Deployment

See the Quick Start guide to getting Fedora up and running as quickly as possible.
Although deploying Fedora is as easy as downloading the WAR file and copying to your servlet container's `webapps` directory, this document details the process.

Downloads

See the latest release for Fedora WAR files to download.

System Requirements

- Java 11

```
java -version
openjdk version "11.0.5" 2019-10-15
OpenJDK Runtime Environment (build 11.0.5+10-post-Ubuntu-0ubuntu1.1)
OpenJDK 64-Bit Server VM (build 11.0.5+10-post-Ubuntu-0ubuntu1.1, mixed mode, sharing)
```

- Servlet 3.0 container such as:
    - Tomcat 9
    - Jetty 9.x
- A case-sensitive filesystem. Fedora 6 stores resources on disk in an OCFL repository, and OCFL requires a case-sensitive filesystem. By default, Windows and Mac systems uses case-insenstive filesystems. If you intend to run Fedora on such a system, you should do some combination of the following:
    - Locate Fedora home on a case-sensitive volume. This is an exercise for the reader.
    - Do not use Archival Groups. Case related resource name collisions can only happen when dealing with resources contained within the same Archival Group.
    - Do not name two different resources the same but with different casing.

Deploying with Tomcat 9

1. Download and install Tomcat
2. Set the Java properties for Tomcat (see: Application Configuration and Catalina Java Properties sections below)
3. Either configure the container auth or disable auth entirely
4. Copy the Fedora WAR file into Tomcat's "webapps" directory, `$CATALINA_HOME/webapps/fcrepo.war`
5. Start the server, `$CATALINA_HOME/bin/startup.sh`
6. Navigate to http://localhost:8080/fcrepo/rest in your browser

Tomcat and Encoded slashes

If you attempt to create a resource with an encoded slash in the path (ie. http://localhost:8080/fcrepo/rest/path%2Fto%2Fresource) Tomcat will throw a 400 error. You can disable this by adding `-Dorg.apache.tomcat.util.buf.UDecoder.ALLOW_ENCODED_SLASH=true` to your CATALINA_OPTS.

Deploying with Jetty 9

1. Download and install Jetty
2. Set the Java properties for Jetty (see: Application Configuration and Catalina Java Properties sections below)

3. Either configure the container auth or disable auth entirely
4. Copy the Fedora WAR file into Jetty's "webapps" directory, `$JETTY_BASE/webapps/fcrepo.war`
5. Start the server, `java -jar $JETTY_HOME/start.jar`
6. Navigate to http://localhost:8080/fcrepo/rest in your browser

The Fedora web-application supports several deploy-time, system-level configuration options. These configuration elements can be set using a properties file or through system properties.

See: Best Practices - Fedora Configuration

Deployments

Four means of deploying Fedora have been verified

- Tomcat 9 servlet container
- Jetty 9 servlet container
- Maven jetty:run plugin - *for testing*
- One-Click Run - *for testing*

Each of these deployment approaches has its own way of setting System Properties.

Tomcat 9

On Debian Linux systems, the typical way of setting System Properties is to update the following file:

```
/etc/default/tomcat9
```

Within that file, new properties can be added per the example below:

```
JAVA_OPTS="${JAVA_OPTS} -Dfcrepo.home=/mnt/fedora-data"
```

Additional information regarding the configuration of System Properties in Tomcat 9 can be found here.

Windows notes

Alternatively on Windows systems you can set the following file:

```
CATALINA_BASE/bin/setenv.bat (windows)
```

Within that file, new properties can be added per the example below:

```
set CATALINA_OPTS=%CATALINA_OPTS% -Dfcrepo.home=/mnt/fedora-data
```

Reverse Proxy

If you have a reverse proxy for serving HTTPS that uses Tomcat's HTTP port, you will also need to set up a RemoteIPValve in your server.xml in order for Tomcat to rewrite links with HTTPS.

To do this you will first need to make sure the X-Forwarded-Proto header is set in your server config (example with Apache):

```
<VirtualHost *:443>
    RequestHeader set X-Forwarded-Proto "https"
    ServerName dummy-host.example.com

    SSLEngine on
    SSLCertificateFile  /etc/ssl/certs/localhost.crt
    SSLCertificateKeyFile /etc/ssl/private/localhost.key

    ProxyPreserveHost On
    ProxyRequests Off
    ProxyPass / http://localhost:8080/
    ProxyPassReverse / http://localhost:8080/

    DocumentRoot "/opt/fedora/apache-tomcat-8.5.66/webapps/fcrepo-webapp"
</VirtualHost>
```

Then you will need to add a Valve to the localhost Engine in Tomcat's server.xml:

```
        <Valve className="org.apache.catalina.valves.RemoteIpValve"
               protocolHeader="X-Forwarded-Proto" />
```

Jetty 9

On Debian Linux systems, one way of setting System Properties is to update the following file:

```
/etc/default/jetty
```

Within that file, new properties can be added per the example below (note the use of JAVA_OPTIONS instead of JAVA_OPTS):

```
JAVA_OPTIONS="${JAVA_OPTIONS} -Dfcrepo.home=/mnt/fedora-data"
```

Additional information regarding the configuration of System Properties in Jetty 9 can be found here.

Windows notes

Alternatively on Windows systems you can set the following file:

```
{JETTY_DIST}/start.ini
```

Within that file, new properties can be added per the example below:

```
--exec
-Dfcrepo.home=/mnt/fedora-data
Maven jetty:run
```

System Properties can be set when using the Maven jetty:run plugin by passing them per the example below:

```
mvn -Dfcrepo.home=/mnt/fedora-data jetty:run
```

One-Click Run

One option is to use the "one click" application, which comes with an embedded Jetty servlet. This can be optionally built by running:

```
mvn install -pl fcrepo-webapp -P one-click
```

and can be started by either double-clicking on the jar file or by running the following command:

```
java -jar ./fcrepo-webapp/target/fcrepo-webapp-<version>-jetty-console.jar
```

By default, a Fedora home directory, fcrepo, is created in the current directory. You can change the default location by passing in an argument when starting the one-click, e.g.:

```
java -Dfcrepo.home=/data/fedora-home -jar fcrepo-webapp-6.0.0-jetty-console.jar
```

Click here for a complete list of configurable properties.

Catalina Java Properties

For a complete inventory of configurable properties please refer to the Properties page.

JVM Tuning Properties

We have a separate page with suggested VM options for general Java tuning.