ActiveState

Solving Dependency Hell at Enterprise Scale

About ActiveState

Used by Millions of Developers and 97% of Fortune 1000

20+ Years of Open Source Language Experience

ActiveState

Introductions

Scott Robertson

CTO

Dana Crane

Product Marketing Manager

ActiveState

Housekeeping

- Session: 30 minutes; Live Q&A: 15 minutes
- You can also ask questions in the Q&A tab
- There will be two polls and a survey afterwards - your feedback is crucial!
- Recording of this will be available and sent to you

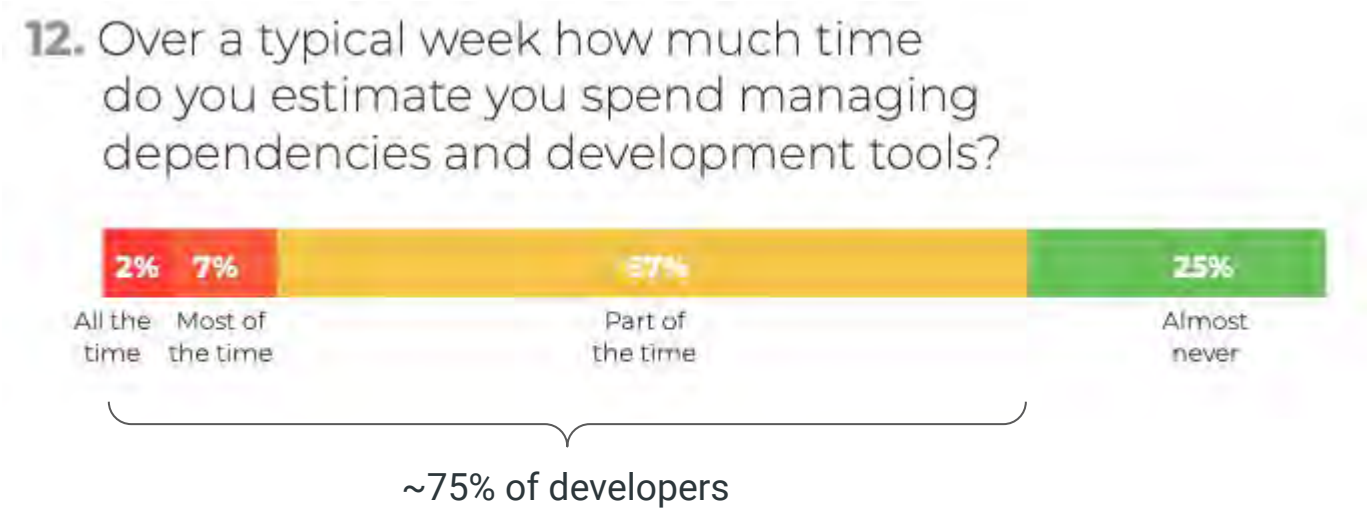
ActiveState

Agenda

- Time Managing Dependencies
- Causes of Dependency Hell
- Best Practices: Pros & Cons
- ActiveState Platform
- Demo
- Q&A

ActiveState

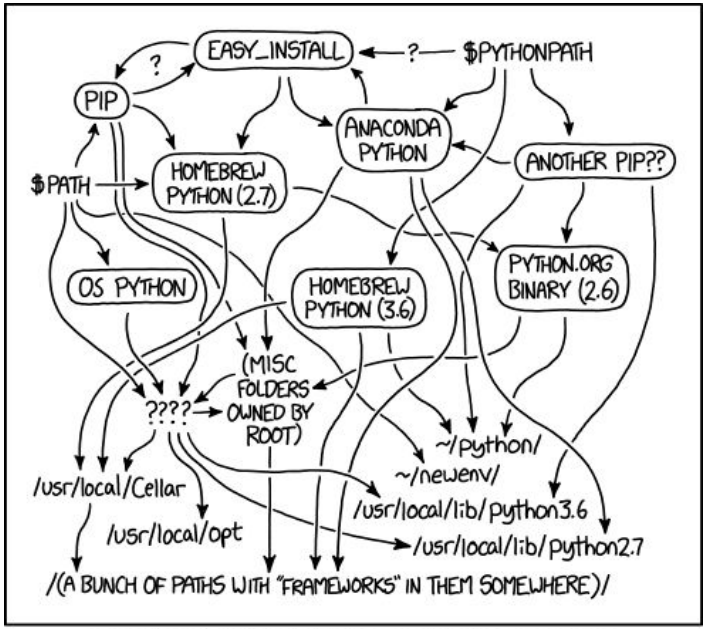
Time Spent Managing Dependencies



ActiveState

How Did We Get Here?

