JFrog Artifactory Fundamentals

Understanding What Binaries Are

Chris B. Behrens
Software Architect

@chrisbbehrens

Course Overview

Understanding what binaries are

- How to version them
- Some theoretical prep so our application is well-grounded

A tour of Artifactory

Integrate with a simple software build

- Globomantics, Inc.

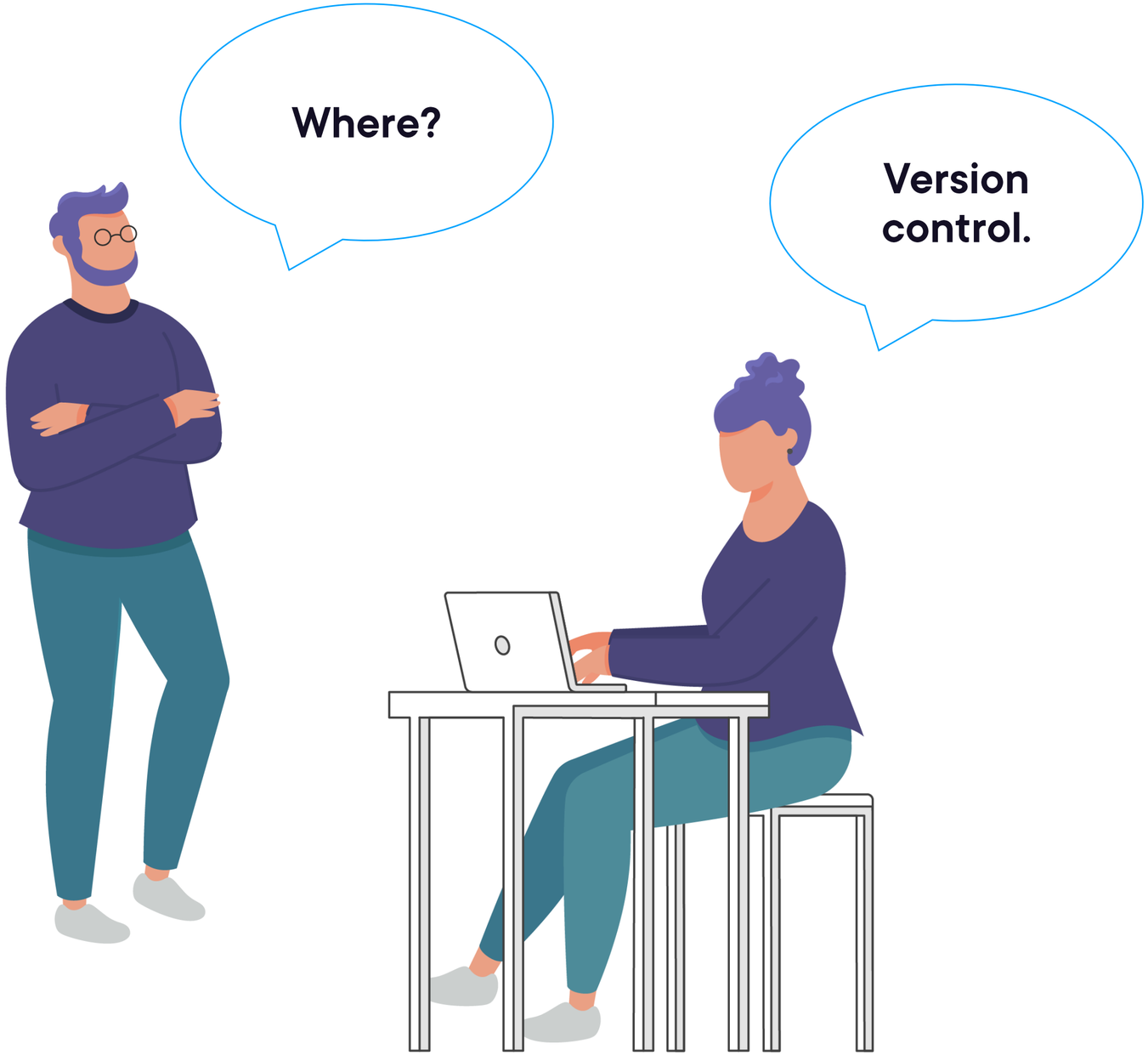
Packaging projects in a solution

Containers?

Packaging wild content

https://github.com/FeynmanFan/artifactory-ps

The Three Places



.gitignore

Excluded Products

Temp files are temporary

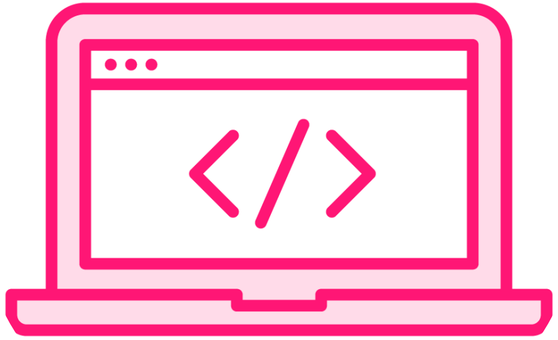
"Just rebuild the code"

Third-party packages

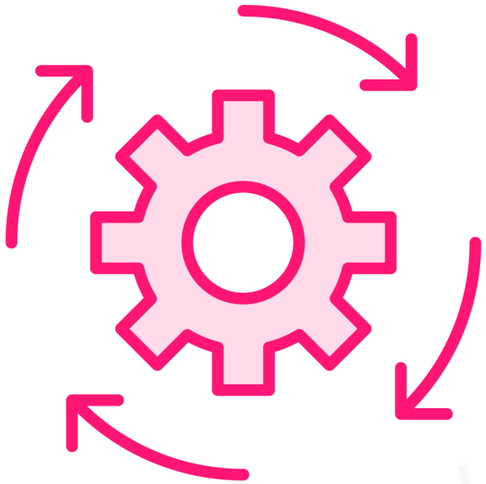
How *painful* is it to rebuild the code?

