodyTarget(wl_bins: Quantity, temp: Quantity, magnitude: Quantity, band: string)
+ checkConfig(conf: Entry): string
FileTarget
+ FluxDensityTarget(file: string, wl_bins: Quantity)
+ checkConfig(conf: Entry): string
Atmosphere
+ Atmosphere(parent: IRadiant, transmittance: string, emission: string, temp: Quantity)
+ Atmosphere(parent: IRadiant, transmittance: string)
+ checkConfig(conf: Entry): string
StrayLight
+ StrayLight(parent: IRadiant, emission: string)
+ checkConfig(conf: Entry): string
CosmicBackground
+ CosmicBackground(parent: IRadiant, temp: Quantity, emissivity: real)
+ checkConfig(conf: Entry): string
ATRAN
+ ATRAN(parent: IRadiant, transmittance: string, altitude: Quantity, wl_min: Quantity, wl_max: Quantity, latitude: Quantity: water_vapor: Quantity, n_layers: int, zenith_angle: Quantity, resolution: int, temp: Quantity)
- call_ATRAN(...)
- parse_ATRAN(table: str)
+ checkConfig(conf: Entry): string
Heterodyne
- aperture_efficiency: real
- main_beam_efficiency: real
- receiver_temp: Quantity
- eta_fss: real
- lambda_line: Quantity
- kappa: float
- common_conf: Entry
- n_on: real
+ Imager(parent: IRadiant, aperture_efficiency: real, main_beam_efficiency: real, receiver_temp: Quantity, eta_fss: real, lambda_line: Quantity, kappa: real, common_conf: Entry, n_on: real)
+ calcSNR(exp_time: Quantity): Quantity
+ calcExpTime(snr: Quantity): Quantity
+ calcSensitivity(exp_time: Quantity, snr: Quantity, target_brightness: Quantity): Quantity
+ check_config(sensor: Entry, conf: Entry): str
- printDetails(t_sys: Quantity, delta_nu: Quantity, t_rms: Quantity, t_signal: Quantity, prefix: string): void
- output(t_signal: SpectralQty, t_background: SpectralQty, r_rms: SpectralQty, name: string, snr: SpectralQty, exp_time: SpectralQty, sensitivity: SpectralQty): void
- calcTemperatures(): (Quantity, Quantity)
Imager
- quantum_efficiency: spectralQty
- pixel_geometry: Quantity
- pixel_size: Quantity
- read_noise: Quantity
- dark_current: Quantity
- well_capacity: Quantity
- f_number: real
- center_offset: Quantity
- shape: string
- contained_energy: string
- contained_pixels: Quantity
- common_conf: Entry
- central_wl: Quantity
+ Imager(parent: IRadiant, quantum_efficiency: Quantity, pixel_geometry: Quantity, pixel_size: Quantity, read_noise: Quantity, dark_current: Quantity, well_capacity: Quantity, f_number: real, common_conf: Entry, center_offset: Quantity, shape: string, contained_energy: string, contained_pixels: Quantity)
+ calcSNR(exp_time: Quantity): Quantity
+ calcExpTime(snr: Quantity): Quantity
+ calcSensitivity(exp_time: Quantity, snr: Quantity, target_brightness: Quantity): Quantity
+ check_config(sensor: Entry, conf: Entry): str
- printDetails(signal: Quantity, background: Quantity, read_noise: Quantity, dark: Quantity, prefix: string): void
- output(signal: Quantity, background: Quantity, read_noise: Quantity, dark: Quantity, prefix: string): void
- exposePixels(): (Quantity, Quantity, Quantity, Quantity)
- calcPhotometricAperture(obstruction: real): Quantity
- calcIncomingElectronCurrent(): (Quantity, string, real, Quantity)
AHotOpticalComponent
+ AHotOpticalComponent(parent: IRadiant, emissivity: SpectralQty, temp: Quantity, obstruction: real, obstructor_temp: Quantity, obstructor_emissivity: real)
# ownNoise(): SpectralQty
- gb_factory(temp: Quantity, em: real): Callable
+ checkConfig(conf: Entry): string
Mirror
+ Mirror(parent: IRadiant, reflectance: string, emissivity: string, temp: Quantity, obstruction: real, obstructor_temp: Quantity, obstructor_emissivity: real)
# propagate(rad: SpectralQty): SpectralQty
+ checkConfig(conf: Entry): string
Lens
+ Lens(parent: IRadiant, transmittance: string, emissivity: string, temp: Quantity, obstruction: real, obstructor_temp: Quantity, obstructor_emissivity: real)
# propagate(rad: SpectralQty): SpectralQty
+ checkConfig(conf: Entry): string
BeamSplitter
+ BeamSplitter(parent: IRadiant, transmittance: string, emissivity: string, temp: Quantity, obstruction: real, obstructor_temp: Quantity, obstructor_emissivity: real)
# propagate(rad: SpectralQty): SpectralQty
+ checkConfig(conf: Entry): string
