

---

# **Gemini AstroData Type Reference**

***Release 1.00***

**Craig Allen**

March 09, 2012



# CONTENTS

<b>1</b>	<b>Introduction</b>	<b>1</b>
1.1	Document Brief . . . . .	1
1.1.1	Revision History . . . . .	1
1.1.2	Document Purpose . . . . .	1
1.1.3	Intended Audience . . . . .	1
1.2	Overview . . . . .	1
<b>2</b>	<b>Gemini Type Graphs</b>	<b>3</b>
2.1	GENERIC . . . . .	3
2.2	GEMINI . . . . .	4
2.3	GMOS . . . . .	4
2.4	GNIRS . . . . .	4
2.5	NICI . . . . .	4
2.6	NIFS . . . . .	5
2.7	NIRI . . . . .	5
2.8	MICHELLE . . . . .	6
2.9	RAW . . . . .	6
2.10	TRECS . . . . .	6
<b>3</b>	<b>Gemini Type Source Reference</b>	<b>7</b>
3.1	ACQUISITION Classification Source . . . . .	7
3.2	BIAS Classification Source . . . . .	7
3.3	CAL Classification Source . . . . .	7
3.4	DARK Classification Source . . . . .	8
3.5	F2 Classification Source . . . . .	8
3.6	F2_CAL Classification Source . . . . .	9
3.7	F2_DARK Classification Source . . . . .	9
3.8	F2_IMAGE Classification Source . . . . .	9
3.9	F2_IMAGE_FLAT Classification Source . . . . .	10
3.10	F2_IMAGE_TWILIGHT Classification Source . . . . .	10
3.11	F2_LS Classification Source . . . . .	10
3.12	F2_LS_ARC Classification Source . . . . .	11
3.13	F2_LS_FLAT Classification Source . . . . .	11
3.14	F2_LS_TWILIGHT Classification Source . . . . .	11
3.15	F2_MOS Classification Source . . . . .	12
3.16	F2_MOS_ARC Classification Source . . . . .	12
3.17	F2_MOS_FLAT Classification Source . . . . .	13
3.18	F2_MOS_TWILIGHT Classification Source . . . . .	13
3.19	F2_SPECT Classification Source . . . . .	13

3.20	FLAT Classification Source . . . . .	14
3.21	FRINGE Classification Source . . . . .	14
3.22	GEMINI Classification Source . . . . .	14
3.23	GEMINI_NORTH Classification Source . . . . .	15
3.24	GEMINI_SOUTH Classification Source . . . . .	15
3.25	GENERIC Classification Source . . . . .	15
3.26	GMOS Classification Source . . . . .	16
3.27	GMOS_BIAS Classification Source . . . . .	16
3.28	GMOS_CAL Classification Source . . . . .	17
3.29	GMOS_DARK Classification Source . . . . .	17
3.30	GMOS_IFU Classification Source . . . . .	17
3.31	GMOS_IFU_ARC Classification Source . . . . .	18
3.32	GMOS_IFU_BLUE Classification Source . . . . .	18
3.33	GMOS_IFU_FLAT Classification Source . . . . .	18
3.34	GMOS_IFU_RED Classification Source . . . . .	19
3.35	GMOS_IFU_TWILIGHT Classification Source . . . . .	19
3.36	GMOS_IFU_TWO Classification Source . . . . .	19
3.37	GMOS_IMAGE Classification Source . . . . .	19
3.38	GMOS_IMAGE_FLAT Classification Source . . . . .	20
3.39	GMOS_IMAGE_TWILIGHT Classification Source . . . . .	20
3.40	GMOS_LS Classification Source . . . . .	21
3.41	GMOS_LS_ARC Classification Source . . . . .	21
3.42	GMOS_LS_FLAT Classification Source . . . . .	21
3.43	GMOS_LS_TWILIGHT Classification Source . . . . .	22
3.44	GMOS_MOS Classification Source . . . . .	22
3.45	GMOS_MOS_ARC Classification Source . . . . .	22
3.46	GMOS_MOS_FLAT Classification Source . . . . .	23
3.47	GMOS_MOS_TWILIGHT Classification Source . . . . .	23
3.48	GMOS_N Classification Source . . . . .	23
3.49	GMOS_NODANDSHUFFLE Classification Source . . . . .	24
3.50	GMOS_RAW Classification Source . . . . .	24
3.51	GMOS_S Classification Source . . . . .	24
3.52	GMOS_SPECT Classification Source . . . . .	24
3.53	GNIRS Classification Source . . . . .	25
3.54	GNIRS_IMAGE Classification Source . . . . .	25
3.55	GNIRS_PINHOLE Classification Source . . . . .	25
3.56	GNIRS_SPECT Classification Source . . . . .	26
3.57	GSAOI Classification Source . . . . .	26
3.58	HOKUPAAQUIRC Classification Source . . . . .	26
3.59	HRWFS Classification Source . . . . .	27
3.60	IFU Classification Source . . . . .	27
3.61	IMAGE Classification Source . . . . .	27
3.62	LS Classification Source . . . . .	28
3.63	MICHELLE Classification Source . . . . .	28
3.64	MICHELLE_IMAGE Classification Source . . . . .	28
3.65	MICHELLE_SPECT Classification Source . . . . .	29
3.66	MOS Classification Source . . . . .	29
3.67	NEEDSFLUXCAL Classification Source . . . . .	29
3.68	NICI Classification Source . . . . .	29
3.69	NICI_ADI_B Classification Source . . . . .	30
3.70	NICI_ADI_R Classification Source . . . . .	30
3.71	NICI_ASDI Classification Source . . . . .	30
3.72	NICI_CAL Classification Source . . . . .	31
3.73	NICI_DARK Classification Source . . . . .	31

3.74	NICI_DARK_CURRENT Classification Source . . . . .	31
3.75	NICI_DARK_OLD Classification Source . . . . .	32
3.76	NICI_FLAT Classification Source . . . . .	32
3.77	NICI_IMAGE Classification Source . . . . .	32
3.78	NICI_SDI Classification Source . . . . .	32
3.79	NIFS Classification Source . . . . .	33
3.80	NIFS_IMAGE Classification Source . . . . .	33
3.81	NIFS_RONCHI Classification Source . . . . .	33
3.82	NIFS_SPECT Classification Source . . . . .	34
3.83	NIRI Classification Source . . . . .	34
3.84	NIRI_IMAGE Classification Source . . . . .	34
3.85	NIRI_SPECT Classification Source . . . . .	34
3.86	NODCHOP Classification Source . . . . .	35
3.87	OSCIR Classification Source . . . . .	35
3.88	PHOENIX Classification Source . . . . .	35
3.89	PREPARED Classification Source . . . . .	36
3.90	PROCESSED_ARC Classification Source . . . . .	36
3.91	PROCESSED_BIAS Classification Source . . . . .	36
3.92	PROCESSED_DARK Classification Source . . . . .	36
3.93	PROCESSED_FLAT Classification Source . . . . .	37
3.94	PROCESSED_FRINGE Classification Source . . . . .	37
3.95	RAW Classification Source . . . . .	37
3.96	SPECT Classification Source . . . . .	38
3.97	TRECS Classification Source . . . . .	38
3.98	TRECS_IMAGE Classification Source . . . . .	38
3.99	TRECS_SPECT Classification Source . . . . .	39
3.100	UNPREPARED Classification Source . . . . .	39



# INTRODUCTION

## 1.1 Document Brief

### 1.1.1 Revision History

- v0.2.(20120310001559) - Revision incremented at 01:41, 5 May 2010 (UTC)

### 1.1.2 Document Purpose

This document is intended as an up to date reference manual for the Gemini AstroData Type library defined within the ADCONFIG\_Gemini configuration package. Where possible the document is created by inspecting the configuration package itself, i.e. to create correct graphs for type trees. This document records the current state of the configuration. For more information on altering it or making your own configuration, see the “AstroData Package Manual”, available at [http://ophiuchus.hi.gemini.edu/ADDOCS/\\_latex\\_build/astrodatadocumentation.pdf](http://ophiuchus.hi.gemini.edu/ADDOCS/_latex_build/astrodatadocumentation.pdf) (**internal to Gemini, external link not yet available**).

### 1.1.3 Intended Audience

This reference is intended for all users of the Gemini AstroData Package, as well as developers maintaining or augmenting type information.

## 1.2 Overview

In order to configure astrodata with the Gemini data types and descriptors, the astrodata package must be able to find the ADCONFIG package in which they are defined. The ADCONFIG packages must be within a subdirectory called “ADCONFIG\_<whatever>”, the Gemini package is “ADCONFIG\_Gemini”. The astrodata package will search the PYTHONPATH for these packages. While they do contain python code, they are not meant to be directly imported, and PYTHONPATH is used to make installation simpler. One can also set ADCONFIGPATH. Note: the PATH environment variable point to the parent directory, which contains the “**ADCONFIG\_...**” subdirectory.





# GEMINI TYPE GRAPHS

Gemini dataset types, or “AstroData Types” include three distinct trees.

- The GEMINI tree which relates to instruments and instrument-modes.
- The RAW tree which is single descendant, and represents processing status. These types are presented mixed with the “typological” types through the `getTypes` interface, but can also be retrieved through the `getStatus` call.
- A generic type tree which relates to the GEMINI tree (i.e. IMAGE is any of the specific INSTR\_IMAGE types).

The graphs below are derived from the ADCONFIG\_Gemini AstroData Configuration Package as of March 09, 2012, and show descriptor and primitive set assignments when present.

## 2.1 GENERIC

The GENERIC type tree relates to abstract data modes which may be used to apply recipes generic to these modes. If generic primitives can be written, these too can be shared, but it is also possible to assign the primitives at instrument-mode specific granularity as well. That is, one can provide generic code at the levels where it makes sense, and type-specific code when that makes more sense or is expedient. We generally visualize an incremental development process where new instrument-modes are first supported in type-specific code where changes to the system are isolated and don’t affect other processing. Subsequently this code can be integrated into, or merely replaced by, more general algorithms that use other AstroData features to abstract away incidental differences between datasets from different devices.