- Newbies & out of date READMEs
- Incompatible packages you require (new features/ bug fixes/security updates)
- New operating environment
- Installers don't check existing dependencies before stepping on them
- Monkey patching third-party code
- Using multiple package managers per language (ie., pip + conda for Python)



MY PYTHON ENVIRONMENT HAS BECOME SO DEGRADED THAT MY LAPTOP HAS BEEN DECLARED A SUPERFUND SITE.

ActiveState

Poll: How much time do your devs spend managing dependencies?

- <10% of a sprint
- 10-25%
- 25-50%
- >50%

ActiveState

Ramifications of Poor Dependency Management

- Don't want to touch the environment since it might break the build
- Updates snowball if not fixed quickly enough, making the problem worse
- “Works on my machine” & environment reproducibility issues
- Auditing open source dependencies becomes increasingly difficult

ActiveState

Tactics for Managing Dependency Hell

1. Rely on package managers and direct repository access
2. Repository Proxy/Caching
3. Standardize Native Deps with VMs/Container Images
4. Vendor code under source control

ActiveState

Rely on Package Managers

PROS

- Easy to set up
- Community support
- Infrastructure as code

CONS

- Tool proliferation: you need O + L tools
- Availability of prebuilt packages vary by OS (they'll have to be built at some point)
- Poor locking support leads to inconsistent states
- Things change on the Internet

ActiveState

Repository Proxy/Caching

PROS

- Keep teams and environments consistent (mostly)
- Modest level of provenance tracking
- Share internally built artifacts
- Central place to audit package use
- Vet packages before others use them

CONS

- **Poor support for native libraries**
- Inconsistent states still possible
- Building still an issue
- **Vetting introduces friction**

ActiveState

VMs/Container Images - (for native libraries)

PROS

- Better support for consistent use of native packages
- Faster deployment speeds
- Reduces the number of tools downstream consumers need to worry about

CONS

- No universal image format. Tool usage is now O+L+I for DevOps
- Building and using images is hard
- Distribution mechanic synchronization, backwards compatibility
- Multiple project issues
- **Even more vetting friction**

ActiveState

Vendoring Dependencies (reduce vetting friction)

PROS

- Building from source is the ultimate way to support Native
- Support for provenance tracking
- Allows for customization and patching that upstream authors may be slow to bring in

CONS

- Need to setup a CI system and toolchains to support building
- Need to be an expert for each supported OS
- Need a local build process tool like Bazel, Make, Scons, etc.
- Need artifact storage

ActiveState

Make it Someone Else's Problem

PROS

- It's someone else problem!

CONS

- Finding something to do with your free time

ActiveState

Poll: How do you manage dependencies?

- Use native package manager to pin everything
- Use a pre-resolved set of dependencies in an artifact repository
- Use VM/Container base images
- Self vendor all dependencies in source control system

ActiveState

The Need for Good Dependency Management

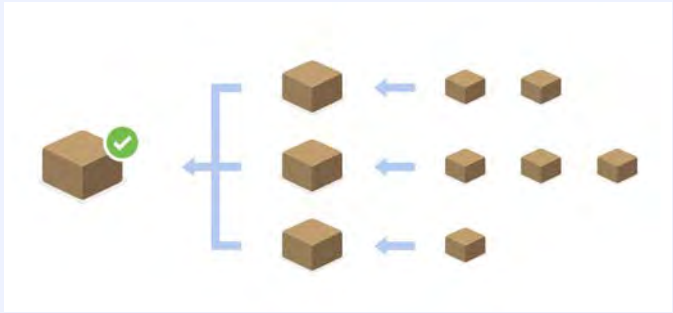
- $$
- Optimal utilization of your most valuable devs
- Maximize dev satisfaction by letting them work on the things they want to work on
- Delayed time to market

ActiveState

ActiveState Platform — Your Team's Supply Chain for Trusted Open Source Artifacts

Includes:

