Psana Developer Documentation

Content

- Content
- Introduction
- Creating Standard Python Package
- Old to New Conda Command Table
- Missing Functionality
- Converting a Release
- References

Introduction

This page covers documentation about the conda release and build system relevant for psana developers. This documentation is also for LCLS users that want to migrate their test releases based on psana C++ modules and the SConsTools scons build system to the conda environments. This page is up to date with our conversion to github, for the older deprecated svn page, see SVN based Psana Developer Documentation.

Creating Standard Python Package

below is information on using SConsTools with conda. However we can now develop standard python packages, please see this github issue for details: https://github.com/slaclab/anarel-manage/issues/39 , note the checking of the changes to make to the anarel-manage code: ...161dd

Old to New Conda Command Table

Below we go over the steps to create, develop, build and manage test releases - comparing old commands from the RPM release system to the new conda based system.

Old commands like newrel, addpkg and relinfo are not available in the conda world. Most all steps are executed using the

condarel

program. Do condarel -h for the latest help on this script.

step	old	conda	notes
get started	source /reg/g/psdm/etc/ana_env.sh or source /reg/g/psdm/etc/ana_env.csh	source /reg/g/psdm/bin/conda_setup	Bash only for conda, no .csh After sourcing conda_setup, /reg/g/psdm/bin is removed from your PATH (if it was there) it is replaced with /reg/g/psdm/sw/conda/manage/bin
create new test release directory	newrel ana-current myrel	condarel --newrel --name myrel	in conda, myrel is based on the current conda environment. The previous conda_setup command activated a conda environment like ana-1.2.3. If you are not in a conda environment with psana-conda installed, condarel will fail. Sourcing conda_setup will automatically activate such an environment. The old myrel directory has the hidden file **.sit_release** with content like ana-0.19.21 In the new myrelease directory, the content will be the psana-conda package name and version. The new myrel directory will also have the hidden file **.sit_conda_env** with the full path of the conda environment myrelease is built against

activate test release	cd myrel sit_setup	cd myrel source conda_setup	conda_setup looks in the **current directory** (like sit_setup) for the special files mentioned above (but see row below). It sets PATH, LD_LIBRARY_PATH and PYTHONPATH to first look for programs, libraries and python modules built in your test release before looking for them in the conda environment. conda_setup will add *tr* to your prompt to indicate that you are in a ***test release.***
activate test release in another directory	sit_setup /path/to/my/release	source conda_setup --reldir /path/to/my/release	As above, but activate a test release in another directory
create new package	newpkg MyPkg	condarel --newpkg --name MyPkg	Creates the directory MyPkg with a minimal structure, include/src/app/data, and SConscript
create new package in psdm svn repo	psvn newpkg MyPkg	**** use git ****	We won't make packages in the svn psdm repo anymore. We have removed psvn. Work directly with github/lcls-psana
create new package in psdm users repo	psvn -u newpkg MyPkg	**** use git ****	With github, bitbucket, slaclab, etc, there is no need for users to create repos in svn. psvn is removed.
checkout new package	addpkg MyPkg	condarel --addpkg --name MyPkg	for packages that are not part of psana-conda gets it from master in lcls-psana
checkout existing package	addpkg XtcInput	condarel --addpkg --name XtcInput	Looks up the appropriate tag for the version of psana-conda. Checks out that tag ****note this puts you in a HEADLESS state, if developing, checkout master****
checkout exisiting using https		condarel --addpkg --name XtcInput --https	We default to ssh keys, but you can generate the https based git clone command with the --https flag
checkout existing package from psdm users repo	addpkg -u MyPkg	condarel --addpkg --user --name MyPkg	condarel takes --user flag to checkout from psdm users repo. You can also use --tag if you maintain tags in your repo.
checkout from HEAD or master	addpkg XtcInput HEAD	condarel --addpkg --name XtcInput --tag HEAD	You can also specify specific tags with the --tag argument, HEAD means master for git
build	scons	scons	same
develop pdsdata/psalg	very awkward	condarel --addpkg --name pdsdata condarel --addpkg --name pdsdata_ext	See also Building the psalg and pdsdata packages With conda, pdsdata and psalg are part of the psana-conda package. They get put in a subdirectory called extpkgs. You need the proxy packages to build/develop.
develop ndarray	very awkward	condarel --addpkg --name ndarray --tag HEAD condarel --addpkg --name ndarray_ext scons tag ndarray when done update version in ndarray recipe meta.yaml: conda package version git url -- your new tag build new ndarray package (admin account): cd /reg/g/psdm/sw/conda/manage git pull (or fetch? Get you edits above) cd recipes/psana ana-rel-admin --cmd bld-pkg ndarray	ndarray live in it's own conda package. The ndarray github repo does not include a SConscript. You need the ndarray_ext to link it into the build system. If ndarray_ext sees you've checked out ndarray, it overrides the ndarray in the conda environment. After tagging your changes to ndarray, a new ndarray conda package must be built. Edit the meta.yaml on github in the recipe in the links to the right. Now from the admin account, update the management code in /reg/g/psdm/sw/conda/manage. Now build a new ndarray package with ana-rel-admin. The next time a release is built, the new ndarray is picked up.
release info	relinfo	**** not implemented ****	this is an important missing feature we need for github should be condarel --relinfo
upgrade release	 relupgrade ana-19.0.20 sit_setup scons -c scons	source activate ana-1.0.8 condarel --chenv source conda_setup scons -c scons	In conda, first activate, using standard conda commands, the environment you want to build against. Then use the --chenv command, it picks up the current conda environment.