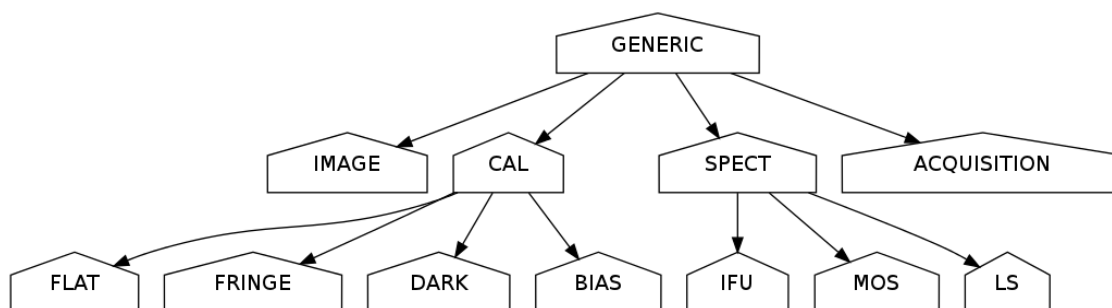


Figure 2.1: The Gemini GENERIC Type Tree

## 2.2 GEMINI

The complete tree of instrument-mode related typological classifications all descend from the GEMINI type, which means of course, the data was from a GEMINI telescope. The figure is difficult to read as all types are present, and will get more so as the instrument trees are filled out. The instrument related graphs are more informative, but this gives an idea of the overall taxonomy.

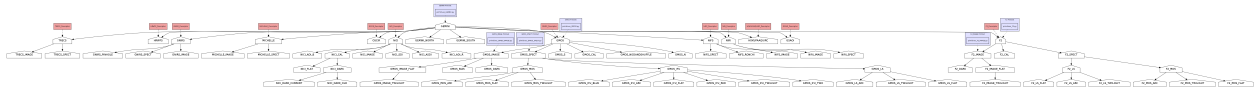


Figure 2.2: The Gemini GEMINI Type Tree

## 2.3 GMOS

GMOS is an optical instrument with an imaging mode, an IFU, and a multi-object spectrograph. We have a complete first revision of the GMOS tree.

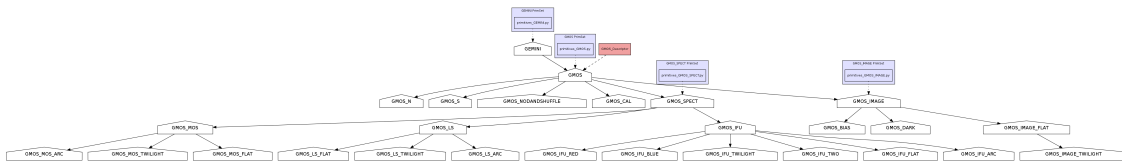


Figure 2.3: The Gemini GMOS Type Tree

## 2.4 GNIRS

GNIRS is a near-infrared spectroscopy instrument currently under repair. The tree is just a stub, which recognizes data from GNIRS.



Figure 2.4: The Gemini GNIRS Type Tree

## 2.5 NICI

NICI is the Near-Infrared Coronagraphic Imager, used at Gemini South. We have a preliminary (development) first revision of the NICI type tree.

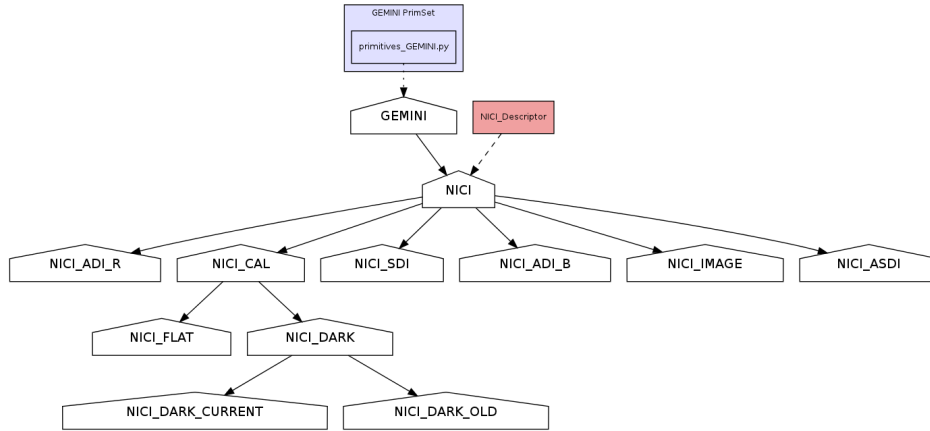


Figure 2.5: The Gemini NICI Type Tree

## 2.6 NIFS

NIFS is a Near-Infrared Integral Field Spectrometer uses at Gemini North. We have a minimal tree in place for NIFS, which recognizes IMAGE and SPECT types.

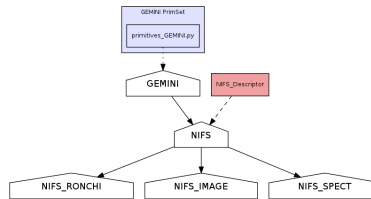


Figure 2.6: The Gemini NIFS Type Tree

## 2.7 NIRI

NIRI is a Near-Infrared Imager in use at Gemini North. We have a minimal tree in place for NIRI, which recognizes IMAGE and SPECT types.

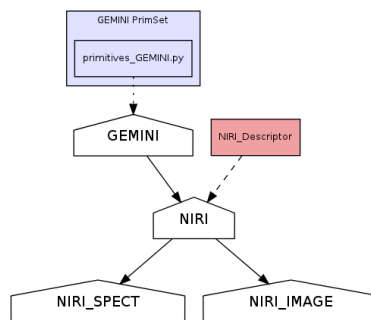


Figure 2.7: The Gemini NIRI Type Tree

## 2.8 MICHELLE

MICHELLE is a mid-infrared image and sepcrometer. We have a minimal tree in place for MICHELLE, which recognizes IMAGE and SPECT types.

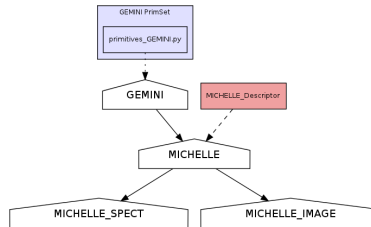


Figure 2.8: The Gemini MICHELLE Type Tree

## 2.9 RAW

The RAW tree contains inherent sequencing, what are show as children are new forms of the data... that is, some transformation(s) will make the data recognized only as the child. These types can be used to check the state of processing, and can have entirely generic recipes associated, as may be needed by some pipelines.

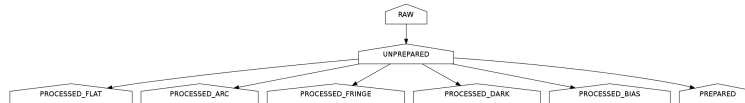


Figure 2.9: The Gemini RAW Type Tree

## 2.10 TRECS

TRECS is a Thermal-Region Camera Spectograph, a mid-infrared imager and long-slit spectrograph built for Gemini South. We have a minimal tree in place for TRECS, which recognizes IMAGE and SPECT.

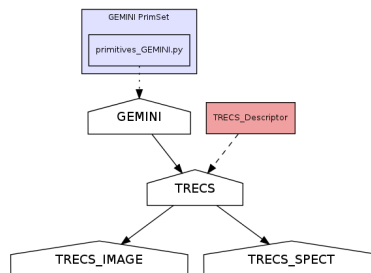


Figure 2.10: The Gemini TRECS Type Tree

# GEMINI TYPE SOURCE REFERENCE

## 3.1 ACQUISITION Classification Source

**Classification** ACQUISITION

**Source Location** ADCONFIG\_Gemini/classifications/types/gemdtype.ACQUISITION.py

```
1 class ACQUISITION(DataClassification):
2     name="ACQUISITION"
3     usage = """
4         Applies to all Gemini acquisitions
5         """
6     parent = "GENERIC"
7     requirement = OR([ PHU(OBSCLASS="acq"),
8                       PHU(OBSCLASS="acqCal") ])
9
10 newtypes.append(ACQUISITION())
```

## 3.2 BIAS Classification Source

**Classification** BIAS

**Source Location** ADCONFIG\_Gemini/classifications/types/generic/gemdtype.BIAS.py

```
1 class BIAS(DataClassification):
2     name="BIAS"
3     # this a description of the intent of the classification
4     # to what does the classification apply?
5     usage = '''
6         Applies to any Gemini dataset which is an instrument bias calibration.'''
7     parent = "CAL"
8     requirement = ISCLASS("GMOS_BIAS")
9
10 newtypes.append( BIAS())
```

## 3.3 CAL Classification Source

**Classification** CAL

**Source Location** ADCONFIG\_Gemini/classifications/types/gemdtype.CAL.py

```
1 class CAL(DataClassification):
2     name="CAL"
3     # this a description of the intent of the classification
4     # to what does the classification apply?
5     usage = """
6         Special parent to group generic types (e.g. IMAGE, SPECT, MOS, IFU)
7         """
8     parent = "GENERIC"
9     requirement = OR([ ISCLASS("F2_CAL"),
10                        ISCLASS("GMOS_CAL"),
11                        ISCLASS("NICI_CAL") ])
12
13 newtypes.append(CAL())
```

## 3.4 DARK Classification Source

### Classification DARK

**Source Location** ADCONFIG\_Gemini/classifications/types/generic/gemdtype.DARK.py

```
1 class DARK(DataClassification):
2     name="DARK"
3     # this a description of the intent of the classification
4     # to what does the classification apply?
5     usage = '''
6         Applies to any dataset that is a Gemini dark current calibration.
7         '''
8     parent = "CAL"
9     requirement = OR(ISCLASS("NICI_DARK"),
10                     ISCLASS("GMOS_DARK"))
11
12 newtypes.append( DARK())
```

## 3.5 F2 Classification Source

### Classification F2

**Source Location** ADCONFIG\_Gemini/classifications/types/F2/gemdtype.F2.py

```
1 class F2(DataClassification):
2     name="F2"
3     usage = """
4         Applies to all datasets from the FLAMINGOS-2 instrument
5         """
6     parent = "GEMINI"
7     # Commissioning data from 28 August 2009 to 20 February 2010 use "Flam" as
8     # the value for the INSTRUME keyword. The final value for the INSTRUME
9     # keyword will be "F2".
10    requirement = OR([ PHU(INSTRUME="Flam"),
11                      PHU(INSTRUME="F2") ])
12
13 newtypes.append(F2())
```

