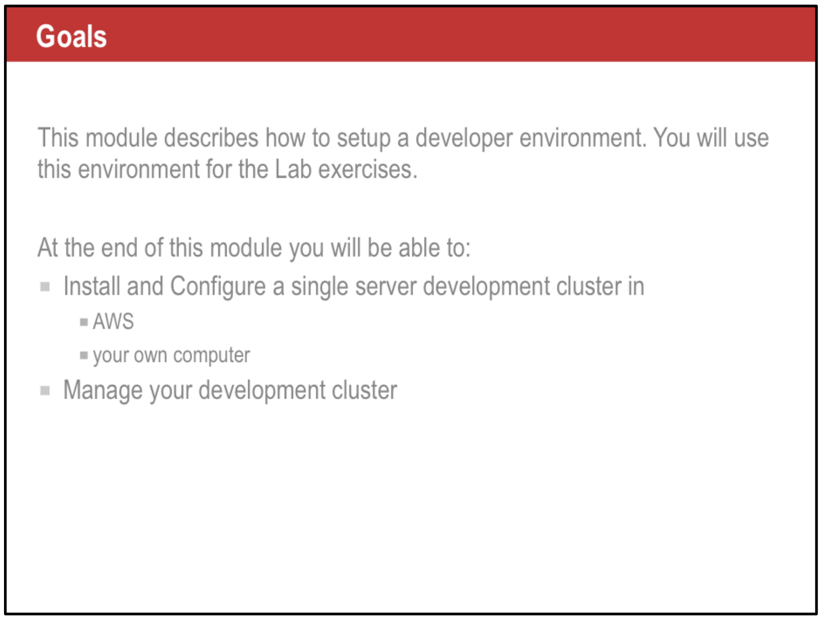
AEROSPIKE

Developer Setup

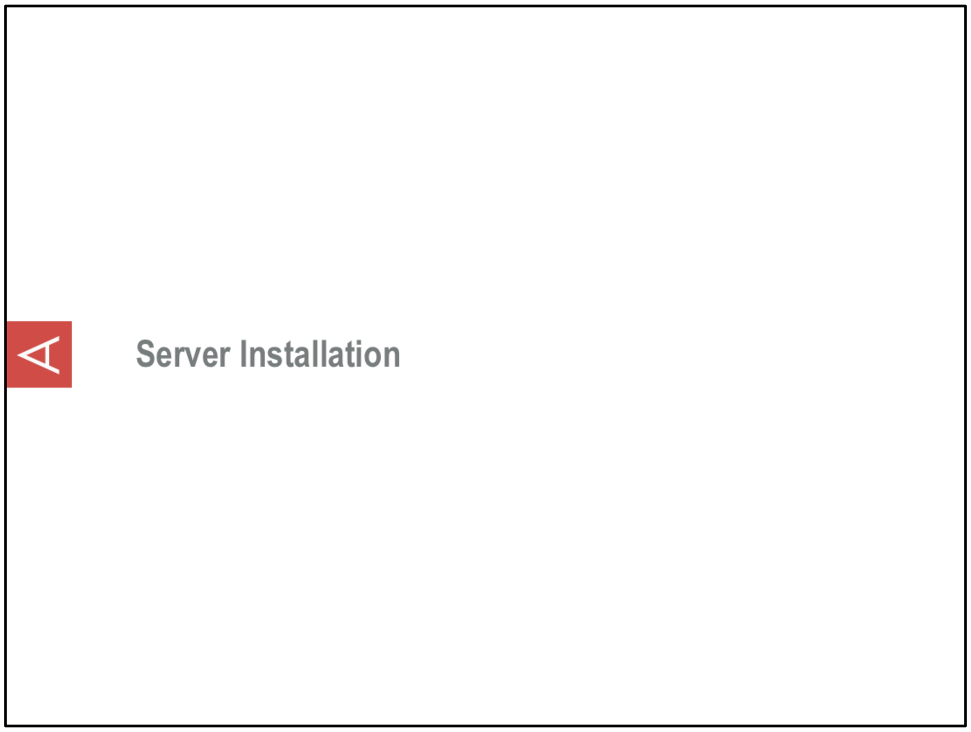


Goals

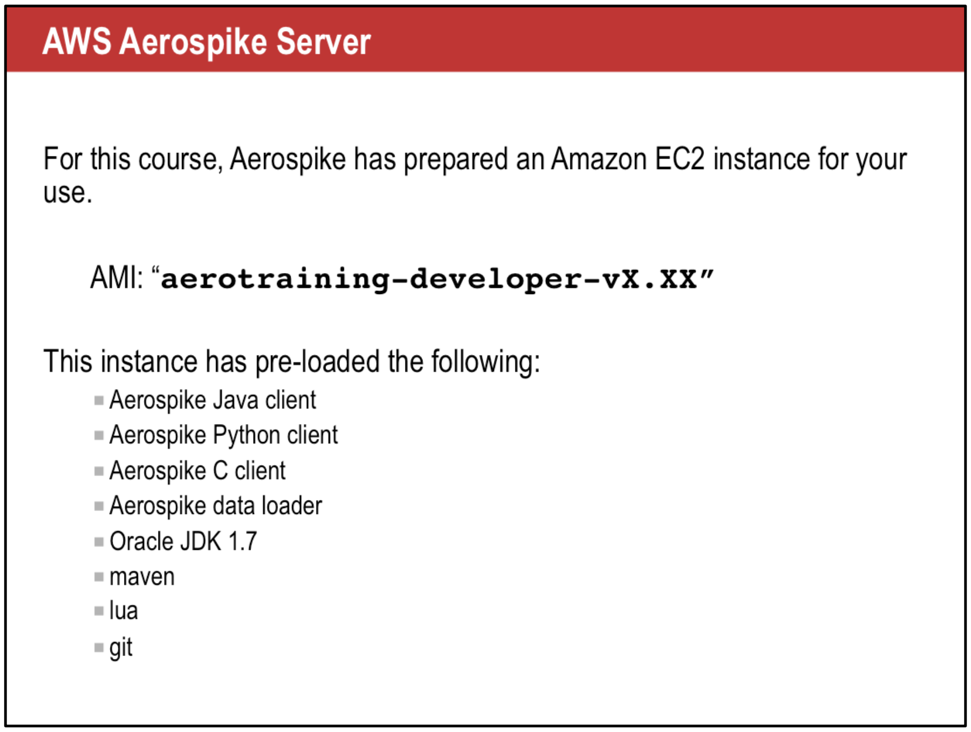
This module describes how to setup a developer environment. You will use this environment for the Lab exercises.