develop Translator	addpkg Translator scons	condarel --addpkg --name Translator condarel --addpkg --name hdf5 condarel --addpkg --name openmpi scons	Since the Translator includes headers using package names, i.e, #include "hdf5/hdf5.h" You must first include the hdf5 and openmpi proxy packages
work with SConsTools add it remove it	 addpkg SConsTools rm -r SConsTools	 condarel --addpkg --name SConsTools rm -r SConsTools rm SConstruct condarel --sconslnk	If you remove SConsTools after checking it out, you have to do additional steps to restore SConstruct. Namely, removing the SConstruct link to the previously checked out version, and then using the --sconslnk command in condarel. This creates the link SConstruct --> SConstruct.main installed in conda env.
Test	scons test	scons test	same
work on package check in new tag to svn psdm repo	addpkg MyPkg HEAD cd MyPkg # modify code svn status # see summary svn diff # see changes svn update svn commit -m "message" psvn tags psvn tag V00-00-00	condarel --addpkg --name MyPkg --tag HEAD same. same git status git diff git pull git commit ... git tag git tag -a ...	We have removed the psvn tool Please see Version control with git for details of using git
track diffs	svn diff -r7810:HEAD file.h	**??**	Please see Version control with git for details of using git
exit conda	--	undo_conda	If you need to get out of the conda world, and go back to where you were before (rpm based psana, if you are sourcing /reg/g/psdm/etc/ana_env.sh) then the undo_conda command does this. **NOTE:** releases built in the conda world will not work in the old RPM world.
list releases	ls $SIT_RELDIR	conda env list	Think of the old RPM based releases as conda environments - use standard conda commands to see them note - this lists your own environments (if you've made any) in addition to the ana environments maintained at LCLS.
identify ana-current	ls -l $SIT_RELDIR/ana-current	more /reg/g/psdm/sw/conda/current/ana/ana-current	changed around April 1 2017, used to be conda/ana-current
identify dm-durrent	ls -l $SIT_RELDIR/current (I think)	more /reg/g/psdm/sw/conda/current/dm/dm-current	
look at source code	ls $SIT_RELDIR/ana-current /<pkg> ls $SIT_RELDIR/<release>/<pkg>	ls /reg/g/psdm/sw/conda/scratch/<release>/<pkg> ls $CONDA_PREFIX/lib/python2.7/site-packages /<pkg>	Package source code is available through the scratch directory. In the conda environment only python code is available.
add pkg to psana	edit the file **ana-tags** in the /reg/g/psdm/sw/releases/buildbot /tags directory	update the file psana-conda-svn-pkgs in the directory /reg/g/psdm/sw/conda/manage/config that is part of the github repo anarel-manage	We should document this more completely in the Admin Documentation

Missing Functionality

Not all functionality of addpkg, relinfo, sit_setup etc have been implemented. If there is a feature that you need, let me know.

If you want to use the old commands, i.e, addpkg instead of condrel --addpkg, we can write new wrappers - but I think while we transition it is good to keep the interfaces distinct as psana developers will be working with both build systems.

Converting a Release

I recommend leaving an old release alone and starting new ones based on conda, however there are two commands in condarel to convert back and forth. This should work for users writing their own C++ psana modules, but won't work for psana developers that have checked out certain external proxy packages. These commands are

condarel --convert2conda

note the name of the old rpm release that you lost. If you want to switch back, use that name with the

condarel --convert2rpm

command. See condarel -h for details.

References

- Version control with git
- https://github.com/lcls-psana/
- SVN based Psana Developer Documentation
- SConsTools