

Java and Memory in IUCLID 6

Changes to this document

Date	Modification
12/01/2018	Updated for IUCLID 6 v2.0.0.
30/03/2017	First version

Table of Contents

1. Introduction to Java and memory in IUCLID 6	1
2. Make sure there is enough memory for all applications	2
3. Assign enough memory to each Java application	3
3.1.1. IUCLID 6 Server and IUCLID 6 Desktop	3
3.1.2. Client for IUCLID 6 Server	4
3.1.3. Updater tool	7
3.1.3.1. Setting the memory for the Updater tool when run with no graphical interface.....	7
3.1.3.2. Setting the memory for the Updater tool when run with a graphical interface.....	7

1. Introduction to Java and memory in IUCLID 6

IUCLID 6 is a Java application and therefore needs Java. The same is true for all the IUCLID 6 tools for example: *Migrator*, *Database patch tool*, *Updater* and *Start-up Fixer*.

IUCLID 6 Server and IUCLID 6 Desktop need JDK 8, whereas for the tools listed above, and the client of IUCLID 6 Server, either JDK 8 or JRE 8 can be used.

The installation packages of IUCLID 6 contain Oracle Java 8 (JDK), and therefore a separate installation of Java is not needed. Although IUCLID 6 tools can be started by running their scripts from any directory, the best practice is to place the installation directory of each tool in the installation directory of IUCLID 6. This ensures that the tools use the installation of Java that is delivered with IUCLID 6. An example is shown below. If a tool is run from some other directory, it will try to use a system installation of Java instead.

For Java applications to run there must be:

- a. Enough memory available in the computer for all the applications.
- b. Enough memory must be assigned to each Java application.

If there is not enough memory, a Java application may not start, or an error may occur during execution, of type, *Out of Memory exception*.

The following can provide the memory described above:

- a. RAM;
- b. Virtual memory, also known as a paging file, or swap, which uses file storage space to emulate RAM.

Virtual memory is much slower than RAM because it involves writing to a file, but it is almost inexhaustible. The operating system can be set so that it uses virtual memory automatically when required. A system should be set up with sufficient RAM so that virtual memory is used only rarely, for example when an unusually large Dossier is created.

First ensure that there is enough memory to run all the applications that must run at the same time, and then ensure that there is enough memory per individual Java application. This is described in the following sections.

2. Make sure there is enough memory for all applications

All applications that run on your computer, including the operating system (OS), require memory. The more applications that are run in parallel, the more memory is required in total. For example if you are executing a migration from IUCLID 5.6 Standalone to IUCLID 6 Desktop and all the components are run on the same computer, you will need to provide enough memory for:

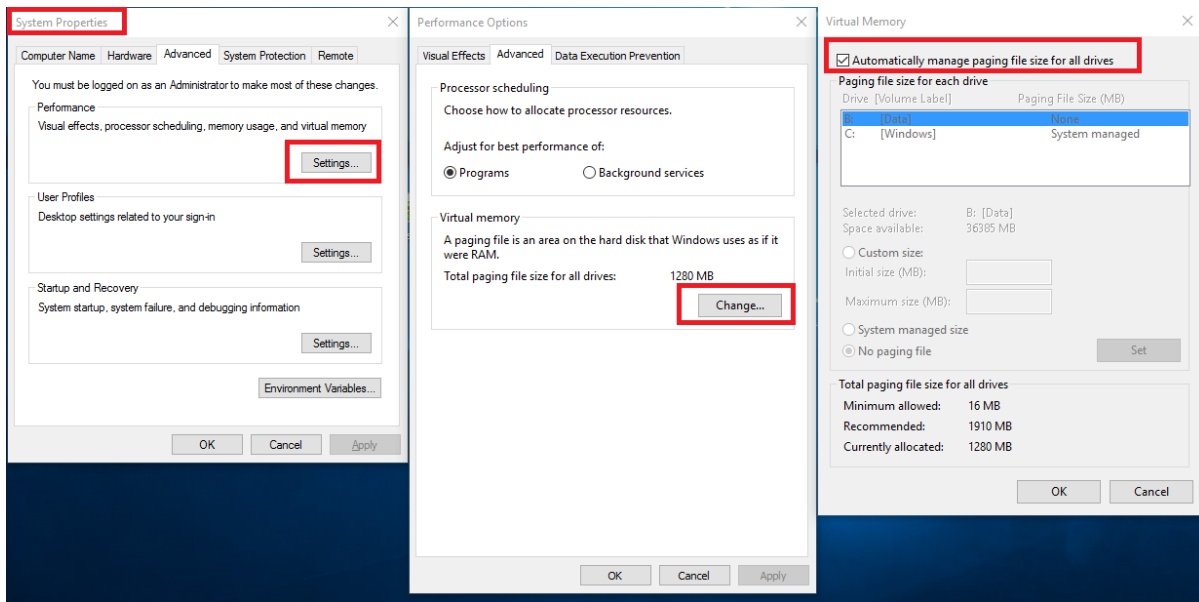
- a. OS;
- b. PostgreSQL that contains the IUCLID 5.6 database;
- c. Installer.