At the end of this module you will be able to:

- Install and Configure a single server development cluster in
  - AWS
  - your own computer
- Manage your development cluster



Server Installation



AWS Aerospike Server

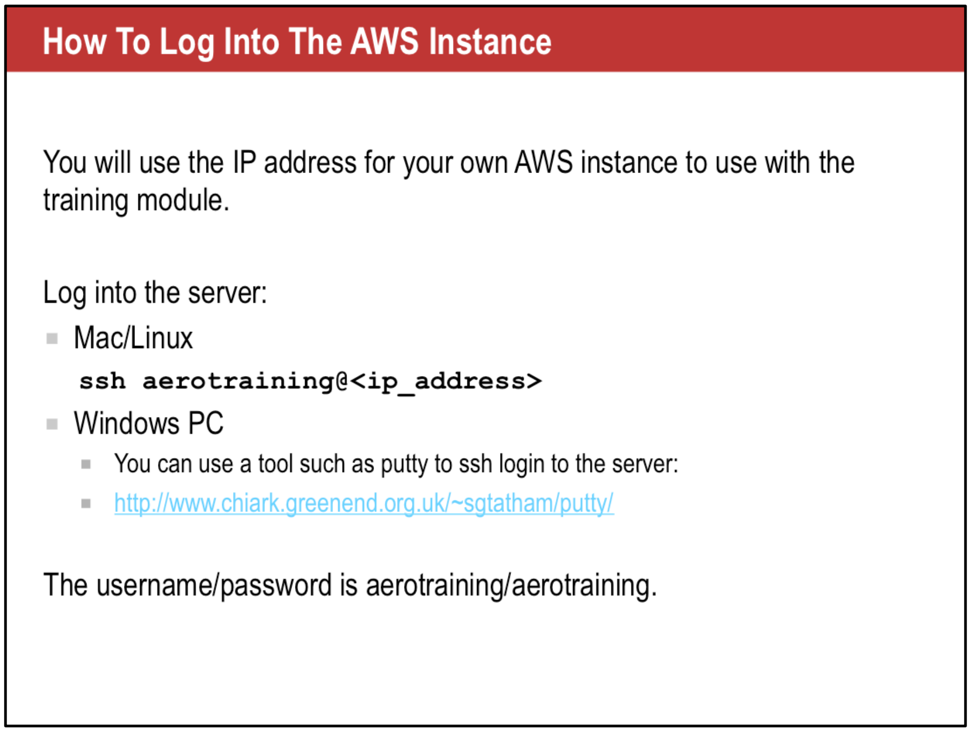
For this course, Aerospike has prepared an Amazon EC2 instance for your use.

AMI: “**`aerotraining-developer-vX.XX`**”

This instance has pre-loaded the following:

- Aerospike Java client
- Aerospike Python client
- Aerospike C client
- Aerospike data loader
- Oracle JDK 1.7
- maven
- lua
- git

You may not need all of these, but they are there for your convenience.



How To Log Into The AWS Instance

You will use the IP address for your own AWS instance to use with the training module.

Log into the server:

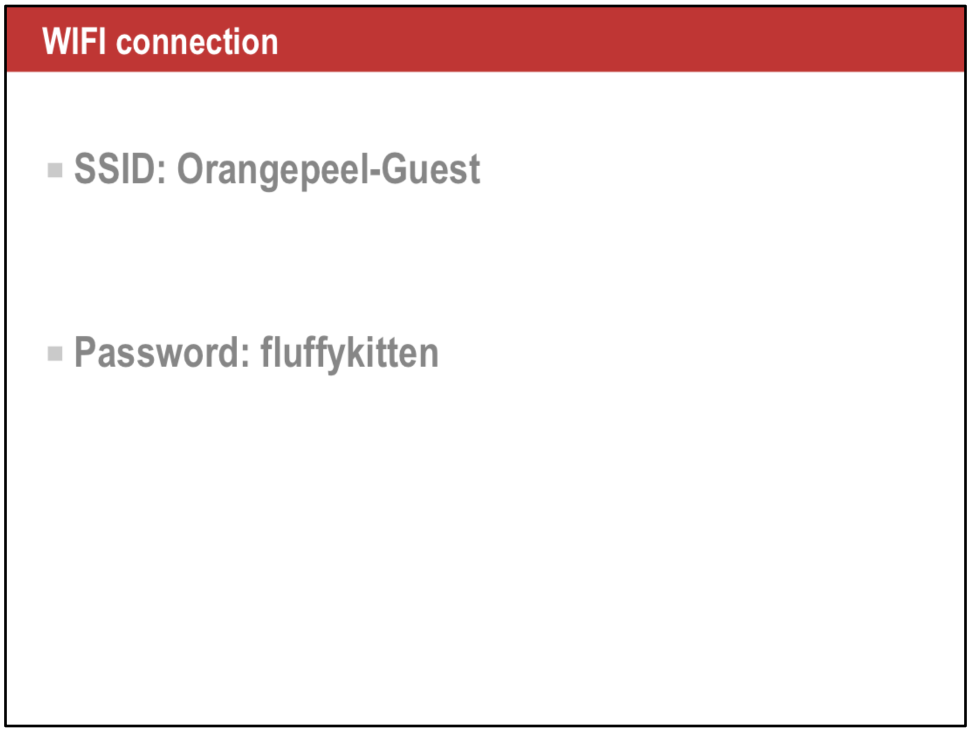
- Mac/Linux

```
ssh aerotraining@<ip_address>
```

- Windows PC
  - You can use a tool such as putty to ssh login to the server:
  - http://www.chiark.greenend.org.uk/~sgtatham/putty/

