FESTIVAL_DATA OBJECT

1/ DESCRIPTION

- The FESTIVAL_DATA object is used to perform the necessary calculation on the raw FITS data in order to get a final projected image.
- It is not a displayable object but a FESTIVAL_VISU_ELEMENT can be build in order to visualize the data (with the CREATE_VISU_ELEMENT method).

- To get the final image, the following steps are necessary :

  a/ FITS image reading and calibration (data -> DATA_PREP)
  The input is a FESTIVAL_SEARCH_RESULT object

  b/ Image rotation (OR NOT) (data -> ROTATE)
  Rotation depends on the required coordinates system

  c/ Calibrated image filtering (OR NOT) (data -> FILTER)

  d/ Filtered image enhancement (data -> ENHANCE)
  At least, a linear stretch (BYTSCL) is applied to the 32 bits filtered image in order to build a 8 bits image.

  e/ Enhanced image projection (OR NOT) (data -> CREATE_PROJECTED_IMAGE)

  f/ Compute the transparency mask projection (data -> BUILD_PROJ_MASK)

  g/ Build a VISU_ELEMENT for visualization (visu_elt = data -> CREATE_VISU_ELEMENT)

- Two data arrays are kept into memory when a FESTIVAL_DATA object is pushed into the HISTORY_STACK :
  - The 32 bits filtered data.
  - The 8 bits projected, enhanced image.

- In case of a data defined as a “difference data” (in order to compute DATA2 – DATA1), the steps are the following (after data creation) :

  a/ Difference on the 32 bits filtered images (data -> MAKE_DIFF(DATA1, DATA2))

  b/ Apply the previous steps e/ f/ and g/

2/ SUPERCLASSES

3/ PROPERTIES

- **_fileSpecification** : full file specification, as the one returned by EIT_CATRD..
- **_date** : image acquisition date.
- **_probe** : probe identifier associated to that image.
- **_instrument** : instrument name associated to that image.
- **_oRefSearchResult** : reference to the corresponding FESTIVAL_SEARCH_RESULT.
- **_pFITSHeader** : pointer to the FITS header structure. NULL pointer is the default.

  From _pFITSHeader, other information like the instrument that produced that image, or the image acquisition date can be derived.
- **_calibrationModel** : structure containing the index of the calibration algorithm used for that image. 0 means no calibration.
- **_pFilteringSequence** : structure containing a pointer to a string array giving the filtering sequence used for that image.
- **_ProjCoord** : structure containing all information concerning the projection used to compute the projected image and mask (projection structure map), flag indicating if the projection is only computed for the polygon coordinates or also for the image). It also contains the index of the coordinate system used.
- **_pCalibratedImage** : pointer to the 32 bits calibrated image. NULL pointer is the default. This property is not saved in the HISTORY_STACK : it is a temporary buffer.
- **_pFilteredImage** : pointer to the 32 bits filtered image. NULL pointer is the default
- **_DeltatOfImageDiff** : Delay between the 2 images used for a difference image (0.0 to indicate that the FilteredImage does not contain a difference).
- **_pEnhancedImage** : pointer to the 8 bits enhanced image. NULL pointer is the default. This property is not saved in the HISTORY_STACK : it is a temporary buffer.
- **_enhancement** : structure containing the index of the enhancement algorithm used for that image, a flag indicating if the autoscale is enabled and the min and max value that the user has specified if the autoscale mode if OFF.
- **_ImageCoordinates** : 4 elements vector [latMin. LonMin, latMax, lonMax] giving the projection limits. Default is [-90, -180, 90, 180].
- **._oProjectedImage** : IDLgrImage defined with ImageData = the 8 bits projected image. If no projection is requested by the user (i.e. when the flag _ProjectionIsDoneOnProjectedImage is FALSE), the projected image contains the enhanced filtered image without projection. The image uses the oPalette defined for the corresponding instrument in the FESTIVAL_COLOR GUI.

