Apache OFBiz Technical Production Setup Guide

Pre Gradle version

This page documents the usage with Gradle, the pre-Gradle documentation is here: Apache OFBiz Technical Production Setup Guide

This page documents the trunk version, the documentation for the R16.11 version is here: Apache OFBiz Technical Production Setup Guide

NOTE: this is the technical setup guide for OFBiz, for the business oriented one, look here.

Data To Gather for Setup

1. Technical Info
   a. Host/domain name to deploy on (and ports if not 80, 443)
   b. Web server setup: direct to Tomcat/Jetty, through Apache, using another (stand-alone) application server?
   c. Which database to use. Is the database installed on a remote server or installed locally ?

Technical Setup Process

Tools Installation

1. **Java SDK**
   Please see System Requirements for details of the correct Java JDK version required for each OFBiz version.

2. **Check/install SVN client**

   > ***Expert Recommendation****: If you intend to only install a release version of OFBiz you will not need the SVN client. But normally you would want to have the svn client in order to be able to upgrade easily.*

OFBiz Installation

1) If you intend to use a development version of OFBiz, please follow the instructions on the Source Repository page to checkout the latest code or the stable branch that you require.

2) Build using "[./] gradlew" (add "./" on *nix systems, to use the embedded Gradle which comes with its wrapper, hence gradlew and not gradle in the command). Do this from the ofbiz home directory (i.e. the directory where you installed OFBiz). Note: you can get more information about Gradle task using "gradlew tasks".

Database Setup

By default OFBiz includes and is configured for an embedded Java database called Derby. This database used to be called Cloudscape but was purchased by IBM, changed to be DB2 specification compliant, and then release as open source. This is a great database for demonstration, testing, development, and even small-scale production environments.

Expert Recommendation: We generally do not recommend the Derby database for production because it is not as easy to administer and optimize, and will generally not perform as well as more substantial or native databases. This is mainly because Derby is a lightweight Java database.

> ***Expert Recommendation:*** *There are many databases that OFBiz can use, both commercial and open source. While there are significant production instances of OFBiz running using Oracle, MS SQLServer, and so on, we generally recommend using an open source database such as Postgres or MySQL. Which database to use is an important decision and may have a future impacts on your implementation. If you do not have a preferred database choice then we strongly recommend discussing options with an experienced consultant before making a final decision.*

1. If you decide not to use the embedded Derby database, check/install your database of choice. See below for some information that may help with using or installing OFBiz with alternative databases

   Changing from Derby to MySQL Database
   OFBiz and Oracle
   Connecting OFBiz to PostGIS spatially enabled database

2. If necessary, put the correct JDBC driver in the following directory:

   a. ${ofbiz install dir}/framework/entity/lib/jdbc (NOTE: download the last mySQL jdbc driver)
   b. Replace the old or incorrect version of the driver (i.e. copy in with same name to avoid problems with update restoring the old driver later)

3. To setup the Entity Engine to use a different database from the default Derby database.
   In the: ${ofbiz install dir}/framework/entity/config/entityengine.xml file:

   - Modify the "localpostgres" (or "localmysql", etc.) datasources elements to connect to your database.
   - Modify the "default" delegator element:
     Change
     <group-map group-name="org.apache.ofbiz" datasource-name="local**derby**"/>
     to
     <group-map group-name="org.apache.ofbiz" datasource-name="local**postgres**"/>
   - The OLAP and Tenant data sources will still use Derby. If you want to change those to use PostgreSQL also, then:
     - Modify the "localpostgresolap" datasource element to connect to your OLAP database.
     - Modify the "localpostgrestenant" datasource element to connect to your Tenant database.
   - Modify the "default" delegator element:

     Change
     <group-map group-name="org.apache.ofbiz.olap" datasource-name="localderbyolap"/>
     to
     <group-map group-name="org.apache.ofbiz.olap" datasource-name="localpostgresolap"/>
     Change
     <group-map group-name="org.apache.ofbiz.tenant" datasource-name="localderbytenant"/>
     to
     <group-map group-name="org.apache.ofbiz.tenant" datasource-name="localpostgrestenant"/>
   - Modify an existing datasource (near the bottom of the file) or create a new one by copying one of the sample datasources already there and giving it a new unique name

1. If using the default OFBiz transaction manager and connection pool then update the data URI, username and password in the inline-jdbc tag

2. If you want your OFBiz tables in a special schema you will first need to create that schema in your database and then set the schema-name attribute on the datasource tag

3. Find the "default" delegator near the top of the file and change the datasource-name attribute value in the group-map tag under it to the name of the data source you modified or created.

