How do you remotely update Java applications?

Asked 16 years, 3 months ago Modified 4 years, 5 months ago

Viewed 18k times

28

We've got a Java server application that runs on a number of computers, all connected to the Internet, some behind firewalls. We need to remotely update the JAR files and startup scripts from a central site, with no noticeable interruption to the app itself.

The process has to be unattended and foolproof (i.e. we can't afford to break the app due to untimely internet outages).

In the past we've used a variety of external scripts and utilities to handle similar tasks, but because they have their own dependencies, the result is harder to maintain and less portable. Before making something new, I want to get some input from the community.

Has anyone found a good solution for this already? Got any ideas or suggestions?

Just to clarify: This app is a server, but not for web applications (no webapp containers or WAR files here). It's just an autonomous Java program.

java release-management

Share

Improve this question

Follow

edited Nov 25, 2014 at 18:04

Robert Harvey

181k ● 48 ● 346 ● 512

asked Sep 24, 2008 at 4:24

David Crow

16.2k ● 8 ● 43 ● 34

@Nzall For the updated bounty proposed on this question, wouldn't using custom JRE solve it to some extent? and this Q&A might be related – Naman Jul 20, 2020 at 10:57

1 @Naman I'm not sure how that'll work. We want to install it remotely from our server machine to a client machine (Karaf core with libraries and custom code), and we want to push this without people having to manually copy over the installer and run the install script. We have some customers with 3 dozen agents that need to be updated, so we want to make upgrading more smooth. – Nzall Jul 20, 2020 at 20:06

In 2020 distributed deployment management and performing rolling releases are two of the big problems that Kubernetes and other container orchestrators solve. Probably too heavyweight to retrofit your servers as a k8s cluster just for one application though. Can you not accomplish automatic redeployment with an ssh script run from a CI server job? – Richard Woods Jul 21, 2020 at 14:41

9 Answers

Sorted by: Highest score (default)

You didn't specify the type of server apps - I'm going to assume that you aren't running web apps (as deploying a WAR already does what you are talking about, and you very rarely need a web app to do pull type updates. If you are talking about a web app, the following discussion can still apply - you'll just implement the update check and ping-pong for the WAR file instead of individual files).

You may want to take a look at jnlp - WebStart is based on this (this is a client application deployment technology), but I'm pretty sure that it could be tailored to performing updates for a server type app. Regardless, jnlp does a pretty good job of providing descriptors that can be used for downloading required versions of required JARs...

Some general thoughts on this (we have several apps in the same bucket, and are considering an auto-update mechanism):

1. Consider having a bootstrap.jar file that is capable of reading a jnlp file and downloading required/updated jars prior to launching the application.
2. JAR files *can* be updated even while an app is running (at least on Windows, and that is the OS most likely to hold locks on running files). You can run into problems if you are using custom class loaders, or you have a bunch of JARs that might be loaded or unloaded at any time, but if you create mechanisms to prevent this, then overwriting JARs

then re-launching the app should be sufficient for update.

3. Even though it is possible to overwrite JARs, you might want to consider a ping-pong approach for your lib path (if you don't already have your app launcher configured to auto-read all jar files in the lib folder and add them to the class path automatically, then that's something you really do want to do). Here's how ping-pong works:

App launches and looks at lib-ping\version.properties and lib-pong\version.properties and determines which is newer. Let's say that lib-ping has a later version. The launcher searches for lib-ping*.jar and adds those files to the CP during the launch. When you do an update, you download jar files into lib-pong (or copy jar files from lib-ping if you want to save bandwidth and the JAR didn't actually change - this is rarely worth the effort, though!). Once you have all JARs copied into lib-pong, the very last thing you do is create the version.properties file (that way an interrupted update that results in a partial lib folder can be detected and purged). Finally, you re-launch the app, and bootstrap picks up that lib-pong is the desired classpath.