If you are using 32 bit version of IUCLID/Java the theoretical maximum amount of memory that can be allocated is 4 GB. However, in practice the limit is less, depending on the details of the machine and the OS. For Windows, limits of less than 2 GB have been reported. To provide more memory you can:

- a. Use 64 bit IUCLID/Java;
- b. Close applications you do not need;
- c. Install more RAM on the machine;
- d. Increase the amount of virtual memory.

The more recent versions of the Windows OS should provide enough memory for all running applications if they are configured to automatically manage the paging size. In this way, the OS increases and decreases the size of the paging file, as and when required. An example of where to configure virtual memory in Windows is shown below:

System Properties > Performance Options > Virtual Memory



3. Assign enough memory to each Java application

In Java 8 the following configuration parameters are available to assign memory:

- Initial heap size: Xms
This is the amount of memory given to the application when it first starts.
- Maximum heap size: Xmx
This is the maximum amount of memory that the application can use.

Xms must be smaller than or equal to Xmx. In a stable system that never requires virtual memory, they may be given the same value.

When a Java application is first started up, Java runtime:

- Checks whether there is enough memory to reserve Xmx. If there is not enough free memory available at that moment, the application will not run.
- If there is Xmx memory available, it reserves Xms memory. Xms is less than or equal to Xmx.

Later, whilst the application is running:

- If more memory than Xms is needed, Java runtime will attempt to reserve more memory, but never more than Xmx. If at that moment there is not enough free memory, the application may experience errors.
- If more than Xmx memory is needed, an error occurs of type, *Out of Memory exception*.

3.1.1. IUCLID 6 Server and IUCLID 6 Desktop

The memory parameters described above can be configured in the following configuration file:

```
<IUCLID 6 installation
folder>\glassfish4\glassfish\domains\domain1\config\domain.xml
```

The section is:

```
iuclid_6_java_memory_en.docx
```

```
<configs>/<config name="server-config">/<java-config>;
```

Note that a value of Xmx is also set in the following section, but it is not read by IUCLID 6, and can therefore be ignored.

```
<configs>/<config name="default-config">/<java-config>;
```

An example for IUCLID 6 Server showing the context is given below:

```
<java-config classpath-suffix="" system-classpath="">
  <jvm-options>-XX:PermSize=64m</jvm-options>
  <jvm-options>-XX:MaxPermSize=192m</jvm-options>
  <jvm-options>-Xmx4096m</jvm-options>
  <jvm-options>-XX:NewRatio=2</jvm-options>
```

If you want to set a value for Xms it can be done next to Xmx, as shown in the example below:

```
<java-config classpath-suffix="" system-classpath="">
  <jvm-options>-XX:PermSize=64m</jvm-options>
  <jvm-options>-XX:MaxPermSize=192m</jvm-options>
  <jvm-options>-Xmx4096m</jvm-options>
  <jvm-options>-Xms4096m</jvm-options>
  <jvm-options>-XX:NewRatio=2</jvm-options>
```

For IUCLID 6 Desktop the default value of Xmx is 1 GB (-Xmx1024m). For IUCLID 6 Server it is 4 GB (-Xmx4096m).

To change the memory parameters:

- a. Stop IUCLID 6.
- b. Create a backup copy of domain.xml.
- c. Edit domain.xml then save it.
- d. Start IUCLID 6.

The best values of Xmx and Xms to use depend upon on the specific system and its parameters, for example for IUCLID 6 Server, the number of simultaneous users/actions and types of actions they execute. A system has enough memory if it is stable, runs quickly enough, and there are no Out of Memory exceptions.

Increasing the memory requirement above 3 GB requires the 64 bit version of IUCLID/Java in most cases.

Be aware that for IUCLID 6 Server a value of Xmx lower than 4096m can cause problems during the importation of data.

3.1.2. Client for IUCLID 6 Server

The default value of the maximum heap size (Xmx) for the client of IUCLID 6 Server is 1 GB (-Xmx1024m). The value is set by a parameter in the jnlp file that is downloaded and run on clicking the Launch button on the webservices page of IUCLID 6 Server. If you would like all clients to use the same customised value of Xmx, edit the jnlp file from within IUCLID 6 Server. This requires a restart of IUCLID 6 Server and affects all users. It is also possible to download the jnlp file locally,