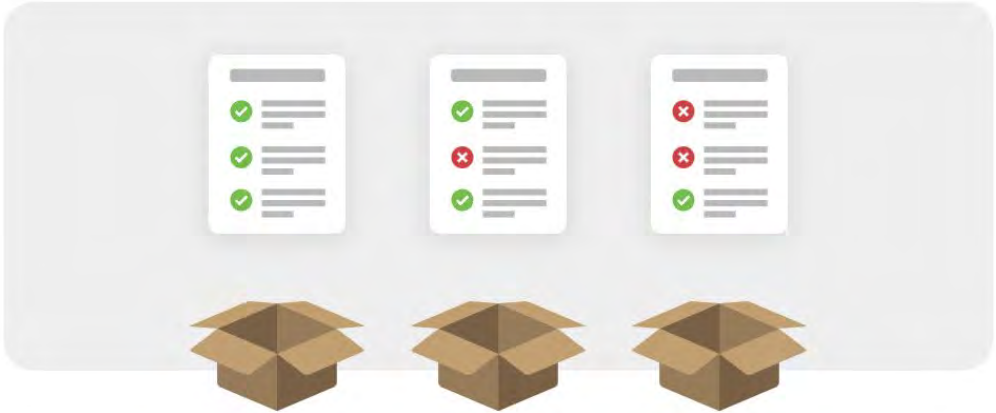
- Catalog of 4M+ Vendored Open Source Components & Recipes
- Universal Dependency Solver
- Hermetically Sealed Multi-OS Build Farm
- Declarative Project Oriented UX/API
- Powerful Revision Control Features



ActiveState

Single Set of Cloud Based Cross Platform Tools for:

- Build
- Distribute
- Maintain
- Monitor



Runtimes for your Open Source based projects

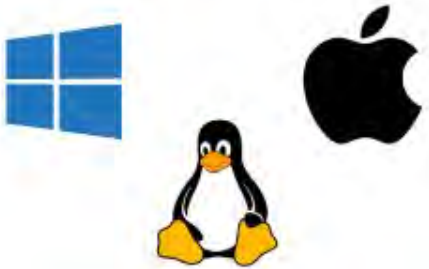
ActiveState

How it Works

ActiveState

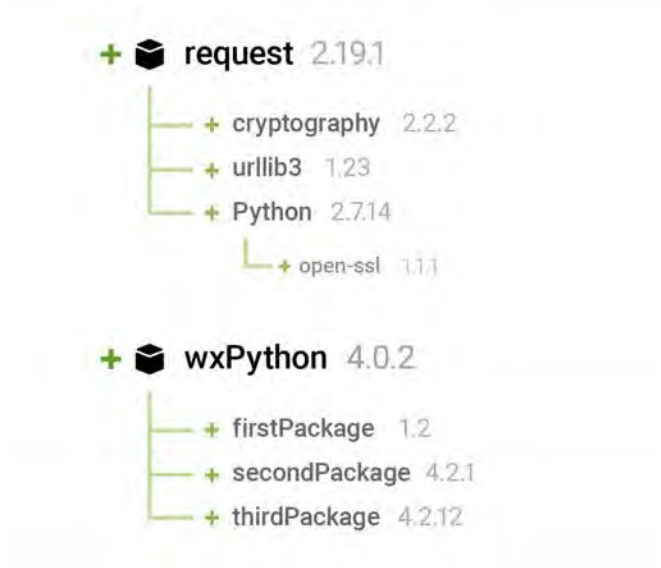
Create a Project with Requirements and Operating Systems

+ request 2.19.1
+ wxPython 4.0.2



ActiveState

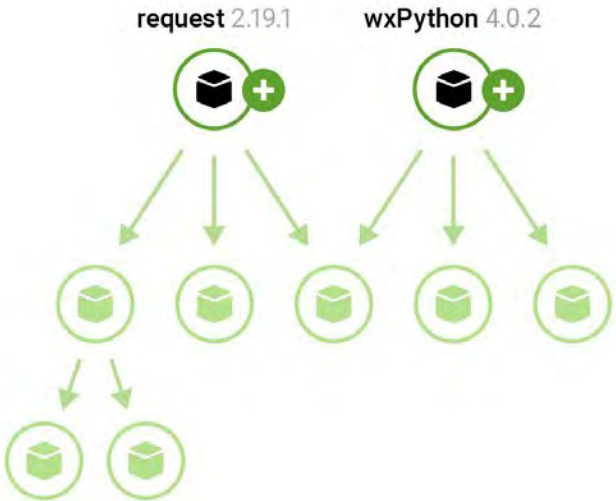
ActiveState Resolves the Complete Set of Dependencies



Including native libraries!

