



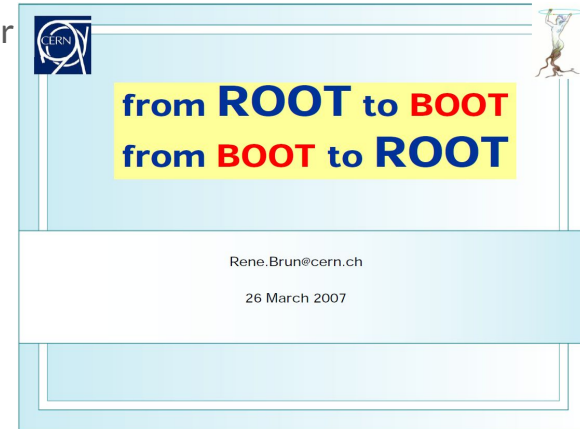
# ROOT Modularization

Oksana Shadura, University of Lincoln-Nebraska

# Goals of modularization project

The project **goals** are:

- Recheck ROOT subsystems and dependencies: CORE, I/O, Math and etc.
- Select “small heart-like” part of ROOT that could be a fundamental for modularization process;
- Make ROOT friendly for plugin new developments/products:
  - Adapt the ROOT source code to enable (more) custom-tailor products.
  - Support custom-tailor products in the build system by
    - i. checking out what is needed only,
    - ii. building what is needed only,
    - iii. re-using intermediate or final deliverables.
- Enable pre-build intermediates and their usage.





# Benefits

- Users can get as a final product - ROOT as a smaller package containing just the services they need and plugging extra whenever they want;
- Making ROOT more friendly for users;
- Possibility to plug and unplug components;
- Help in efficient managing dependencies (ROOT libraries and external libraries);
- Simplify the adapting process for a new/external contributions in ROOT;



# Start At The Beginning

- Talk to 4 developers what “modularization” means and you’ll get 5 answers.
- Trying to start at the base layer - “how would ROOT define a module?” - and trying to a few examples of simple ROOT functionality to see how we could recast them as a module.
- Would like to get a good feeling for how a module or package would work *before* a thorough treatment of how they are managed.
- Presenting in forums like this one provide a mechanism for feedback from a wide set of viewpoints.

---

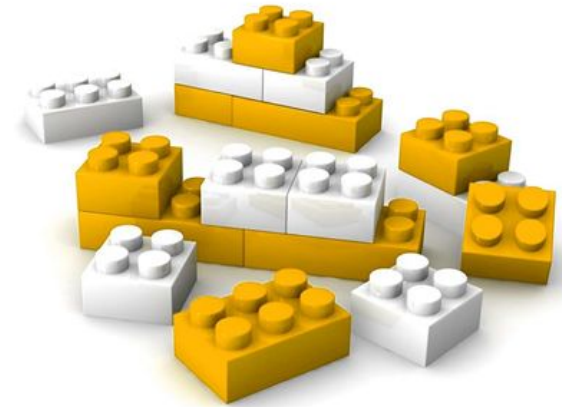
# Examples of products modularization

---

# Examples of modularization: LibreOffice & Xorg

<https://wiki.openoffice.org/wiki/Modularization.org>

<https://www.x.org/wiki/ModularizationProposal/>



---

# Background studies on modularization



# Two directions of ROOT modularization project

- **First:** add external support to plug new contributions that doesn't require changes in a ROOT core;
- **Second:** change or add new interfaces to be able easily plug new developments;

We need to add package manager or alternative classes and work on both directions together!