## 3.6 F2\_CAL Classification Source

**Classification** F2\_CAL

**Source Location** ADCONFIG\_Gemini/classifications/types/F2/gemdtype.F2\_CAL.py

```

1 class F2_CAL(DataClassification):
2     name="F2_CAL"
3     usage = """
4         Applies to all calibration datasets from the FLAMINGOS-2 instrument
5         """
6     parent = "F2"
7     requirement = ISCLASS("F2") & OR([ ISCLASS("F2_IMAGE_FLAT"),
8                                         ISCLASS("F2_IMAGE_TWILIGHT"),
9                                         ISCLASS("F2_DARK"),
10                                        ISCLASS("F2_LS_FLAT"),
11                                        ISCLASS("F2_LS_TWILIGHT"),
12                                        ISCLASS("F2_LS_ARC"),
13                                        ISCLASS("F2_MOS_FLAT"),
14                                        ISCLASS("F2_MOS_TWILIGHT"),
15                                        ISCLASS("F2_MOS_ARC") ])
16
17 newtypes.append(F2_CAL())

```

## 3.7 F2\_DARK Classification Source

**Classification** F2\_DARK

**Source Location** ADCONFIG\_Gemini/classifications/types/F2/gemdtype.F2\_DARK.py

```

1 class F2_DARK(DataClassification):
2     name="F2_DARK"
3     usage = """
4         Applies to all dark datasets from the FLAMINGOS-2 instrument
5         """
6     parent = "F2_IMAGE"
7     requirement = AND([ ISCLASS("F2_IMAGE"),
8                         PHU(OBSTYPE="DARK") ])
9
10 newtypes.append(F2_DARK())

```

## 3.8 F2\_IMAGE Classification Source

**Classification** F2\_IMAGE

**Source Location** ADCONFIG\_Gemini/classifications/types/F2/gemdtype.F2\_IMAGE.py

```

1 class F2_IMAGE(DataClassification):
2     name="F2_IMAGE"
3     usage = """
4         Applies to all imaging datasets from the FLAMINGOS-2 instrument
5         """
6     parent = "F2"
7     # Commissioning data from 28 August 2009 to 20 February 2010 use the

```

```
8      # MASKNAME keyword to specify whether the data is imaging, longslit or
9      # mos. The final keyword to use will be DCKERPOS or MOSPOS.
10     requirement = ISCLASS("F2") & OR([ PHU(MASKNAME="imaging"),
11                                         PHU(DECKER="Open"),
12                                         PHU(MOSPOS="Open")  ])
13
14 newtypes.append(F2_IMAGE())
```

## 3.9 F2\_IMAGE\_FLAT Classification Source

**Classification** F2\_IMAGE\_FLAT

**Source Location** ADCONFIG\_Gemini/classifications/types/F2/gemdtype.F2\_IMAGE\_FLAT.py

```
1 class F2_IMAGE_FLAT(DataClassification):
2     name="F2_IMAGE_FLAT"
3     usage = """
4         Applies to all imaging flat datasets from the FLAMINGOS-2 instrument
5         """
6     parent = "F2_IMAGE"
7     requirement = AND([ ISCLASS("F2_IMAGE"),
8                         OR([ PHU(OBSTYPE="FLAT"),
9                             OR([ PHU(OBJECT="Twilight"),
10                                 PHU(OBJECT="twilight")  ])  ])  ])
11
12 newtypes.append(F2_IMAGE_FLAT())
```

## 3.10 F2\_IMAGE\_TWILIGHT Classification Source

**Classification** F2\_IMAGE\_TWILIGHT

**Source Location** ADCONFIG\_Gemini/classifications/types/F2/gemdtype.F2\_IMAGE\_TWILIGHT.py

```
1 class F2_IMAGE_TWILIGHT(DataClassification):
2     name="F2_IMAGE_TWILIGHT"
3     usage = """
4         Applies to all imaging twilight flat datasets from the FLAMINGOS-2
5         instrument
6         """
7     parent = "F2_IMAGE_FLAT"
8     requirement = AND([ ISCLASS("F2_IMAGE_FLAT"),
9                         OR([ PHU(OBJECT="Twilight"),
10                             PHU(OBJECT="twilight")  ])  ])
11
12 newtypes.append(F2_IMAGE_TWILIGHT())
```

## 3.11 F2\_LS Classification Source

**Classification** F2\_LS

**Source Location** ADCONFIG\_Gemini/classifications/types/F2/gemdtype.F2\_LS.py



```

1 class F2_LS(DataClassification):
2     name="F2_LS"
3     usage = """
4         Applies to all longslit datasets from the FLAMINGOS-2 instrument
5         """
6     parent = "F2_SPECT"
7     requirement = AND([ ISCLASS("F2_SPECT"),
8                         OR([ PHU(DECKER="Long_slit"),
9                             PHU(DCKERPOS="Long_slit"),
10                             PHU(MOSPOS="?.pix-slit") ] ) ] )
11
12 newtypes.append(F2_LS())

```

## 3.12 F2\_LS\_ARC Classification Source

**Classification** F2\_LS\_ARC

**Source Location** ADCONFIG\_Gemini/classifications/types/F2/gemdtype.F2\_LS\_ARC.py

```

1 class F2_LS_ARC(DataClassification):
2     name="F2_LS_ARC"
3     usage = """
4         Applies to all longslit arc datasets from the FLAMINGOS-2 instrument
5         """
6     parent = "F2_LS"
7     requirement = AND([ ISCLASS("F2_LS"),
8                         PHU(OBSTYPE="ARC") ] )
9
10 newtypes.append(F2_LS_ARC())

```

## 3.13 F2\_LS\_FLAT Classification Source

**Classification** F2\_LS\_FLAT

**Source Location** ADCONFIG\_Gemini/classifications/types/F2/gemdtype.F2\_LS\_FLAT.py

```

1 class F2_LS_FLAT(DataClassification):
2     name="F2_LS_FLAT"
3     usage = """
4         Applies to all longslit flat datasets from the FLAMINGOS-2 instrument
5         """
6     parent = "F2_LS"
7     requirement = AND([ ISCLASS("F2_LS"),
8                         PHU(OBSTYPE="FLAT"),
9                         NOT(ISCLASS("F2_LS_TWILIGHT")) ] )
10
11 newtypes.append(F2_LS_FLAT())

```

## 3.14 F2\_LS\_TWILIGHT Classification Source

**Classification** F2\_LS\_TWILIGHT

**Source Location** ADCONFIG\_Gemini/classifications/types/F2/gemdtype.F2\_LS\_TWILIGHT.py

```
1 class F2_LS_TWILIGHT(DataClassification):
2     name="F2_LS_TWILIGHT"
3     usage = """
4         Applies to all longslit twilight flat datasets from the FLAMINGOS-2
5         instrument
6         """
7     parent = "F2_LS"
8     requirement = AND([ ISCLASS("F2_LS"),
9                         PHU(OBSTYPE="FLAT"),
10                        PHU(OBJECT="Twilight") ])
11
12 newtypes.append(F2_LS_TWILIGHT())
```

## 3.15 F2\_MOS Classification Source

**Classification** F2\_MOS

**Source Location** ADCONFIG\_Gemini/classifications/types/F2/gemdtype.F2\_MOS.py

```
1 class F2_MOS(DataClassification):
2     name="F2_MOS"
3     usage = """
4         Applies to all MOS datasets from the FLAMINGOS-2 instrument
5         """
6     parent = "F2_SPECT"
7     requirement = AND ([ ISCLASS("F2_SPECT"),
8                         PHU(OBSTYPE="OBJECT"),
9                         OR([ PHU(DECKER="mos"),
10                            PHU(DCKERPOS="mos"),
11                            PHU(MOSPOS="mos.?.") ]) ])
12
13 newtypes.append(F2_MOS())
```

## 3.16 F2\_MOS\_ARC Classification Source

**Classification** F2\_MOS\_ARC

**Source Location** ADCONFIG\_Gemini/classifications/types/F2/gemdtype.F2\_MOS\_ARC.py

```
1 class F2_MOS_ARC(DataClassification):
2     name="F2_MOS_ARC"
3     usage = """
4         Applies to all MOS arc datasets from the FLAMINGOS-2 instrument
5         """
6     parent = "F2_MOS"
7     requirement = AND([ ISCLASS("F2_MOS"),
8                         PHU(OBSTYPE="ARC") ])
9
10 newtypes.append(F2_MOS_ARC())
```

## 3.17 F2\_MOS\_FLAT Classification Source

**Classification** F2\_MOS\_FLAT

**Source Location** ADCONFIG\_Gemini/classifications/types/F2/gemdtype.F2\_MOS\_FLAT.py

```

1  class F2_MOS_FLAT(DataClassification):
2      name="F2_MOS_FLAT"
3      usage = """
4          Applies to all MOS flat datasets from the FLAMINGOS-2 instrument
5          """
6      parent = "F2_MOS"
7      requirement = AND([ ISCLASS("F2_MOS"),
8                          PHU(OBSTYPE="FLAT"),
9                          NOT(ISCLASS("F2_MOS_TWILIGHT"))  ])
10
11  newtypes.append(F2_MOS_FLAT())

```

## 3.18 F2\_MOS\_TWILIGHT Classification Source

**Classification** F2\_MOS\_TWILIGHT

**Source Location** ADCONFIG\_Gemini/classifications/types/F2/gemdtype.F2\_MOS\_TWILIGHT.py

```

1  class F2_MOS_TWILIGHT(DataClassification):
2      name="F2_MOS_TWILIGHT"
3      usage = """
4          Applies to all MOS twilight flat datasets from the FLAMINGOS-2
5          instrument
6          """
7      parent = "F2_MOS"
8      requirement = AND([ ISCLASS("F2_MOS"),
9                          PHU(OBSTYPE="FLAT"),
10                         PHU(OBJECT="Twilight")  ])
11
12  newtypes.append(F2_MOS_TWILIGHT())

```

## 3.19 F2\_SPECT Classification Source

**Classification** F2\_SPECT

**Source Location** ADCONFIG\_Gemini/classifications/types/F2/gemdtype.F2\_SPECT.py

```

1  class F2_SPECT(DataClassification):
2      name="F2_SPECT"
3      # this a description of the intent of the classification
4      # to what does the classification apply?
5      usage = """
6          Applies to all spectroscopic datasets from the FLAMINGOS-2 instrument
7          """
8      parent = "F2"
9      requirement = ISCLASS("F2") & OR([ PHU(DCKERPOS="Long_slit"),
10                                         PHU(MOSPOS="?.pix-slit"),
11                                         PHU(DCKERPOS="mos"),

