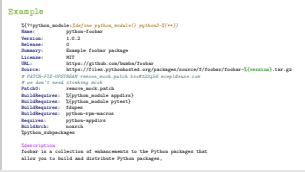
1. The other solution is hard-coded, so whenever anything changes, everything has to be changed again. Other solution is the same as what we used to use for Python 2 (and we still use for Python2 in some enterprise channels), and it is the same now when we use it for simultaneous support of Python 3.8, 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.

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.

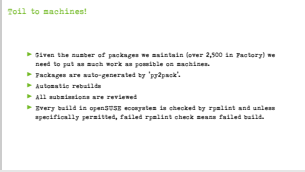
And this is still not correct, because it must be done for all Python versions separately (and those --ignore=build* must be arranged accordingly). And we need to be able to take into consideration existing PYTHONPATH.

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.



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.

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



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