4. ping-pong as described above allows for a roll-back. If you design it properly, you can have one piece of your app that you test the heck out of and then never change that checks to see if it should roll-back a given version. That way if you do mess up and deploy something that breaks the app, you can

invalidate the version. This part of the application just has to delete the version.properties file from the bad lib-* folder, then re-launch. It's important to keep this part dirt simple because it's your fail safe.

5. You can have more than 2 folders (instead of ping/pong, just have lib-yyyymmdd and purge all but the newest 5, for example). This allows for more advanced (but more complicated!) rollback of JARs.

Share Improve this answer

Follow

answered Sep 24, 2008 at 5:18

Kevin Day

16.4k ● 8 ● 48 ● 71

Yes, your assumption is correct--we're not using WAR deployment. The app is a server, but not for web applications. Thanks for the info on JNLP. I'm now looking into it...

– David Crow Sep 24, 2008 at 5:32

2 2020 UPDATE: for anyone coming across this answer. Java Web Start (jnlp) is now depreciated. oracle.com/java/technologies/javase/v9-deprecated-features.html – Philip Attisano Jul 24, 2020 at 11:32

6

You should definitely take a look at OSGi, it was created just for these cases (especially for embedded products) and is used by a large number of companies. You can update jar "bundles", add and remove them, while the app is running. I haven't used it myself, so I don't know about the quality of the open source frameworks/servers, but here is a bunch of useful links to get you started:

http://www.osgi.org/Main/HomePage
http://www.aqute.biz/Code/Bnd
http://blog.springsource.com/2008/02/18/creating-osgi-bundles/
http://blog.springsource.com/
http://www.knopflerfish.org/
http://felix.apache.org/site/index.html

Share Improve this answer

Follow

answered Sep 24, 2008 at 6:27

Lars Westergren
2,129 ● 15 ● 26

4

I'd recommend Capistrano for multi-server deployment. While it is built for deploying Rails apps, I've seen it used successfully to deploy Java applications.

Link: Capistrano 2.0 Not Just for Rails

Share Improve this answer

Follow

answered Sep 24, 2008 at 4:29

Redbeard
1,008 ● 6 ● 8

4

It's very hard to make the update atomic, especially if you have any database updating to do.

But, if you don't, what you can do is first make sure your application can run from a relative path. That is, you can put your app in some directory, and all of the important files are found relative to that location, so that that your actual installation location is not really important.

Next, duplicate your installation. Now, you have the "running" version and you have the "new" version.

Update the "new" version using whatever tech you like (FTP, rsync, paper tape, whatever floats your boat).

Verify your installation (checksums, quick unit tests, whatever you need -- even start it up on a test port if you like).

When you are happy with the new installation, down the original running instance, RENAME the original directory (mv application application_old), rename the NEW directory (mv application_new application), and start it back up.

Your down time is reduced to server shut down and start up time (since rename is "free").

If, by chance, you detect a critical error, you have your original version still there. Stop the new server, rename it back, restart the old. Very fast fall back.

The other nice thing is that your service infrastructure is static (like your rc scripts, cron jobs, etc.) since they point to the "application" directory, and it doesn't change.

It can also be done with soft links instead of renaming directories. either way is fine.

But the technique is simple, and near bullet proof if your application cooperates.

Now, if you have DB changes, well, that's completely different nasty issue. Ideally if you can make your DB changes "backward compatible", then hopefully the old application version can run on the new schema, but that's not always possible.

Share Improve this answer

Follow

answered Sep 24, 2008 at 6:04

Will Hartung

118k ● 20 ● 133 ● 207

3

Jars cannot be modified while the JVM is running on top of it and will result in errors. I have tried similar tasks and the best I came up with is making a copy of the updated Jar and transition the start up script to look at that Jar. Once you have the updated Jar, start it up and wait on the old Jar to end after giving it the signal to do so. Unfortunately this means a loss of GUI etc. for a sec but serializing most of the structures in java is easy and the current GUI could be transferred to the updated application before actually closing (some things may not be serializable though!).