```

```
12         PHU(MOSPOS="mos.?" ) ] )
13
14 newtypes.append(F2_SPECT())
```

## 3.20 FLAT Classification Source

### Classification FLAT

**Source Location** ADCONFIG\_Gemini/classifications/types/gemdtype.FLAT.py

```
1 class FLAT(DataClassification):
2     name="FLAT"
3     # this a description of the intent of the classification
4     # to what does the classification apply?
5     usage = '''
6         Applies to all Gemini FLATS.
7     '''
8     parent = "CAL"
9     requirement = OR( ISCLASS("GMOS_FLAT"),
10                      ISCLASS("NICI_FLAT"))
11
12 newtypes.append( FLAT())
```

## 3.21 FRINGE Classification Source

### Classification FRINGE

**Source Location** ADCONFIG\_Gemini/classifications/types/gemdtype.FRINGE.py

```
1 class FRINGE(DataClassification):
2     name="FRINGE"
3     usage = "A processed fringe."
4     parent = "CAL"
5     requirement = PHU(GIFRINGE='(.*)')
6
7 newtypes.append(FRINGE())
```

## 3.22 GEMINI Classification Source

### Classification GEMINI

**Source Location** ADCONFIG\_Gemini/classifications/types/gemdtype.GEMINI.py

```
1 class GEMINI(DataClassification):
2     name="GEMINI"
3     # this a description of the intent of the classification
4     # to what does the classification apply?
5     usage = '''
6         Applies to all data from either GMOS-North or GMOS-South instruments in any mode.
7     '''
8     # Added the instrument names directly, so that when we get engineering data that does
9     # not have telescope headers in, and thus doesn't identify as GEMINI_NORTH or _SOUTH
```

```

10     # then it does identify as GEMINI, so that the gemini descriptors associate with it.
11     requirement = OR(ISCLASS("GEMINI_NORTH"),
12                     ISCLASS("GEMINI_SOUTH"),
13                     ISCLASS("GMOS"),
14                     ISCLASS("NIRI"),
15                     ISCLASS("GNIRS"),
16                     ISCLASS("MICHELLE"),
17                     ISCLASS("NICI"),
18                     ISCLASS("F2"),
19                     ISCLASS("NIFS"),
20                     ISCLASS("TRECS"),
21                     ISCLASS("GSAOI"))
22
23
24 newtypes.append( GEMINI() )

```

### 3.23 GEMINI\_NORTH Classification Source

**Classification** GEMINI\_NORTH

**Source Location** ADCONFIG\_Gemini/classifications/types/gemdtype.GEMINI\_NORTH.py

```

1 class GEMINI_NORTH(DataClassification):
2     name="GEMINI_NORTH"
3     usage = "Data taken at Gemini North upon Mauna Kea."
4
5     parent = "GEMINI"
6     requirement = PHU({'TELESCOP': 'Gemini-North', 'OBSERVAT': 'Gemini-North'})
7
8 newtypes.append(GEMINI_NORTH())

```

### 3.24 GEMINI\_SOUTH Classification Source

**Classification** GEMINI\_SOUTH

**Source Location** ADCONFIG\_Gemini/classifications/types/gemdtype.GEMINI\_SOUTH.py

```

1 class GEMINI_SOUTH(DataClassification):
2     name="GEMINI_SOUTH"
3     usage = "Applies to datasets from instruments at Gemini South."
4
5     parent = "GEMINI"
6     requirement = PHU(TELESCOP='Gemini-South',OBSERVAT='Gemini-South')
7
8 newtypes.append(GEMINI_SOUTH())

```

### 3.25 GENERIC Classification Source

**Classification** GENERIC

**Source Location** ADCONFIG\_Gemini/classifications/types/gemdtype.GENERIC.py

```
1 class GENERIC(DataClassification):
2     name="GENERIC"
3     # this a description of the intent of the classification
4     # to what does the classification apply?
5     usage = """
6         Special parent to group generic types (e.g. IMAGE, SPECT, MOS, IFU)
7         """
8     parent = None
9     requirement = False # no type is "GENERIC"
10
11 newtypes.append(GENERIC())
```

## 3.26 GMOS Classification Source

**Classification** GMOS

**Source Location** ADCONFIG\_Gemini/classifications/types/GMOS/gemdtype.GMOS.py

```
1 class GMOS(DataClassification):
2     name="GMOS"
3     # this a description of the intent of the classification
4     # to what does the classification apply?
5     usage = '''
6         Applies to all data from either GMOS-North or GMOS-South instruments in any mode.
7         '''
8
9     parent = "GEMINI"
10    requirement = ISCLASS("GMOS_N") | ISCLASS("GMOS_S")
11    # equivalent to...
12    #requirement = OR(
13    #                                ClassReq("GMOS_N"),
14    #                                ClassReq("GMOS_S")
15    #                                )
16
17    newtypes.append( GMOS())
```

## 3.27 GMOS\_BIAS Classification Source

**Classification** GMOS\_BIAS

**Source Location** ADCONFIG\_Gemini/classifications/types/GMOS/gemdtype.GMOS\_BIAS.py

```
1 class GMOS_BIAS(DataClassification):
2     name="GMOS_BIAS"
3     usage = """
4         Applies to all dark datasets from the GMOS instruments
5         """
6     parent = "GMOS_IMAGE"
7     requirement = PHU(OBSTYPE="BIAS")
8
9    newtypes.append(GMOS_BIAS())
```

## 3.28 GMOS\_CAL Classification Source

**Classification** GMOS\_CAL

**Source Location** ADCONFIG\_Gemini/classifications/types/GMOS/gemdtype.GMOS\_CAL.py

```

1  class GMOS_CAL(DataClassification):
2      name="GMOS_CAL"
3      usage = """
4          Applies to all calibration datasets from the GMOS instruments
5          """
6      parent = "GMOS"
7      requirement = ISCLASS("GMOS") & OR([ ISCLASS("GMOS_IMAGE_FLAT"),
8                                           ISCLASS("GMOS_IMAGE_TWILIGHT"),
9                                           ISCLASS("GMOS_BIAS"),
10                                          ISCLASS("GMOS_LS_FLAT"),
11                                          ISCLASS("GMOS_LS_TWILIGHT"),
12                                          ISCLASS("GMOS_LS_ARC"),
13                                          ISCLASS("GMOS_MOS_FLAT"),
14                                          ISCLASS("GMOS_MOS_TWILIGHT"),
15                                          ISCLASS("GMOS_MOS_ARC"),
16                                          ISCLASS("GMOS_IFU_FLAT"),
17                                          ISCLASS("GMOS_IFU_TWILIGHT"),
18                                          ISCLASS("GMOS_IFU_ARC") ])
19
20  newtypes.append(GMOS_CAL())

```

## 3.29 GMOS\_DARK Classification Source

**Classification** GMOS\_DARK

**Source Location** ADCONFIG\_Gemini/classifications/types/GMOS/gemdtype.GMOS\_DARK.py

```

1  class GMOS_DARK(DataClassification):
2      name="GMOS_DARK"
3      usage = ""
4      parent = "GMOS_IMAGE"
5      requirement = ISCLASS('GMOS') & PHU( OBSTYPE = 'DARK')
6
7  newtypes.append(GMOS_DARK())

```

## 3.30 GMOS\_IFU Classification Source

**Classification** GMOS\_IFU

**Source Location** ADCONFIG\_Gemini/classifications/types/GMOS/gemdtype.GMOS\_IFU.py

```

1  class GMOS_IFU(DataClassification):
2      name="GMOS_IFU"
3      usage = """
4          Data taken in the IFU instrument mode with either GMOS instrument
5          """
6      parent = "GMOS_SPECT"
7      requirement = AND([ ISCLASS("GMOS_SPECT"),

```

```
8             PHU(MASKTYP="-1"),
9             PHU(MASKNAME="IFU*")    ] )
10
11 newtypes.append(GMOS_IFU())
```

### 3.31 GMOS\_IFU\_ARC Classification Source

**Classification** GMOS\_IFU\_ARC

**Source Location** ADCONFIG\_Gemini/classifications/types/GMOS/gemdtype.GMOS\_IFU\_ARC.py

```
1 class GMOS_IFU_ARC(DataClassification):
2     name="GMOS_IFU_ARC"
3     usage = """
4         Applies to all IFU arc datasets from the GMOS instruments
5         """
6     parent = "GMOS_IFU"
7     requirement = AND([ ISCLASS("GMOS_IFU"),
8                         PHU(OBSTYPE="ARC")    ] )
9
10 newtypes.append(GMOS_IFU_ARC())
```

### 3.32 GMOS\_IFU\_BLUE Classification Source

**Classification** GMOS\_IFU\_BLUE

**Source Location** ADCONFIG\_Gemini/classifications/types/GMOS/gemdtype.GMOS\_IFU\_BLUE.py

```
1 class GMOS_IFU_BLUE(DataClassification):
2     name="GMOS_IFU_BLUE"
3     usage = ""
4     parent = "GMOS_IFU"
5     requirement = ISCLASS('GMOS_IFU') & PHU(MASKNAME='(IFU-B) | (IFU-B-NS)')
6
7 newtypes.append(GMOS_IFU_BLUE())
```

### 3.33 GMOS\_IFU\_FLAT Classification Source

**Classification** GMOS\_IFU\_FLAT

**Source Location** ADCONFIG\_Gemini/classifications/types/GMOS/gemdtype.GMOS\_IFU\_FLAT.py

```
1 class GMOS_IFU_FLAT(DataClassification):
2     name="GMOS_IFU_FLAT"
3     usage = """
4         Applies to all IFU flat datasets from the GMOS instruments
5         """
6     parent = "GMOS_IFU"
7     requirement = AND([ ISCLASS("GMOS_IFU"),
8                         PHU(OBSTYPE="FLAT"),
9                         NOT(ISCLASS("GMOS_IFU_TWILIGHT"))    ] )
10
11 newtypes.append(GMOS_IFU_FLAT())
```