- **_UVRange** : range [U min, V min, U max, V max] of the UV coordinates of the projected image pProjectedImage. This will define the position of the polygon FESTIVAL_VISU_ELEMENT used to visualize the oProjectedImage.
- **_pMaskOfNonMissing** : pointer to a byte array containing the mask of the "good" values after a projection was performed. NULL pointer is the default.
- **_pProjectedUserDefinedMaskCoord** : array of the n coordinates of the projected mask defined by the user (in the FESTIVAL_MASKING GUI).
- **_user_mask_parameters** : mask parameters contained in a MASKING_STRUCTURE as defined in DATA_TYPES (the information is : is the mask ON or OFF, the center of the mask, the radius of the mask)
- **_SizeInMemory** : approximated memory used (integer, expressed in Mo)

4/ METHODS

4.1/ FESTIVAL_DATA::INIT

Syntax :

Result = Obj->[FESTIVAL_DATA::]Init (FESTIVAL_SEARCH_RESULT=festivalSearchResult])

Description :

The FESTIVAL_DATA::INIT function method constructs a FESTIVAL_DATA object from a specified festivalSearchResult.

If the keyword FESTIVAL_SEARCH_RESULT is not set, an object is created but it will not be associated to any result of query.

Return Value :

The FESTIVAL_DATA::INIT function method returns 1 if initialization was successful (i.e. if the FITS file that defines the raw data can be found), or 0 otherwise.

Arguments :

None.

Keywords :

festivalSearchResult : the reference to a FESTIVAL_SEARCH_RESULT object

4.2/ FESTIVAL_DATA::DATA_PREP

Syntax :

Result = Obj->[FESTIVAL_DATA::]DATA_PREP (FestivalCalibrationGUI)

Description :

The FESTIVAL_DATA::DATA_PREP method reads the FITS file passed to

FESTIVAL_DATA::INIT method and performs if necessary the calibration of the data.

Return Value :

The FESTIVAL_DATA::DATA_PREP function method 1 if the FITS file could be read, 0 otherwise.

It puts the computed image in the _pCalibratedImage array property which will be used by the FESTIVAL_DATA::ROTATE and FESTIVAL_DATA::FILTER methods. It updates the _calibrationModel property. It updates the _imageCoordinates property.

It stores the FITS header in a property of the FESTIVAL_DATA object.

Arguments :

FestivalCalibrationGUI : the reference to the FESTIVAL_CALIBRATION_GUI object that contains all user defined data concerning the requested calibrations.

Keywords :

None

4.3/ FESTIVAL_DATA::ROTATE

Syntax :

Result = Obj->[FESTIVAL_DATA::]Rotate (FestivallProjCoordGUI)

Description :

The FESTIVAL_DATA::ROTATE method applies (if necessary) the rotation necessary to express the data in the coordinate system defined in the FestivallProjCoordGUI object to the calibrated image _pCalibratedImage, and updates the _PROJCOORD property accordingly.

Return Value :

The FESTIVAL_DATA::DATA_PREP function method 1 if the rotation has been successful, 0 otherwise.

Arguments :

FestivalCalibrationGUI : the reference to the FESTIVAL_CALIBRATION_GUI object that contains all user defined data concerning the requested calibrations.

Keywords :

None

4.4/ FESTIVAL_DATA::FILTER

Syntax :

Result = Obj->[FESTIVAL_DATA::]Filter (FestivalFilterGUI)

Description :

The FESTIVAL_DATA::FILTER function method applies the given filtering sequence to the calibrated image _pCalibratedImage and puts the resulting filtered image in the _pFilteredImage property.

Return Value :

The FESTIVAL_DATA:: FILTER function method returns 1 if the filtering was successful, 0 otherwise.

Arguments :

FestivalFilterGUI

This object describes the filtering sequence to be applied to any calibrated image _pCalibratedImage.

Keywords :

None

4.5/ FESTIVAL_DATA::ENHANCE

Syntax :

Result = Obj->[FESTIVAL_DATA::]Enhance (FestivalEnhancementGUI)

Description :

The FESTIVAL_DATA::ENHANCE function method applies one of the available enhancement algorithm to _pFilteredImage and stores the resulting 8 bits array in _pEnhancedImage. The property _enhancement is also updated.