Expert Recommendation: if you are using an external application server or want to setup more advanced Entity Engine configurations, see the Entity Engine Configuration Guide or other online documentation. The training videos from Undersun Consulting are highly recommend if you get into more advanced usage of OFBiz.

Initial Data Loading

To load the initial data just use the OFBiz install routine through gradlew or directly with Java and the build/libs/ofbiz.jar executable JAR file. By default the install routine will load the "seed" and "demo" sets of data files, as defined in the entityengine.xml file and in the ofbiz-component.xml file in each component.

Run one of the following two options from the command line in the ofbiz home directory to run the default install routine:

```
 [./]gradlew loadAll (add "./" on *nix systems, to use the embedded Gradle which comes with its wrapper, hence
gradlew and not gradle in the command).
Do this from the ofbiz home directory (i.e. the directory where you installed OFBiz). Note: you can get more
information about Gradle task using "gradlew tasks".
or
java -jar build/libs/ofbiz.jar --load-data
```

Help can be obtained by by the following command:

```
gradlew "ofbiz -help"
or
java -jar build/libs/ofbiz.jar ?
```

Note that you can choose to only load the basic "seed" data and not load the "demo" set of data files. To do this run something like:

```
gradlew "ofbiz --load-data readers=seed,seed-initial"
or
java -jar build/libs/ofbiz.jar --load-data readers=seed,seed-initial
```

> *Definition: **Seed Data** is data that an application requires in order to run. It is created and maintained along with the code and should be loaded into the database when the code is updated. It should not be changed or removed in the application database without first having done so in the Seed Data XML files. In OFBiz these are the source of the Seed Data and they are maintained in the code repository along with the code. Add in components should do the same for this sort of data.*

Definition: Seed Initial Data is data that is only required the first time the system is loaded like the password of the admin user. Later if you want to update the seed data, you would normally not want the password of admin to be set back to 'ofbiz'.

> ***Expert Recommendation**: For smaller installations we recommend loading the demo data and modifying it through the OFBiz applications rather than creating all of the data from scratch.*

For larger installations we recommend creating data files with your own settings, copied from the demo data, or simply creating all data from scratch. These can be added to the "ext" set of install data files, which is done by referencing those files in an ofbiz-component.xml file entity-resource tag with the attribute reader-name="ext".

To load the "seed" and "ext" groups run something like:

```
gradlew "ofbiz --load-data readers=seed,ext"
or
java -jar build/libs/ofbiz.jar --load-data readers=seed,ext
```

The OFBiz demo data includes a Party with ID "admin" and it has a number of UserLogin accounts associated with it that have varying permissions.

Note the following configuration changes:

1. For production systems the passwords on all UserLogin accounts associated with the "admin" party **should be changed** from their default values.
2. The default password for the "admin" userLoginId is "ofbiz" which is also the same for the "flexadmin" and "demoadmin" accounts.
3. The "1" account is meant to be used with the POS system so the new password should include only numbers to make it easier to use with the POS client.

> ***Expert Recommendation**: While the "admin" party is the most critical user you need to make sure the passwords are changed for, there are other parties you should also check and change the password for all User Login accounts associated with these parties. This includes "ltdadmin" and "externaluser". If the demo data is loaded (done by default) then also change the passwords for the users "DemoBuyer", "DemoRepAll", and "DemoRepStore".*

OFBiz Configuration

Cache Settings

${ofbiz insall dir}/framework/base/config/cache.properties

Fresh from SVN, this file is configured to be development friendly to reload resources frequently. For production use the expire times for the caches in the last section of the file should all be commented out by putting a hashtag sign "#" at the beginning of each line. The properties in question start with "minilang.", "script.", "webapp.", "widget." And "template.".

Debug Settings

${ofbiz install dir}/framework/base/config/debug.properties

This file is used to enable/disable different logging levels and used to configure Log4J logging level and output settings. For most production use it can stay as-is.

Disabling the "info" and "timing" level properties can save some resources on the server, but we recommend leaving all of the others enabled. To do this just change the values of the "print.info" and "print.timing" properties to anything except "true".

For more advanced logging setup you can edit the Log4J configuration file located under framework/base/config/.