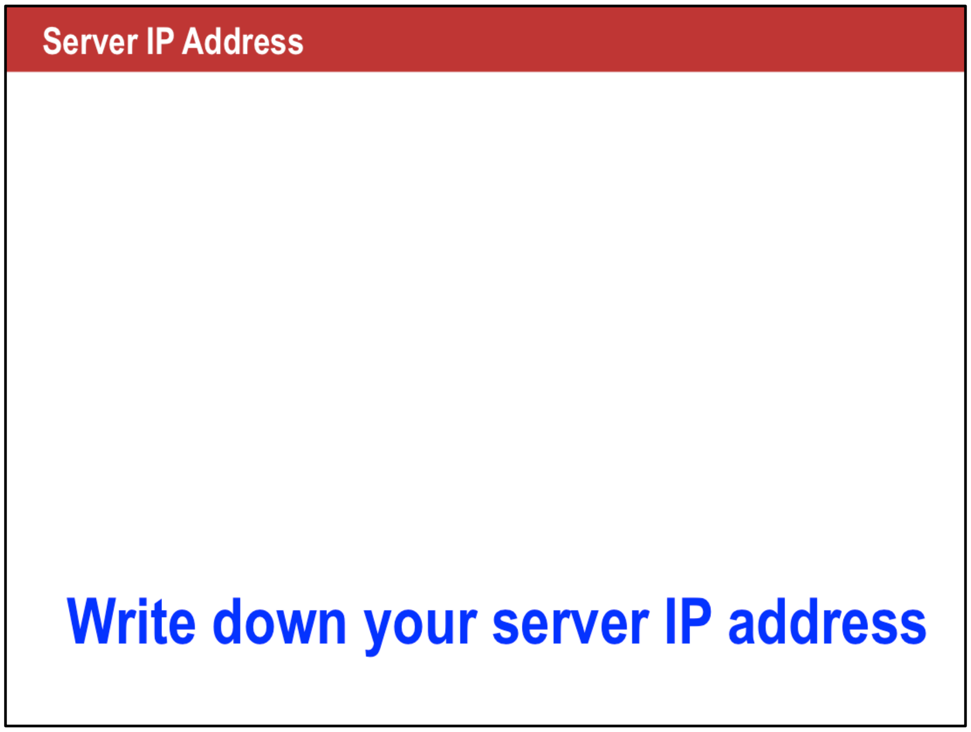
The username/password is aerotraining/aerotraining.

You each should have your own IP address to log into your own instance. You can use whichever tool you are comfortable with to log in. You will not need to have a good knowledge of Linux for this class.



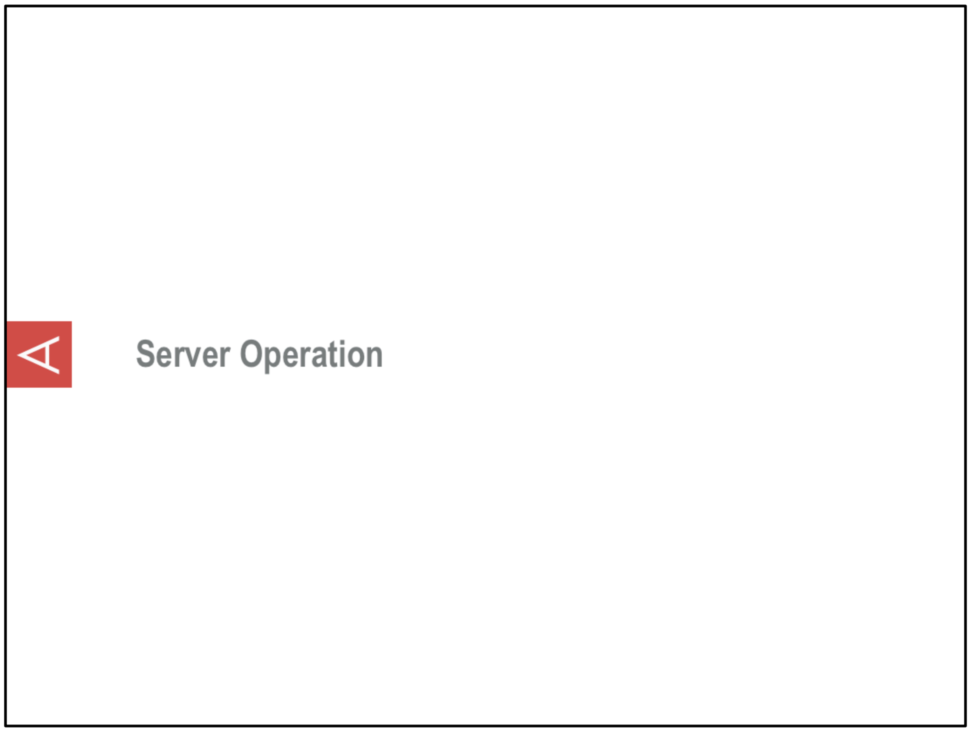
WIFI connection

- SSID: Orangepeel-Guest
- Password: fluffykitten



Server IP Address

Write down your server IP address



Server Operation

Starting And Stopping Aerospike Server

Controlling the server requires you to be root or have sudo privileges, which the aerotraining user has.

Start server

```
   sudo service aerospike start
```

Check on server status

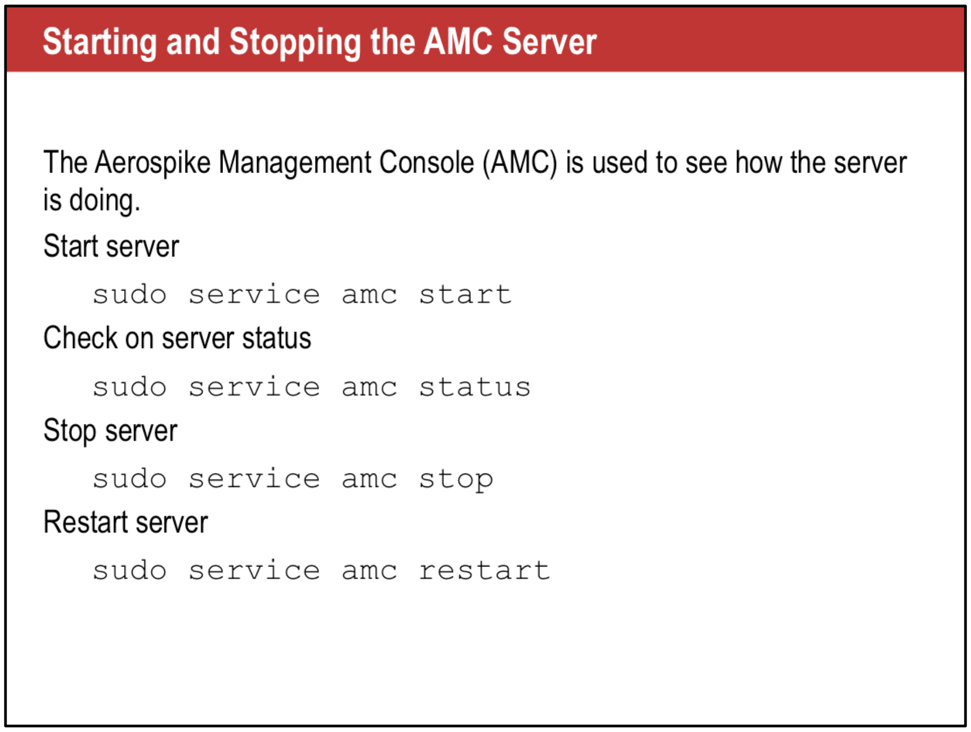
```
   sudo service aerospike status
```

Stop server

```
   sudo service aerospike stop
```

Restart server

```
   sudo service aerospike restart
```



Starting and Stopping the AMC Server

The Aerospike Management Console (AMC) is used to see how the server is doing.