### 3.34 GMOS\_IFU\_RED Classification Source

**Classification** GMOS\_IFU\_RED

**Source Location** ADCONFIG\_Gemini/classifications/types/GMOS/gemdtype.GMOS\_IFU\_RED.py

```

1 class GMOS_IFU_RED(DataClassification):
2     name="GMOS_IFU_RED"
3     usage = ""
4     parent = "GMOS_IFU"
5     requirement = ISCLASS('GMOS_IFU') & PHU(MASKNAME='(IFU-R)|(IFU-R-NS)')
6
7 newtypes.append(GMOS_IFU_RED())

```

### 3.35 GMOS\_IFU\_TWILIGHT Classification Source

**Classification** GMOS\_IFU\_TWILIGHT

**Source Location** ADCONFIG\_Gemini/classifications/types/GMOS/gemdtype.GMOS\_IFU\_TWILIGHT.py

```

1 class GMOS_IFU_TWILIGHT(DataClassification):
2     name="GMOS_IFU_TWILIGHT"
3     usage = ""
4     """
5     Applies to all IFU twilight flat datasets from the GMOS instruments
6     """
7     parent = "GMOS_IFU"
8     requirement = AND([ ISCLASS("GMOS_IFU"),
9                        PHU(OBSTYPE="FLAT"),
10                       PHU(OBJECT="Twilight") ])
11
12 newtypes.append(GMOS_IFU_TWILIGHT())

```

### 3.36 GMOS\_IFU\_TWO Classification Source

**Classification** GMOS\_IFU\_TWO

**Source Location** ADCONFIG\_Gemini/classifications/types/GMOS/gemdtype.GMOS\_IFU\_TWO.py

```

1 class GMOS_IFU_TWO(DataClassification):
2     name="GMOS_IFU_TWO"
3     usage = ""
4     parent = "GMOS_IFU"
5     requirement = ISCLASS('GMOS_IFU') & PHU(MASKNAME='(IFU-2)|(IFU-2-NS)')
6
7 newtypes.append(GMOS_IFU_TWO())

```

### 3.37 GMOS\_IMAGE Classification Source

**Classification** GMOS\_IMAGE

**Source Location** ADCONFIG\_Gemini/classifications/types/GMOS/gemdtype.GMOS\_IMAGE.py

```
1 class GMOS_IMAGE(DataClassification):
2     name="GMOS_IMAGE"
3     usage = """
4         Applies to all imaging datasets from the GMOS instruments
5         """
6     parent = "GMOS"
7     requirement = AND([ ISCLASS("GMOS"),
8                         PHU(GRATING="MIRROR"),
9                         NOT(ISCLASS("GMOS_BIAS")) ])
10
11 newtypes.append(GMOS_IMAGE())
```

### 3.38 GMOS\_IMAGE\_FLAT Classification Source

**Classification** GMOS\_IMAGE\_FLAT

**Source Location** ADCONFIG\_Gemini/classifications/types/GMOS/gemdtype.GMOS\_IMAGE\_FLAT.py

```
1 class GMOS_IMAGE_FLAT(DataClassification):
2     name="GMOS_IMAGE_FLAT"
3     usage = """
4         Applies to all imaging flat datasets from the GMOS instruments
5         """
6     parent = "GMOS_IMAGE"
7     requirement = AND(NOT(PHU({"re}FILTER.*?": "Hartmann.*?"})),
8                       OR(AND([ ISCLASS("GMOS_IMAGE"),
9                               PHU(OBSTYPE="FLAT") ]),
10                          AND([ ISCLASS("GMOS_IMAGE"),
11                              OR([PHU(OBJECT="Twilight"),
12                                  PHU(OBJECT="twilight")] ])))))
13
14 newtypes.append(GMOS_IMAGE_FLAT())
```

### 3.39 GMOS\_IMAGE\_TWILIGHT Classification Source

**Classification** GMOS\_IMAGE\_TWILIGHT

**Source Location** ADCONFIG\_Gemini/classifications/types/GMOS/gemdtype.GMOS\_IMAGE\_TWILIGHT.py

```
1 class GMOS_IMAGE_TWILIGHT(DataClassification):
2     name="GMOS_IMAGE_TWILIGHT"
3     usage = """
4         Applies to all imaging twilight flat datasets from the GMOS instruments
5         """
6     parent = "GMOS_IMAGE_FLAT"
7     requirement = AND([ ISCLASS("GMOS_IMAGE_FLAT"),
8                       OR([PHU(OBJECT="Twilight"),
9                           PHU(OBJECT="twilight")] ]))
10
11 newtypes.append(GMOS_IMAGE_TWILIGHT())
```

## 3.40 GMOS\_LS Classification Source

**Classification** GMOS\_LS

**Source Location** ADCONFIG\_Gemini/classifications/types/GMOS/gemdtype.GMOS\_LS.py

```

1  class GMOS_LS(DataClassification):
2      name="GMOS_LS"
3      usage = """
4          Applies to all longslit datasets from the GMOS instruments
5          """
6      parent = "GMOS_SPECT"
7      requirement = AND([ ISCLASS("GMOS_SPECT"),
8                          PHU(MASKTYP="1"),
9                          PHU(MASKNAME=".*arcsec")  ])
10
11  newtypes.append(GMOS_LS())

```

## 3.41 GMOS\_LS\_ARC Classification Source

**Classification** GMOS\_LS\_ARC

**Source Location** ADCONFIG\_Gemini/classifications/types/GMOS/gemdtype.GMOS\_LS\_ARC.py

```

1  class GMOS_LS_ARC(DataClassification):
2      name="GMOS_LS_ARC"
3      usage = """
4          Applies to all longslit arc datasets from the GMOS instruments
5          """
6      parent = "GMOS_LS"
7      requirement = AND([ ISCLASS("GMOS_LS"),
8                          PHU(OBSTYPE="ARC")  ])
9
10  newtypes.append(GMOS_LS_ARC())

```

## 3.42 GMOS\_LS\_FLAT Classification Source

**Classification** GMOS\_LS\_FLAT

**Source Location** ADCONFIG\_Gemini/classifications/types/GMOS/gemdtype.GMOS\_LS\_FLAT.py

```

1  class GMOS_LS_FLAT(DataClassification):
2      name="GMOS_LS_FLAT"
3      usage = """
4          Applies to all longslit flat datasets from the GMOS instruments
5          """
6      parent = "GMOS_LS"
7      requirement = AND([ ISCLASS("GMOS_LS"),
8                          PHU(OBSTYPE="FLAT"),
9                          NOT(ISCLASS("GMOS_LS_TWILIGHT"))  ])
10
11  newtypes.append(GMOS_LS_FLAT())

```

### 3.43 GMOS\_LS\_TWILIGHT Classification Source

**Classification** GMOS\_LS\_TWILIGHT

**Source Location** ADCONFIG\_Gemini/classifications/types/GMOS/gemdtype.GMOS\_LS\_TWILIGHT.py

```
1 class GMOS_LS_TWILIGHT(DataClassification):
2     name="GMOS_LS_TWILIGHT"
3     usage = """
4         Applies to all longslit twilight flat datasets from the GMOS
5         instruments
6         """
7     parent = "GMOS_LS"
8     requirement = AND([ ISCLASS("GMOS_LS"),
9                         PHU(OBSTYPE="FLAT"),
10                        PHU(OBJECT="Twilight") ])
11
12 newtypes.append(GMOS_LS_TWILIGHT())
```

### 3.44 GMOS\_MOS Classification Source

**Classification** GMOS\_MOS

**Source Location** ADCONFIG\_Gemini/classifications/types/GMOS/gemdtype.GMOS\_MOS.py

```
1 class GMOS_MOS(DataClassification):
2     name="GMOS_MOS"
3     usage = """
4         Applies to all MOS datasets from the GMOS instruments
5         """
6     parent = "GMOS_SPECT"
7     requirement = AND([ ISCLASS("GMOS_SPECT"),
8                         PHU(MASKTYP="1"),
9                         PHU({"{prohibit}MASKNAME": ".*arcsec"}),
10                        PHU({"{prohibit}MASKNAME": "IFU*"})) ])
11
12 newtypes.append(GMOS_MOS())
```

### 3.45 GMOS\_MOS\_ARC Classification Source

**Classification** GMOS\_MOS\_ARC

**Source Location** ADCONFIG\_Gemini/classifications/types/GMOS/gemdtype.GMOS\_MOS\_ARC.py

```
1 class GMOS_MOS_ARC(DataClassification):
2     name="GMOS_MOS_ARC"
3     usage = """
4         Applies to all MOS arc datasets from the GMOS instruments
5         """
6     parent = "GMOS_MOS"
7     requirement = AND([ ISCLASS("GMOS_MOS"),
8                         PHU(OBSTYPE="ARC") ])
9
10 newtypes.append(GMOS_MOS_ARC())
```

## 3.46 GMOS\_MOS\_FLAT Classification Source

**Classification** GMOS\_MOS\_FLAT

**Source Location** ADCONFIG\_Gemini/classifications/types/GMOS/gemdtype.GMOS\_MOS\_FLAT.py

```

1  class GMOS_MOS_FLAT(DataClassification):
2      name="GMOS_MOS_FLAT"
3      usage = """
4          Applies to all MOS flat datasets from the GMOS instruments
5          """
6      parent = "GMOS_MOS"
7      requirement = AND([ ISCLASS("GMOS_MOS"),
8                          PHU(OBSTYPE="FLAT"),
9                          NOT(ISCLASS("GMOS_MOS_TWILIGHT")) ])
10
11  newtypes.append(GMOS_MOS_FLAT())

```

## 3.47 GMOS\_MOS\_TWILIGHT Classification Source

**Classification** GMOS\_MOS\_TWILIGHT

**Source Location** ADCONFIG\_Gemini/classifications/types/GMOS/gemdtype.GMOS\_MOS\_TWILIGHT.py

```

1  class GMOS_MOS_TWILIGHT(DataClassification):
2      name="GMOS_MOS_TWILIGHT"
3      usage = """
4          Applies to all MOS twilight flat datasets from the GMOS instruments
5          """
6      parent = "GMOS_MOS"
7      requirement = AND([ ISCLASS("GMOS_MOS"),
8                          PHU(OBSTYPE="FLAT"),
9                          PHU(OBJECT="Twilight") ])
10
11  newtypes.append(GMOS_MOS_TWILIGHT())

