Developer’s Guide

# Pre-requisites

1. Maven
   1. Download from <https://maven.apache.org/download.cgi>
      1. Add to path if on windows
   2. Or install through brew
      1. brew install maven
   3. Or install through apt-get
      1. Sudo apt-get install maven
2. Git
   1. Download through <https://git-scm.com/>
   2. Or install through brew
      1. Brew install git
   3. Or install through apt-get
      1. Sudo apt-get install git

# Building A-Expression dependency

1. Clone repository
   1. git clone <https://github.com/KRAETS/a-expression>
2. Move into the cloned directory
   1. cd a-expression
3. Build through maven
   1. mvn clean install
4. Done

# Building KQL

1. Clone repository
   1. git clone <https://github.com/KRAETS/kql.git>
2. Move into the cloned directory
   1. Cd kql
3. Import the project into eclipse indigo and build the grammar files
   1. Download eclipse indigo here <https://eclipse.org/downloads/packages/eclipse-ide-java-ee-developers/indigosr2>
   2. Install the antlr plugin in eclipse indigo
      1. In eclipse marketplace search for ANTLR IDE
      2. Install the corresponding packages
   3. Configure the plugin
      1. Window->Preferences->ANTLR->Builder
      2. Click Add… under installed packages
      3. Click in the Directory button and search for /Path/To/KQL/Project/src/resources
      4. Select antlr-3.5.2-complete.jar
      5. Click apply
      6. Click Ok
   4. Install Maven Indigo integration
      1. Add the repository
         1. Help->Install New Software->Work With <http://download.eclipse.org/releases/indigo/>
      2. In type filter write maven
      3. Under collaboration click m2e
      4. Click next
      5. Follow installation procedure
      6. Alternately, install older version of m2e from http://download.eclipse.org/technology/m2e/releases/1.3/1.3.1.20130219-1424
   5. Import KQL project
      1. File->Import->Maven->Existing Maven Projects
      2. Under root directory select the KQL directory
      3. Select next
   6. Set project as ANTLR project
      1. Right click the project on the left sidebar
      2. Configure->Convert to ANTLR project…
   7. Optional
      1. If there is an Unbound JRE problem, right click the project on the left sidebar
         1. Properties->Java Build Path->Libraries
         2. Remove the library with a red X
         3. Click on the Add Library button
         4. Select JRE System Library
         5. Point it to a Java 1.7 JDK
            1. It can be downloaded here <http://www.oracle.com/technetwork/java/javase/downloads/jdk7-downloads-1880260.html>
   8. Perform a project->clean and project->build to generate language files (not tested yet, grammar files were already built)
4. Add unityjdbc as a Maven dependency
5. Build the project through maven
   1. mvn clean install assembly:single –DskipTests

# Building the Elasticsearch-KQL plugin

1. Clone the repository
   1. git clone <https://github.com/KRAETS/elasticsearch-kql.git>
2. Change into the cloned directory
   1. cd elasticsearch-sql
3. Copy unityjdbc.jar as unityjdbc-1.jar into ~/.m2/mongodb/jdbc/1/
4. Build through maven
   1. mvn clean install assembly:single –DskipTests

# Building the NADS project

1. Clone the repository
   1. git clone <https://github.com/KRAETS/nads_example.git>
2. Change into the cloned repository
   1. cd nads
   2. cd Platform\_Module
3. Build through maven
   1. mvn clean install assembly:single –DskipTests