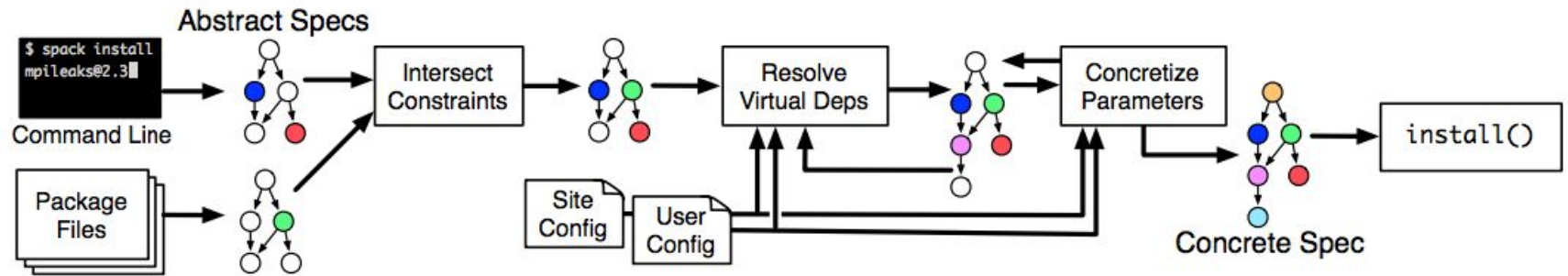
# Package managers as an inspiration for a new tool

- Homebrew
- Portage (*It has GIT plug-in module for portage which performs various git actions and checks on repositories.*)
- Spack



# Spack

- Spack is a package management tool designed to **support multiple versions and configurations of software on a wide variety of platforms and environments.**
- It is widely used in EP-SFT for LCGPackages, could be the ideas used in Spack, is useful for project?





# Python packaging PEP425 and PEP427

- Wheel format <https://wheel.readthedocs.io/en/latest/>

## Wheel Slogans

- *Because 'newegg' was taken.*
- *Python packaging - reinvented.*
- *A container for cheese.*
- *It makes it easier to roll out software.*

To quote [PEP 427](#), wheel is:

*A wheel is a ZIP-format archive with a specially formatted file name and the .whl extension. It contains a single distribution nearly as it would be installed according to PEP 376 with a particular installation scheme. Although a specialized installer is recommended, a wheel file may be installed by simply unpacking into site-packages with the standard 'unzip' tool while preserving enough information to spread its contents out onto their final paths at any later time.*

To put it simply, wheels are a way to have your own local instant pypi.



# PEP 425 and wheel

`{distribution}-{version}{(-{build tag})?}-{python tag}-{abi tag}-{platform tag}.whl`

## Usage:

```
pip wheel [options] <requirement specifier> ...
pip wheel [options] -r <requirements file> ...
pip wheel [options] [-e] <vcs project url> ...
pip wheel [options] [-e] <local project path> ...
pip wheel [options] <archive url/path> ...
```



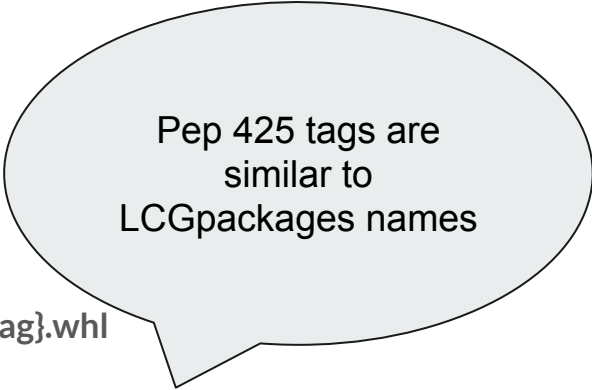
## Description:

Build Wheel archives for your requirements and dependencies.

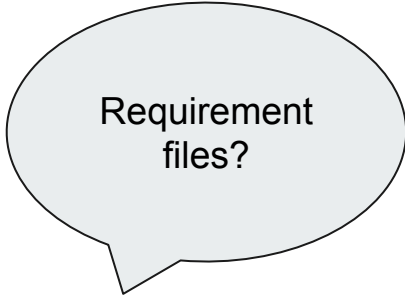
Wheel is a built-package format, and offers the advantage of not recompiling your software during every install. For more details, see the wheel docs: <https://wheel.readthedocs.io/en/latest/>

Requirements: `setuptools>=0.8`, and `wheel`.

'pip wheel' uses the `bdist_wheel` `setuptools` extension from the `wheel` package to build individual wheels.



Pep 425 tags are similar to LCGpackages names



Requirement files?



