Installing VIVO

- Installing from Distribution
  - Overview
  - Preparing the Installation Settings
  - Permissions
  - Installing VIVO
- Installing from GitHub
  - Preparing the Repositories
  - Preparing the Installation Settings
  - Permissions
  - Installing VIVO
- Completing the Installation
  - Configure the Database Schema
  - Configure the Home Directory
  - Configure and Start Solr
  - Configure and Start Tomcat
- Verify Your Installation

Be sure to review System Requirements before installing VIVO.

Installing from Distribution

Overview

Download the distribution from the VIVO repository on GitHub. The standard distribution consists of the projects required to create a home directory for VIVO, and to copy the web application and search index. All the compiled code and dependencies are resolved from the Maven central repository at the time you run Maven.

The standard distribution is laid out as follows:

```
vivo-1.11.1/
  pom.xml
  home/
    pom.xml
    src
  webapp/
    pom.xml
    src
  installer/
    example-settings.xml
```

Preparing the Installation Settings

In order to fully install VIVO, you need to create a settings file that provides some essential information:

app-name

vivo-dir

tomcat-dir

This file needs to be created following the Maven Settings Reference. A template file called example-settings.xml can be found in the installer directory of the distribution. You may copy this file (it can be called anything you like), and edit the contents to fit your requirements / system configuration.

Permissions

Make sure:

- The maven user has write permission to the Tomcat webapps directory. Maven will fail silently if it cannot copy files to tomcat.
- The tomcat user has write permission to the <vivo-dir>/home directory.

Installing VIVO

Once you have an appropriate settings file (these instructions will assume that you are using example-settings.xml - replace this with your actual file), you simply need to run Maven, specifying the install goal and your settings file.

```
$ cd VIVO
VIVO$ mvn install -s installer/example-settings.xml
[INFO] Scanning for projects...
[INFO] ------------------------------------------------------------------------
[INFO] Reactor Build Order:
[INFO]
[INFO] Vitro
[INFO] Vitro Dependencies
[INFO] Vitro API
[INFO] VIVO
[INFO] VIVO API
[INFO] Vitro Web App
[INFO] VIVO Web App
[INFO] Vitro Home
[INFO] VIVO Home
[INFO] VIVO Installer
[INFO] VIVO Prepare Home
[INFO] VIVO Prepare Web App
[INFO]
....
```

The VIVO home directory will now be created and the VIVO application installed to Tomcat.

In order to run VIVO, please read the section below "Completing the Installation".

Installing from GitHub

Preparing the Repositories

In order to install the development code from GitHub, you need to clone both the Vitro and VIVO repositories from the vivo-project organization. These clones should be in sibling directories called "Vitro" and "VIVO" respectively:

```
$ git clone https://github.com/vivo-project/Vitro.git Vitro
$ git clone https://github.com/vivo-project/VIVO.git VIVO
```

If you do not place the Vitro code in a sibling directory called "Vitro", then you will have to supply the "vitro-core" property to Maven - e.g. mvn package -Dvitro-core=~/Vitro

It is expected that the Maven project numbers are kept in sync between the Vitro / VIVO projects, however, depending on when you update / sync your repositories, you may need to adjust the project version numbers for the build to work.

Preparing the Installation Settings

In order to fully install VIVO, you need to create a settings file that provides some essential information:

app-name

vivo-dir

tomcat-dir

This file needs to be created following the Maven Settings Reference. A template file already exists in the "installer" directory within the VIVO project, called "example-settings.xml". You may copy this file (it can be called anything you like), and edit the contents to fit your requirements / system configuration.

Permissions

Make sure:

- The maven user has write permission to the Tomcat webapps directory. Maven will fail silently if it cannot copy files to tomcat.
- The tomcat user has write permission to the <vivo-dir>/home directory.

Installing VIVO

Default Installer

Once you have an appropriate settings file (these instructions will assume that you are using installer/example-settings.xml - replace this with your actual file), you simply need to run Maven, specifying the install goal and your settings file.

```
$ cd VIVO
VIVO$ mvn install -s installer/example-settings.xml
[INFO] Scanning for projects...
[INFO] ------------------------------------------------------------------------
[INFO] Reactor Build Order:
[INFO]
[INFO] Vitro
[INFO] Vitro Dependencies
[INFO] Vitro API
[INFO] VIVO
[INFO] VIVO API
[INFO] Vitro Web App
[INFO] VIVO Web App
[INFO] Vitro Home
[INFO] VIVO Home
[INFO] VIVO Installer
[INFO] VIVO Prepare Home
[INFO] VIVO Prepare Web App
[INFO]
....
```

The VIVO home directory will now be created and the VIVO application installed to Tomcat. To run VIVO, please read the section below "Completing the Installation".

