Catalogs and MAF

NOTICE! The simulations packages have migrated to rubin_sim (https://github.com/lsst/rubin_sim) away from the sims_* or lsst_sims package. This means lsst_sims packages are NO LONGER SUPPORTED.

See https://github.com/lsst/rubin_sim for more information. Rubin_sim includes all of the standard lsst_sims packages EXCEPT for sims_catUtils and sims_GalsimInterface, and imports are similar to the previous lsst.sim.XXX imports (rubin_sim.XXX) so your code should need very little modification.

Installing rubin_sim is significantly simpler than installing the lsst_sims packages. Installation instructions are given in the README of the rubin_sim package. Online documentation for the software package is at https://rubin-sim.lsst.io

---

Installing old lsst_sims packages

If you do need to install lsst_sims for whatever reason, the last available version is sims_w_2021_19, which should be installed with w_2021_19 of the LSST DM software.

A report from a user building sims_w_2021_19 suggest that you may need to do the following to get a successful build:

> Running newinstall from the same tag (sims_w_2021_19) – see https://pipelines.lsst.io/install/newinstall.html for general newinstall.sh guidelines and requirements.
> Let it build until it breaks on daf_butler
> Then downgrade the 'click' package to a 7.x version
> Then continuing the build (this is a problem with the sims_w_2021_19 build from DM .. they may have fixed the issue by now, so if your build doesn't break at daf_butler, it's fine!)

This means:

```
cd ~/lsst
curl -OL https://raw.githubusercontent.com/lsst/lsst/w.2021.19/scripts/newinstall.sh
bash newinstall.sh -ct
```

-- Set up the environment and install the sims packages.--

Source the appropriate shell script and use eups (see here for more info) to install the software and data. Easiest install is using bash.

```
source ~/lsst/loadLSST.bash
eups distrib install lsst_sims -t sims_w_2021_19
curl -sSL https://raw.githubusercontent.com/lsst/shebangtron/master/shebangtron | python
```

This will install all packages currently in the catalogs simulations framework (CatSim) and metrics analysis framework (MAF) and all dependencies. The installation should take on the order of 1 hour, with a final required installation size of 10GB.

Any of the individual packages and all their dependencies can be installed by replacing lsst_sims with the appropriate package name in the above code snippet (e.g. sims_maf). Installation is now complete. See package specific pages for documentation.

-- Setup installed packages--

You have now downloaded and built all of the packages in lsst_sims, with the command "eups distrib install <packagename> -t <tag>". These packages are designed to be totally self-contained. They have been built in the directories

```
$LSST_HOME/yourOperatingSystem/yourPackageName/
```

where yourOperatingSystem is 'DarwinX86' for Mac users and 'Linux64' for Linux users. In order to use the packages you installed, you must 'activate' them in your environment, which is essentially adding these directories to your $PYTHONPATH. This is done using

```
setup your_package_name -t your_package_tag
```

"setup <packagename> -t <tag>" is an eups command that activates the packages and their dependencies. your_package_name is the name of one of the packages you have installed (e.g. sims_maf or sims_catUtils). your_package_tag is a tag eups uses to keep track of the versions of each package you have built on your machine. This corresponds to the argument of '-t' in the 'eups distrib install' command above. So, if you wanted to setup the version of MAF you just downloaded, you would use

```
setup sims_maf -t sims_w_2021_19
```

When you setup a package, eups inspects it and determines what other packages it depends on. Eups will also setup those prerequisite packages, preferring versions tagged with the tag you specified and defaulting to versions with the tag 'current'. If you do not want to default to 'current', you can specify more than one tag.

```
setup sims_maf -t $USER -t sims_weekly_tag
```

will setup sims_maf and all of its dependencies, and then using versions matching the tags (resolved from left to right). This system of eups tags allows you to have multiple versions of the stack built on your system simultaneously. You will only ever be using the one that eups has setup. To see which versions of a package exist on your system (and which has been setup) use

```
eups list -v your_package_name
```

or

```
eups list -s
```

Known issues:

- If you have issues with installation, first check that your system meets the minimum requirements listed here: https://pipelines.lsst.io/install/newinstall.html#prerequisites. Note that you need cmake – if installation of 'mariadb' fails, you probably do not have cmake installed ('conda install cmake' is an easy way to get it).
- Check for other known issues here: https://pipelines.lsst.io/known-issues.html#installation-issues
- You can also search for similar problems on https://community.lsst.org/
- If you are having issues specifically with pyephem or healpy on a Mac, check for the existence of a /Developer directory. This directory is obsolete after upgrading to newer versions of XCode, but not removed by the XCode installer. Rename the /Developer directory and pyephem will install.
- If you are using your own python, be sure to check the Using Your Own Python page. In particular, on Linux, some lsst_apps packages will currently fail to build if the "nomkl" package is not installed in anaconda.
- On a Mac, make sure you have accepted the terms on XCode. You can do this by opening the Xcode.app (should be in your Applications folder).
- git can fail, complaining about not having an https helper. If your native git version is > 1.7, you can probably use that rather than the LSST installed git.
- If all else fails, it's usually an issue with some environment variables interfering with the installation. You can create a new user and install the stack there. You can quickly login/out of a new user account as follows: First make a new admin-level user in System Preferences->Users and Groups, and then click on your name in the top right hand corner of the screen. A drop-down menu should appear, offering you a choice of other users to log in as. You might have to toggle the check box in System Preferences->Users and Groups->Login Options first though.