

Building a VIVO distribution for other servlet containers

- Before the build
 - Required software
 - Logging properties for VIVO
 - Logging properties for Solr
 - The build.properties file
 - Running the build script
 - 3. Deploying Solr
 - 4. Deploying VIVO

Some sites prefer to use a container like GlassFish or JBoss, instead of Tomcat

The simple installation process for VIVO tells how to build and deploy VIVO into a Tomcat servlet container. This document tells how to build VIVO so it can be used in any servlet container that supports the Java Servlet 2.4 Specification.

This process will produce:

- a WAR file for the VIVO application,
- a WAR file for the Solr application,
- a TAR file for the basic Vitro home directory, ready for configuring,
- a TAR file for the Solr home directory, configured to work with VIVO.

These artifacts can then be installed into a servlet container (or more than one), and configured to work together.

The configuration includes the customary `runtime.properties` file in the Vitro home directory. It also requires items that tell the VIVO application and the Solr application how to find their respective home directories. These items are specific to the servlet container. However, they are described so you can translate them to your container of choice.

Start by reading the customary [instructions for installing VIVO](#), taking note of the Tomcat-specific sections. This document will consist mostly of comparisons to those instructions.

Before the build

Required software

Tomcat is not required. You can use any servlet container that supports the Java Servlet 2.4 Specification.

Logging properties for VIVO

The logging properties for VIVO are determined by the file `[vitro-core]/webapp/config/log4j.properties`. (*Note: if `debug.log4j.properties` exists, it will override `log4j.properties`*).

Notice how the location of the log file is determined:

```
log4j.appender.AllAppender.File=${catalina.home}/logs/${webapp.name}.all.log
```

The filename of the log file is based on the `webapp.name` property found in the `build.properties` file. This substitution is made during the build process. The path to the log file is based on the system property `catalina.home` which is set by Tomcat at runtime.

You will likely want to change this line -- making it an absolute path, or basing it on some other system property. Notice that ant will substitute properties such as `${webapp.name}` during the build process. The name of a system property requires a second dollar sign, e.g. `${catalina.home}` to protect it from ant. Ant will remove the second dollar sign, but will not try to substitute a value for the property. At runtime, when Log4J reads the properties file, it will substitute the matching system property.

Logging properties for Solr

The logging properties for Solr are determined by the file `[vitro-core]/webapp/config/solr/logging.properties`.

As with VIVO, the location of the log file is based on the system property `catalina.home` which is set by Tomcat at runtime.

```
org.apache.juli.FileHandler.directory = ${catalina.base}/logs
```

Note that only one dollar sign is used, since Ant does not attempt to substitute properties in this file.

The syntax of the Solr logging properties is different from the syntax for VIVO. This is because VIVO uses Log4J as a back end for the Commons Logging framework, while Solr uses JULI as a back end for SLF4J.

Note: Solr is able to do this because Tomcat provides the JULI framework by default. Other servlet containers may require JULI to be installed. If you encounter this as an issue, please share your experience with the VIVO Development mailing list (vivo-dev-all@lists.sourceforge.net), so we can improve our distribution.

The build.properties file

The standard installation instructions specify that these properties are required in build.properties

- vitro.core.dir
- webapp.name
- tomcat.home
- vitro.home

However, if you are building with ant distribute, then only these are required:

- vitro.core.dir
- webapp.name

tomcat.home is ignored by the distribute target. You may choose to specify vitro.home in build.properties, or later, when you deploy VIVO (see [Deploying VIVO](#)). If you specify vitro.home in build.properties, you can override it when you deploy, but you will receive a warning when VIVO starts, saying that vitro.home has been specified twice.

Running the build script

To build VIVO for other servlet containers, you will use one of these commands:

- ant distribute -- to incorporate changes since your previous build.
- ant clean distribute -- to do a full build from scratch

The build will produce a file named distribution.tar.gz, in the .build sub-directory of your VIVO distribution directory. This compressed archive contains these files:

- vivo.war -- a WAR file for the main VIVO application.
- vivosolr.war -- a WAR file for the Solr application.
- vitrohome.tar -- a Vitro home directory that is ready for configuring.
- solrhome.tar -- a Solr home directory that is configured for use with VIVO.

The WAR files should be deployed to your servlet container. They may be renamed as desired when deployed. The TAR files will be unpacked to become your VIVO home directory and your Solr home directory.

3. Deploying Solr

The Solr application is packaged in vivosolr.war (see [Running the build script](#)). Deploy this file as required by your servlet container. The filename is not significant, and the file may be renamed as required by your container.

The Solr home directory is packaged in solrhome.tar (see [Running the build script](#)). Create a Solr home directory on your machine, and unpack this file into that directory. It is customary to use a solr sub-directory in your Vitro home directory, but this is not required. Note that the Solr home directory will contain VIVO's search index, so it may grow to be quite large.

You must tell Solr where to find the home directory. You can use one of two methods:

1. Set the system property solr.solr.home to the path of your Solr home directory.
2. Set a JNDI value at [java:comp/env/solr/home](#) to the path of your Solr home directory. For servlet containers, a JNDI prefix of [java:comp/env/](#) is assumed for all environment entries, so you will likely just specify a value for solr/home.

Which of these methods should you use? In general, it is easier to set a system property than an environment entry. However, a system property applies across the entire servlet container. If you want to deploy two instances of Solr in the same container, you will need to use environment entries to give each instance its own home directory.

The Solr application must be authorized to read and write to the Solr home directory.

4. Deploying VIVO

The VIVO application is packaged in vivo.war (see [Running the build script](#)). Deploy this file as required by your servlet container. The filename is not significant, and the file may be renamed as required by your container.

The Vitro home directory is packaged in vitrohome.tar. You must create a runtime.properties file in the Vitro home directory. The contents of this file are exactly as specified in the standard installation instructions. Pay attention to the value of vitro.local.solr.url. This must point to the base of the Solr application, as you have deployed it.

You must tell VIVO where to find the Vitro home directory. If you did not specify this in build.properties, you can use one of two methods:

1. Set the system property vitro.vitro.home to the path of your Vitro home directory.

2. Set a JNDI value at `java:comp/env/vitro/home` to the path of your Vitro home directory. For servlet containers, a JNDI prefix of `java:comp/env/` is assumed for all environment entries, so you will likely just specify a value for `vitro/home`.

Which of these methods should you use? In general, it is easier to set a system property than an environment entry. However, a system property applies across the entire servlet container. If you want to deploy two instances of VIVO in the same container, you will need to use environment entries to give each instance its own home directory.

The VIVO application must be authorized to read and write to the Vitro home directory.

Note: Session object in VIVO are not serializable, and therefore cannot be made persistent. The standard build process tells Tomcat not to attempt to persist Sessions. You may need to set a similar configuration option in your servlet container.