Python packaging in openSUSE

Advantages against other distributions

Matěj Cepl
mcepl@cepl.eu

openSUSE Conference, June 2022

2022-06-02

Python packaging in openSUSE

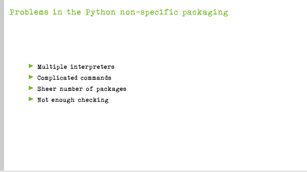
Python packaging in openSUSE
Advantages against other distributions

Matěj Cepl
mcepl@cepl.eu

openSUSE Conference, June 2022

Problems in the Python non-specific packaging

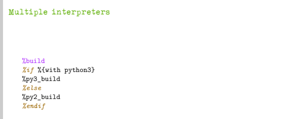
- Multiple interpreters
- Complicated commands
- Sheer number of packages
- Not enough checking


Problems in the Python non-specific packaging

- Multiple interpreters
- Complicated commands
- Sheer number of packages
- Not enough checking

Multiple interpreters

```
%build
%if

%else

%endif
```



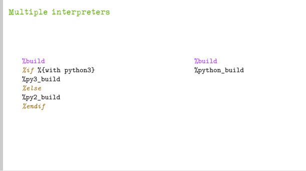
Multiple interpreters

```
%build
%if %{with python3}
%py3_build
%else
%py2_build
%endif
```

1. The other solution is hard-coded, so whenever anything changes, everything has to be changed again. Other solution is the same as what we used to use for Python 2 (and we still use for Python2 in some enterprise channels), and it is the same now when we use it for simultaneous support of Python 3.8, 3.9, 3.10 and possibly soon 3.11 on Tumbleweed, and it would be the same if somebody finally fixed pypy3 or if a miracle happened and somebody got jythohn to the functional state.

Multiple interpreters

```
%build                                      %build
%if

%else

%endif
```



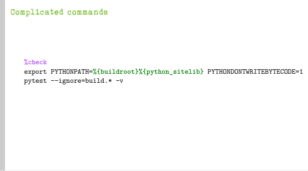
Multiple interpreters

```
%build                                      %build
%if %{with python3}                         %python_build
%py3_build
%else
%py2_build
%endif
```

1. The other solution is hard-coded, so whenever anything changes, everything has to be changed again. Other solution is the same as what we used to use for Python 2 (and we still use for Python2 in some enterprise channels), and it is the same now when we use it for simultaneous support of Python 3.8, 3.9, 3.10 and possibly soon 3.11 on Tumbleweed, and it would be the same if somebody finally fixed pypy3 or if a miracle happened and somebody got jythohn to the functional state.

Complicated commands

```
%check
                    %{buildroot}%{python_sitelib}
```



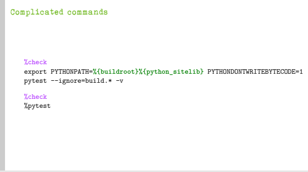
Complicated commands

```
%check
export PYTHONPATH=%{buildroot}%{python_sitelib} PYTHONDONTWRITEBYTECODE=1
pytest --ignore=build.* -v
```

And this is still not correct, because it must be done for all Python versions separately (and those --ignore=build* must be arranged accordingly). And we need to be able to take into consideration existing PYTHONPATH.

Complicated commands

```
%check
                  %{buildroot}%{python_sitelib}


%check
```



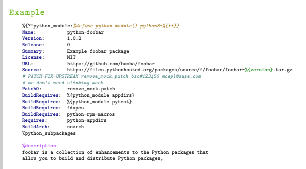
Complicated commands

```
%check
export PYTHONPATH=%{buildroot}%{python_sitelib} PYTHONDONTWRITEBYTECODE=1
pytest --ignore=build.* -v

%check
%pytest
```

And this is still not correct, because it must be done for all Python versions separately (and those --ignore=build* must be arranged accordingly). And we need to be able to take into consideration existing PYTHONPATH.

Example

```
                    %define python_module() python3-%{**}
Name
Version
Release
Summary
License
URL
Source                                                                    %{version}
# PATCH-FIX-UPSTREAM remove_mock.patch bsc#123456 mcepl@suse.com
# we don't need stinking mock
Patch0
BuildRequires
BuildRequires
BuildRequires
BuildRequires
Requires
BuildArch


%description
```



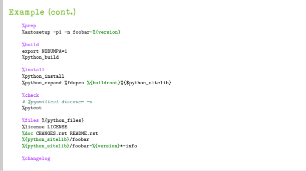
Example

```
%{?!python_module:%define python_module() python3-%{**}}
Name:           python-foobar
Version:        1.0.2
Release:        0
Summary:        Example foobar package
License:        MIT
URL:            https://github.com/bumba/foobar
Source:         https://files.pythonhosted.org/packages/source/f/foobar/foobar-%{version}.tar.gz
# PATCH-FIX-UPSTREAM remove_mock.patch bsc#123456 mcepl@suse.com
# we don't need stinking mock
Patch0:         remove_mock.patch
BuildRequires:  %{python_module appdirs}
BuildRequires:  %{python_module pytest}
BuildRequires:  fdupes
BuildRequires:  python-rpm-macros
Requires:       python-appdirs
BuildArch:      noarch
%python_subpackages

%description
foobar is a collection of enhancements to the Python packages that
allow you to build and distribute Python packages,
```

1. Eliminate as much boiler-plate as possible. We have only two lines now, which are same every time: that python_module definition and Release. And in both cases we are just forced to have them there by the mechanics of RPM.
2. Note that macro python_subpackages, that's the root of all machinations. In effect this SPEC file is just a foundation, I don't want to use the term "template„ because that would be misleading, for multiple generated ones.

Example (cont.)

```
%prep
                         %{version}

%build



%install
                       %{buildroot}

%check
# %pyunittest discover -v


%files

%doc
%{python_sitelib}
%{python_sitelib}        %{version}

%changelog
```

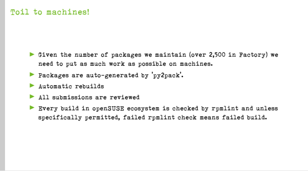


```
%prep
%autosetup -p1 -n foobar-%{version}

%build
export NOBUMPA=1
%python_build

%install
%python_install
%python_expand %fdupes %{buildroot}%{$python_sitelib}

%check
# %pyunittest discover -v
%pytest

%files %{python_files}
%license LICENSE
%doc CHANGES.rst README.rst
%{python_sitelib}/foobar
%{python_sitelib}/foobar-%{version}*-info

%changelog
```

1. Using autosetup to avoid dealing with each patch again.

Toil to machines!

- Given the number of packages we maintain (over 2,500 in Factory) we need to put as much work as possible on machines.
- Packages are auto-generated by 'py2pack'.
- Automatic rebuilds
- All submissions are reviewed
- Every build in openSUSE ecosystem is checked by rpmlint and unless specifically permitted, failed rpmlint check means failed build.


Toil to machines!

- Given the number of packages we maintain (over 2,500 in Factory) we need to put as much work as possible on machines.
- Packages are auto-generated by 'py2pack'.
- Automatic rebuilds
- All submissions are reviewed
- Every build in openSUSE ecosystem is checked by rpmlint and unless specifically permitted, failed rpmlint check means failed build.

1. They are auto-generated, but they are more like a ready-to-cook food, they need to be finished.
2. And yes, it is similar to the Fedora's pyp2rpm, but we have never managed to unify two code-bases. If anybody is willing to do the work, it would be lovely.
3. I may add an anecdote about

Thank you!

Thank you!

My email is mcepl@cepl.eu.

LaTeX source of this presentation is available on https://gitlab.com/mcepl/opensuseconf.git.