Return Value :

The FESTIVAL_DATA:: ENHANCE function method returns a pointer to the enhanced image if the enhancement was successful, NULL pointer otherwise.

Arguments :

FestivalEnhancementGUI

The FESTIVAL_ENHANCEMENT_GUI object that specifies the enhancement algorithm to be used on each image. It can be one of :

- 0 : Linear enhancement
- 1 : Square root + linear enhancement
- 2 : Log + linear enhancement

Default is 0 (Linear enhancement).

Keywords :

None

4.6/ FESTIVAL_DATA::CREATE_PROJECTED_IMAGE

Syntax :

Result = Obj->[FESTIVAL_DATA::]Create_Projected_Image (PROJCOORDGUI = ProjcoordGUI, COLORGUI=ColorGUI [,MISSING=missing]

[,NO_PROJECTION_NEEDED=No_Projection_Needed])

Description :

The FESTIVAL_DATA:: Create_Projected_Image method applies a given projection (defined by the MapStructure included in the PROJCOORDGUI object) to the enhanced image _pEnhancedImage.

The resulting projected image is stored inside the IDLgrImage _oProjectedImage property of the FESTIVAL_DATA object. Also, this method initializes the _pUVrange and _pMask properties of the FESTIVAL_DATA object.

The image is associated to the IDLgrPalette defined in the FestivalColorGUI object.

If requested by the user in the PROJCOORDGUI, the projection of the image is not done, but the IDLgrImage _oProjectedImage will be created anyway and UVRange value will be computed even in this case. The property _ProjectionIsDoneOnProjectedImage is updated accordingly.

If the keyword NO_PROJECTION_NEEDED is set, no projection is applied on the image (even if the PROJCOORDGUI object requests to apply a projection). This is used during the process of a difference image creation.

Return Value :

The FESTIVAL_DATA::Create_Projected_Image method returns 1 is the projection was successful, 0 otherwise.

Arguments :

None

Keywords :

PROJCOORDGUI (input, mandatory keyword)

This object FESTIVAL_PROJCOORD_GUI contains in particular the MapStructure that has been created before (with the Map_Proj_Init function).

COLORGUI (input, mandatory argument)

This FESTIVAL_COLOR_GUI object defines the palette to be used for any image.

MISSING

Set this keyword equal to an integer value to be used for pixels that fall outside of the valid map coordinates. The default value is -1.

4.7/ FESTIVAL_DATA::GETUVRANGE

Syntax :

Result = Obj->[FESTIVAL_DATA::]GetUVRange ()

Description :

The FESTIVAL_DATA::GETUVRANGE returns the range of the UV coordinates of the projected image _pProjectedImage.

Return Value :

The FESTIVAL_DATA::GETUVRANGE returns a pointer to the UV coordinates or a NULL pointer.

Arguments :

None

Keywords :

None

4.8/ FESTIVAL_DATA::GETMISSINGMASK

Syntax :

Result = Obj->[FESTIVAL_DATA::]GetMissingMask)

Description :

The FESTIVAL_DATA::GETMISSINGMASK returns the mask of the "good values" to be applied to the projected image _pProjectedImage.

Return Value :

The FESTIVAL_DATA::GETMISSINGMASK returns a pointer to the mask or a NULL pointer.

Arguments :

None

Keywords :

None

4.9/ FESTIVAL_DATA::GETHEADER

Syntax :

Result = Obj->[FESTIVAL_DATA::]GetHeader()

Description :

The FESTIVAL_DATA::GETHEADER function method returns the FITS header associated with the FESTIVAL_DATA object.

Return Value :

The FESTIVAL_DATA::GETHEADER function method returns a pointer the FITS header structure, or NULL pointer.

