Upgrading SHRINE to 1.20

In most cases, upgrading an existing instance of SHRINE is a relatively quick process. Exceptions to this rule include older versions of SHRINE that contained substantial changes to configuration files and other portions of the file structure. The instructions here specifically describe an upgrade path from SHRINE 1.19.x to SHRINE 1.20.1. For instructions on upgrading to SHRINE 1.19.x, check the child pages underneath the "Archived Upgrade Instructions" section.

This guide makes the following assumptions of a current 1.19.2 user. Make sure all of these conditions are satisfied before proceeding:

- i2b2 1.7.05 or newer is installed and operational.
- SHRINE 1.19.2 is installed and operational. (along with Tomcat 7)
- The SHRINE Data Steward included with 1.19.2 has been installed and configured properly.

Shut Down SHRINE

Before starting the upgrade process, make sure SHRINE's Tomcat is not running. Leaving it running during this process can cause problems. Simply run the following command:

```
$ shrine_shutdown
```

If the above command is not found, try the following instead:

```
$ /opt/shrine/tomcat/bin/shutdown.sh
```

Create Backups

Now that SHRINE is stopped, it is a good idea to back up the current versions of the components we will be upgrading. The exact method for making this backups may vary, but these instructions will place the backups in a folder called /opt/shrine/upgrade-backups.

Start by creating a folder to contain these backups:

```
$ mkdir /opt/shrine/upgrade-backups
```

Remove the old .war files with this command:

```
$ rm /opt/shrine/tomcat/webapps/*.war
```

Next, move the current SHRINE webapp folder to the backup location:

```
$ mv /opt/shrine/tomcat/webapps/shrine /opt/shrine/upgrade-backups/shrine
```

Make sure to also back up the other existing SHRINE components (shrine-proxy and steward), just in case:

```
$ mv /opt/shrine/tomcat/webapps/shrine-proxy /opt/shrine/upgrade-backups/shrine-proxy
$ mv /opt/shrine/tomcat/webapps/steward /opt/shrine/upgrade-backups/steward
```

Make especially sure that the **shrine-webclient/** folder is backed up. Later on, we will be restoring important webclient configuration files from this backup. If you choose not to make any backups, make sure to at least keep a copy of **i2b2_config_data.js and js-i2b2/cells/SHRINE/cell_config_data.js**!

```
$ mv /opt/shrine/tomcat/webapps/shrine-webclient /opt/shrine/upgrade-backups/shrine-webclient
```

Modify Tomcat

You will need to create a file called **setenv.sh** in Tomcat's **bin/** folder to increase the PermGen size given to it. This will prevent OutOfMemoryError messages caused by one of the underlying application libraries**.** Make sure this file is owned by the same user that runs SHRINE, and make sure it is executable:

creating setenv.sh

```
$ cd /opt/shrine/tomcat/bin
$ vi setenv.sh
$ sudo chown shrine:shrine setenv.sh
$ sudo chmod 755 setenv.sh
```

The contents of **setenv.sh** should look like this:

setenv.sh contents

```
export CATALINA_OPTS=" -XX:MaxPermSize=256m -Dakka.daemonic=on "
```

If using Windows, the file should be named **setenv.bat** instead, and the contents should look something like this:

setenv.bat contents

```
set CATALINA_OPTS=-XX:MaxPermSize=256m -Dakka.daemonic=on
```

Deploy New shrine.war

Next, we will retrieve the new SHRINE webapp from the HMS Sonatype Nexus server at http://repo.open.med.harvard.edu/nexus/content/groups/public/net/shrine/shrine-war/. Navigate to the folder for 1.20.1. From there, download **shrine-war-1.20.1.war** to the **webapps/** directory on the SHRINE server and rename it to **shrine.war**.

