

## └ 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 jython to the functional state.

## └ Multiple interpreters

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

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 jython to the functional state.

2022-06-02

## Python packaging in openSUSE

└─ Complicated commands

Complicated commands

└─ check

```
export PYTHONPATH=${builddir}/${python_sitelib} PYTHONDOCTWRITETECODE=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`.

## Python packaging in openSUSE

└─ Complicated commands

Complicated commands

```
❌check
export PYTHONPATH=${builddir}/${python_sitelib} PYTHONDOCTWRITETECODE=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`.



## Python packaging in openSUSE

## └ Example (cont.)

## Example (cont.)

```
type
  autosetup -p -> /usr-1/$(uname)

build
export RPMQA=1
python_build

install
python_install
python_remove %{buildroot}/%{python_install}

check
# Specific test discover ->
pytest

files %{python_files}
license LICENSE
doc CHANGES.txt README.txt
%python_install%files
%python_install%files
%python_install%files+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 'pyp2rpm'.
- ▶ 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