Start server

```
sudo service amc start
```

Check on server status

```
sudo service amc status
```

Stop server

```
sudo service amc stop
```

Restart server

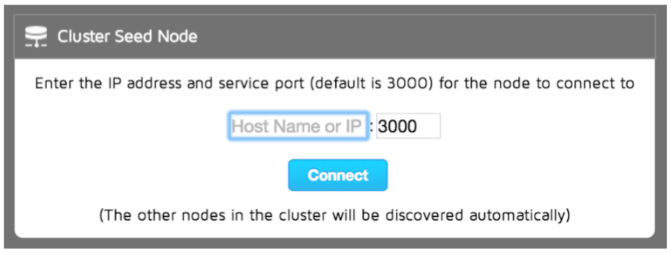
```
sudo service amc restart
```

Aerospike Management Console

Connect to the AMC using

- http://<your ip address>:8081

You will be prompted for a node address in the cluster, enter your IP address:



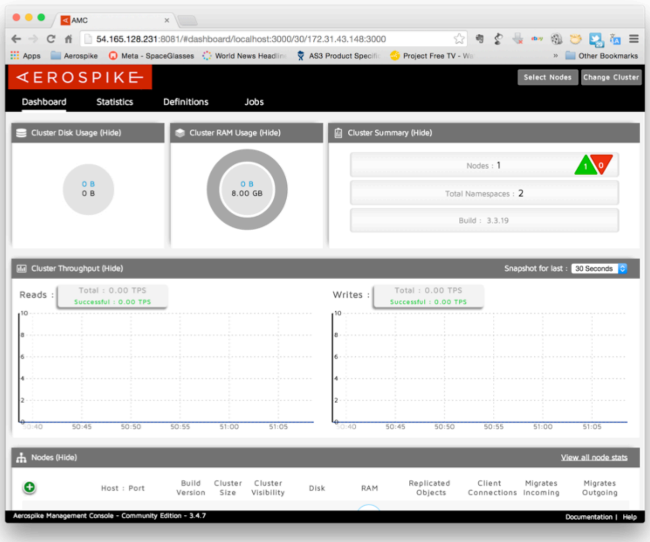
Cluster Seed Node

Enter the IP address and service port (default is 3000) for the node to connect to

Host Name or IP : 3000

(The other nodes in the cluster will be discovered automatically)

