9 Reasons

To Use a Binary Repository for Front-End Development with Bower

White Paper

Introduction

The availability of packages for front-end web development has somewhat lagged behind back-end systems. Getting a package (such as jquery, or Bootstrap) was an arduous process that included scouring the internet, downloading a package and verifying compatibility with other packages in a project, manually unzipping and inserting the relevant files into projects, and then repeating the whole process to manage updates. The emergence of Bower has taken the pain out of doing front-end web development with packages. Now all you need to do is configure the packages you need in your *bower.json* file and Bower does the rest. Or does it?

Bower may be able to find the right packages for you, but what happens if you have a connectivity problem at 2 a.m. when you're trying to beat a deadline? And why does Bower seem to have so many broken pointers? How do you make sure that the ten people in your organization using a package are all using the same version? It's issues like these that Artifactory solves for you.

Artifactory's support for version control systems enables it to proxy the Bower registry for you in order to overcome the issues related to using packages. All you need to do is create a remote repository to proxy any Git repository you want to access, and then collect all those remote repositories under a single **virtual repository**. Now, just point your Bower commands at Artifactory. Artifactory implements the Bower API and transparently replaces the Bower Registry for you. When you request a component, Artifactory will delegate that request to Bower, get the right pointer to GitHub, and then download and zip the requested files into a single package along with its corresponding *bower.json* file. For you, the developer, all this happens transparently in the background so you can continue to be productive while Artifactory does all the work of getting your packages and keeping them updated.

This white paper describes the benefits of using Bower together with Artifactory.

Virtual Repositories

A virtual repository encapsulates any number of local and remote repositories and represents them as a unified repository accessed from a single URL. It gives you a way to manage which repositories are accessed by developers since you have the freedom to mix, match and modify the actual repositories included within the virtual repository. You can also optimize artifact resolution by defining the underlying repository order so that Artifactory will first look through local repositories, then remote repository caches, and only then Artifactory will go through the network and request the artifact directly from the remote resource. For the developer it's simple. Just request the package, and Artifactory will safely and optimally access it according to your organization's policies.

Learn more >

#1 Reliable and consistent access to components

Bower has several potential points-of-failure. If either the Bower Registry or GitHub go down you can't download or update your packages. Similarly, networking issues may bring your work to a grinding halt.

Artifactory overcomes these issues by caching your front-end packages on-demand in a **remote repository** so you are independent of GitHub, Bower Registry and the network. If any of these go down, your packages are still available and you may continue to work oblivious of any external issues.

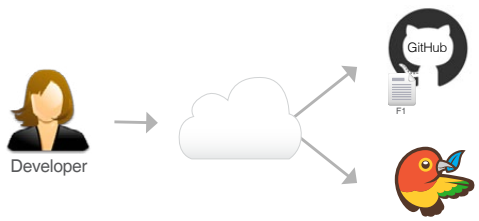
Remote Repositories

A remote repository serves as a caching proxy for a repository managed at a remote site like GitHub. Artifacts are stored and updated in remote repositories according to various configuration parameters that control the caching and proxying behavior.

Learn more >

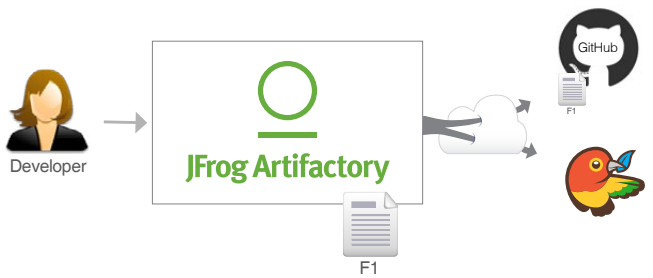
Before

Developer accesses Bower Registry and GitHub through the internet



After

First time a file accessed, Artifactory downloads it. From then on, it is available locally to all developers