## Put wheels under your feet!

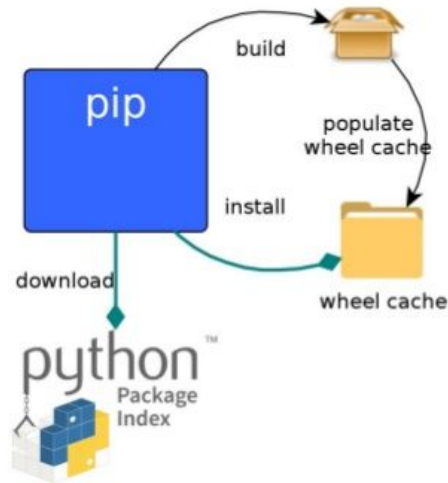
PEP 427

setuptools  $\geq$  0.8

pip can populate  
wheel cache

Labels eg  
linux\_x86\_64  
py3

Package retrieved  
from cache



Simple configuration and metadata files!

```
~/usr/local/lib/python2.7/site-packages/Keras-2.0.2.dist-info/METADATA
Metadata-Version: 2.0
Name: Keras
Version: 2.0.2
Summary: Deep Learning for Python
Home-page: https://github.com/fchollet/keras
Author: Francois Chollet
Author-email: francois.chollet@gmail.com
License: MIT
Download-URL: https://github.com/fchollet/keras/tarball/2.0.2
Platform: UNKNOWN
Requires-Dist: pyyaml
Requires-Dist: six
Requires-Dist: theano
Provides-Extra: h5py
Requires-Dist: h5py; extra == 'h5py'
Provides-Extra: tests
Requires-Dist: pytest; extra == 'tests'
Requires-Dist: pytest-pep8; extra == 'tests'
Requires-Dist: pytest-xdist; extra == 'tests'
Provides-Extra: visualize
Requires-Dist: pydot-ng; extra == 'visualize'
UNKNOWN
```

```
~/usr/local/lib/python2.7/site-packages/Keras-2.0.2.dist-info/WHEEL
Wheel-Version: 1.0
Generator: bdist_wheel (0.29.0)
Root-Is-Purelib: true
Tag: cp27-none-any
```

```
~/usr/local/lib/python2.7/site-packages/Keras-2.0.2.dist-info/metadata.json 748/748 100%
{"download_url": "https://github.com/fchollet/keras/tarball/2.0.2", "extensions": {"python.details": {"contacts": [{"email": "francois.chollet@gmail.com", "name": "Francois Chollet", "role": "author"}], "document_names": {"DESCRIPTION": "DESCRIPTION.rst"}, "project_urls": {"Home": "https://github.com/fchollet/keras"}}, "extras": ["h5py", "tests", "visualize"], "generator": "bdist_wheel (0.29.0)", "license": "MIT", "metadata_version": "2.0", "name": "Keras", "run_requires": [{"extra": "h5py", "requires": ["h5py"]}, {"extra": "visualize", "requires": ["pydot-ng"]}, {"extra": "tests", "requires": ["pytest-pep8", "pytest-xdist", "pytest"]}, {"requires": ["pyyaml", "six", "theano"]}], "summary": "Deep Learning for Python", "version": "2.0.2"}
```


```
~/usr/local/lib/python2.7/site-packages/Keras-2.0.2.dist-info/top_level.txt
keras
```

```
~/usr/local/lib/python2.7/site-packages/Keras-2.0.2.dist-info/RECORD
Keras-2.0.2.dist-info/DESCRIPTION.rst,sha256=0CTu0N6LcWuLhHS3d5FfjdsQtw22n7HENFRh6jC6ego,10
Keras-2.0.2.dist-info/METADATA,sha256=Ho3tru0PVy0YK1hcIgg_PHY9HbgvpacZnMkzcBoQ2PE,644
Keras-2.0.2.dist-info/RECORD,,
Keras-2.0.2.dist-info/WHEEL,sha256=BtVfdXUcEYLcFj0kbIrcFRyXU4qszVPT-E9o3RwKSNw,93
Keras-2.0.2.dist-info/metadata.json,sha256=UW-NijLZ8rPeIK3xWU-NcfKd3RHG4EEnyh3pKiz6R0I,748
Keras-2.0.2.dist-info/top_level.txt,sha256=ptcw_-0uGZ4ZDjMdwI_Z0cLZm8QAqFdvzzFnDEOTs9o,6
keras/__init__.py,sha256=YuUk-6EFC-ug-_eoPehZ3gTsD5XB6ZFu3-i0xmRR1I,488
keras/activations.py,sha256=05gQ2fLM_KDnwmnJT1fcdKNHqHvEpd1SVc-Rpsyr7L9E,1875
keras/callbacks.py,sha256=cju08s0cwQ4H-7WdjgZLNxAPnFpRLCt40SD8T-X45QI,33865
keras/constraints.py,sha256=KMLXmS0Hfc1x4v6JLAWXZGKc0FzXGwQY-qexwnpqs,6048
keras/initializers.py,sha256=7nBUT0mqgKjT9Bja04Y55TFvlpKpkmPlccA6rkv6jK,14871
keras/losses.py,sha256=0VEj9WoxKqrfjLbKqupkIM7IZUCotZ50cvYDpBi08,2784
keras/metrics.py,sha256=Pv7W9nPKFKZ1PcW8aikwoKca0Cst9t4vjfo8DumKvM,2112
keras/models.py,sha256=A4duST0dsJImZ79Tq35c-_IbHJSTisK1yordg8VGC2qw,49574
keras/objectives.py,sha256=ck4Iw_cAo6AbKrOPBLBLFnSiVmIcvhAbm9Ax-Fty8PM,138
keras/optimizers.py,sha256=d5U3xGREB-X13jLgJmjbNybM8-G-f-Ztqvubi_qkOA,24892
keras/regulators.py,sha256=TJ3qrfNsXBkVhIStDE5z2jF4WtkXhBBsQSQG6mrHno,2034
keras/applications/__init__.py,sha256=wu75Dk9-m58K_pnga-0Wdx0v3xvm4k5Z0kq04nYVa,150
keras/applications/imagenet_utils.py,sha256=ygAP4z0vYmQR0KYC01N8H-my4b7SIznGh7fAg3mhy8k,5426
keras/applications/inception_v3.py,sha256=bN_jjIppRu081oT2z2d_8MLKqFkXbl6kOf_0oEC2JL4UM,14499
```