AMC



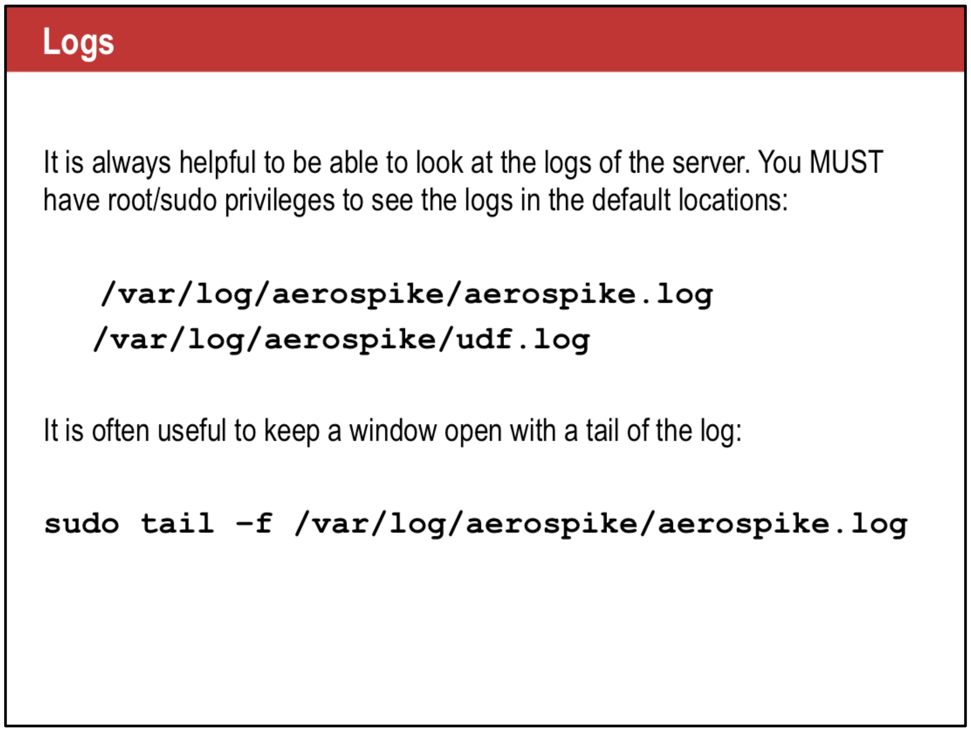
Nodes : 1

Total Namespaces : 2

Build : 3.3.19

Reads : Total : 0.00 TPS
Successful : 0.00 TPS

Writes : Total : 0.00 TPS
Successful : 0.00 TPS



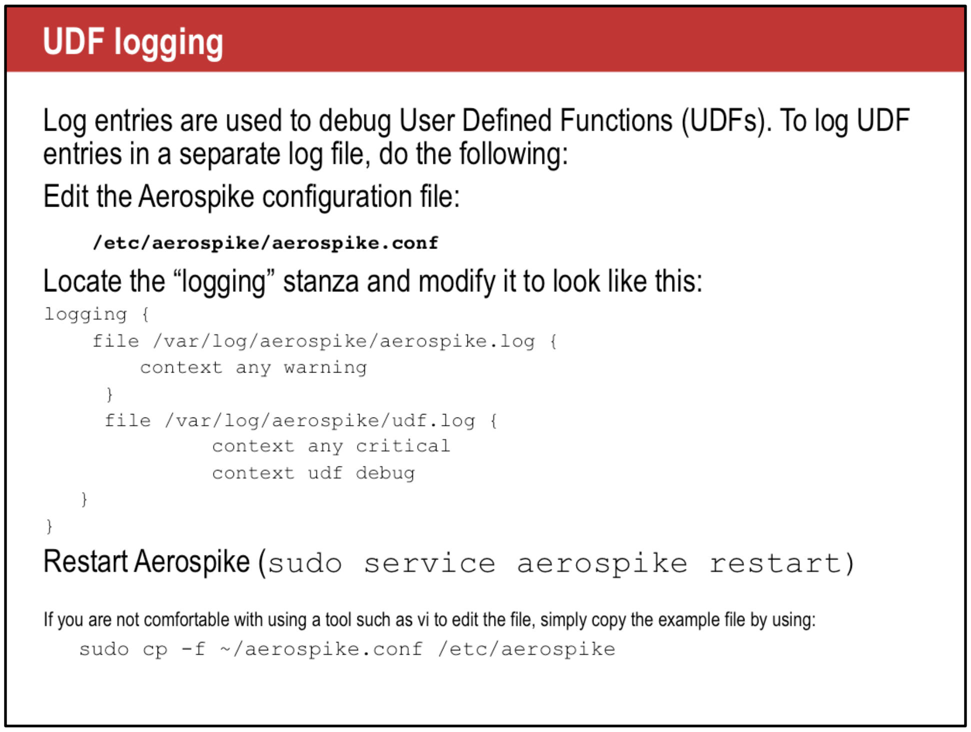
Logs

It is always helpful to be able to look at the logs of the server. You MUST have root/sudo privileges to see the logs in the default locations:

```
 /var/log/aerospike/aerospike.log
/var/log/aerospike/udf.log
```

It is often useful to keep a window open with a tail of the log:

```
sudo tail -f /var/log/aerospike/aerospike.log
```



UDF logging

Log entries are used to debug User Defined Functions (UDFs). To log UDF entries in a separate log file, do the following:

Edit the Aerospike configuration file:

```
    /etc/aerospike/aerospike.conf
```

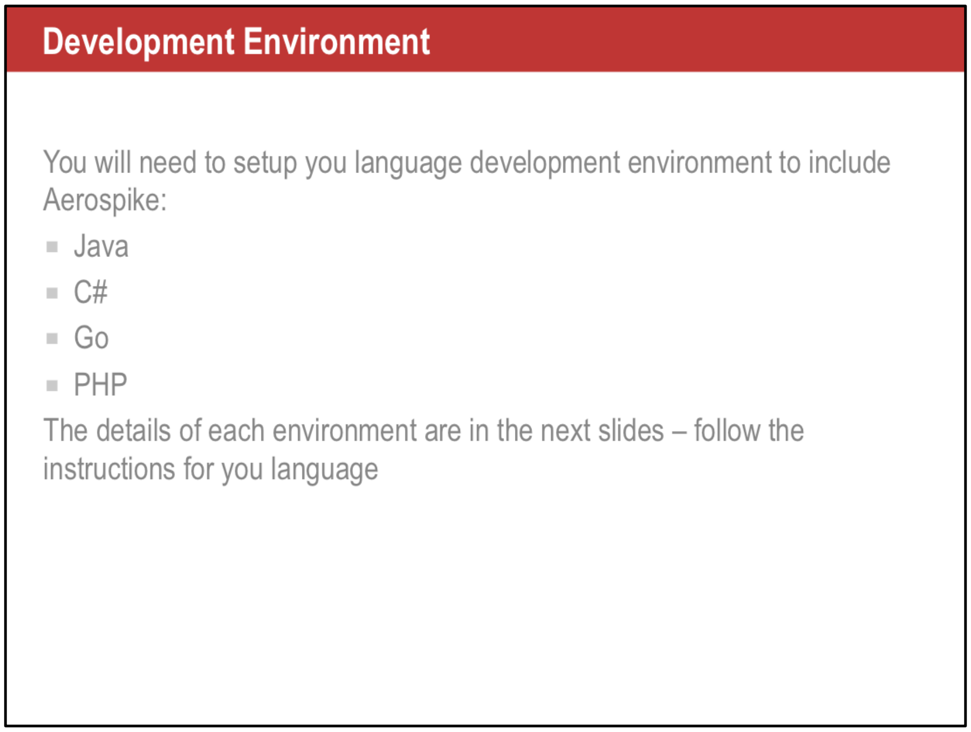
Locate the “logging” stanza and modify it to look like this:

```
logging {
    file /var/log/aerospike/aerospike.log {
        context any warning
     }
     file /var/log/aerospike/udf.log {
              context any critical
              context udf debug
   }
}
```

Restart Aerospike (`sudo service aerospike restart)`

If you are not comfortable with using a tool such as vi to edit the file, simply copy the example file by using:

```
   sudo cp -f ~/aerospike.conf /etc/aerospike
```



