

# CES Installation

- General prerequisites
- Standard (and easy) setup
- High-availability setup
  - Installing GraphDB and the EUF plugin
  - Setting up a Coordinator
  - Setting up a Worker node
  - Adding a Worker node into the Coordinator

## General prerequisites

- Java 8
- Credentials to our Nexus publishing repos

## Standard (and easy) setup

1. Get a Semantic Pipeline and unzip it to a directory. For the purpose of this guide, we'll assume you have your pipeline contents unpacked in

```
/home/user/pipeline
```



Note (Jan, 2017). The Extractor-Web webapp, expects the gate init file to be located at `/home/user/pipeline/application.xgapp`. In the latest pipeline, this file was instead in the `xgapp/` folder. In this case, or if the pipeline is in different place, one can specify it explicitly via the Java option `"-Dgate.app.location=file:/home/<user>/pipeline/xgapp/application.xgapp"`.

2. Install a web application container. If you already have one - great, if not - we use [Apache Tomcat 7](#)
3. You will need to set a few JVM params, in Tomcat this is done from `*/apache-tomcat/bin/setenv.sh`. See the [worker configuration page](#).
4. Download [extractor-web.war](#)
5. You can now start your webapp container
6. Deploy the war you just downloaded. In Tomcat you simply need to move it to its `/webapps` sub-directory and it will get picked up.
7. Now go to <http://localhost:8080/extractor-web/apidocs> for live documentation.



Due to [Swagger](#) limitations, the most important endpoint, namely `/extract`, cannot have live documentation. This is why it's explained [here](#).

## High-availability setup


The high-availability setup architecture includes several components, communicating through RESTful calls. Each component has its own role in the environment. Here's a list with brief explanation of each module:

- GraphDB with EUF plugin – the **GraphDB** module maintains a semantic database, containing RDF data used within the system. Its **EUF** plugin (EUF stands for *Entity Update Feed*) is responsible for providing the outer world with notifications about every entity (concept) within the database that has been modified in any way (added, removed, edited)
- Concept Extraction API Coordinator – the **Coordinator** module accepts annotation requests and dispatches them towards a group of Concept Extraction **Workers** (see below). The Coordinator communicates with the semantic database in order to track for changes leading to updates in every Worker's Dynamic Gazetteer.
- Concept Extraction API Worker – a **Worker** module evaluates annotation requests. It maintains a pool of GATE pipeline instances, used for text analysis and concept extraction.

## Installing GraphDB and the EUF plugin

Information about installing and using the GraphDB semantic database can be found on the official [GraphDB documentation page](#).

In order to install the Entity Update Feed plugin, check the [CES Components](#) page.

 OPTIONAL: insert a single random statement having `rdfs:label` as predicate in order to activate the EUF plugin

## Setting up a Coordinator

We will be using **Apache Tomcat** as a web application container for this example.

1. Download the [Coordinator web application](#) from our Nexus instance
2. Add the coordinator-specific parameters to the Tomcat setup – use the `<tomcat-home>/bin/setenv.sh` file; example:

```
coordinator setenv.sh


#!/bin/bash

# general options -- name, storage directory location, URL (required)
export GENERAL_OPTS="-Dcoordinator.name=master
-Dcoordinator.stateDirectory=/path/to/storage/dir/coordinator
-Dcoordinator.baseUrl=http://the.base.url:7070/coordinator"

# sparql endpoint options -- location
export
ENDPOINT_OPTS="-Dcoordinator.sparql.endpoint=http://sparql.endpoint.be:8080/graphdb
VM options -- heap size, etc
export JVM_OPTS="-XX:+UseConcMarkSweepGC -XX:+TieredCompilation -Xmx1g"

export CATALINA_OPTS="$GENERAL_OPTS $ENDPOINT_OPTS $JVM_OPTS"
```

3. deploy the coordinator web application in Tomcat's `webapps` directory
4. (Re-)start the Tomcat instance

 More information about all Coordinator configuration parameters can be found [here](#).

## Setting up a Worker node

We will be using **Apache Tomcat** as a web application container for this example.

1. Download the [CES Worker web application](#) from our Nexus instance
2. Add the worker-specific parameters to the Tomcat setup – use the `<tomcat-home>/bin/setenv.sh` file; example:

```
worker setenv.sh


#!/bin/bash

# garbage collection and compiler
export J_OPTS="-XX:+UseConcMarkSweepGC -XX:+TieredCompilation -Xmx4g"


# worker options -- path to pipeline, size of pipeline pool
export W_OPTS="-Dworker.name=st-worker
-Dgate.app.location=file:/path/to/pipeline/application.xgapp
-Dpipeline-pool-max-size=2"

export CATALINA_OPTS="$J_OPTS $W_OPTS"
```

3. Deploy the `extractor-web.war` in Tomcat's `webapps` directory
4. (Re-)start the Tomcat instance

 More information about all Worker configuration parameters can be found [here](#)

## Adding a Worker node into the Coordinator

 Assumptions:


- the **coordinator** instance is located at `http://coordinator.url:7070/coordinator`
- the **worker** instance is located at `http://worker.url:6060/worker`
- the **worker** instance has a pipeline pool with size 2

Using a REST client, execute the following request to the **coordinator** instance (`http://coordinator.url:7070`):

```
POST /coordinator/workers
Content-type: application/json

[{"capacity":2, "url":"http://worker.url:6060/worker"}]
```

- `capacity` is the number of pipeline instances in the worker pool.
- `url` is the location of the worker instance.

 Instead of a REST client, one could use the Coordinator' Swagger Documentation endpoint, located at `http://coordinator.url:7070/coordinator/apidocs` and the specific **POST** or **PUT** requests.