Arguments :

None

Keywords :

None

4.10/ FESTIVAL_DATA::GETPROPERTY

Syntax :

Result =

Obj>[FESTIVAL_DATA::]GetProperty([,PROBE=probe][,INSTRUMENT=instrument][,IMAGE_INDEX=imageIndex][,ABSOLUTE_INDEX=absoluteIndex][,DATE=date][,FILESPECIFICATION=fileSpecification][,CALIBRATION=calibration][,PROJECTION_NAME=projName][,PROJECTION_LIMITS=projLimits][,FILTERING_SEQUENCE=filteringSequence][,ENHANCEMENT=enhancement][,FITSHEADER=fitsHeader])

Description :

The FESTIVAL_DATA::GETPROPERTY procedure method retrieves the value of a property for the FESTIVAL_DATA object.

Arguments :

None

Keywords :

PROBE

Set this keyword to a named variable that will contain the probe name for that image.

INSTRUMENT

Set this keyword to a named variable that will contain the instrument name for that image.

IMAGE_INDEX

Set this keyword to a named variable that will contain the relative image index.

ABSOLUTE_INDEX

Set this keyword to a named variable that will contain the absolute image index.

DATE

Set this keyword to a named variable that will contain the image acquisition date.

FILESPECIFICATION

Set this keyword to a named variable that will contain the full file specification for that image.

CALIBRATION

Returns the name of the calibration algorithm used to calibrate the raw image. Can be “None” if the image was not calibrated.

PROJECTION_NAME

Returns the name of the projection used. Can be “None” if the calibrated image was not projected.

PROJECTION_LIMITS

Returns a vector [latMin, lonMin, latMax, lonMax] giving the limits of the projection. Can be -1 if the calibrated image was not projected.

FILTERING_SEQUENCE

Returns a string array giving the filtering sequence applied to the projected image _pProjectedImage. Can be “None” if no filtering sequence was applied.

ENHANCEMENT

Returns an integer value giving the enhancement algorithm applied to the filtered image. Can be -1 f no enhancement algorithm was applied.

FITSHEADER

Returns a reference to the fits header pointer property _pFITSHeader of the FESTIVAL_DATA object.

4.11/ FESTIVAL_DATA::BUILD_PROJ_MASK

Syntax :

Result = Obj->[FESTIVAL_DATA::]Build_Proj_Mask (ProjcoordGUI , MaskingGUI)

Description :

The FESTIVAL_DATA:: Build_Proj_Mask method computes the UV coordinates of the mask border that will be used to define the transparent zone in the projected image.

The result is put in the _pProjectedUserDefinedMaskCoord.

In order to do this, the function checks if a mask has been requested by the user in the ProjCoordGUI object. If a user mask has been requested, its UV coordinates are computed. Otherwise, the mask is build using the _pMaskOfNonMissing array.

Return Value :

The FESTIVAL_DATA::BUILD_PROJ_MASK method returns 1 if the build was successful, 0 otherwise.

Arguments :

ProjcoordGUI

Keywords :

TBD.

4.12/ FESTIVAL_DATA::CREATE_VISU_ELEMENT

Syntax :

Result = Obj->[FESTIVAL_DATA::]Create_Visu_Element (TBD)

Description :

The FESTIVAL_DATA:: Create_Visu_Element method creates a new FESTIVAL_VISU_ELEMENT that will allow to visualize the FESTIVAL_DATA object. In order to do this, it takes the existing image _oProjectedImage as texture map of a new IdlgrPolygon which vertices are defined by the _pUVRange property. The position of the mask of transparency (TEXT_COORD) is defined using the _pProjectedUserDefinedMaskCoord property.

Return Value :

The FESTIVAL_DATA::Create_Visu_Element method returns a reference to the new FESTIVAL_VISU_ELEMENT if create was successful, NULL otherwise.

Arguments :

None

Keywords :

TBD.

4.13/ FESTIVAL_DATA::CLEAN_BEFORE_PUSH

Syntax :

Result = Obj->[FESTIVAL_DATA::]Clean_Before_Push (SIZE_OF_DATA=size_of_data)

Description :

The FESTIVAL_DATA:: Clean_Before_Push method is used to destroy all properties that are not saved in the HISTORY_STACK. Therefore, this function shall be called before pushing a data into the history stack.