# R

- Packages are the fundamental units of reproducible R code. They include reusable R functions, the documentation that describes how to use them, and sample data.
- R does not have responsibility for plugged packages



You can further provide examples (which can also serve as tests) and links to related functions. The examples need to be **fast** ( $\ll 5$  sec), because they're run frequently. (CRAN checks every package every day on multiple systems.)

```
RSkittleBrewer/  
  
DESCRIPTION  
NAMESPACE  
  
R/RSkittleBrewer.R  
R/plotSkittles.R  
R/plotSmarties.R  
  
man/RSkittleBrewer.Rd  
man/plotSkittles.Rd  
man/plotSmarties.Rd
```

## An .Rd file

```
\name{RSkittleBrewer}  
\alias{RSkittleBrewer}  
\title{Candy-based color palettes}  
\description{Vectors of colors corresponding to different  
candies.}  
\usage{RSkittleBrewer(flavor = c("original", "tropical",  
"wildberry", "M&M", "smarties"))}  
}  
\arguments{  
  \item{flavor}{Character string for candy-based color  
palette.}  
}  
\value{Vector of character strings representing the chosen  
set of colors.}  
\examples{  
plotSkittles()  
plotSmarties()  
}  
\keyword{hplot}  
\seealso{ \code{\link{plotSkittles}},  
          \code{\link{plotSmarties}} }
```

---

# Ideas about how to help to modularize ROOT





# Proposal about Cling integrated tool from Vassil V.

```
[root]#include "TXXX.h"
```

“No such package XXX. Installing it for you..”

```
[root]TXXX a;
```

```
[root]//works
```



# Ideas for a tool TInstaller from Pere Mato

- TInstaller is a class that help ROOT to work with external packages (from cvmfs/lcgexternals) or help build packages from scratch
- Similar to Python package management tool?
- Where to put TInstaller: could be or external or inside Core or inside Cling?



## Next steps

- Review ROOT minimal option ("Minimal Installation")
- Review dependencies introduced in Core and etc.
- Test functionality of ROOT minimal
- Make a prototype allowing to upgrade ROOT minimal with extra options/dependencies/services

Thank you!  
Looking forward to see your ideas  
and suggestions!

---