```

## 3.48 GMOS\_N Classification Source

**Classification** GMOS\_N

**Source Location** ADCONFIG\_Gemini/classifications/types/GMOS/gemdtype.GMOS\_N.py

```

1  class GMOS_N(DataClassification):
2      name="GMOS_N"
3      usage = ""
4      typeReqs= []
5      phuReqs= {}
6      parent = "GMOS"
7      requirement = PHU(INSTRUME='GMOS-N')
8
9  newtypes.append(GMOS_N())

```

## 3.49 GMOS\_NODANDSHUFFLE Classification Source

**Classification** GMOS\_NODANDSHUFFLE

**Source Location** ADCONFIG\_Gemini/classifications/types/GMOS/gemdtype.GMOS\_NODANDSHUFFLE.py

```
1 class GMOS_NODANDSHUFFLE(DataClassification):
2     name="GMOS_NODANDSHUFFLE"
3     usage = ""
4     typeReqs= []
5     phuReqs= {}
6     parent = "GMOS"
7     requirement = PHU(NODPIX='.*')
8
9 newtypes.append(GMOS_NODANDSHUFFLE())
```

## 3.50 GMOS\_RAW Classification Source

**Classification** GMOS\_RAW

**Source Location** ADCONFIG\_Gemini/classifications/status/gemdtype.GMOS\_RAW.py

```
1 class GMOS_RAW(DataClassification):
2     editprotect=True
3     name="GMOS_RAW"
4     usage = 'Applies to RAW GMOS data.'
5     typeReqs= ['GMOS']
6     requirement = ISCLASS("RAW") & ISCLASS("GMOS")
7
8 newtypes.append(GMOS_RAW())
```

## 3.51 GMOS\_S Classification Source

**Classification** GMOS\_S

**Source Location** ADCONFIG\_Gemini/classifications/types/GMOS/gemdtype.GMOS\_S.py

```
1 class GMOS_S(DataClassification):
2     name="GMOS_S"
3     usage = "For data from GMOS South"
4
5     parent = "GMOS"
6     requirement = PHU(INSTRUME='GMOS-S')
7
8 newtypes.append(GMOS_S())
```

## 3.52 GMOS\_SPECT Classification Source

**Classification** GMOS\_SPECT

**Source Location** ADCONFIG\_Gemini/classifications/types/GMOS/gemdtype.GMOS\_SPECT.py

```

1  class GMOS_SPECT(DataClassification):
2      name="GMOS_SPECT"
3      # this a description of the intent of the classification
4      # to what does the classification apply?
5      usage = """
6          Applies to all spectroscopic datasets from the GMOS instruments
7          """
8      parent = "GMOS"
9      requirement = AND([ ISCLASS("GMOS"),
10                          PHU({"{prohibit}MASKTYP": "0"}),
11                          PHU({"{prohibit}MASKNAME": "None"}),
12                          PHU({"{prohibit}GRATING": "MIRROR"}),
13                          NOT(ISCLASS("GMOS_BIAS")) ])
14
15  newtypes.append(GMOS_SPECT())

```

### 3.53 GNIRS Classification Source

**Classification** GNIRS

**Source Location** ADCONFIG\_Gemini/classifications/types/GNIRS/gemdtype.GNIRS.py

```

1  class GNIRS(DataClassification):
2      name="GNIRS"
3      usage = "Applies to all datasets from the GNIRS instrument."
4      parent = "GEMINI"
5      requirement = PHU(INSTRUME='GNIRS')
6
7  newtypes.append(GNIRS())

```

### 3.54 GNIRS\_IMAGE Classification Source

**Classification** GNIRS\_IMAGE

**Source Location** ADCONFIG\_Gemini/classifications/types/GNIRS/gemdtype.GNIRS\_IMAGE.py

```

1  class GNIRS_IMAGE(DataClassification):
2      name="GNIRS_IMAGE"
3      usage = "Applies to any IMAGE dataset from the GNIRS instrument."
4      parent = "GNIRS"
5      requirement = ISCLASS('GNIRS') & PHU(ACQMIR='In')
6
7  newtypes.append(GNIRS_IMAGE())

```

### 3.55 GNIRS\_PINHOLE Classification Source

**Classification** GNIRS\_PINHOLE

**Source Location** ADCONFIG\_Gemini/classifications/types/GNIRS/gemdtype.GNIRS\_PINHOLE.py

```

1  class GNIRS_PINHOLE(DataClassification):
2      name="GNIRS_PINHOLE"

```

```
3     usage = "Applies to GNIRS Pinhole mask calibration observations"
4     parent = "GNIRS"
5     requirement = AND(ISCLASS('GNIRS'), PHU(OBSTYPE='FLAT'), OR( PHU(SLIT='LgPinholes_G5530'), PHU(SLIT='LgPinholes_G5530')))
6
7 newtypes.append(GNIRS_PINHOLE())
```

## 3.56 GNIRS\_SPECT Classification Source

**Classification** GNIRS\_SPECT

**Source Location** ADCONFIG\_Gemini/classifications/types/GNIRS/gemdtype.GNIRS\_SPECT.py

```
1 class GNIRS_SPECT(DataClassification):
2     name="GNIRS_SPECT"
3     usage = "Applies to any SPECT dataset from the GNIRS instrument."
4     parent = "GNIRS"
5     requirement = ISCLASS('GNIRS') & PHU(ACQMIR='Out')
6
7 newtypes.append(GNIRS_SPECT())
```

## 3.57 GSAOI Classification Source

**Classification** GSAOI

**Source Location** ADCONFIG\_Gemini/classifications/types/GSAOI/gemdtype.GSAOI.py

```
1 class GSAOI(DataClassification):
2     name="GSAOI"
3     usage = "Applies to any data from the GSAOI instrument."
4     parent = "GEMINI"
5
6     requirement = PHU(INSTRUME='GSAOI')
7
8 newtypes.append(GSAOI())
```

## 3.58 HOKUPAAQUIRC Classification Source

**Classification** HOKUPAAQUIRC

**Source Location** ADCONFIG\_Gemini/classifications/types/QUIRC/gemdtype.QUIRC.py

```
1 class HOKUPAAQUIRC(DataClassification):
2     name="HOKUPAAQUIRC"
3     usage = "Applies to datasets from the HOKUPAA+QUIRC instrument"
4     parent = "GEMINI"
5     requirement = PHU(INSTRUME='Hokupaa\+QUIRC')
6
7 newtypes.append(HOKUPAAQUIRC())
```



## 3.59 HRWFS Classification Source

**Classification** HRWFS

**Source Location** ADCONFIG\_Gemini/classifications/types/gemdtype.HRWFS.py

```

1 class HRWFS(DataClassification):
2     name="HRWFS"
3     usage = "Applies to all datasets from the HRWFS instrument."
4     parent = "GEMINI"
5     requirement = PHU(INSTRUME='hrwfs')
6
7 newtypes.append(HRWFS())

```

## 3.60 IFU Classification Source

**Classification** IFU

**Source Location** ADCONFIG\_Gemini/classifications/types/generic/gemdtype.IFU.py

```

1 class IFU(DataClassification):
2     name="IFU"
3     # this a description of the intent of the classification
4     # to what does the classification apply?
5     usage = '''
6         Applies to all Gemini IFU data.
7     '''
8     parent = "SPECT"
9     requirement = ISCLASS("GMOS_IFU")
10
11 newtypes.append( IFU())

```

## 3.61 IMAGE Classification Source

**Classification** IMAGE

**Source Location** ADCONFIG\_Gemini/classifications/types/gemdtype.IMAGE.py

```

1 class IMAGE(DataClassification):
2     name="IMAGE"
3     # this a description of the intent of the classification
4     # to what does the classification apply?
5     usage = """
6         Applies to all Gemini imaging datasets
7     """
8     parent = "GENERIC"
9     requirement = OR([ ISCLASS("F2_IMAGE"),
10                        ISCLASS("GMOS_IMAGE"),
11                        ISCLASS("GNIRS_IMAGE"),
12                        ISCLASS("MICHELLE_IMAGE"),
13                        ISCLASS("NICI_IMAGE"),
14                        ISCLASS("NIFS_IMAGE"),
15                        ISCLASS("NIRI_IMAGE"),
16                        ISCLASS("TRECS_IMAGE")])

```

```
17
18 newtypes.append(IMAGE())
```

## 3.62 LS Classification Source

**Classification** LS

**Source Location** ADCONFIG\_Gemini/classifications/types/generic/gemdtype.LS.py

```
1 class LS(DataClassification):
2     name="LS"
3     # this a description of the intent of the classification
4     # to what does the classification apply?
5     usage = '''
6         Applies to all long slit spectral datasets.
7     '''
8     parent = "SPECT"
9     requirement= ISCLASS("GMOS_MOS")
10
11 newtypes.append(LS())
```

## 3.63 MICHELLE Classification Source

**Classification** MICHELLE

**Source Location** ADCONFIG\_Gemini/classifications/types/MICHELLE/gemdtype.MICHELLE.py

```
1 class MICHELLE(DataClassification):
2     name = "MICHELLE"
3     usage = "Applies to all datasets from the MICHELLE instrument"
4     parent = "GEMINI"
5     requirement = PHU(INSTRUME="michelle")
6
7 newtypes.append(MICHELLE())
```

## 3.64 MICHELLE\_IMAGE Classification Source

**Classification** MICHELLE\_IMAGE

**Source Location** ADCONFIG\_Gemini/classifications/types/MICHELLE/gemdtype.MICHELLE\_IMAGE.py

```
1 class MICHELLE_IMAGE(DataClassification):
2     name = "MICHELLE_IMAGE"
3     usage = "Applies to all imaging datasets from the MICHELLE instrument"
4     parent = "MICHELLE"
5     requirement = ISCLASS("MICHELLE") & PHU(CAMERA="imaging")
6
7 newtypes.append(MICHELLE_IMAGE())
```

## 3.65 MICHELLE\_SPECT Classification Source

**Classification** MICHELLE\_SPECT

**Source Location** ADCONFIG\_Gemini/classifications/types/MICHELLE/gemdtype.MICHELLE\_SPECT.py

```

1 class MICHELLE_SPECT(DataClassification):
2     name = "MICHELLE_SPECT"
3     usage = """
4         Applies to all spectroscopic datasets from the MICHELLE instrument
5         """
6     parent = "MICHELLE"
7     requirement = ISCLASS("MICHELLE") & PHU(CAMERA="spectroscopy")
8
9 newtypes.append(MICHELLE_SPECT())