Development Environment

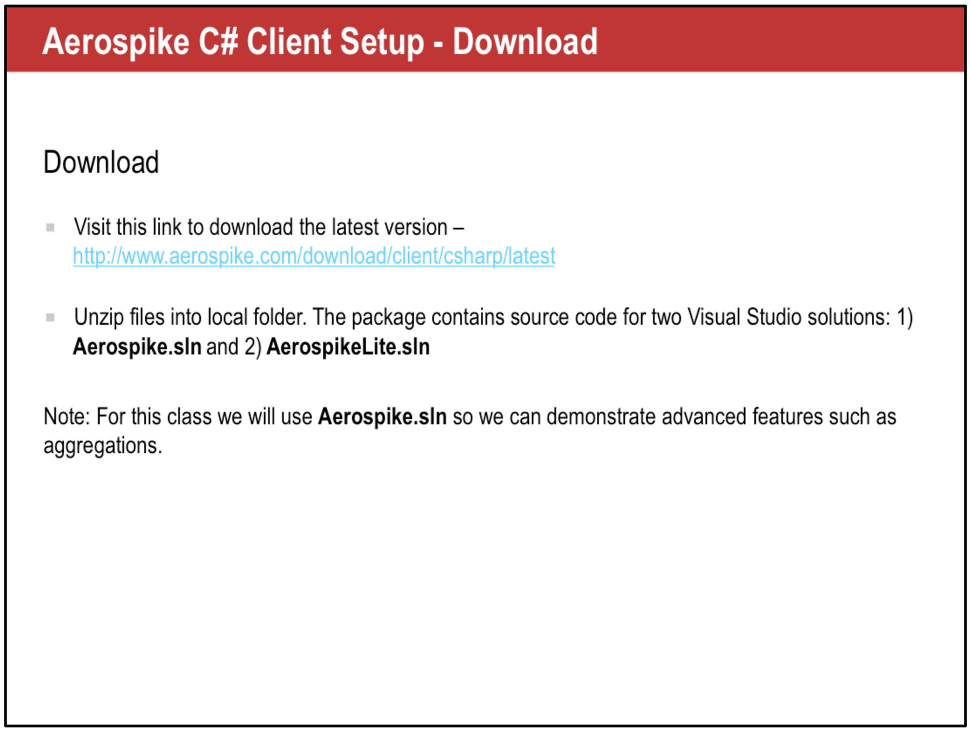
You will need to setup you language development environment to include Aerospike:

- Java
- C#
- Go
- PHP

The details of each environment are in the next slides – follow the instructions for you language

Aerospike C# Client Setup

C# Client is one of the several client libraries that can be used to access Aerospike's database from an application. For a full list of client libraries, please visit http://www.aerospike.com/develop/



Aerospike C# Client Setup - Download

Download

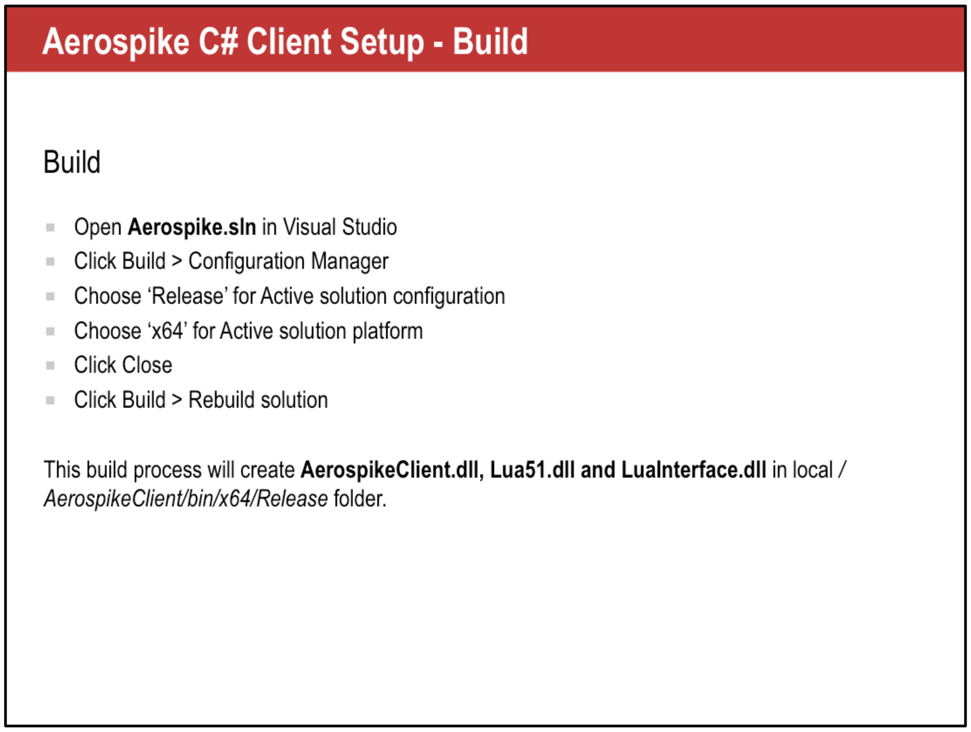
- Visit this link to download the latest version – http://www.aerospike.com/download/client/csharp/latest
- Unzip files into local folder. The package contains source code for two Visual Studio solutions: 1) **Aerospike.sln** and 2) **AerospikeLite.sln**

Note: For this class we will use **Aerospike.sln** so we can demonstrate advanced features such as aggregations.

Aerospike.sln vs AerospikeLite.sln:

Aerospike.sln includes advanced features that depend on Lua interpreter. For example, secondary index query with user-defined aggregation. Supported compile targets are x64 (64-bit) and x86 (32-bit)

AerospikeLite.sln does *not* include advanced features such as aggregations. Effectively eliminating the dependency on Lua interpreter. Supported compile targets are AnyCPU, x64 (64-bit) and x86 (32-bit)



Aerospike C# Client Setup - Build

Build

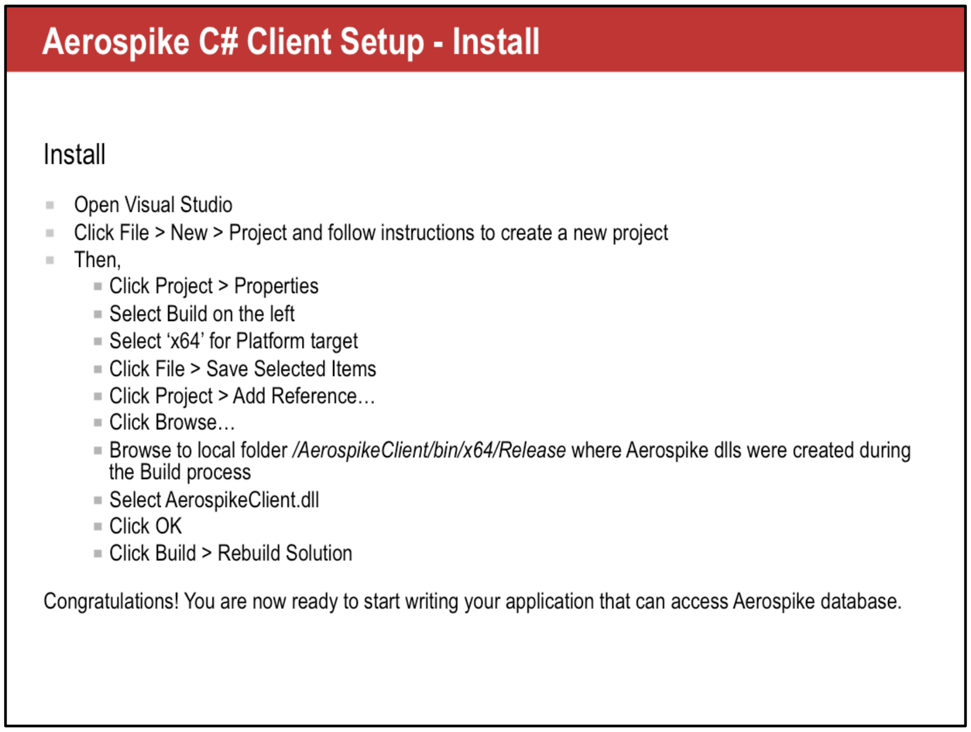
- Open **Aerospike.sln** in Visual Studio
- Click Build > Configuration Manager
- Choose 'Release' for Active solution configuration
- Choose 'x64' for Active solution platform
- Click Close
- Click Build > Rebuild solution

