IVrixDB Quick Start

1. Download IVrixDB Repository + Resources
2. Setup OpenJDK 12
3. Setup Deployment Environment
   a. Setup External Zookeeper
   b. Setup Solr Cluster
4. Setup Development Environment
   a. Compile Lucene, Solr, and IVrixDB
5. Start Deployment Environment
6. Using IVrixDB

Download IVrixDB Repository + Resources

1. Download the IVrixDB repo from GitHub (it provided lots of configs and resources to get started)
2. Download data files for index ingestion/testing from this link: https://drive.google.com/drive/folders/10K0YTadhjRVHAOcqRTO5HfhCG7cd41yh?usp=sharing
   a. I suggest downloading the files one at a time. Otherwise, Google Drive behaves inconsistently

OpenJDK 12 Setup

1. Download OpenJDK 12 (I used version 12.0.1)
2. Set the home folder as JAVA_HOME environment variable
3. Place the JDK bin folder (within the home folder) into your PATH environment variable

Validation Checkpoint:

- Run command "java -version"
  - Should output JDK version 12.x.x (whichever one you installed)

Deployment Environment Setup

External Zookeeper Setup

1. Download Zookeeper (Any version after 3.5.5 works fine, though I used version 3.6.1)
2. Place the Zookeeper's bin folder your PATH environment variable
3. In zookeeper's conf folder:
   a. Delete the "zoo_sample.cfg" file
   b. Copy the "zoo.cfg" file found in "IvrixDB Resources/configs/zookeeper/…" into zookeeper's conf folder
   c. Replace the "dataDir" parameter within the "zoo.cfg" according to where you would want the data location to be
      i. I suggest creating a "zoo_data" folder within the zookeeper folder for this

        ii. For Windows: Use forward slashes and remove the drive prefix ("C:\", for example)

Validation Checkpoint:

1. Run Zookeeper with command "zkServer.cmd"
    a. Should successfully run Zookeeper on localhost:9983
    b. Should successfully create a snapshot into a folder named "version-2" within the folder of the "dataDir" parameter
2. Run Zookeeper with command "zkCli.cmd -server localhost:9983" – once connected, run "ls /"
    a. Should successfully connect to Zookeeper
    b. "ls /" should output "[zookeeper]"

Solr Cluster Setup

1. Place the folder ".../IVrixDB/solr/server/bin" into your PATH environment variable
2. For every Solr Node you want to configure in your Cluster – create a folder and copy the "solr.xml" config file found in "IVrixDB Resources/configs/solr/..." into the newly created folder (change the port as you see fit)

Development Environment Setup

1. Install Postman and Import the IVrixDB Postman configs found in "IVrixDB Resources/configs/postman/..." to easily interact with IVrixDB
    a. I also highly recommend importing the Solr Postman configs, as I put a lot of Solr's commands in it

2. Install IntelliJ IDEA (the IVrixDB repo is setup with IntelliJ to easily get started)
3. Copy the ".idea" folder in "IVrixDB Resources/configs/intelij/..." into the main folder
4. Open the repo with IntelliJ
5. Set JDK 12 as the java environment within Project Structure (Ctrl+Alt+Shift+S)
6. Run *prerequisite-ivy-bootstrap*
    a. This is to prevent a compilation error regarding missing Ivy. Sometimes you may face issues with Ivy, like an incomplete downloaded artifact. Cleaning up the Ivy cache and retrying is a workaround for most of such issues: "rm -rf ~/.ivy2/cache"

Compiling Lucene, Solr, and IVrixDB

As IVrixDB is currently embedded within Solr, the compilation of Solr compiles IVrixDB as well. So, when developing IVrixDB, I simply make changes inside the ivrixdb package, and then recompile with *compile-solr*. I used to also use IntelliJ's standard build feature to double-check the Java compilation, although this can lead to numerous issues, so I just left it out of the IntelliJ config. Compile the code as follows:

1. Run *compile-lucene*
2. Run *compile-solr*

Validation Checkpoint:

- Should successfully conclude *compile-lucene* and *compile-solr* with “Ant build completed with…”. Ignore the errors ***if-and-only-if*** a successful conclusion has occurred. Those errors are almost certainly about missing JAR files that the compilation downloads right after they were not found. (Though do not take my word for it, double check this yourself.)

Start Deployment Environment

1. Run Zookeeper on localhost:9983 with command “zkServer.cmd”
2. For every Solr Node in your cluster, start it with “ solr start -c -p {port} -s "{solr_node _directory}" -z localhost:9983 ”
    a. change {port} and {solr_node _directory} as required
    b. *This will start the Solr Node with IVrixDB code embedded within*

3. Execute Postman request “ADD IVRIX CONFIGSET” to upload the “ivrix_configset” into solr
4. Execute Postman request “CREATE IVRIX CONTROLLER COLLECTION”, which will create an empty "ivrixdb" controller collection
5. Execute Postman request “CREATE IVRIX CONTROLLER REPLICA ON NODE” for each node you want IVrixDB to run on. From here, IVrixDB will automatically boot up.

Using IVrixDB

Indexing

1. Execute Postman request “CREATE INDEX” (change the name to your liking)
2. Execute Postman request “LOAD-BALANCED INDEXING OF EVENTS”. I like to use the large test file from the download link above

Searching

1. Execute Postman request “CREATE SEARCH JOB (ALL)”, which will quickly return a Search ID
2. Place the given Search ID into the IVrixDB environment variable “search_job”
3. Poll any component to your liking

Some Notes and Heads-Up

Development

- If you want logging, override the “/solr/server/resources/log4j2.xml” with the file log4j config in “IVrixDB Resources”
- If you are struggling booting up solr, try overriding some of the files in “/solr/bin” with the solr bin config files in “IVrixDB Resources”
- when developing IVrixDB, make changes inside the ivrixdb package, stop the Solr nodes, recompile with *compile-solr*, and restart the nodes.

- Configsets in Solr must be zipped (from inside the conf folder) and POSTed to Solr. So, when making changes to “ivrix_configset”, delete the old version from Solr, and then re-POST the new version back in. This kind-of sucks because you also must delete all the Solr Cores and start over, as I couldn’t get them to move to the new versions of the configset.

Debugging

- To debug IVrixDB within Solr (or just Solr in general):
    - Set Run Configuration in IntelliJ – solr-server-remote-x (remote JVM Attach-at-port-{{address}})
    - Start Solr with "-a" parameter with value "-agentlib:jdwp=transport=dt_socket,server=y,suspend=n,address=*:{{address}}"
        - change {{address}} to the port for debug connection
- I recommend that when applying breakpoints, use it on a thread-by-thread basis, to ***not*** kill the zookeeper session. I made this mistake in the past, and it ruins the entire debugging session.

Indexing

Sadly, IVrixDB is hard coded to only support the type of data format as presented in the test data for ingestion. This is due to hard-coded STFE, no automatic timestamp extractor, and only a JSON stream reader. In the future this will change.

Searching

- Within the search decorator, the collection name is treated as the name of the IVrixDB index you are requesting to search
- fl and sort are not hard-coded, but they should be – ***DO NOT*** change them, or the search will not run properly
- There is a lot more examples of different types of searches – just look at the TESTS request folder, under section 2 in Postman