Filter
+ Filter(parent: IRadiant, band: string, em: string, temp: real, obstruction: real, obstructor_temp: real, obstructor_emissivity: real)
+ Filter(parent: IRadiant, tr: string, em: string, temp: real, obstruction: real, obstructor_temp: real, obstructor_emissivity: real)
+ Filter(parent: IRadiant, start: real, end: real, em: string, temp: real, obstruction: real, obstructor_temp: real, obstructor_emissivity: real)
# propagate(rad: SpectralQty): SpectralQty
+ checkConfig(conf: Entry): string
Entry
val: string
+ Entry(kwargs)
+ parse(xml: Element)
+ __call__(): str
+ check_quantity(name: string, unit: Unit): string
+ check_selection(name: string, choices: string[]): string
+ check_file(name: string): string
+ check_path(name:string): string
+ check_float(name:string): string
Configuration
+ conf: Entry
+ Configuration(file: string)
- parser(parent: Element)
- calc_metaoptions()
- calc_metaoption_wl_delta()
- check_config()
- check_optical_components(conf: Entry[]): string
helpers
+ isLambda(obj: object): bool
+ rasterizeCircle(grid: ndarray, radius: real, xc: real, yc: real)
+ readCSV(file: str, units: Unit[], format_: str): Table
+ plot(sqty: spectralQty): void
SpectralQty
+ wl: Quantity
+ qty: Quantity
# fill_value: real
+ SpectralQty(wl: Quantity, qty: Quantity, fill_value: real)
+ fromFile(file: string, wl_unit_default: Quantity, qty_unit_default: Quantity, fill_value: real): SpectralQty
+ rebin(wl: Quantity): SpectralQty
- __eq__(other: SpectralQty): bool
- __add__(other: SpectralQty): SpectralQty
- __sub__(other: SpectralQty): SpectralQty
- __mul__(other: SpectralQty): SpectralQty
- __str__(precision: int): string
- __truediv__(other: SpectralQty): SpectralQty
- __pow__(other: real): Spectral Qty
+ integrate(): u.Quantity
logging
Handler
SpinnerHandler
spinner: Halo
spinning: bool
+ SpinnerHandler(spinner: Halo, level: int)
+ filter(record: Record): bool
+ emit(record: Record): Record
output
+ printSNR(exp_time: Quantity, snr: Quantity): void
+ printExposureTime(exp_time: Quantity, snr: Quantity): void
+ printSensitivity(exp_time: Quantity, snr: Quantity, sensitivity: Quantity): void
numpy
ndarray
PixelMask
+ pixel_geometry: Quantity
+ pixel_size: Quantity
+ center_ind: [real, real]
+ psf_center_ind: [real, real]
+ PixelMask(pixel_geometry: Quantity, pixel_size: Quantity, center_offset: Quantity)
+ __array_finalize__(obj: object)
+ createPhotometricAperture(shape: string, radius: Quantity, center_offset: Quantity)
1
<<Interface>>
IPSF
+ calcReducedObservationAngle(contained_energy: string, jitter_sigma: Quantity, obstruction: real): Quantity
+ mapToPixelMask(mask: PixelMask, jitter_sigma: Quantity, obstruction: real): PixelMask
# rebin(arr: ndarray, factor: real)
Airy
- f_number: real
- wl: Quantity
- d_aperture: Quantity
- osf: real
- pixel_size: Quantity
- psf_jitter: ndarray
+ Airy(f_number: real, wl: Quantity, d_aperture: Quantity, osf: real, pixel_size: Quantity)
+ calcReducedObservationAngle(contained_energy: string, jitter_sigma: Quantity, obstruction: real): Quantity
+ mapToPixelMask(mask: PixelMask, jitter_sigma: Quantity, obstruction: real): PixelMask
- airy(x: ndarray, obstruction: real): ndarray
- airy_int(x: real, obstruction: real): real
AGriddedPSF
- f_number: real
- wl: Quantity
- d_aperture: Quantity
- osf: real
- pixel_size: Quantity
- psf: ndarray
- grid_delta: Quantity
- center_point: int[]
- center_point_os: int[]
- psf_os: ndarray
- psf_osf: real
+ AGriddedPSF(psf: ndarray, f_number: real, wl: Quantity, d_aperture: Quantity, osf: real, pixel_size: Quantity, grid_delta: Quantity, center_point: float[])
+ calcReducedObservationAngle(contained_energy: string, jitter_sigma: Quantity, obstruction: real): Quantity
+ mapToPixelMask(mask: PixelMask, jitter_sigma: Quantity, obstruction: real): PixelMask
- calcPSF(jitter_sigma: Quantity)
Zemax
+ Zemax(file: string, f_number: real, wl: Quantity, d_aperture: Quantity, osf: real, pixel_size: Quantity)
FITS
+ FITS(file: string, f_number: real, wl: Quantity, d_aperture: Quantity, osf: real, pixel_size: Quantity)