This build process will create **AerospikeClient.dll, Lua51.dll and LuaInterface.dll** in local /*AerospikeClient/bin/x64/Release* folder.

Aerospike.sln vs AerospikeLite.sln:

Aerospike.sln includes advanced features that depend on Lua interpreter. For example, secondary index query with user-defined aggregation. Supported compile targets are x64 (64-bit) and x86 (32-bit)

AerospikeLite.sln does *not* include advanced features such as aggregations. Effectively eliminating the dependency on Lua interpreter. Supported compile targets are AnyCPU, x64 (64-bit) and x86 (32-bit)



Aerospike C# Client Setup - Install

Install

- Open Visual Studio
- Click File > New > Project and follow instructions to create a new project
- Then,
  - Click Project > Properties
  - Select Build on the left
  - Select 'x64' for Platform target
  - Click File > Save Selected Items
  - Click Project > Add Reference…
  - Click Browse…
  - Browse to local folder */AerospikeClient/bin/x64/Release* where Aerospike dlls were created during the Build process
  - Select AerospikeClient.dll
  - Click OK
  - Click Build > Rebuild Solution

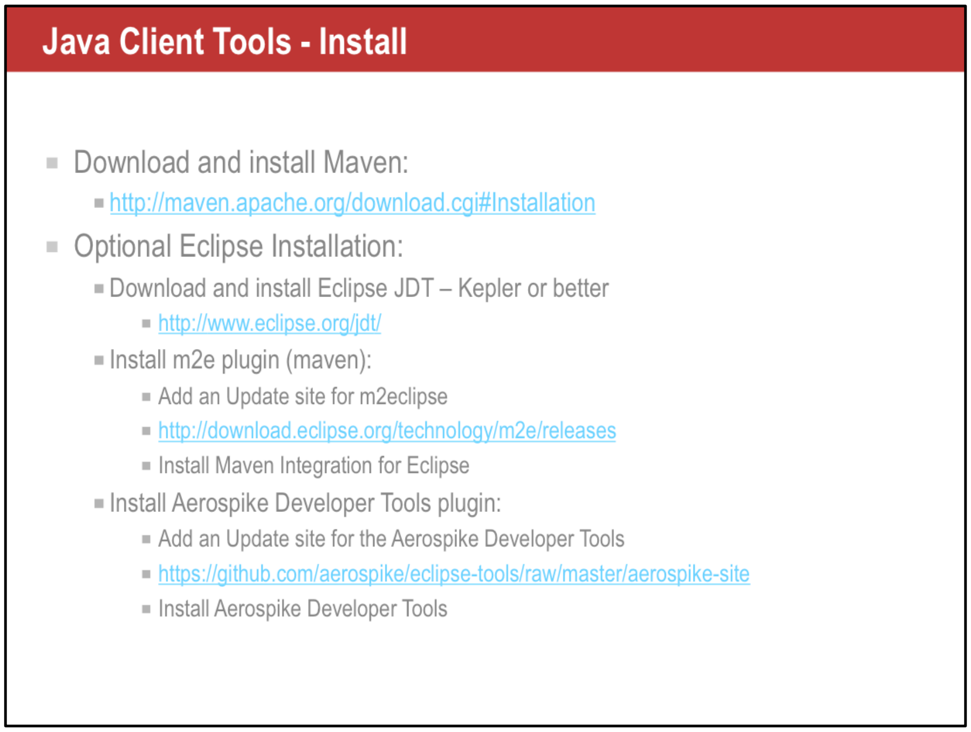
Congratulations! You are now ready to start writing your application that can access Aerospike database.

Two ways to install:

1) By adding reference to AerospikeClient DLL in an application. This is the norm.

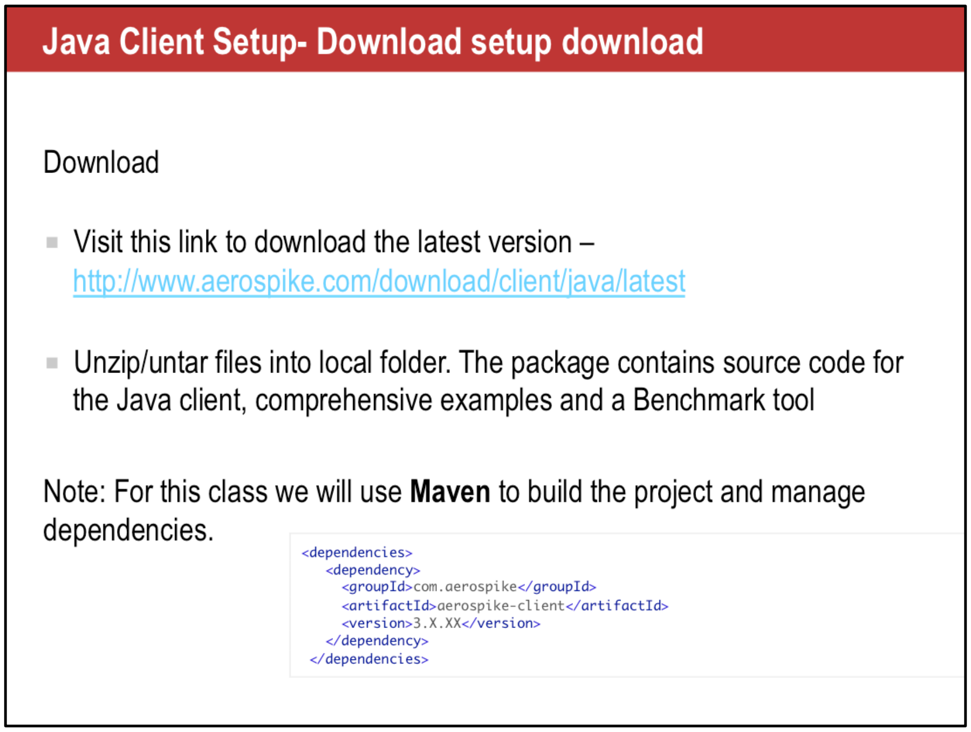
2) By referencing the AerospikeClient project in an application. Unless absolutely necessary for very specific reason(s), this is not how applications normally reference external libraries.