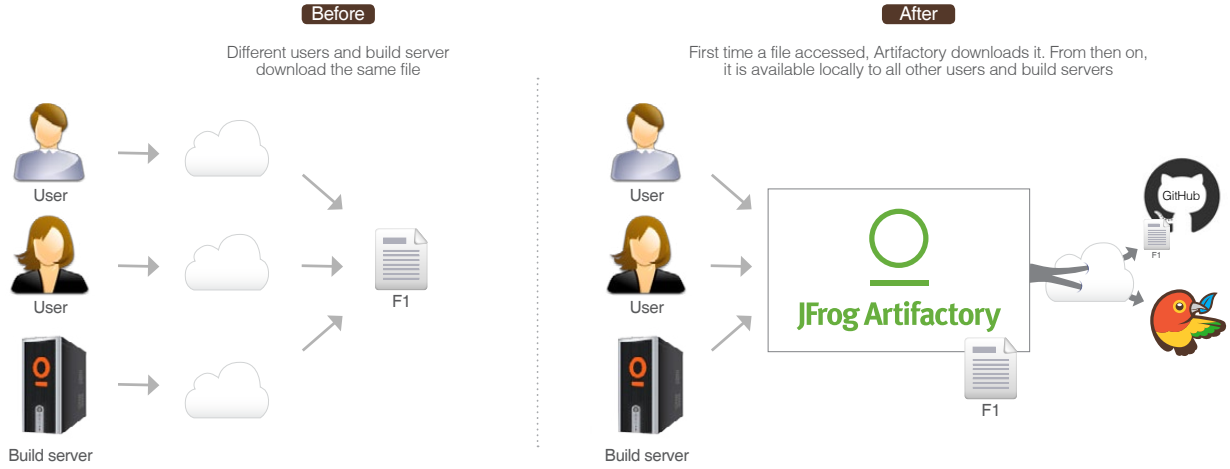


#2 Reduce network traffic and optimize builds

Since much of your code is likely to be assembled rather than built, you want to make sure that your usage of packages downloaded from external resources is optimized. It makes no sense for two (or two hundred) developers using the same package to download it separately.

In addition to reliability, another benefit of remote repositories is reduced networking. Once a package has been downloaded, it is then locally available to all other developers in the organization (thus reducing network traffic). Naturally, this is all transparent to the individual developer. Once packages are accessed through Artifactory, the developer can get on with what she does best and leave the rest to Artifactory.

If we look at network traffic from the point of view of a **build server**, the benefits are clear. A typical project may depend on tens if not hundreds of packages from external resources. For the server to build these projects, all remote artifacts must be available to the server environment. Downloading all those required artifacts may generate Gigabytes of data traffic on the network which takes a significant amount of time delaying the build process. By caching remote artifacts locally, the build process is much quicker and incurs much less networking.



#3 Full Support for Docker

As Docker technology continues to evolve, its usage continues to grow. If you are not yet using Docker in your organization, it is likely you will do so soon. So now, in addition to managing Bower packages, you also need to manage Docker images. But there's no need to onboard and maintain another tool. Artifactory is a fully-fledged Docker repository supporting all Docker Registry APIs. This allows the Docker client to work with Artifactory directly, presenting several benefits for enterprise Docker users.

Using local repositories, you can **distribute and share images** within your organization to make managing images between different teams easy. You can even replicate your Artifactory Docker repositories to remote instances of Artifactory to share images with colleagues in geographically distant sites.

Artifactory offers **fine-grained access control** to your organization's images with secure "docker push" and "docker pull" effectively providing **secure, private Docker repositories** that exceed the security offered by Docker Trusted Registry.

Using Artifactory, instead of private repositories on Docker Hub, removes any issues related to internet connectivity resulting in **reliable and consistent access to images**. And with Artifactory running in a **High Availability configuration** you get system stability and availability of your Docker images that is unmatched in the industry.

Artifactory's **smart search** makes it easy to find any Docker image stored in your system. Full support for the Docker Registry API supports basic search with the Docker client, but Artifactory offers much more. Built in searches answer common needs with single-click operations, custom properties provide the flexibility to meet a variety of specific needs, and Artifactory Query Language offers a simple way to formulate complex queries letting you find images based on any set of criteria.

Whether you're already on board with Docker or just evaluating how to introduce it to your organization, once you're using Artifactory to manage your Bower packages, you're already covered for Docker images.

#4 Security

The Bower Registry does not inherently provide security or privacy. If you want to control access to your components you need to use private repositories on GitHub. But the result of working like this is that there are many pointers in the Bower Registry that come up blank for most users since they point to someone else's private repository. And then, do you really want to make pointers to your private GitHub repositories publicly available?

Artifactory offers a more complete security solution for Bower. As a first line of defense, Artifactory lets you use naming patterns to define "Excludes" and "Includes" for access so you can control which packages can even be cached in any particular remote repository. Then you can assign different sets of permissions to users and groups to control access to each repository. You can even use Artifactory's integration with LDAP, Active Directory, SAML, Crowd and others to control access to your servers. Effectively Artifactory becomes your own private and secure internal Bower registry.

#5 Smart Search and Artifactory Query Language (AQL)

Given the multitude of packages in your system, finding something specific can sometimes get quite complex.

Artifactory provides you with flexible search capabilities to help you find the packages stored in your system. First, Artifactory supports the Bower API so your most basic search for packages is done in exactly the same way you are used to using the Bower Registry. Then, you can find packages based on any combination of inherent attributes such as name, version, timestamp, **checksum** and more. But Artifactory offers much more with a set of common built-in searches. For example, you can easily find the "latest" version of any package without having to specify a version number. Artifactory also lets you assign any set of custom properties to your components, which can later be used for search. For example, you can tag all the specific versions of components used in a product release with a "released" property to easily reproduce the released version later on. But the full power of search comes with the complete flexibility of AQL. Using **AQL**, you can define search queries to any level of complexity needed to extract just the right packages you are looking for.