> Log4J is the engine used by OFBiz to manage its log output. The releases 13.07.*, 12.04.*, 11.04.*, 10.04.* use Log4J 1 (the configuration file is named log4j.xml) while newer releases are bundled with Log4j2 (the configuration file is named log4j2.xml).
>
> For information about advanced configuration of Log4j refer to:
>
> - Log4J 1: http://logging.apache.org/log4j/1.2/manual.html
> - Log4J2: http://logging.apache.org/log4j/2.x/manual/configuration.html

Security Settings

${ofbiz install dir}/framework/security/config/security.properties

There are various settings you might want to configure in this file, but for most production use it can remain unchanged.

If you want to use LDAP for user authentication:

- Set the security.ldap.enable property to true.
- Using the ${ofbiz install dir}/framework/security/config/jndiLdap.properties file:
  - Change the java.naming.provider.url property to point to your LDAP server.
  - Change the ldap.dn.template to use the Distinguished Name appropriate for your LDAP installation.

If your LDAP installation is simple (all of your users are in a single organization unit) then these are all the changes needed. If your users are in more than one organizational unit, then you will need to set up each user's distinguished name using the Party Manager View Profile -> Update UserLogin Security Settings screen.

By default, OFBiz will authenticate the user to LDAP first, and if successful it will synchronize the user's OFBiz password to the user's LDAP password, and then proceed to authenticate the user to OFBiz (using OFBiz's internal permissions logic). If LDAP authentication is unsuccessful, the user is still authenticated to OFBiz. This is the behavior appropriate for most installations.

If you want the entire authentication process to fail if LDAP authentication fails, then set the security.ldap.fail.login property in ${ofbiz install dir}/framework /security/config/security.properties to true. Only users who are in the LDAP directory will be able to use OFBiz. This would be appropriate for installations that use only the back office (manager) applications.

> **Vulnerabilities**
>
> Be sure to always check the Keeping OFBiz secure page for possible vulnerabilites

URL and Port Settings

The following configuration files contain port configurations that may configured for custom OFBiz installations.
You may also use the global portoffset Java properties. Use **"gradlew "ofbiz ?"** or **"java -jar build/libs/ofbiz.jar -h"** to know more about this option.

- ${ofbiz install dir}/framework/base/config/ofbiz-containers.xml (The Containers Configuration File)

- ${ofbiz install dir}/framework/webapp/config/url.properties (The URL Properties File)
  - These properties are used to configure the settings to send to the client browser. The port and host settings may be different than the port and host settings of the local machine if it is running through Apache through AJP, through any proxy, or through a load balancer of any sort. The settings in this file should be set to what the client browser will see when communicating with your system.
  - Note that the settings in this file can be overridden on the WebSite record with the ID specified by the webSiteId field in the web.xml file in each webapp, if applicable. These can be viewed and modified in the WebSites tab of the Content Manager application.

- ${ofbiz install dir}/framework/base/config/jndiservers.xml (JNDI Servers XML File)
  - Generally only the "default" server is used which has automatic configuration through the Java standard JNDI facility, so no changes are needed for most production deployments in this file.

- ${ofbiz install dir}/framework/base/config/jndi.properties (JNDI Properties file)
  - This is a standard Java JNDI configuration properties file and is used to configure the local JNDI server to be used. Note that this is how the "default" JNDI server in the JNDI Servers XML File is configured. If this file is not present the Java standard JNDI classes will use various defaults instead.

- ${ofbiz install dir}/framework/service/config/serviceengine.xml (Service Engine XML file)
  - Most of the settings in this file can remain unchanged for production deployments, but there are some default server locations that refer to port 1099 (JRMP, for RMI) and 8080 (HTTP). These are in the "service-location" tags in the file.

- ${ofbiz install dir}/framework/jotm/config/iiop.properties (or jrmp.properties) (Carol IIOP or JRMP Only Properities)
  - The default transaction manager in OFBiz is called JOTM and it includes a remote communication container called Carol that implements various standard protocols.

    - There are two configuration files here as examples of two different ways of configuring Carol in JOTM. The iiop.properties file sets up the IIOP and the JRMP protocols, while the jrmp.properties file only sets up the JRMP protocols. The properties file to use for Carol in JOTM is specified in the ofbiz-containers.xml file, or whichever containers XML file you are using, in the "jndi-config" property of the "jotm-container" container.

- ${ofbiz install dir}/framework/webapp/config/fop.xconf (FOP configuration file)
    - the default settings in this file can remain unchanged for production deployments, but you may need to change the "base" element (the default value is "http://localhost:8080") to point to the modified base url (it is used by FOP to resolve relative paths, e.g. to images) .