Custom Installer

If you want to use the source code / GitHub clone with your own customizations, you can exclude the supplied installer project, and use your own customized installer project instead. To do so, you need to supply the location of your custom installer project as the "vivo-installer-dir" property. This can be done on the command line or in the settings.xml. If you are supplying a relative path, it should be relative to the location of the VIVO/pom.xml.

```
$ cd VIVO
VIVO$ mvn install -s installer/example-settings.xml -Dvivo-installer-dir=../myedu-vivo
[INFO] Scanning for projects...
[INFO] ------------------------------------------------------------------------
[INFO] Reactor Build Order:
[INFO]
[INFO] Vitro
[INFO] Vitro Dependencies
[INFO] Vitro API
[INFO] VIVO
[INFO] VIVO API
[INFO] Vitro Web App
[INFO] VIVO Web App
[INFO] Vitro Home
[INFO] VIVO Home
[INFO] Custom VIVO Installer
[INFO] Custom VIVO Prepare Home
[INFO] Custom VIVO Prepare Web App
[INFO]
....
```

The VIVO home directory will now be created and the VIVO application installed to Tomcat, including any customizations that are defined in your local installer project. To run VIVO, please read the section below "Completing the Installation".

Completing the Installation

Configure the Database Schema

SDB vs TDB

Note, VIVO v1.11.0 uses SDB by default, however v1.11.1+ uses TDB by default. If you are using v1.11.1+ you can skip the 'Configuring the Database Schema' section.

The default configuration of VIVO is to use MySQL as a backing store for Jena SDB. Whilst VIVO / Jena will create the necessary tables for the triple store, a database (schema) and authentication details need to have been created first. To do so, log in to MySQL as a superuser (e.g. root)

```
$ mysql -u root -p
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 2
Server version: 5.7.9 MySQL Community Server (GPL)
Copyright (c) 2000, 2015, Oracle and/or its affiliates. All rights reserved.
Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.
Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql> CREATE DATABASE vitrodb CHARACTER SET utf8;
mysql> GRANT ALL ON vitrodb.* TO 'vitrodbUsername'@'localhost' IDENTIFIED BY 'vitrodbPassword';
mysql> FLUSH PRIVILEGES;
```

For MySQL 8+, the second command must be split into two commands like so:

```
mysql> CREATE USER 'vitrodbUsername'@'localhost' IDENTIFIED BY 'vitrodbPassword';
mysql> GRANT ALL PRIVILEGES ON vitrodb.* TO 'vitrodbUsername'@'localhost';
mysql> FLUSH PRIVILEGES;
```

Configure the Home Directory

There are two configuration files that are required to be in the home directory. By default, the installer does not create them so that they are not overwritten when you redeploy the application. Instead, example files are created in the home directory, which can be copied and used as the basis for your installation.

```
$ cd /usr/local/vivo/home/config
/usr/local/vivo/home/config$ cp example.runtime.properties runtime.properties
/usr/local/vivo/home/config$ cp example.applicationSetup.n3 applicationSetup.n3
```

Minimum Configuration of runtime.properties Required

In order for your installation to work, you will need to edit `runtime.properties` and ensure that the VitroConnection properties are correct for your database engine. They should look something like this.

```
VitroConnection.DataSource.url = jdbc:mysql://localhost/vitrodb
VitroConnection.DataSource.username = vitrodbUsername
VitroConnection.DataSource.password = vitrodbPassword
```

Configure and Start Solr

As of the 1.11.0 release of VIVO, Solr must be independently deployed and configured with the schema expected by VIVO. Then VIVO must be configured to connect to the external Solr.

1. Download and install the latest 7.x version of Solr (installation instructions)
    a. The directory in which Solr is installed is referenced below as ${SOLR_HOME} (e.g. /opt/solr/solr-7.7.2)
    b. Package managers may result in ${SOLR_HOME} being different than the installation directory. You can confirm the value of ${SOLR_HOME} by visiting http://localhost:8983/solr and checking the value of "solr.solr.home" in the Java Properties list, or running

    ```
    curl --silent http://localhost:8983/solr/admin/info/system | grep -e "solr_home"
    ```

2. Add the vivocore directory of the vivo-solr GitHub repository and its contents into ${SOLR_HOME}/server/solr
    a. The end result should be a directory structure such as:

```
${SOLR_HOME}/server/solr/vivocore/core.properties
                                conf/
                                     currency.xml
                                     elevate.xml
                                     ...
```

3. Start Solr

```
${SOLR_HOME}/bin/solr start
```

4. Remove schema.xml from ${SOLR_HOME}/server/solr/vivocore/conf
    a. When solr was started it created the managed-schema automatically from the schema.xml and is no longer needed
5. Update VIVO runtime.properties as below to point to the URL of your Solr

```
vitro.local.solr.url = http://localhost:8983/solr/vivocore
```

6. Start VIVO!
    a. Note: If VIVO was started before connecting to Solr, please restart VIVO.

Configure and Start Tomcat

Set JVM parameters

VIVO copies small sections of your RDF database into memory in order to serve Web requests quickly (the in-memory copy and the underlying database are kept in synch as edits are performed).