For example:

```
$ cd /opt/shrine/tomcat/webapps
$ wget --no-check-certificate http://repo.open.med.harvard.edu/nexus/content/groups/public/net/shrine/shrine-war/1.20.1/shrine-war-1.20.1.war -O shrine.war
```

Deploy New steward.war

NOTE: The previous bug with the SHRINE Data Steward turning all usernames into lowercase has been fixed. Any i2b2 usernames that use both uppercase and lowercase letters will now be able to use the SHRINE Data Steward properly. However, for the sake of consistency, please continue using all lowercase letters in your usernames if they were already changed.

Much like shrine.war, the SHRINE Data Steward can be found on the HMS Sonatype Nexus server at http://repo.open.med.harvard.edu/nexus/content/groups/public/net/shrine/steward/. Navigate to the folder for 1.20.1. From there, download **steward-1.20.1.war** to the **webapps/** directory on the SHRINE server and rename it to **steward.war**.

For example:

```
$ cd /opt/shrine/tomcat/webapps
$ wget --no-check-certificate https://repo.open.med.harvard.edu/nexus/content/groups/public/net/shrine/steward/1.20.1/steward-1.20.1.war -O steward.war
```

Deploy New shrine-proxy.war

Like other SHRINE artifacts, the SHRINE proxy can be found on the HMS Sonatype Nexus server at https://repo.open.med.harvard.edu/nexus/content/groups/public/net/shrine/shrine-proxy/. Navigate to the folder for 1.20.1. From there, download **shrine-proxy-1.20.1.war** to the **webapps/** directory on the SHRINE server and rename it to **shrine-proxy.war**.

For example:

```
$ cd /opt/shrine/tomcat/webapps
$ wget --no-check-certificate https://repo.open.med.harvard.edu/nexus/content/groups/public/net/shrine/shrine-proxy/1.20.1/shrine-proxy-1.20.1.war -O shrine-proxy.war
```

Deploy New SHRINE Webclient

NOTE: Installation procedures for this component have changed! Please be careful. The SHRINE webclient is now packaged as a zip file, and no longer requires a checkout from the (no-longer-used) SHRINE SVN repository.

The SHRINE webclient can now be found on the HMS Sonatype Nexus server at https://repo.open.med.harvard.edu/nexus/content/groups/public/net/shrine/shrine-webclient/. Navigate to the folder for 1.20.1. From there, download **shrine-webclient-1.20.1-dist.zip** file to the **webapps/** directory on the SHRINE server and rename it to **shrine-webclient.zip**. Then, unzip the shrine-webclient.zip file.

For example:

```
$ cd /opt/shrine/tomcat/webapps
$ wget --no-check-certificate https://repo.open.med.harvard.edu/nexus/content/groups/public/net/shrine/shrine-webclient/1.20.1/shrine-webclient-1.20.1-dist.zip -O shrine-webclient.zip
$ unzip shrine-webclient.zip
```

```
Restore Webclient Backups
```

After this, restore the previous **i2b2_config_data.js** and **cell_config_data.js** files from your backup and place them in the new shrine-webclient folder:

```
$ cp /opt/shrine/upgrade-backups/shrine-webclient/i2b2_config_data.js /opt/shrine/tomcat/webapps/shrine-webclient/i2b2_config_data.js
```

```
$ cp /opt/shrine/upgrade-backups/shrine-webclient/js-i2b2/cells/SHRINE/cell_config_data.js /opt/shrine/tomcat/webapps/shrine-webclient/js-i2b2/cells/SHRINE/cell_config_data.js
```

Deploy New SHRINE Dashboard

NOTE: This component is a working prototype, and simply mirrors information from the existing SHRINE Happy module in its current state. Additional functionality is coming in a future release.

Like other SHRINE artifacts, the SHRINE Dashboard can be found on the HMS Sonatype Nexus server at https://repo.open.med.harvard.edu/nexus/content/groups/public/net/shrine/dashboard-war/. Navigate to the folder for 1.20.1. From there, download **dashboard-war-1.20.1.war** to the **webapps/** directory on the SHRINE server and rename it to **shrine-dashboard.war**.

