Installation Guide - UnifiedViews 3.X

There are basically two ways how to install UnifiedViews - using Docker images, or from sources.

If you do not need to have local version of the whole UnifiedViews, but just need to develop DPUs, see "Installation (for DPU developers)" below.

General prerequisites

- Recommended RAM memory for the machine running UnifiedViews - at least 4 GB.
- Installed Java 8+ JDK. (Java 8+ JRE should be enough as well, if you do not plan to e.g. develop DPUs on that machine)
- Browser which supports HTML5 canvas element, see minimum versions of browsers supporting canvas element.
    - So that you can use Unifiedviews UI.
- If you plan to also develop DPUs, you should also have IDE, such as Netbeans or Eclipse or Intellij

Installing using Docker Images

See the images & readme: https://github.com/tenforce/docker-unified-views/tree/feature/compactmodular-3.0-releasebuild

Build & Install from sources

When installing from sources, you have to install UnifiedViews - Core. Please note that after installing UnifiedViews - Core, there will be no plugins (DPUs) available - plugins must be installed separately, see below.

Prerequisities:

- See the general prerequisites above
- Installed Apache Tomcat 7+
    - Installation (from Apache): https://tomcat.apache.org/tomcat-7.0-doc/setup.html
        - You can install multiple instances of Tomcat running as a service
- Installed Mysql for relational database used for storing configuration of pipelines, DPUs, ...
    - As an alternative, PostgreSQL, MS SQL, or Oracle may be also used
    - For Mac OS X:
        - Download: https://dev.mysql.com/downloads/mysql/ DMG archive, Open it, install
        - To run mysql: sudo /usr/local/mysql-5.7.19-macos10.12-x86_64/bin/mysqld_safe --user=mysql --max_allowed_packet=64M
        - To connect via mysql console: sudo mysql -uroot -pXXX
        - SET PASSWORD = PASSWORD('newpass');
        - `create database unifiedviews;`
- Installed Apache Maven 3.
    - You may skip installation of Git, you can also just copy the relevant files from a different computer (but make sure that code build using Java 7 is not executed by Java 8 RE and vice versa.)
- Installed Git version control tool It may be provided together with certain IDEs
    - You may skip installation of Git, you can also just copy the relevant files from a different computer (but make sure that code build using Java 7 is not executed by Java 8 RE and vice versa.)

UnifiedViews - Plugins-devEnv + Core

1. Obtain the source code from UV/Plugins-devEnv (latest release can be found at releases page). Read the release notes.
    a. Clean and build the downloaded code by running mvn install inside root folder.

2. Obtain the source code from UV/Core, latest release can be found at releases page. Read the release notes.
    a. Clean and build the downloaded code by running `mvn install` on the downloaded `Core` folder, further denoted as `{root}`.

3. Prepare relational database. Use mysql.

    a. log to the mysql database console using `mysql -u{user}`
    b. create new database for UnifiedViews, switch to that database (i.e., type in the mysql console `create database unifiedviews; use unifiedviews;`, which will create new database unifiedviews)
    c. DB schema is automatically created and initial data automatically inserted **as the backend is executed for the first time.**

4. Adjust configuration file of UnifiedViews
    a. Copy `{root}/conf/config.sample.properties` to the `{root}/conf/config.properties` folder (or any other location you prefer)
        i. The path to your config.properties file is further denoted as `{configFileLocation}/config.properties`
    b. Adjust at least the "Core configuration" section

        i. `general.workingdir` - the directory where the working directory of the executed pipelines is located, i.e., `{root}/backend/working`
            1. Please double-check that the user under which backend/frontend runs has write access to `{root}/backend/working`.
        ii. `module.path` - the directory where the data processing units are located, i.e., `{root}/target`
            1. Warning: If you specify different directory then `{root}/target`, you have to copy `{root}/target/lib` folder (which is generated after running `mvn install`) to `{module.path}/lib` folder, so that shared libraries are available to DPUs.
            2. Please double-check that the user under which backend/frontend will be running has write access to `{module.path}/dpu`, `{module.path}/lib folders`.
        iii. Specify the connection to the relational database where configuration of pipelines is stored using `database.sql.*` properties
        iv. Specify the connection to RDF store used for intermediate results of the pipelines' executions using `database.rdf.*` properties. There are three choices where intermediate RDF data may be stored: localRDF (rdf4j Native repository), remoteRDF (using Sesame, native repositories), and GraphDB; based on the selected choice, please uncomment the set of related `database.rdf.*` properties.
5. To run backend - engine executing data processing tasks:
    a. **Run backend** using `nohup java -DconfigFileLocation={configFileLocation}/config.properties -jar {root}/backend/target/backend-{version}.jar &`
        i. Note: Parameter `-DconfigFileLocation` specifies where the config.properties file is located.
    b. Check write access of backend to certain folders:
        i. Check that backend has write access to `{root}/backend/working, where backend stores data about executions.`
        ii. Check that backend has write access to the directory to which backend is logging (which is specified in `{configFileLocation}/config.properties`).
    c. **Note: Running backend will also create DB schema**
6. To deploy & run frontend - management GUI of the tool:
    a. Adjust the config option `-DconfigFileLocation` within `CATALINA_OPTS` in `{tomcatHome}/bin/setenv.sh`, so that tomcat knows where `{configFileLocation}/config.properties` should be searched. You can also directly edit `{tomcatHome}/bin/catalina.bat/catalina.sh` (i.e., CATALINA_OPTS="$CATALINA_OPTS -DconfigFileLocation={configFileLocation}/config.properties -server -Xms1024m -Xmx2048m")
    b. Deploy frontend war file -- `{root}/frontend/target/unifiedviews.war –` to the application server (Tomcat), to the `{webapp}` folder.
    c. Start the application unifiedviews.war
    d. Check that the application is running under the context `/unifiedviews`. Google Chrome is the preferred browser. The default passwords are `admin/test` for admin account and `user/test` for user account
        i. Note: before being able to log into the UI, you have to run backend first to create the DB schema with initial data
    e. Check write access of frontend to certain folders:
        i. Check that frontend has write access to `{root}/target/dpu` (needed when frontend imports DPUs)
        ii. If you specified certain log directory to which frontend is logging (in config.propeties of fronted), check that frontend has write access to such folder.

UnifiedViews - Plugins

After finishing the steps above, the installed instance of UnifiedViews still does not contain any plugin (DPU). To install plugins please follow these steps:

1. Obtain the source code from UV/Plugins (latest release can be found at releases page). Read the release notes.
    a. Clean and build the downloaded code by running mvn install inside root folder.
        i. Do not forget to use "mvn install -P java8" if you want to build code for Java8
2. As a result, all DPUs should be built, the resulting JAR files (DPU bundles) are in {dpuName}/target folder.
3. You may then
    a. individually import selected DPUs (JAR files) via GUI of UnifiedViews (DPUTemplate -> Create DPU template") or
    b. pack the generated DPUs inside a zip archive, which may be then imported via GUI (DPUTemplate -> Create DPU template -> zip folder) - when importing zip archive, it basically tries to import all JAR files (DPUs) in the zip archive regardless of the folder in which they are

Advanced installation Options:

Installation (for DPU developers)

1. Obtain the source code from https://github.com/UnifiedViews/Plugin-DevEnv, release or master/develop branch (depends on to which version you want to develop).
2. Clean and build the project by running `mvn clean install.`
3. `Java IDE (Eclipse, Netbeans), New Project, from Maven archetype, uv-dpu-template-base. Define basic metadata for the DPU. As a result, skeleton for the DPU is prepared.`