ActiveState®

Package Management for Teams

Today, organizations are forced to manage multiple tools, processes and systems for each language in their tech stack separately. And then manage each language runtime for each of their deployment environments separately, as well.

But if organizations had a single package management solution that could work with multiple OS's and languages, and even coordinate deployment across multiple systems, they could significantly reduce overhead and friction across their entire software development lifecycle.

The issue stems from that fact that open source language package managers (from Perl's CPAN to Python's pip to Ruby's RubyGems, etc) have all been designed to fulfill a very narrow use case: automate the process of installing, upgrading, configuring, and removing language components for *a single system* in a consistent manner.

But what if a package manager could perform its traditional role across *multiple languages for multiple users on multiple platforms?* After all:

Most developers work in teams

Most team members work on their preferred OS

Most applications are written in more than one language

The Evolution of Package Management

Once package management's scope is extended to multiple systems, OS's and languages, traditional use cases broaden to become:

Managing Team vs Individual Runtime Environments

Traditionally, each developer installs and manages their own runtime environment. But out-of-date Readme files, differing system requirements, and inconsistently applied patches/updates mean that a team's runtimes are rarely in sync, resulting in "works on my machine" issues. The result is a loss of productivity as developers spend time troubleshooting environments.

ActiveState lets you manage an entire team by providing a single source of truth for a project's runtime environment on the ActiveState Platform that can be shared, installed and updated with a single command. The result is a reduction in "works on my machine" issues so developers can spend more time coding and less time troubleshooting.

Managing End to End Deployments vs Each System

Without the right tools, there can often be differences between runtimes in your development vs test vs production environments, which can result in productivity loss and shipping delays. While containers help, you still need to ensure each container is consistently built with the correct runtime environment.

ActiveState's "single source of truth" approach can ensure all your containers and environments are built with the proper runtime environment in a consistent, reproducible manner.

Enabling Custom vs Standard Runtimes for CI/CD

CI/CD support for custom runtime environments is often poor, forcing DevOps to create work arounds with the unintended consequence that a CI/CD runtime environment may differ from the development and production runtime environments. The resulting inconsistencies create a proliferation of "works on my machine" issues.

By pre-building the runtime on the ActiveState Platform and using our Command Line Interface (CLI) the State Tool to deploy it seamlessly into a container, DevOps no longer needs to wrestle with custom runtimes across their CI/CD chain.
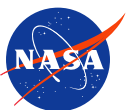
ActiveState's Package Management Solution

The ActiveState Platform provides a cross-platform build environment that can automatically build language runtimes (including linked C libraries), resolve dependency conflicts, and package them for Windows, Mac, and Linux. Today, you can use it to create a Python, Perl or Tcl runtime that acts as the single source of truth for all developers working on your projects, as well as DevOps managing your CI/CD processes. Create a free account at **platform.activestate.com.**

The ActiveState Platform's CLI, the State Tool simplifies common package and environment management tasks, including:

Runtime Management - configure and update runtimes managed on the ActiveState Platform, and then share them with a single command, ensuring everyone always has an up-to-date runtime environment.

Environment Management - install runtimes into virtual environments, isolating dependencies and eliminating conflicts so developers can more easily manage multiple projects without corrupting their system installation.

ActiveState®