With these capabilities, Artifactory lets you search for packages using virtually any set of rules relevant to your workflow.

Checksum-based search

Searching for a package by its checksum is a powerful feature supported by Artifactory thanks to a unique method of storing files by their checksum. Even if a binary has been renamed, moved or even deployed outside of your organization, you can trace it back to the original version and obtain its complete build information. Simply run the package through a checksum tool (both MD5 and SHA1 are supported) and run a "Checksum" search in Artifactory to retrieve the original version.

Learn more >

Artifactory Query Language (AQL)

AQL is flexible query language that offers a simple way to formulate complex queries to search through your repositories using any number of search criteria, filters, sorting options and output fields. It takes full advantage of the database underlying Artifactory's unique architecture, and gives you unlimited degrees of freedom to formulate exactly the right query to find those very specific packages you are searching for. This is something that no other Binary Repository can offer.

Learn more >

#6 Distribute and share packages across your organization

As already mentioned, you want to make sure that packages originating from remote resources only get downloaded once, and are then shared across your organization. The same principle holds true for your own proprietary packages which you want to be able to securely share with others in your organization.

Using **local repositories**, Artifactory gives you a central location to store your internal packages. When all teams know that any package can be accessed from a single URL, access to local packages and managing dependencies between the different teams becomes very easy. But what if you want to share your packages with colleagues who are in geographically remote sites of your organization?

Artifactory supports replication of your repositories to another instance of Artifactory which is outside of your local network. Replicated repositories are automatically synchronized with their source periodically so that your packages can be made available to different teams wherever they may be located around the world.

Local Repositories

Local repositories are physical, locally-managed repositories into which you can deploy artifacts. Typically these are used to deploy internal and external releases as well as development builds, but they can also be used to store packages that are not widely available on public repositories such as 3rd party commercial components. Using local repositories, all of your internal resources can be made available from a single access point across your organization from one common URL.

Learn more >

#7 System stability and reliability with Artifactory High Availability

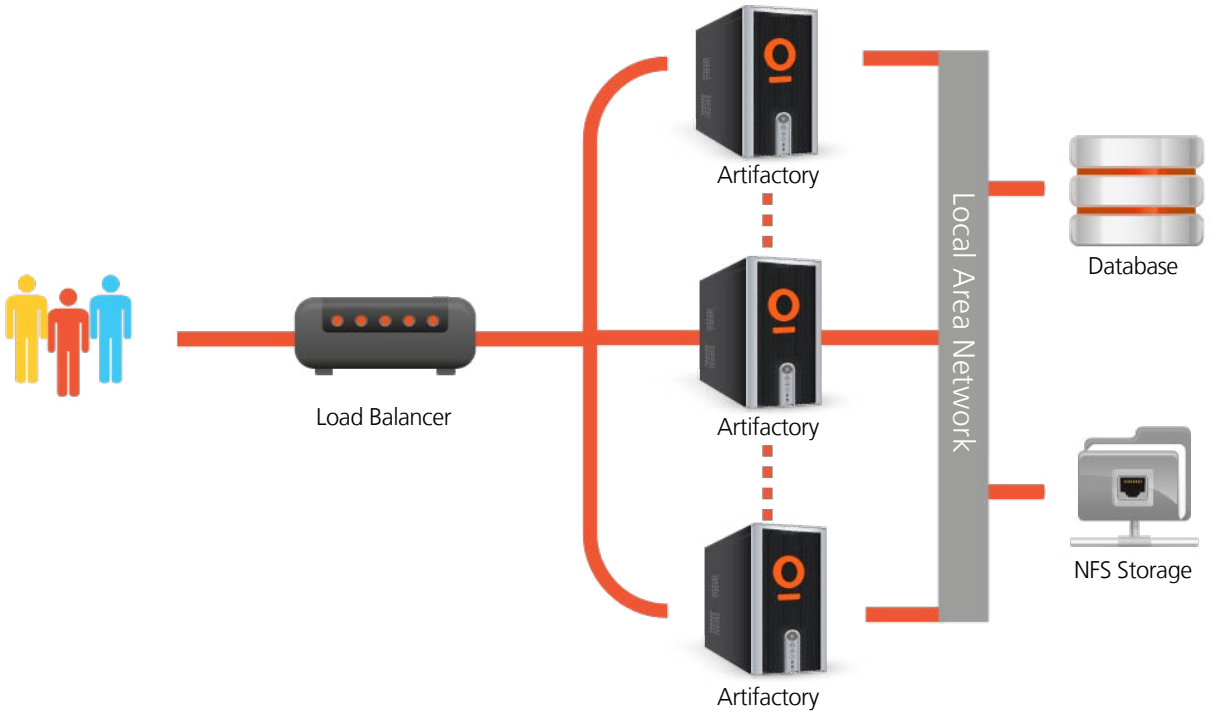
Playing such a central role in the management of binaries, your Binary Repository can become a mission-critical component of your organization meaning that any downtime can have severe consequences.