Widgets setting

${ofbiz install dir}/framework/widget/config/widget.properties

- To ease development (Out Of The Box, OFBiz is configured in development mode) the compressHTML properties in general.properties if OFF. If you prefer to strip unnecessary whitespace from HTML output you can set it on ON (uncomment the line "#compress.HTML=true"). Note that non html output screens like for example tab delimited exports can be then messed up. Note also that it is NOT the same thing as gzipped HTTP 1.1 compression.

- For the same reason (development mode) the line "widget.verbose=true" is uncommented. If you do not want to see any informations about widget boundaries in the generated HTLM code, you should comment this line.

Ports, Default Values

- Admin Port - 10523
    - Configured in: start.properties File
    - Referenced in: Config.java File
- HTTP - 8080
    - Configured in: ofbiz-containers XML File
    - Referenced in: Service Engine XML File, Client Browser (if running direct), URL Properties File
- HTTPS - 8443
    - Configured in: ofbiz-containers XML File
    - Referenced in: Client Browser (if running direct)
- AJP13 - 8009
    - Configured in: ofbiz-containers XML File
    - Referenced in: Apache mod_jk plugin configuration
- JRMP (JNDI, RMI, etc) - 1099
    - Configured in: Carol Properties (carol.jrmp.url)
    - Referenced in: Containers XML File, JNDI Properties File, Service Engine XML File
- BSH Client - 9989, 9990 (this should generally be disabled or at least protected by a firewall)
    - Configured in: ofbiz-containers XML File
- Multicast ports - 45564, etc. (Tomcat clustering)
    - Configured in: ofbiz-containers XML File

SSL Certificate Setup

Choose a password to enter later when prompted. This same password will be used for the keystore password and for another question a bit later as the key password for.

1. Run: "keytool -genkey -keyalg RSA -alias ssl -keystore [keystore name]"

Go through and answer the following questions:

> Enter keystore password: [password]
>
> What is your first and last name?
> [Unknown]: www.mydomain.com (example)
>
> What is the name of your organizational unit?
> [Unknown]: Undersun Testing (example)
>
> What is the name of your organization?
> [Unknown]: Undersun Testing (example)
>
> What is the name of your City or Locality?
> [Unknown]: New York (example)
>
> What is the name of your State or Province?
> [Unknown]: New York (example)
>
> What is the two-letter country code for this unit?
> [Unknown]: US (example)

Is CN=www.mydomain.com, OU=Undersun Testing, O=Undersun Testing, L=New York, ST=New York, C=US correct?
[no]: yes

Enter key password for
(RETURN if same as keystore password): [password]

2. Run: "keytool -certreq -alias ssl -keyalg RSA -file certreq.csr -keystore [keystore name]"

The following will be prompted/shown:

> Enter keystore password: [password]
>
> The CSR will be saved in the current directory: BEGIN NEW CERTIFICATE REQUEST and END NEW CERTIFICATE REQUEST

3. Submit the CSR to a signing authority (Thawte, Verisign, etc)

4. Download your certificate from the signing authority. Please remember to download the Certificate in PKCS#7 format. If you get a certificate in pem format don't convert to PKCS#7/P7B Format but der format

5. Import the Certificate into the keystore by running:

> "keytool -import -alias ssl -trustcacerts -file mysignedcert.cer -keystore [keystore name]"

6. Configure the framework\catalina\ofbiz-component.xml file to point to your new keystore and password:

- If using Tomcat (Catalina), which is the default, find the "catalina-container" -> "https-connector" -> "keystoreFile" and "keystorePass" properties and set them.
- If using Jetty find the "jetty-container" -> "https-listener" -> "keystore" and "password" properties and set them.
- For other Servlet containers, see the documentation for that container to find out how to set the HTTPS keystore and password settings.

Currency, Locale, Time Zone Settings

Currency settings may be configured in the following configuration file:

${ofbiz install dir}/framework/common/config/general.properties

- Default Currency Code (**Note**: This must exist in Uom table, three letter code, see seed data in the CurrencyData.xml file): currency.uom.id.default=USD
- Default Country Code (**Note**: This must exist in Geo table, is three letter ISO country code, see seed data in the GeoData.xml file): country.geo.id.default=USA
- Note that these settings can be overridden for a Store in the Edit Product Store page.

You can constrain the available locale using locales.available.
For instance
    # -- locales made available separated by comma's
    locales.available=en,fr,de,it,nl,es,ja,zh

The default locale and time zone are configured in the following file:

${ofbiz install dir}/framework/start/src/org/ofbiz/base/start/start.properties

- Default Locale: ofbiz.locale.default
- Default Time Zone: ofbiz.timeZone.default

Email Server Settings

e-mail settings may be configured in the following, OFBiz file:

${ofbiz install dir}/framework/common/config/general.properties

- SMTP Server (relay host): mail.smtp.relay.host
- SMTP Username (if needed): mail.smtp.auth.user
- SMTP Password (if needed): mail.smtp.auth.password
- Turn on email notifications by setting the mail.notifications.enabled property to "Y".