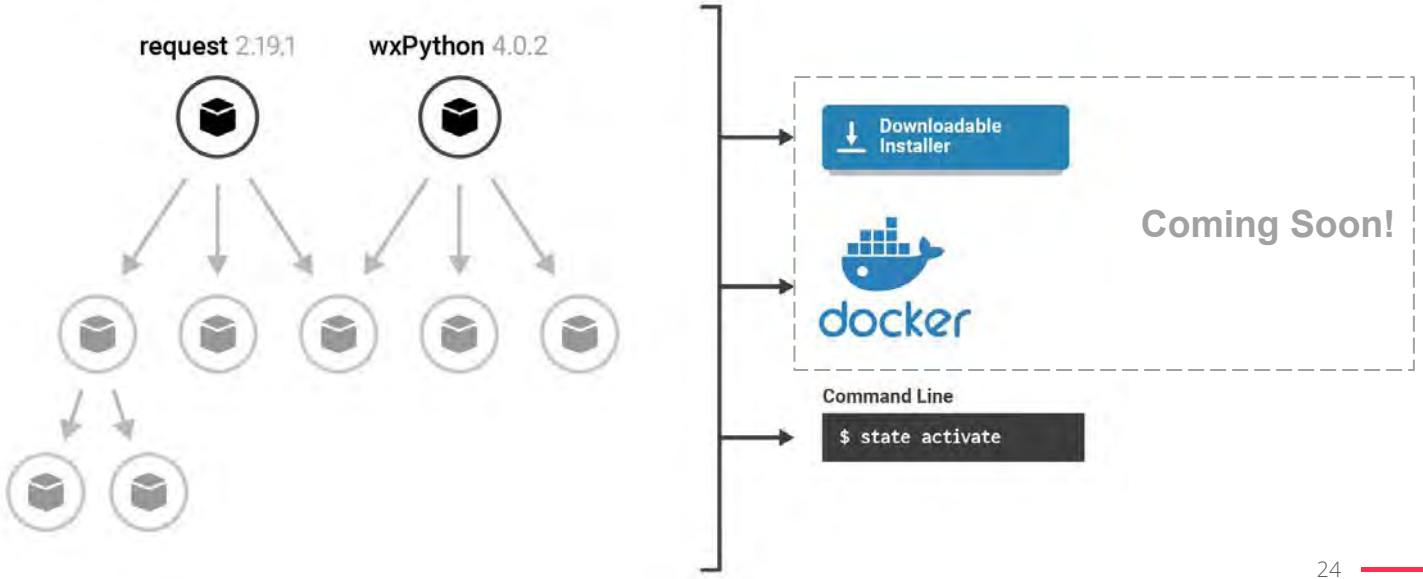
ActiveState

All Components are Built, Hermetically Sealed, in the Cloud from Source!