Share Improve this answer

Follow

answered Sep 24, 2008 at 4:30

bmeck

416 ● 2 ● 4

3

I would use ansible to distribute jars to multiple servers and run update scripts.

For an update to be unnoticeable for users the application would have to stop the old instance and bring up a new one very quickly, in which java apps aren't very good.

There are ways to make update without any downtime, but none of them is free. Choosing a solution you should consider what downtime is acceptable to you and what cost can you accept to achieve it.

I see two options:

1. Rolling update, which means each time you update you spin up an instance with the new version, then validate if it is working correctly, and if it is, you kill the old version. This solution will require some kind of proxy (eg. haproxy) which will direct the traffic to the valid instances.
2. Optimizing application start-up time.

As Will Hartung mentioned you will also have to consider external dependencies like database schema which you might also want to update. In order to get seamless updates, you might also want to do a rolling update on a database. It would require you to do not introduce any breaking changes between any two consecutive releases of the application. Eg. when you want to delete a column, in the first release you remove all references to the column in an application and in next release you are allowed to delete the column in the database.

 answered Jul 21, 2020 at 18:04

Follow

Cezary Butler
857 • 7 • 23

We don't mind if the application has temporary downtime. These are agents (as in software agents) that are used to remotely start scripts for devops purposes (automatic tests, deploying other software). They aren't expected to be up full time because they're only used when the customers need them, and our clients will hopefully only have to update them every couple of months. – Nzall Jul 21, 2020 at 19:34

In that case, Ansible seems a reasonable choice. The general idea of how it works is that you write a playbook which describes the desired system state, and the ansible-playbook command brings all hosts within inventory to that state. Ansible has very few requirements, at least when running on Unix systems. Ansible has to be installed only on the host that initiates the update process, on the other end only ssh and Python is required. However certain plugins may have their own requirements. We use Ansible to manage different kinds of applications and are happy with it.
– Cezary Butler Jul 22, 2020 at 7:59

Can we ship Ansible as part of a commercial product though? And can we use it to deploy to a Windows environment?
– Nzall Jul 22, 2020 at 19:40

It is possible to use Ansible to deploy to a Windows environment, however, I don't have experience doing so. Here you'll have some information on how it is different from deploying to Unix systems: docs.ansible.com/ansible/latest/user_guide/windows_usage.html – Cezary Butler Jul 22, 2020 at 20:57

Regarding licensing. It depends on how the deployment would work. It's absolutely fine if you would use ansible as a command-line tool to deploy to your customers' environment. Ansible doesn't have to be installed on your customers' devices. However, if you'd like to deliver some kind of tool to

your customers, so they'd be able to deploy on their environments. Such tools would've to be open-sourced on GPL license. – Cezary Butler Jul 22, 2020 at 21:07

2

I believe you can hot-deploy JAR files if you use an OSGi-based app server like SpringSource dm Server. I've never used it myself, but knowing the general quality of the Spring portfolio, I'm sure it's worth a look.

Share Improve this answer

Follow

answered Sep 24, 2008 at 5:25

Andrew Swan

13.6k ● 23 ● 73 ● 99

0

We use Eclipse which the update system of OSGi, and our experience is very good.

Recommended!

Share Improve this answer

Follow

answered Sep 24, 2008 at 7:36

Lars A Frøyland

126 ● 2

0

The latest version of Java Web Start allows for injecting an application in the local cache without actually invoking the program and it can be marked as "offline". Since the cache is what is used to invoke the program, it will be updated for the next run only. For this to work you will

most likely need jars with version numbers in their name (e.g. our-library-2009-06-01.jar).

Share Improve this answer

Follow

answered Jun 1, 2009 at 16:03

Thorbjørn Ravn Andersen

75.3k ● 34 ● 199 ● 352