```

## 3.66 MOS Classification Source

**Classification** MOS

**Source Location** ADCONFIG\_Gemini/classifications/types/gemdtype.MOS.py

```

1 class MOS(DataClassification):
2     name="MOS"
3     # this a description of the intent of the classification
4     # to what does the classification apply?
5     usage = """
6         Applies to all MOS data which conformed to the required Gemini Generic
7         MOS Standard
8         """
9     parent = "SPECT"
10    requirement = OR([ ISCLASS("F2_MOS"),
11                      ISCLASS("GMOS_MOS"), ])
12
13 newtypes.append(MOS())

```

## 3.67 NEEDSFLUXCAL Classification Source

**Classification** NEEDSFLUXCAL

**Source Location** ADCONFIG\_Gemini/classifications/status/gemdtype.NEEDSFLUXCAL.py

```

1 class NEEDSFLUXCAL(DataClassification):
2     editprotect=False
3     name="NEEDSFLUXCAL"
4     usage = 'Applies to data ready for flux calibration.'
5     requirement = ISCLASS("IMAGE") & ISCLASS("PREPARED")
6
7 newtypes.append(NEEDSFLUXCAL())

```

## 3.68 NICI Classification Source

**Classification** NICI

**Source Location** ADCONFIG\_Gemini/classifications/types/NICI/gemdtype.NICI.py

```
1 class NICI(DataClassification):
2     name="NICI"
3     usage = "Applies to all datasets taken with the NICI instrument."
4     parent = "GEMINI"
5     requirement = PHU(INSTRUME='NICI')
6
7 newtypes.append(NICI())
```

## 3.69 NICI\_ADI\_B Classification Source

**Classification** NICI\_ADI\_B

**Source Location** ADCONFIG\_Gemini/classifications/types/NICI/gemdtype.NICI\_ADI\_B.py

```
1 class NICI_ADI_B(DataClassification):
2     name="NICI_ADI_B"
3     usage = "Applies to imaging datasets from the NICI instrument."
4     parent = "NICI"
5     # DICHROIC PHU keyword value contains the string 'Mirror'
6     requirement = ISCLASS('NICI') & PHU( {'re'}.*?DICHROIC': ".*?Mirror*" })
7
8 newtypes.append(NICI_ADI_B())
```

## 3.70 NICI\_ADI\_R Classification Source

**Classification** NICI\_ADI\_R

**Source Location** ADCONFIG\_Gemini/classifications/types/NICI/gemdtype.NICI\_ADI\_R.py

```
1 class NICI_ADI_R(DataClassification):
2     name="NICI_ADI_R"
3     usage = "Applies to imaging datasets from the NICI instrument."
4     parent = "NICI"
5     # DICHROIC PHU keyword value contains the string 'Open'
6     requirement = ISCLASS('NICI') & PHU( {'re'}.*?DICHROIC': ".*?Open*" })
7
8 newtypes.append(NICI_ADI_R())
```

## 3.71 NICI\_ASDI Classification Source

**Classification** NICI\_ASDI

**Source Location** ADCONFIG\_Gemini/classifications/types/NICI/gemdtype.NICI\_ASDI.py

```
1 class NICI_ASDI(DataClassification):
2     name="NICI_ASDI"
3     usage = "Applies to imaging datasets from the NICI instrument."
4     parent = "NICI"
5     # DICHROIC PHU keyword value contains the string '50/50'
6     requirement = ISCLASS('NICI') & PHU( {'re'}.*?DICHROIC': ".*?50/50.*" }) & \
7         PHU(CRMODE='FIXED') & PHU({'prohibit'}OBSTYPE': 'FLAT' )
```

```

8
9 newtypes.append(NICI_ASDI())

```

## 3.72 NICI\_CAL Classification Source

**Classification** NICI\_CAL

**Source Location** ADCONFIG\_Gemini/classifications/types/NICI/gemdtype.NICI\_CAL.py

```

1 class NICI_CAL(DataClassification):
2     name="NICI_CAL"
3     # this a description of the intent of the classification
4     # to what does the classification apply?
5     usage = '''
6         Applies to all NICI flats
7     '''
8     parent = "NICI"
9     requirement = ISCLASS('NICI_FLAT') | ISCLASS('NICI_DARK')
10
11 newtypes.append( NICI_CAL())

```

## 3.73 NICI\_DARK Classification Source

**Classification** NICI\_DARK

**Source Location** ADCONFIG\_Gemini/classifications/types/NICI/gemdtype.NICI\_DARK.py

```

1 class NICI_DARK(DataClassification):
2     name="NICI_DARK"
3     # this a description of the intent of the classification
4     # to what does the classification apply?
5     usage = '''
6         Applies to all dark current calibration datasets for NICI instrument.
7     '''
8     parent = "NICI_CAL"
9     requirement = ISCLASS("NICI_DARK_CURRENT", "NICI_DARK_OLD")
10
11 newtypes.append( NICI_DARK())

```

## 3.74 NICI\_DARK\_CURRENT Classification Source

**Classification** NICI\_DARK\_CURRENT

**Source Location** ADCONFIG\_Gemini/classifications/types/NICI/gemdtype.NICI\_DARK\_CURRENT.py

```

1 class NICI_DARK_CURRENT(DataClassification):
2     name="NICI_DARK_CURRENT"
3     usage = "Applies to current dark current calibrations for the NICI instrument."
4     parent = "NICI_DARK"
5     requirement = ISCLASS('NICI') & PHU(OBSTYPE='DARK')
6
7 newtypes.append(NICI_DARK_CURRENT())

```

## 3.75 NICI\_DARK\_OLD Classification Source

**Classification** NICI\_DARK\_OLD

**Source Location** ADCONFIG\_Gemini/classifications/types/NICI/gemdtype.NICI\_DARK\_OLD.py

```
1 class NICI_DARK_OLD(DataClassification):
2     name="NICI_DARK_OLD"
3     usage = "Applies to OLD NICI dark current calibration datasets."
4     parent = "NICI_DARK"
5     requirement = ISCLASS('NICI') & PHU(OBSTYPE='FLAT',
6                                         GCALSHUT='CLOSED')
7
8 newtypes.append(NICI_DARK_OLD())
```

## 3.76 NICI\_FLAT Classification Source

**Classification** NICI\_FLAT

**Source Location** ADCONFIG\_Gemini/classifications/types/NICI/gemdtype.NICI\_FLAT.py

```
1 class NICI_FLAT(DataClassification):
2     name="NICI_FLAT"
3     # this a description of the intent of the classification
4     # to what does the classification apply?
5     usage = '''
6         Applies to all NICI flats
7     '''
8     parent = "NICI_CAL"
9     requirement = ISCLASS('NICI') & PHU(OBSTYPE='FLAT')
10
11 newtypes.append( NICI_FLAT())
```

## 3.77 NICI\_IMAGE Classification Source

**Classification** NICI\_IMAGE

**Source Location** ADCONFIG\_Gemini/classifications/types/NICI/gemdtype.NICI\_IMAGE.py

```
1 class NICI_IMAGE(DataClassification):
2     name="NICI_IMAGE"
3     usage = "Applies to imaging datasts from the NICI instrument."
4     parent = "NICI"
5     requirement = ISCLASS('NICI') & PHU(INSTRUME='NICI')
6
7 newtypes.append(NICI_IMAGE())
```

## 3.78 NICI\_SDI Classification Source

**Classification** NICI\_SDI

**Source Location** ADCONFIG\_Gemini/classifications/types/NICI/gemdtype.NICI\_SDI.py

```

1 class NICI_SDI(DataClassification):
2     name="NICI_SDI"
3     usage = "Applies to imaging datasets from the NICI instrument."
4     parent = "NICI"
5     # DICHROIC PHU keyword value contains the string '50/50'
6     requirement = ISCLASS('NICI') & PHU( {'re'}.*?DICHROIC': ".*?50/50.*?" }) & \
7         PHU(CRMODE='FOLLOW') & PHU({'prohibit'}OBSTYPE:'FLAT')
8
9 newtypes.append(NICI_SDI())

```

## 3.79 NIFS Classification Source

**Classification** NIFS

**Source Location** ADCONFIG\_Gemini/classifications/types/NIFS/gemdtype.NIFS.py

```

1 class NIFS(DataClassification):
2     name="NIFS"
3     usage = "Applies to datasets from NIFS instrument"
4     parent = "GEMINI"
5     requirement = PHU(INSTRUME='NIFS')
6
7 newtypes.append(NIFS())

```

## 3.80 NIFS\_IMAGE Classification Source

**Classification** NIFS\_IMAGE

**Source Location** ADCONFIG\_Gemini/classifications/types/NIFS/gemdtype.NIFS\_IMAGE.py

```

1 class NIFS_IMAGE(DataClassification):
2     name="NIFS_IMAGE"
3     usage = "Applies to any image dataset from the NIFS instrument."
4     parent = "NIFS"
5
6     requirement = ISCLASS("NIFS") & PHU( FLIP='In')
7
8 newtypes.append(NIFS_IMAGE())

```

## 3.81 NIFS\_RONCHI Classification Source

**Classification** NIFS\_RONCHI

**Source Location** ADCONFIG\_Gemini/classifications/types/NIFS/gemdtype.NIFS\_RONCHI.py

```

1 class NIFS_RONCHI(DataClassification):
2     name="NIFS_RONCHI"
3     usage = "Applies to NIFS Ronchi mask calibration observations"
4     parent = "NIFS"
5     requirement = AND(ISCLASS('NIFS'), PHU(OBSTYPE='FLAT'), PHU(APERTURE='Ronchi_Screen_G5615'))
6
7 newtypes.append(NIFS_RONCHI())

```

## 3.82 NIFS\_SPECT Classification Source

**Classification** NIFS\_SPECT

**Source Location** ADCONFIG\_Gemini/classifications/types/NIFS/gemdtype.NIFS\_SPECT.py

```
1 class NIFS_SPECT(DataClassification):
2     name="NIFS_SPECT"
3     usage = "Applies to any spectroscopy dataset from the NIFS instrument."
4     parent = "NIFS"
5
6     requirement = ISCLASS("NIFS") & PHU( FLIP='Out' )
7
8 newtypes.append(NIFS_SPECT())
```

