ElbaAWS Guide

ggresham3@gatech.edu

1 Contents

Within this repository are the files for generating both Java libraries and the documentation to go along with them, including this file. The Java projects may be imported to Eclipse. A sample project (ElbaCLI) is included to show how the API can be used.

2 Dependencies

The dependencies are within the Eclipse *.project* file, but are absolute paths to libraries on my computer. What you will need to compile this project are the AWS SDK for Java, available as an Eclipse plugin or libraries, and several Apache Commons libraries. Additionally, Jaxb is used for XML parsing and is included in the repository.

3 Documentation

The documentation was done to Doxygen standards, and should be compatible with Javadoc as well. To generate files for Doxygen, use config file titled *Doxyfile* at the root of this repository.

4 Things to be changed

There are several things specific to my AWS account. The AMI, for example, is tied to my account and should be made public and copied to an account of yours. There are also several usages of a private key to authenticate with the instances that, while I can provide to you, should probably be replaced with your own key-pair. The AMI will have to be modified to reflect these changes.

It's likely that I left something hardcoded that I shouldn't have. I started using fixed directories instead of dynamically parsing them, and every instance may have not been replaced.

Load balancing is a manual process in the code still. The information is available from the parsed XML configuration, but I did not get around to implementing the automation. Regardless, Professor Pu mentioned that Apache is rarely the bottleneck, and my MySQL load balancing is, of course, incorrect as it does not use any real clustering software. You may just want to comment out any load balancer references.

5 Usage

The actual use of the API should be very simple. You need to import the package that I've created, use the Utils class to load an **AWSCredentials**, and then instantiate an **EC2Manager** using it.

```
import elbaEC2.EC2Manager;
import elbaEC2.Utils;

AWSCredentials cred =
        Utils.getCredentials("awsAccess.properties");
EC2Manager ec2 = new EC2Manager(cred);
```

The format of a AWSCredentials file using information provided by Amazon is:

```
accessKey=<AccessKey>
secreteKey=<SecretKey>
```

At this point the easiest thing to do is invoke **runExperiment** method. However, if you would like more control of the experiment you can call all the methods **runExperiment** uses directly. The uses of these are documented in the Doxygen/Javadoc format.