The Problem

273 projects
in the solution



You couldn't
debug locally



The build server
was a bottleneck

The Solution

Package these projects

Most had not been changed in years

I packaged the projects from the bottom up

Build-per-project

Push to Artifactory

From then on, pull from Artifactory instead of building

The three places

- Version control
- A secrets store
- An artifact repository

Only build when things *change*

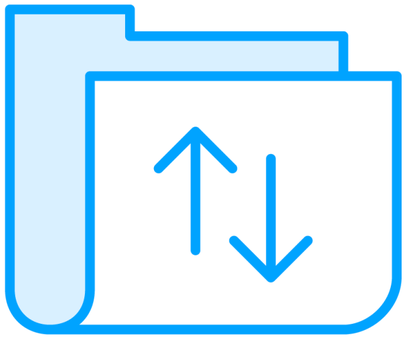
Why Not Version Control?

Why not have the build check them in?

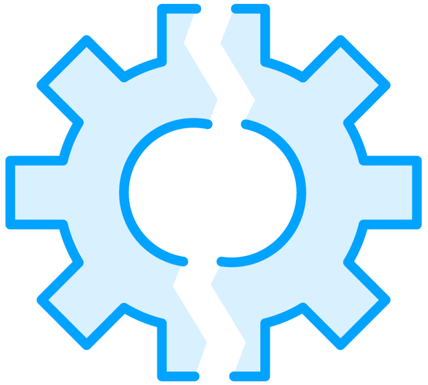
VCS are highly optimized for *text*

Binaries bog down your VCS

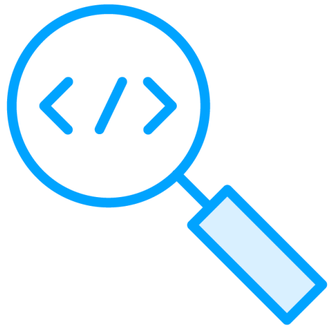
Another Reason



Restore the package



VCS are not built to restore big binaries



Figuring out the version is non-trivial

We need an artifact store because we want to create binary products at only one time – when the code that generates them has changed.

Many projects hadn't changed in years...

But some were changing often

Different projects can have different lifecycles

Breaking them out can yield better final build durations

By making them happen at different times

What Packages Are

Zip Files

A compressed file

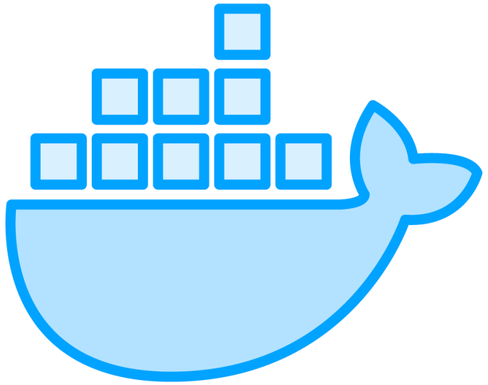
You can generally just rename the packages to .zip and examine the contents

The point of the files is transport

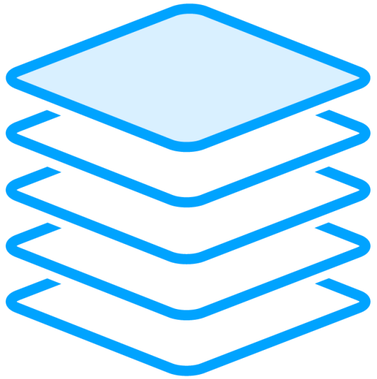
Manifest - a file which describes the contents

Subfolders which contain the binaries and other associated resources

Container Formats



Docker images are like zip files



Each layer is a zip, more or less

A compressed file full of binaries, folders, and text files which supply information about the definition of the package.

Demo

A NuGet package

A C# project in Visual Studio

Rename the package to dot zip

Drill into what we find inside

Package Wrap-up

Compressed file optimized for transport from the package feed to a client

Package manifest file which outlines the structure of the package

Possible metadata to list dependencies that the package depends on

Package Feed Basics

Primary Package Feeds

nuget.org

npmjs.org

pypi.org

rubygems.org

Package Feeds and Intellectual Property

These are the biggest feeds

Your packages may contain proprietary information

With the big feeds, it's free and open to the world

And most of the time, your packages are useless to everyone else

Artifactory – your private artifact store

Push to Artifactory, and configure your tools to pull from it

Dependency Confusion

Project

BigSoftware.Application.Provider → nuget.org

Globomantics.Internal.ProprietaryLogic → Private feed

The Attack

The attacker gains a list of dependencies

A poison version of the internal package uploaded to the public feed

The package manager confuses the poison package for the real one

The Effect

npm looks for the highest version of a package to resolve the confusion – that's bad

https://bit.ly/45H48d5

A White-hat attack – the good guys

I've never seen it happen in the wild

We've been lucky so far

Don't use multiple feeds

But what about our public dependencies?

"Upstream feed" or "Downstream feed" – I'll 'splain later

Demo

Look at our Globomantics project

Get it introduced for the course

Look at the Visual Studio Package Manager

Check on a package it relies on

Update its version

A Quick Note on What We're Focusing On

Windows and Linux "Package Managers"

APT – Advanced Packaging Tool

winget – Windows' very tardy entry into the field

Not this kind of package manager

Those are OS package managers

Managing the packages of a software project

That's what Artifactory provides

Summary

What we mean by “binaries”

The three places

- Version control
- A secrets manager
- An artifact repository

Learned what packages are

A few demos on working with packages in Visual Studio