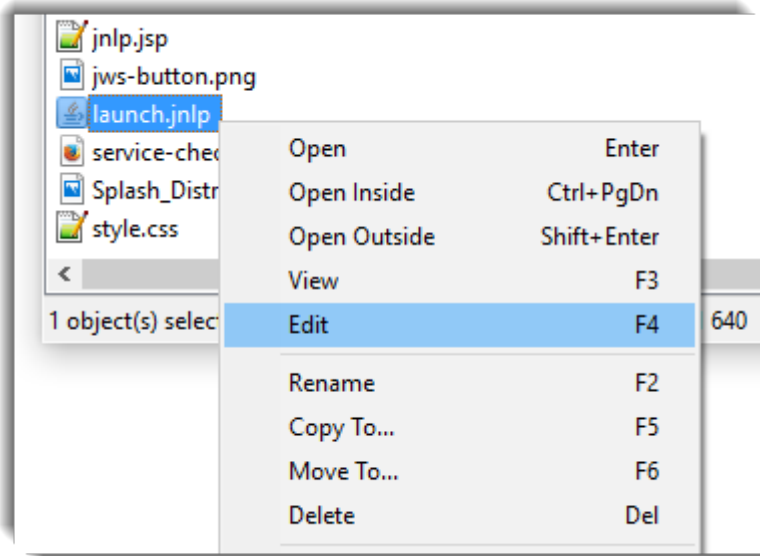
edit it, and then run it manually. This affects only the client run from the edited jnlp file, and does not require a restart of IUCLID 6 Server.

To edit the file `launch.jnlp` from within IUCLID 6 Server, carry out the following procedure:

1. Stop IUCLID 6 Server.
2. Edit the file `launch.jnlp` from within the archive file:

```
<installation directory>  
\glassfish4\glassfish\domains\domain1\iuclid6\iuclid6-war-  
<version>.war
```

For example, this can be done in Windows by opening the `war` archive using the application 7-zip, right-clicking on the `launch.jnlp` file, and then selecting edit, as shown below:



3. Change the value of the parameter `max-heap-size`, for example: `max-heap-size="4096m"`.
4. Save the file `launch.jnlp`.
5. Start IUCLID 6 Server.

To edit and use a local copy of `launch.jnlp`, carry out the following procedure:

1. Right-click on the button labelled *Launch* and then save locally the file `launch.jnlp`, as shown in the example below:



2. Open the file `launch.jnlp` in your favourite text editor.
3. Change the value of the parameter `max-heap-size`, for example, the value `max-heap-size="2048m"` set in the editor Notepad++ looks like the following:

```
<resources>
  <j2se version="1.5+" max-heap-size="2048m"
  java-vm-args="-client -da
  -Dsun.java2d.dpiaware=false"/>
```

4. Save the file `launch.jnlp`.

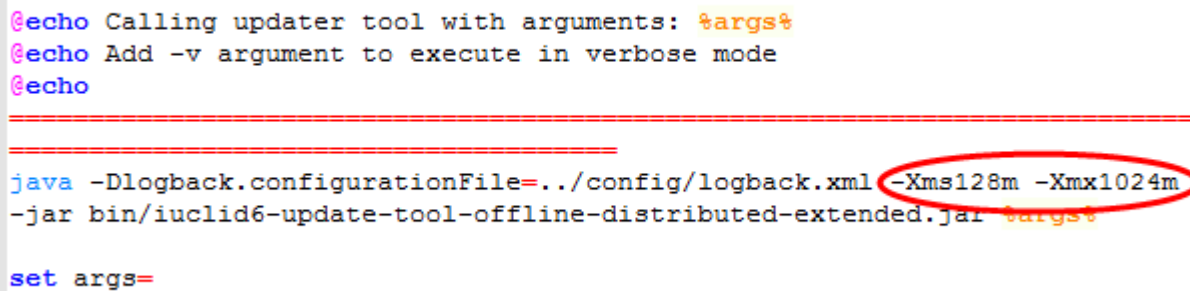
5. Run the file `launch.jnlp`, either by double-clicking on it, or from the command line using the command:

```
javaws.exe launch.jnlp
```

3.1.3. Updater tool

The *Updater tool* can be run either with or without a graphical user interface. In both cases, the maximum heap space (Xmx) is defined in the following file:

```
<IUCLID updater tool>\bin\internal\updater.cmd
```



```
<IUCLID updater tool>
@echo Calling updater tool with arguments: %args%
@echo Add -v argument to execute in verbose mode
@echo

java -Dlogback.configurationFile=../config/logback.xml -Xms128m -Xmx1024m
-jar bin/iuclid6-update-tool-offline-distributed-extended.jar %args%

set args=
<IUCLID updater tool>
```

For example, to set the maximum amount of memory to 4 GB, use the value `-Xmx4096m`.

3.1.3.1. Setting the memory for the Updater tool when run with no graphical interface

Change the value of **Xmx** as shown above.

Start the *Updater tool* by running either `iuclid6-update.cmd` or `iuclid6-update.sh`. Do not close the command window whilst the *Updater tool* is running.

3.1.3.2. Setting the memory for the Updater tool when run with a graphical interface

Change the value of **Xmx** as shown above.

Convert the script that is used to run the *Updater tool* with no graphical user interface, in to one that uses a graphical interface and reads `updater.cmd` or `updater.sh`. To do that, edit the script `iuclid6-update.cmd`, changing the text **cli-based** to **ui-based**. Save the script with a new name, such as `iuclid6-update-gui.cmd`.

Start the *Updater tool* by running the new script, for example either `iuclid6-update-gui.cmd` or `iuclid6-update-gui.sh`. Do not close the command window whilst the *Updater tool* is running.