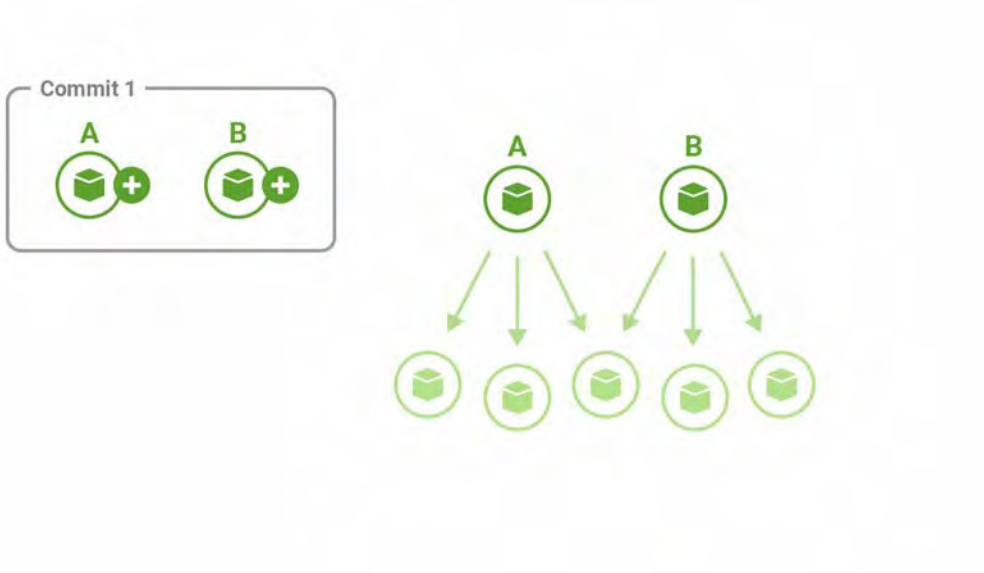
ActiveState

An Array of Distribution Mechanisms



ActiveState

We've Marries Dependency Management with Revision Control



ActiveState

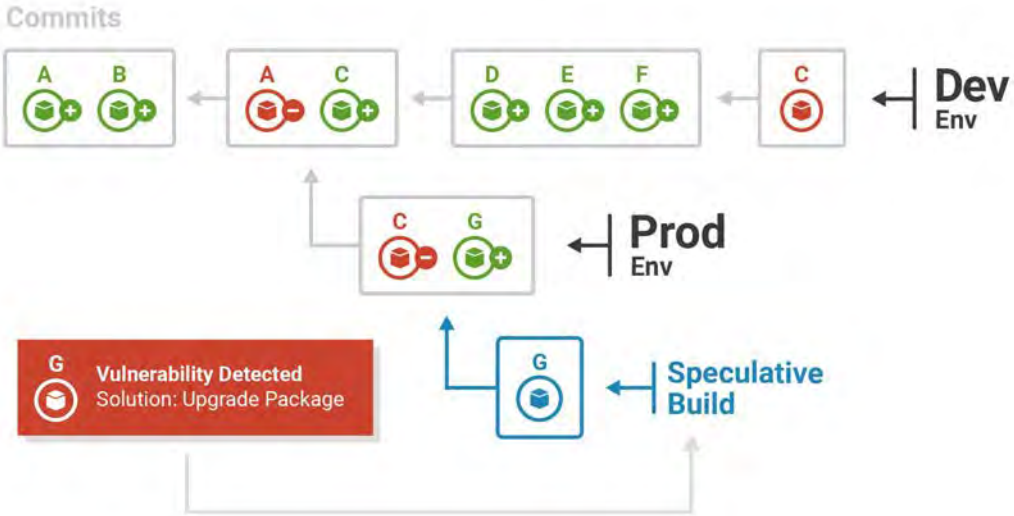
Every Dependency Change is Tracked, Including who Made it and Why

ActiveState

Includes Fine Grain Control of Dependencies by Operating Environment

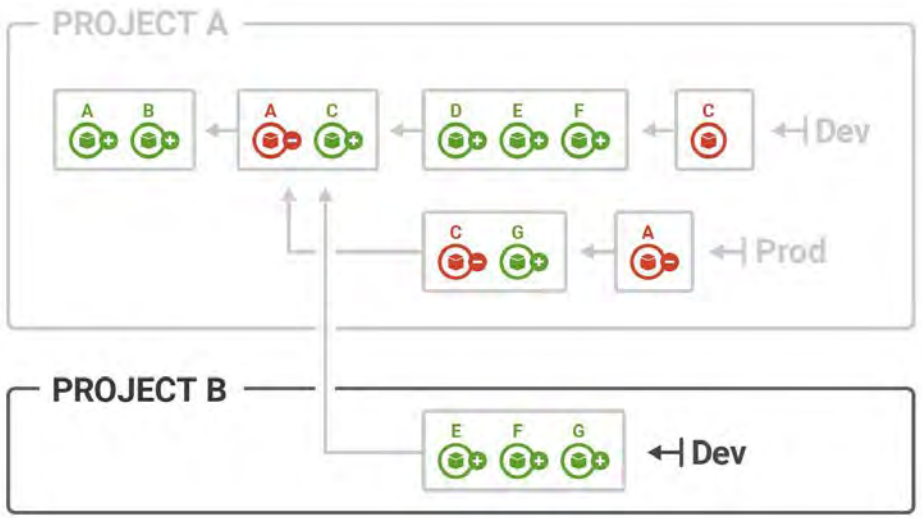
ActiveState

ActiveState Monitors for Risks and Suggests Upgrades Where Appropriate



ActiveState

Existing PRojects Can be Used as Templates



ActiveState

Platform Demo

ActiveState

Demo: Getting Out of Dependency Hell

https://www.youtube.com/watch?v=4WIL_cNVZ1E

ActiveState

Next Steps

Schedule a demo with our product experts:
https://www.activestate.com/get-demo/

Learn more about dependency management:
https://www.activestate.com/blog/dependency-resolution-optimization-activestates-approach/

Try the ActiveState Platform for free:
https://platform.activestate.com/

ActiveState

Webinar Feedback

Take our quick survey!
https://www.surveymonkey.com/r/dependency-hell