VIVO may require more memory than allocated to Tomcat by default. With most installations of Tomcat, the setenv.sh or setenv.bat file in Tomcat's bin dire ctory is a convenient place to set the memory parameters. *If this file does not exist in Tomcat's bin directory, you can create it.*

For example:

```
export CATALINA_OPTS="-Xms512m -Xmx512m -XX:MaxPermSize=128m"
```

This tells Tomcat to allocate an initial heap of 512 megabytes, a maximum heap of 512 megabytes, and a PermGen space of 128 megs. Larger values may be required, especially for production installations in large enterprises. In general, VIVO runs more quickly if given more memory.

If an OutOfMemoryError occurs during VIVO execution, increase the heap parameters and restart Tomcat.

Set security limits

VIVO is a multithreaded web application that may require more threads than are permitted under the default configuration of your operating system. Ensure that your installation can support the required number of threads for your application. For a Linux production environment you may wish to make the following edits to /etc/security/limits.conf, replacing apache and tomcat with the appropriate user or group name for your setup :

```
apache hard nproc 400

tomcat hard nproc 1500
```

Set URI encoding

In order for VIVO to correctly handle international characters, you must configure Tomcat to conform to the URI standard by accepting percent-encoded UTF-8.

Edit Tomcat's conf/server.xml and add the following attribute to each of the Connector elements: URIEncoding="UTF-8".

```
<Server ...>
  <Service ...>
    <Connector ... URIEncoding="UTF-8"/>
      ...
    </Connector>
  </Service>
</Server>
```

Some versions of Tomcat already include this attribute as the default.

Take care when creating Context elements

The webapp (VIVO) in the VIVO distribution includes a "context fragment" file, containing some of the deployment information for that webapp.

Tomcat allows you to override these context fragments by adding Context elements to server.xml . If you decide to do this, be sure that your new Context element includes the necessary deployment parameters from the overridden context fragment.

Starting Tomcat

If everything has been completed successfully, then you should simply be able to start Tomcat at this point, and VIVO will be available. If you are using a Tomcat supplied by your operating system / package manager, then use your normal means for starting the application server.

Otherwise, start Tomcat by running the startup script - e.g.

```
$ /usr/local/tomcat/bin/startup.sh
```

Verify Your Installation

If you have completed the previous steps, you have good indications that the installation was successful.

- When you Start tomcat , you see that Tomcat recognizes the webapp, and that the webapp is able to present the initial page.
- The startup status will indicate if the basic configuration of the system was successful. If there were any serious errors, you will see the status screen and will not be allowed to continue with VIVO. If there are warnings, you will see the status screen when you first access VIVO, but after that you may use VIVO without hinderance. In this case, you can review the startup status from **siteAdmin -> Startup status**.
- Log in as root. Your root username is vivo_root@mydomain.edu (or the email you configured in runtime.properties). The first time root password is rootPassword. You will be asked to change it.

Here is a simple test to see whether the ontology files were loaded:

- Click on the "Index" link on the upper right, below the logo. You should see a "locations" section, with links for "Country" and "Geographic Location." The index is built in a background thread, so on your first login you may see an empty index instead. Refresh the page periodically to see whether the index will be populated. This may take some time: with VIVO installed on a modest laptop computer, loading the ontology files and building the index took more than 5 minutes from the time that Tomcat was started.
- Click on the "Country" link. You should see an alphabetical list of the countries of the world.

Here is a test to see whether your system is configured to serve linked data:

- Point your browser to the home page of your website and click the "Log in" link near the upper right corner. Log in with the rootUser.emailAddress you set in runtime.properties . If this is your first time logging in, you will be prompted to change the password.
- After you have successfully logged in, click "site admin" in the upper right corner. In the drop down under "Data Input" select "Faculty Member (core)" and click the "Add individual of this class" button.
- Enter the name "test individual" under the field "Individual Name," scroll to the bottom, and click "Create New Record." You will be taken to the "Individual Control Panel." Make note of the value of the field "URI" - it will be used in the next step.
- Open a new web browser or browser tab to the page http://lodview.it/. Enter the URI of the individual you created in the previous step and click "go."
- In the resulting page search for the URI of the "test individual." The page should display information for the individual, including an rdfs: label and rdf:type. This indicates that you are successfully serving linked RDF data. If the service returns an error you are not successfully serving linked data.

Finally, test the search index.

- Type the word "Australia" into the search box, and click on the Search button.You should see a page of results, with links to countries that border Australia, individuals that include Australia, and to Australia itself. To trigger a rebuild of the search index, you can log in as a site administrator and go to **Site Admin -> Rebuild search index**.