How to document a Task

Tasks should be documented in the .py file, adding an entry to the doxygen group LSST_Task_Documentation group, *e.g.*

```
## \addtogroup LSST_task_documentation
## \{
## \page sourceDetectionTask
## \ref SourceDetectionTask_ "SourceDetectionTask"
## \copybrief SourceDetectionTask
## \}
```

It is acceptable to make documentation only changes on master. If you feel the need to clean things up you may do it as part of the documentation, but as a short-lived branch.

There are a couple of tricks:

- Remember that only """! ... """ comments are fully processed by doxygen
- I use \copydoc to get the method docs into the "right place". Russell points out that you have to write __init__

Task documentation will appear at http://lsst-web.ncsa.illinois.edu/~buildbot/doxygen/x_masterDoxyDoc/group___l_s_s_t__task__documentation.html as you check it in (and as soon as buildbot notices). Please check that your documentation appears the way that you expected.

To remind you we need:

1. The Task's purpose
2. How to initialise the Task object
3. How to invoke a Task
4. The config parameters
5. Debugging options enabled with "import debug"
6. A complete example.

The input description may not just be something like "sensorRef of the needed inputs". As most of you know, I am unhappy about using blobs/kwargs /sensorRefs as inputs to Tasks, but this mail is not a request to fix this. However, if your task *does* use a blob then the documentation must list all of its members, both optional and required (and enough information for the user to decide which is which for their application). Also, some tasks don't have a run method (e.g. SourceMeasurementTask.measure), but there's no need to fix this now.

The "complete example" may be difficult. I'm mostly concerned about constructor arguments and the case that the inputs include a blob, in which case I want the docs to include how that blob should be built in the calling code -- not from the pipe_base argument parser. If important information is provided by the config, then the user should be told how to build that config in their code; if the arguments are a blob of some sort the example must tell the user how to build it. Examples may not assume the use of the cmdLineTask to build sensorRefs as that can't be used by generic user python scripts.

Think about your users, which includes you! Please try to make these examples as helpful as possible.

There are example docs in python source code in:

- python/lsst/meas/algorithms/measurement.py (SourceMeasurementTask)

- python/lsst/meas/algorithms/detection.py (SourceDetectionTask)

if you're looking for inspiration. I don't claim that these are the best way to achieve the goal, but let's get the docs written and the haggle over how we should have done it -- i.e. please do it this way for now.

The P1 tasks are:

Name	Issue
DipoleMeasurementTask(SourceMeasurementTask)	⚠ DM-911 - Jira project doesn't exist or you don't have permission to view it.
PsfMatch	⚠ DM-912 - Jira project doesn't exist or you don't have permission to view it.

ImagePsfMatchTask(PsfMatch)	⚠ DM-913 - Jira project doesn't exist or you don't have permission to view it.
SnapPsfMatchTask(ImagePsfMatchTask)	⚠ DM-914 - Jira project doesn't exist or you don't have permission to view it.
AssembleCcdTask(pipeBase.Task)	⚠ DM-915 - Jira project doesn't exist or you don't have permission to view it.
FringeTask(Task)	⚠ DM-917 - Jira project doesn't exist or you don't have permission to view it.
IsrTask(pipeBase.CmdLineTask)	⚠ DM-916 - Jira project doesn't exist or you don't have permission to view it.
SourceDetectionTask(pipeBase.Task)	⚠ DM-918 - Jira project doesn't exist or you don't have permission to view it.
SourceMeasurementTask(pipeBase.Task)	⚠ DM-919 - Jira project doesn't exist or you don't have permission to view it.
ReplaceWithNoiseTask(pipeBase.Task)	⚠ DM-925 - Jira project doesn't exist or you don't have permission to view it.
PhotoCalTask(pipeBase.Task)	⚠ DM-920 - Jira project doesn't exist or you don't have permission to view it.

SourceDeblendTask(pipeBase.Task)	⚠ DM-926 - Jira project doesn't exist or you don't have permission to view it.
CmdLineTask(Task)	⚠ DM-927 - Jira project doesn't exist or you don't have permission to view it.
AstrometryTask(pipeBase.Task)	⚠ DM-921 - Jira project doesn't exist or you don't have permission to view it.
CalibrateTask(pipeBase.Task)	⚠ DM-922 - Jira project doesn't exist or you don't have permission to view it.
InterpImageTask(pipeBase.Task)	⚠ DM-923 - Jira project doesn't exist or you don't have permission to view it.
MeasurePsfTask(pipeBase.Task)	⚠ DM-924 - Jira project doesn't exist or you don't have permission to view it.
RepairTask(pipeBase.Task)	⚠ DM-928 - Jira project doesn't exist or you don't have permission to view it.
How to write your own command line task, including how-to-retarget sub-tasks	⚠ DM-929 - Jira project doesn't exist or you don't have permission to view it.

And example source code is at

- python/lsst/pipe/tasks/astrometry.py
- python/lsst/meas/photocal/PhotoCal.py ([still] in meas_astrom)
- **python/lsst/meas/algorithms/measurement.py**
- python/lsst/meas/algorithms/detection.py

if you're looking for inspiration.