For example:

```
$ cd /opt/shrine/tomcat/webapps
$ wget --no-check-certificate https://repo.open.med.harvard.edu/nexus/content/groups/public/net/shrine/dashboard-war/1.20.1/dashboard-war-1.20.1.war -O shrine-dashboard.war
```

Create dashboard.conf File

A new .conf file is required for the new SHRINE Dashboard component. Its contents are relatively simple, and can be mostly copied from an existing **steward.conf**.

1. Create a file named **dashboard.conf** in the **tomcat/lib/** folder.
2. The first line of the file should be "**shrine {**", and the last line of the file should be "**}**".
3. In between these two lines, copy the contents of the **pmEndpoint{}** block from **shrine.conf** or **steward.conf**.
4. Create an **authenticate{}** block, and within that block, create a **usersource{}** block. Within that block, there should be one variable, "domain", which should be set to the same i2b2 domain referenced in the **hiveCredentials** block of **shrine.conf**.
5. Afterwards, copy the contents of the **keystore{}** block from **steward.conf** into this file.
6. Save this file (of course).

dashboard.conf should look something like this:

```
shrine {
  pmEndpoint {
    //URL for the PM Service, used to authenticate users
    url = "http://our.site.edu/i2b2/services/PMService/getServices"
  }

  authenticate {
    usersource {
      domain = "i2b2demo"
    }
  }

  keystore {
    file = "/opt/shrine/shrine.keystore"
    password = "PASSWORD"
    privateKeyAlias = "our.site.edu"
  }
}
```

(Optional) For Users of Oracle or SQL Server

If you experience issues in getting the SHRINE Data Steward (or the new audit logging functionality) working with Oracle or SQL Server, please try the instructions in this article written by a SHRINE developer: Using SHRINE Data Steward with Oracle or SQL Server

This article contains links to alternative database drivers, as well as links to alternative database scripts for creating tables for the SHRINE Data Steward. It also includes information on the necessary changes to make to **shrine.xml**, **steward.xml**, **shrine.conf**, and **steward.conf**.

Also note that due to the new query audit database functionality added in SHRINE 1.20.1, you will also need to modify shrine.conf. There are audit{} blocks that must be added to both the adapter{} and queryEntryPoint{} sections of shrine.conf. By default, these are configured to use MySQL, so users of MySQL do not need to add anything to their shrine.conf. However, users of Oracle or SQL Server for SHRINE must add them. See the below for an example:

```
shrine {
[...]
  queryEntryPoint {
    audit {
      database {
        slickProfileClassName="freeslick.driverNameHere$"
      }
    }
    [...]
  }
[...]

  adapter {
    audit {
      database {
        slickProfileClassName="freeslick.driverNameHere$"
      }
    }
    [...]
  }
[...]
}
```

Replace "freeslick.driverNameHere$" with the name of the driver you are using ("freeslick.OracleProfile$" for Oracle, and "freeslick.MSSQLServerProfile$" for SQL Server).

Create Query Audit Databases

SHRINE 1.20.1 adds query audit databases, which will need to be created in order for SHRINE 1.20.1 to function.

1. Create a database named **adapterAuditDB** on the same database server used for storing SHRINE query history.
2. Download and run the schema creation script for the **adapterAuditDB** database, available here: https://open.med.harvard.edu/stash/projects/SHRINE/repos/shrine/browse/adapter/adapter-service/src/main/sql/mysql.ddl?at=tags/1.20.1&raw
    a. Other schema creation scripts can be found in the same folder: https://open.med.harvard.edu/stash/projects/SHRINE/repos/shrine/browse/adapter/adapter-service/src/main/sql/?at=tags/1.20.1
3. Create a database named **qepAuditDB** on the same database server used for storing SHRINE query history.
4. Download and run the schema creation script for the **qepAuditDB** database, available here: https://open.med.harvard.edu/stash/projects/SHRINE/repos/shrine/browse/qep/service/src/main/sql/mysql.ddl?at=tags/1.20.1&raw
    a. Other schema creation scripts can be found in the same folder: https://open.med.harvard.edu/stash/projects/SHRINE/repos/shrine/browse/qep/service/src/main/sql?at=tags/1.20.1
5. In **tomcat/conf/Catalina/localhost/shrine.xml**, add two new **<Resource>** elements for each of these new databases. You can copy and paste the lines used for **shrine_query_history** Resource, making sure that both the "**shrineDB**" part of the name attribute and the "**shrine_query_history**" part of the url attribute both change to "**adapterAuditDB**" for the first copy and "**qepAuditDB**" for the second copy.