If you are having trouble getting OFBiz to connect to your mail server, try disabling your anti-virus software (temporarily) as it may block attempts to send emails from unknown applications because it thinks they are being sent by a virus.

Another possible issue is the presence in database of **SystemProperty** data that could overload general.properties configuration. Be sure to check the absence of mail property in SystemProperty table.

Mounting a Root WebApp

It is often desirable to have one of the webapps mounted on the root. This is often either the ecommerce webapp or your own web site, which is created as a webapp in an OFBiz component would be setup the same way.

1. To set the mount point to root (or "/") for a webapp, find the corresponding "webapp" tag in the appropriate ofbiz-component.xml file and change the "mount-point" attribute on that tag to "".
2. For the default OFBiz ecommerce webapp, this is found in the file:

${ofbiz install dir}/specialpurpose/ecommerce/ofbiz-component.xml

Running OFBiz

There are various ways to run OFBiz, and they all come down to some variation of the executing the "build/libs/ofbiz.jar" executable JAR file.

On the command line this can be as simple as (but beware you might need to pass specific JVM arguments, like "-server" if you develop on Windows, see OFBIZ-7321 - Getting issue details... STATUS):

```
gradlew ofbiz
or
java -jar build/libs/ofbiz.jar
```

To access the application from your browser follow advice given in the Demo and Test Setup Guide

For production use you will want to setup a start script that contains special settings for things like memory heap size, and so on. There are example start and stop scripts in the root ofbiz directory in the startofbiz.sh and stopofbiz.sh files.

> ***Expert Recommendation**: Instead of running the startup and shutdown scripts manually it might be better to set them up as services on the system, or to use something like daemontools to make sure the process is restarted automatically if it goes down for some unexpected reason.*

Running OFBiz Automated Tests

Each component running in OFBiz can have its own set of tests. These are usually defined in the "testdef" directory in each component, and the test set XML files are specified in the ofbiz-component.xml file for each component.

To run all automated tests use the following command:

```
gradlew testIntegration
or
java -jar build/libs/ofbiz.jar -t
```

To run just the tests for one component run something like (for the entity component):

```
gradlew "ofbiz --test component=entity"
or
 java -jar build/libs/ofbiz.jar --test component=entity
```

Performance Monitoring and Tuning

> ***Expert Recommendation:** While some steps can be taken at this point in the configuration process, the real performance testing and tuning should be done after you have setup your catalog, categories and products, and after you have customized any templates you plan to.*

As mentioned above a critical part of performance tuning is turning off the expire times in the cache.properties by commenting out the lines near the bottom of that file that set those values.

General Cache Information

You can get a lot of good information on database and configuration file (XML, properties, etc) performance by looking at the statistics in the cache management page in Web Tools. They are all listed there with statistics about hits and misses, and about why cache misses happened.

There are three causes for cache misses:

1. Not Found
2. Soft Reference clearing (is done during garbage collection)
3. Expire Time reached for an entry in the cache

Memory Settings

If you are running into problems with Soft References (as seen on the cache management page in WebTools), the case is generally that your heap memory settings are less than ideal. Every time Java does a memory allocation and finds it needs to grow the heap size it first does a garbage collection and clears out soft references. The Entity Engine caches all use Soft References to avoid overrunning memory with cached database data.

In other words, you may have 2Gb of memory on the box, but you are probably only starting with 64Mb (or even less) and have a max heap of 128Mb. These are done with the following java command line arguments: -Xms64M for the low size, and -Xmx128M for the max size.

We usually recommend a max heap size of around 75% of the memory on the box, but that depends on the operating system of course... The small size should be pretty high, perhaps even as high as the max, to avoid the caches being cannibalized during allocations and garbage collections as described above. Last recent versions (say since R10.04) require more permanent generation space (also know as perm gen space). For instance in the trunk demo server we currently (16 May 2010) use 512 MB of perm gen space. And our memory parameters are -Xms128M -Xmx1024M -XX:MaxPermSize=512m.

Database Intensive Operations
The comments above on memory settings, caches, etc. are for category browsing pages and such where just about everything should be cached. For database intensive operations, like the product searching, it's much better to focus on how the database is managing with the queries.