## 3.83 NIRI Classification Source

**Classification** NIRI

**Source Location** ADCONFIG\_Gemini/classifications/types/NIRI/gemdtype.NIRI.py

```
1 class NIRI(DataClassification):
2     name="NIRI"
3     usage = "Applies to any data from the NIRI instrument."
4     parent = "GEMINI"
5
6     requirement = PHU(INSTRUME='NIRI')
7
8 newtypes.append(NIRI())
```

## 3.84 NIRI\_IMAGE Classification Source

**Classification** NIRI\_IMAGE

**Source Location** ADCONFIG\_Gemini/classifications/types/NIRI/gemdtype.NIRI\_IMAGE.py

```
1 class NIRI_IMAGE(DataClassification):
2     name="NIRI_IMAGE"
3     usage = "Applies to any IMAGE dataset from the NIRI instrument."
4     parent = "NIRI"
5     requirement = ISCLASS('NIRI') & PHU({"{prohibit}FILTER3": "(.*)grism(.*)"})
6
7
8 newtypes.append(NIRI_IMAGE())
```

## 3.85 NIRI\_SPECT Classification Source

**Classification** NIRI\_SPECT

**Source Location** ADCONFIG\_Gemini/classifications/types/NIRI/gemdtype.NIRI\_SPECT.py



```

1 class NIRI_SPECT(DataClassification):
2     name="NIRI_SPECT"
3     usage = "Applies to any spectra from the NIRI instrument."
4     parent = "NIRI"
5     requirement = ISCLASS('NIRI') & PHU(FILTER3='(.*?)grism(.*?)')
6
7 newtypes.append(NIRI_SPECT())

```

## 3.86 NODCHOP Classification Source

**Classification** NODCHOP

**Source Location** ADCONFIG\_Gemini/classifications/types/gemdtype.NODCHOP.py

```

1 class NODCHOP(DataClassification):
2     name="NODCHOP"
3     # this a description of the intent of the classification
4     # to what does the classification apply?
5     usage = '''
6         Applies to data marked with NOD and CHOP keywords.
7         TEST TYPE!!!
8         Made to test Structures Feature.
9     '''
10    requirement = PHU(DATATYPE="marked-nodandchop")
11
12 newtypes.append( NODCHOP() )

```

## 3.87 OSCIR Classification Source

**Classification** OSCIR

**Source Location** ADCONFIG\_Gemini/classifications/types/OSCIR/gemdtype.OSCIR.py

```

1 class OSCIR(DataClassification):
2     name="OSCIR"
3     usage = "Applies to datasets from the OSCIR instrument"
4     parent = "GEMINI"
5     requirement = OR(PHU(INSTRUME='oscir'), PHU(INSTRUME='OSCIR'))
6
7 newtypes.append(OSCIR())

```

## 3.88 PHOENIX Classification Source

**Classification** PHOENIX

**Source Location** ADCONFIG\_Gemini/classifications/types/PHOENIX/gemdtype.PHOENIX.py

```

1 class PHOENIX(DataClassification):
2     name="PHOENIX"
3     usage = ""
4     requirement = PHU(INSTRUME='PHOENIX')
5
6 newtypes.append(PHOENIX())

```

## 3.89 PREPARED Classification Source

**Classification** PREPARED

**Source Location** ADCONFIG\_Gemini/classifications/status/gemdtype.PREPARED.py

```
1 class PREPARED(DataClassification):
2
3     name="PREPARED"
4     usage = 'Applies to all "prepared" data.'
5     parent = "UNPREPARED"
6     requirement = PHU( {'{re}.*?PREPARE': ".*?" })
7
8 newtypes.append(PREPARED())
```

## 3.90 PROCESSED\_ARC Classification Source

**Classification** PROCESSED\_ARC

**Source Location** ADCONFIG\_Gemini/classifications/status/gemdtype.PROCESSED\_ARC.py

```
1 class PROCESSED_ARC(DataClassification):
2
3     name="PROCESSED_ARC"
4     usage = 'Applies to all data processed by storeProcessedArc.'
5     parent = "UNPREPARED"
6     requirement = PHU( {'{re}.*?PROCARC': ".*?" })
7
8 newtypes.append(PROCESSED_ARC())
```

## 3.91 PROCESSED\_BIAS Classification Source

**Classification** PROCESSED\_BIAS

**Source Location** ADCONFIG\_Gemini/classifications/status/gemdtype.PROCESSED\_BIAS.py

```
1 class PROCESSED_BIAS(DataClassification):
2
3     name="PROCESSED_BIAS"
4     usage = 'Applies to all "gbias"ed data.'
5     parent = "UNPREPARED"
6     requirement = OR([PHU( {'{re}.*?GBIAS': ".*?" }),
7                        PHU( {'{re}.*?PROCBIAS': ".*?" })])
8
9 newtypes.append(PROCESSED_BIAS())
```

## 3.92 PROCESSED\_DARK Classification Source

**Classification** PROCESSED\_DARK

**Source Location** ADCONFIG\_Gemini/classifications/status/gemdtype.PROCESSED\_DARK.py

```

1 class PROCESSED_DARK(DataClassification):
2
3     name="PROCESSED_DARK"
4     usage = 'Applies to all dark data stored using storeProcessedDark.'
5     parent = "UNPREPARED"
6     requirement = PHU( {'{re}.*?PROCDDARK': ".*?" })
7
8 newtypes.append(PROCESSED_DARK())

```

### 3.93 PROCESSED\_FLAT Classification Source

**Classification** PROCESSED\_FLAT

**Source Location** ADCONFIG\_Gemini/classifications/status/gemdtype.PROCESSED\_FLAT.py

```

1 class PROCESSED_FLAT(DataClassification):
2
3     name="PROCESSED_FLAT"
4     usage = 'Applies to all "giflat"ed flat data, or data stored using storeProcessedFlat.'
5     parent = "UNPREPARED"
6     requirement = OR([PHU( {'{re}.*?GIFLAT': ".*?" }),
7                          PHU( {'{re}.*?PROCFLAT': ".*?" })])
8
9 newtypes.append(PROCESSED_FLAT())

```

### 3.94 PROCESSED\_FRINGE Classification Source

**Classification** PROCESSED\_FRINGE

**Source Location** ADCONFIG\_Gemini/classifications/status/gemdtype.PROCESSED\_FRINGE.py

```

1 class PROCESSED_FRINGE(DataClassification):
2
3     name="PROCESSED_FRINGE"
4     usage = 'Applies to all "gifringe"ed data.'
5     parent = "UNPREPARED"
6     requirement = OR([PHU( {'{re}.*?GIFFRINGE': ".*?" }),
7                          PHU( {'{re}.*?PROCFRNG': ".*?" })])
8
9 newtypes.append(PROCESSED_FRINGE())

```

### 3.95 RAW Classification Source

**Classification** RAW

**Source Location** ADCONFIG\_Gemini/classifications/status/gemdtype.RAW.py

```

1 class RAW(DataClassification):
2     editprotect=True
3     name="RAW"
4     usage = 'Applies to data that has not been modified by the gemini package in any way (looks for C'
5     requirement = PHU({'{prohibit}GEM-TLM': ".*?" })

```

```
6
7 newtypes.append(RAW())
```

## 3.96 SPECT Classification Source

**Classification** SPECT

**Source Location** ADCONFIG\_Gemini/classifications/types/gemdtype.SPECT.py

```
1 class SPECT(DataClassification):
2     name="SPECT"
3     # this a description of the intent of the classification
4     # to what does the classification apply?
5     usage = """
6         Applies to all Gemini spectroscopy datasets
7         """
8     parent = "GENERIC"
9     requirement = OR([ ISCLASS("F2_SPECT"),
10                        ISCLASS("GMOS_SPECT"),
11                        ISCLASS("GNIRS_SPECT"),
12                        ISCLASS("MICHELLE_SPECT"),
13                        ISCLASS("NIFS_SPECT"),
14                        ISCLASS("NIRI_SPECT"),
15                        ISCLASS("TRECS_SPECT") ])
16
17 newtypes.append(SPECT())
```

## 3.97 TRECS Classification Source

**Classification** TRECS

**Source Location** ADCONFIG\_Gemini/classifications/types/TRECS/gemdtype.TRECS.py

```
1 class TRECS(DataClassification):
2     name = "TRECS"
3     usage = "Applies to all datasets from the TRECS instrument"
4     parent = "GEMINI"
5     requirement = PHU(INSTRUME="TReCS")
6
7 newtypes.append(TRECS())
```

## 3.98 TRECS\_IMAGE Classification Source

**Classification** TRECS\_IMAGE

**Source Location** ADCONFIG\_Gemini/classifications/types/TRECS/gemdtype.TRECS\_IMAGE.py

```
1 class TRECS_IMAGE(DataClassification):
2     name = "TRECS_IMAGE"
3     usage = "Applies to all imaging datasets from the TRECS instrument"
4     parent = "TRECS"
5     requirement = ISCLASS("TRECS") & PHU(GRATING="(.*?)[mM]irror")
```

```

6
7 newtypes.append(TRECS_IMAGE())

```

### 3.99 TRECS\_SPECT Classification Source

**Classification** TRECS\_SPECT

**Source Location** ADCONFIG\_Gemini/classifications/types/TRECS/gemdtype.TRECS\_SPECT.py

```

1 class TRECS_SPECT(DataClassification):
2     name = "TRECS_SPECT"
3     usage = "Applies to all spectroscopic datasets from the TRECS instrument"
4     parent = "TRECS"
5     requirement = ISCLASS("TRECS") & NOT(ISCLASS("TRECS_IMAGE"))
6
7 newtypes.append(TRECS_SPECT())

```

### 3.100 UNPREPARED Classification Source

**Classification** UNPREPARED

**Source Location** ADCONFIG\_Gemini/classifications/status/gemdtype.UNPREPARED.py

```

1 class UNPREPARED(DataClassification):
2     editprotect=True
3     name="UNPREPARED"
4     usage = 'Applies to un-"prepared" datasets, datasets which have not had the prepare task run on t
5     parent = "RAW"
6     requirement= PHU({'prohibit,re'.*?PREPARE': ".*?" })
7
8 newtypes.append(UNPREPARED())

```

The following documentation lists current Gemini types and shows their source files in raw python form. When types are defined in the same source, their section is combined. Entries are alphabetized.