Update ERROR_RESULT Table Structure

In order to accommodate improvements made to error reporting in the SHRINE webclient, additional columns have been added to the ERROR_RESULT table in the **shrine_query_history** database. Run the following queries on your SHRINE query history database:

mysql -u shrine -p -D shrine_query_history

```
alter table ERROR_RESULT add column CODEC varchar(256) not null default "Pre-1.20 Error";
alter table ERROR_RESULT add column STAMP varchar(256) not null default "Unknown time and machine";
alter table ERROR_RESULT add column SUMMARY text not null;
alter table ERROR_RESULT add column PROBLEM_DESCRIPTION text not null;
alter table ERROR_RESULT add column DETAILS text not null;
```

shrine.conf Changes

(Optional) New Property - create

The hub{}, adapter{}, and queryEntryPoint{} components in shrine.conf have an additional (optional) boolean property named "create". This will enable or disable creation of that component. For example, a site that serves solely as a data source (no webclient, no querying users at their site) could disable the hub{} and queryEntryPoint{} components.

In SHRINE 1.20.1, the default value for this is true, but this is subject to change in a future release.

```
hub {
  ...
  create = false // this is solely an example, not a mandatory change.
  ...
}
```

steward.conf Changes

Change slickProfileClassName

Due to an underlying library upgrade, the names of the database drivers used for this variable have changed. If your current value for slickProfileClassName starts with "scala.", remove "scala." from it. For example, change this:

```
slickProfileClassName = "scala.slick.driver.MySQLDriver$"
```

To this:

```
slickProfileClassName = "slick.driver.MySQLDriver$"
```

Wrapper for usersource block

Surround the usersource{} block with a new block called authenticate{}. For example:

```
  authenticate {
    usersource {
      domain = "i2b2demo"
    }
  }
```

(Optional) Increase pmEndpoint timeout

If you experience issues with the SHRINE webclient occasionally failing to retrieve a list of approved topics and are certain the SHRINE Data Steward application is otherwise functioning properly, try increasing the timeout for the SHRINE Data Steward's interactions with the i2b2 PM cell. By default, the SHRINE Data Steward will only wait 1 second for a response from i2b2. To increase this limit, add the following inside the pmEndpoint{} block:

```
  pmEndpoint {
    [...]
    timeout {
      seconds = 10
    }
  }
```

Start SHRINE

The only thing left to do at this point is start SHRINE back up. Simply do the following:

```
$ shrine_startup
```

If the above command is not found, try the following instead:

```
$ /opt/shrine/tomcat/bin/startup.sh
```

Verify SHRINE Upgrade

After starting SHRINE up, verify that the upgrade was properly deployed by checking the SHRINE Dashboard. The exact address you will need to go to depends on your configuration, but the general format looks like the following:

```
https://your.shrine.host:6443/shrine-dashboard/client/index.html#/dashboard/diagnostics
```

It may take a few seconds for the page to load, but after it does load, log in with your SHRINE credentials (any user will do, regardless of role) and verify that the value for "SHRINE Version" is 1.20.1. If it is still displaying an old version, repeat the instructions in the "**Deploy New shrine.war**" section, restart SHRINE, and try again.