Artifactory supports a **High Availability** network configuration with a cluster of 2 or more Artifactory servers on the same Local Area Network. A redundant network architecture means that there is no single-point-of-failure, and your system can continue to operate as long as at least one of the Artifactory nodes is operational. This maximizes your uptime and can take it to levels of up to “five nines” availability. Moreover, your system can accommodate larger load bursts with no compromise to performance. With horizontal server scalability, you can easily increase your capacity to meet any load requirements as your organization grows. And by using an architecture with multiple servers, Artifactory HA lets you perform most maintenance tasks with no system downtime.

High Availability Systems

Systems that are considered mission-critical to an organization can be deployed in a **High Availability** configuration to increase stability and reliability. This is done by replicating nodes in the system and deploying them as a redundant cluster to remove the complete reliability on any single node. In a High Availability configuration there is no single point-of-failure. If any specific node goes down the system continues to operate seamlessly and transparently to its users through the remaining, redundant nodes with no down time or degradation of performance of the system as a whole.

Learn more >



#8 Maintenance and Monitoring

With current use of build servers and CI systems, the number of artifacts you generate can grow very quickly. Without proper management, your systems can quickly get clogged with old and irrelevant artifacts.

Artifactory keeps your system organized and free of clutter with automatic, timed cleanup processes. With a few simple settings, you can schedule tasks to clean up old builds and unused packages. You can set restrictions on and monitor disk space usage or define “watches” to receive an alert whenever there is a change to your most critical binaries. And with an extensive REST API, Artifactory can support virtually any rule-based cleanup protocol you would want to implement in your organization’s scripts.

#9 A Universal, End-to-End Solution For All Binaries

No single packaging format or technology is sufficient to support development in a modern organization. There is a multitude of formats, a variety of build tools, different continuous integration systems and other technologies that go into building a flexible and maintainable software development ecosystem. Managing binaries for all the different packaging formats and integrating with all the moving parts of the ecosystem can become a maintenance nightmare.

Artifactory was designed from the ground up to fit in with any development ecosystem. Uniquely built on checksum-based storage, Artifactory supports any repository layout and can, therefore, provide native-level support for any packaging format. Essentially, regardless of the packaging format you are using, Artifactory can store and manage your binaries, and is transparent to the corresponding packaging client. The client works with Artifactory in exactly the same way it would work with its native repository. For example, If you are working with Bower, Artifactory proxies GitHub (or any other public Bower repository), lets you store and manage your own images in local Docker repositories, and works transparently with the Docker client. If you are working with Bower, Artifactory proxies any public remote Bower repository, lets you store your own packages in local Bower repositories, and works transparently with the Bower client. Similarly for npm, Vagrant, NuGet, Ruby, Debian, YUM, Python and more.

But development is only one end of the software delivery pipeline. Before a package makes it into a product, it needs to go through processes of build and integration. There are many build and integration tools on the market, but there is only one product that works with them all. Through a set of plugins, Artifactory provides tight integration with popular CI systems available today such as Jenkins, Bamboo and TeamCity. These systems use Artifactory to supply artifacts and resolve dependencies when creating a build, and also as a target to deploy build output. And to support cloud-based CI systems on which you are not able to apply plugins, Artifactory provides plugins for the build tools you use (such as Maven and Gradle) which ultimately provides the same level of build automation. That takes care of development and deployment, but what about distributing your software once it's ready for consumption. That's where Bintray comes in.

Bintray is JFrog's download center in the cloud offering rapid downloads, fine-grained access control, detailed stats and logs and an extensive REST API. Promoting releases for distribution from Artifactory is a matter of a single-click or API call. Like Artifactory, Bintray is package-agnostic and works seamlessly with all the different package clients, so it can be fully integrated into any continuous integration/ continuous delivery ecosystem.

Artifactory is a universal repository. It is the single tool that sits in the center of your development ecosystem and "talks" to all the different technologies, increasing productivity, reducing maintenance efforts and promoting automated integration between the different parts. Together, Artifactory and Bintray are the central components of a fully-automated software distribution pipeline.

Summary

Artifactory's support for Bower streamlines front-end development letting you concentrate on writing code without having to worry about security or availability of the Bower Registry. Additional features such as smart search, AQL, High Availability, maintenance, monitoring and more help you work more efficiently and speed up development cycles, ultimately getting your product out to market as quickly as possible.

For more information on how Artifactory can boost your organization's performance please contact us at info@jfrog.com