The properties that will be erased are the _pCalibratedImage and _pEnhancedImage properties.

The property _SizeInMemory is updated.

Return Value :

The FESTIVAL_DATA::Clean_Before_Push method returns 1 if cleaning has been successful, 0 otherwise.

Arguments :

None

Keywords :

SIZE_OF_DATA : a named variable (integer) that will contain the approximated memory used by this object once it has been cleaned (expressed in Mo).

4.14/ FESTIVAL_DATA::MAKE_DIFF

Syntax :

Result = Obj->[FESTIVAL_DATA::]Make_Diff (FESTIVAL_DATA1, FESTIVAL_DATA2, FestivalFilterGUI, FestivalCalibrationGUI, ProjCoordGUI)

Description :

The FESTIVAL_DATA:: Make_Diff method is used to compute the difference FESTIVAL_DATA2 – FESTIVAL_DATA1 (but not the enhanced projection of this difference).

This function supposes that the DATA has already been initialized with Data::Init()

The function verifies that DATA1 and DATA2 have been computed with the requested user options (calibration model, coordinate system, filter)

Almost all properties of this data are set to the value contained by FESTIVAL_DATA2 (date, file specification, refSearchResult, filtering sequence, etc...)

The function computes the difference DATA2 – DATA1 using the _pFilteredImage property of each data and puts the difference in the _pFilteredImage data property.

The property _DeltatOfImageDiff is set to _date(DATA2) – _date (DATA1).

Return Value :

The FESTIVAL_DATA::Make_Diff method returns 1 if the operation is successful, 0 otherwise.

Arguments :

FESTIVAL_DATA1, FESTIVAL_DATA2

The 2 FESTIVAL_DATA objects that will be used in order to compute the difference.

FESTIVAL_FILTER_GUI

This object describes the filtering sequence to be applied to any calibrated image _pCalibratedImage.

FESTIVAL_CALIBRATION_GUI

the reference to the FESTIVAL_CALIBRATION_GUI object that contains all user defined data concerning the requested calibrations.

FESTIVAL_CALIBRATION_GUI

This object FESTIVAL_PROJCOORD_GUI contains in particular the coordinate system requested for the difference.

Keywords :

None

4.15/ FESTIVAL_DATA::COPY_DATA

Syntax :

Result = Obj->[FESTIVAL_DATA::]Copy_data (FESTIVAL_DATA,
WITH_FILTERED_IMAGE=with_Filtered_Image,
WITH_ENHANCED_PROJECTED_IMAGE=with_Enhanced_Projected_Image,
ALL_DATA=all_data)

Description :

The FESTIVAL_DATA:: COPY_DATA function method is used to copy an existing data into a new one.

This function creates and initializes a new DATA. By default, all images and masks are NOT copied. The result is a reference to the new data. The new data is not pushed into the HISTORY_STACK yet.

Return Value :

The FESTIVAL_DATA::COPY_DATA method returns a reference to the created FESTIVAL_DATA if the operation is successful, 0 otherwise.

Arguments :

FESTIVAL_DATA,

The existing FESTIVAL_DATA object that will be copied.

Keywords :

WITH_FILTERED_IMAGE

The property _FilteredImage is also copied.

WITH_ ENHANCED_PROJECTED_IMAGE

The property _oProjectedImage is also copied.

ALL_DATA

All properties are copied (including the projected mask pProjectedUserDefinedMaskCoord which is not projected with the previous keywords).

4.16/ FESTIVAL_DATA::CLEANUP

Syntax :

OBJ_DESTROY, Obj

or

Obj->[FESTIVAL_DATA::]Cleanup   (In a lifecycle method only.)

Description :

The FESTIVAL_DATA::CLEANUP procedure method performs all cleanup on the object.

Arguments :

None

Keywords :

None