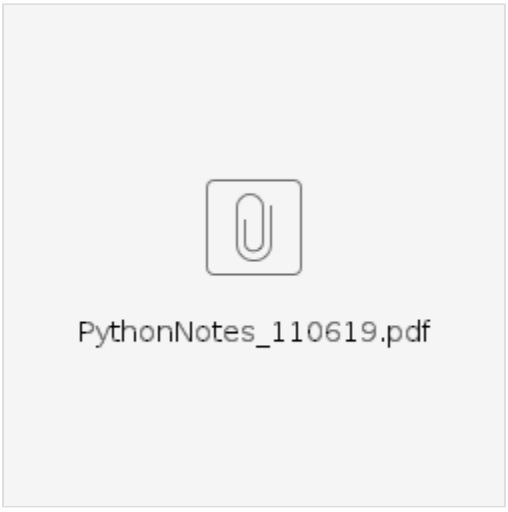
2019-11-07: Python support in JEDI

Yannick opened the meeting by announcing the topic of the day, namely how we might support python applications within JEDI as we move forward. Specific issues up for discussion include the deprecation of python 2 at the end of the year, the preferred method for packaging and distributing JEDI python dependencies (e.g. pip or miniconda), and restrictions that may exist on some platforms, for example security policies at operational centers that may restrict some third-party applications. These issues are summarized in the following slides, which were presented by Steve.



Steve and Mark M then opened the discussion with the issue of python 2. Mark asked if we were to remove python 2 from the containers, would it break anything in JEDI. Steve V responded that it would break a few of his converter scripts in the ioda-converters repository but he and Mark O agreed that this would be easily remedied. Travis commented that they have achieved identical results between their python 2 and 3 scripts for SOCA and commented that these could be used as templates for others to convert their applications in a way that doesn't change results. Mark O added that python installations typically include tools to convert python 2 scripts to python 3.

Yannick then asked the group if there is anyone that requires python 2. There were no responses. Mark then asked for the perspective of some visitors to our meeting that were joining us from NOAA/EMC/EIB. Kate Friedman responded and said that they are in the process of converting their python 2 scripts to python 3 but they really just started with this. She said they don't use much python (in the global workflow) so it is not a high priority. Python2 is used in the HWRF workflow scripts. Stylianos confirmed that python3 is already available on WCOSS. However, Cory said that the Cray still has python 2. In an email, Arun Chawla (EIB) informed Mark M that "We are moving everything to Python 3.6. It will take some time but we are transitioning to that. It took a year but all the operational HPC platforms now have Python 3.6 and that was the delay in transitioning over. "

Having confirmed that a move to python 3 should not be a problem for JEDI users, Yannick and Mark O asked if there is a particular version that we should require. Mark said the current stable release is 3.8 but 3.7 might be safer. His jedi-rapids workflow requires at least 3.5.

Rahul then brought up another issue about how the dependent modules are used and managed. This brought the discussion to the second discussion topic mentioned above about packaging and distributing python dependencies for JEDI.

Mark O advocated for the python virtualenv tool that can be used to define, install and provide access to a selected set of python modules. This can build on pre-existing system modules but it can optionally replace these or extend them with a user-specified list. He said this functions in a similar way to environment modules in that they install python modules in a user-specified directory and grant the user access to that directory. So, it would integrate well with the module-based jedi-stack build system.

Rahul approved of this option, adding that third-party applications like miniconda might not be allowed by the administrators of secure systems like WCOSS.

Mark O added that virtualenv works by removing conflicts from python packages, creates an isolated environment, and then uses pip to install packages into it.

Rahul added that one could even modify jedi-stack to build some python dependencies like matplotlib from source.

Mark O then reported that another advantage of virtualenv is that it does not require administrator (root) privileges to install. However, certain base packages like numpy might show improved performance if they are installed on the system. But these are the exception - many python packages are not compiled and should be easy to install, without any need for optimization.

virtualenv works by having the user/developer specify a list of dependencies in a text file. Rahul mentioned that, when running at operational centers, it may be necessary to get this pre-approved. Steve H and Yannick asked how to know what packages are acceptable - is there is list somewhere? Nobody was aware of one but it was mentioned that we should ask the EIB lead (Arun). Sarah K volunteered to check if the Navy also has restrictions on python packages.

There was general agreement that this strategy of using virtualenv to manage python dependencies is the best option.

Yannick then addressed the group and asked if there were any other constraints that we should be aware of. Steve S wanted a clarification of what we are talking about - is it just a question of what is included in the containers and jedi-stack? Yannick said that is part of it, but it's also our python use in general. For example, if we introduce jedi components such as jedi-rapids that require a particular python version (in this case 3.5), will this cause problems for anyone?

Yannick then asked how we should proceed. The first step is to decide on a minimum release version and make sure everyone has access to it. For this, it was decided to tentatively set that minimum release to python 3.6.

So, if you do not have access to python 3.6 please let us know

Then, the next step would be to write a script that can be added to jedi-stack and that includes all that is needed to install and distribute jedi python dependencies using virtualenv. Then this could be added to the containers and other platforms, including AWS, Discover, Hera, Cheyenne, S4, etc. virtualenv works by running a startenv script that sets up environment variables for subsequent installations. We can specify install directories on HPC systems that can be shared among JEDI users, much like the environment modules.

Then we opened the floor to ask if there were any other questions about this approach. Mark M asked if users need a particular minimum version of python to run virtualenv. Mark O and Yannick answered that it has been around for a while - a decade or more - so availability should not be an issue.

Mark M then asked if this approach would conflict with other python management systems that jedi users might use. For example, if a jedi user is accustomed to installing python packages on their laptop with miniconda, will the JEDI virtualenv approach lead to conflicts?

Mark O responded that virtualenv, like pip and (it was expected) conda all use the setuptools utility to install and upgrade packages. setuptools is a standard feature of most python installations and is used to provide standard functionality such as python eggs and wheels. So, the JEDI virtualenv should not conflict with any other python packages or package managers on a user's system.

With general agreement that this is a good path forward, the meeting was adjourned.