Java Client Setup



Java Client Tools - Install

- Download and install Maven:
  - http://maven.apache.org/download.cgi#Installation
- Optional Eclipse Installation:
  - Download and install Eclipse JDT – Kepler or better
    - http://www.eclipse.org/jdt/
  - Install m2e plugin (maven):
    - Add an Update site for m2eclipse
    - http://download.eclipse.org/technology/m2e/releases
    - Install Maven Integration for Eclipse
  - Install Aerospike Developer Tools plugin:
    - Add an Update site for the Aerospike Developer Tools
    - https://github.com/aerospike/eclipse-tools/raw/master/aerospike-site
    - Install Aerospike Developer Tools



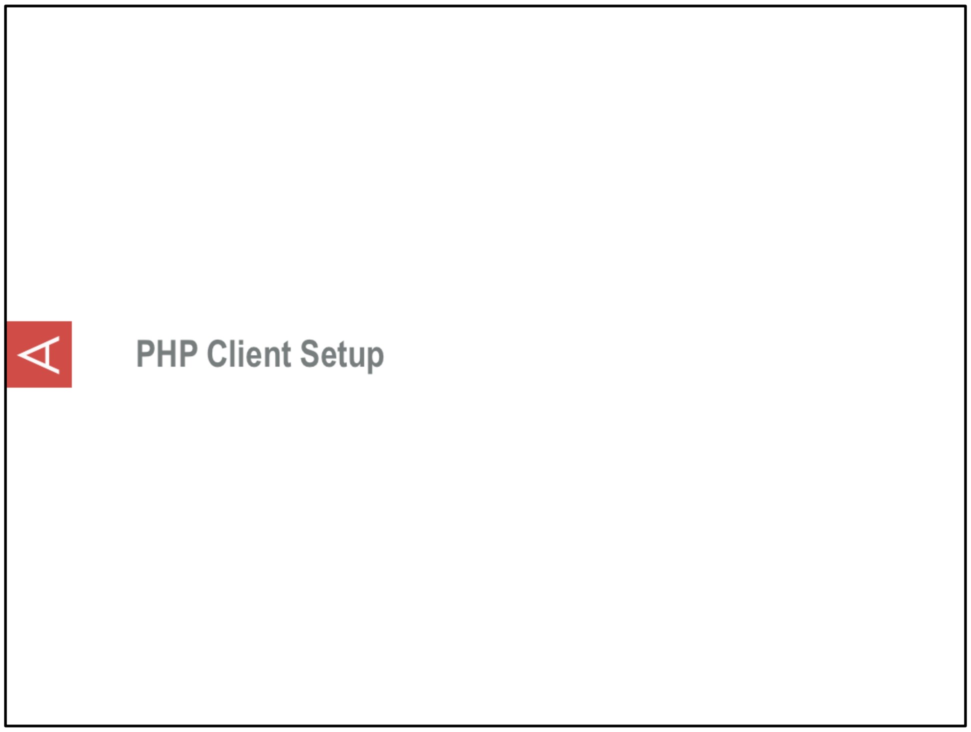
Java Client Setup- Download setup download

Download

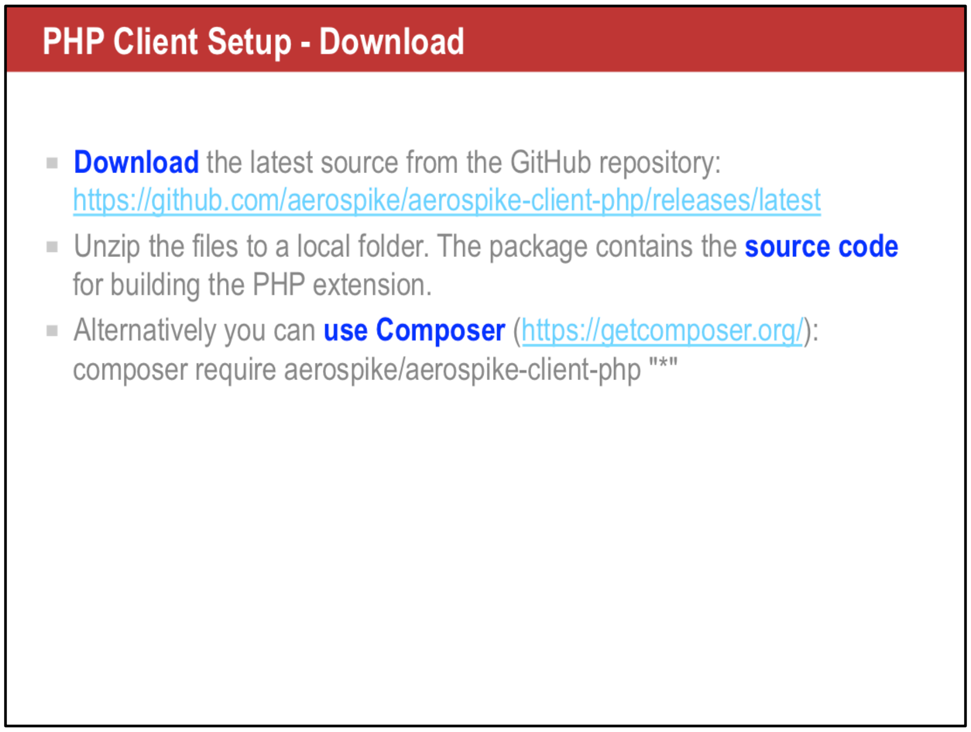
- Visit this link to download the latest version – http://www.aerospike.com/download/client/java/latest
- Unzip/untar files into local folder. The package contains source code for the Java client, comprehensive examples and a Benchmark tool

Note: For this class we will use **Maven** to build the project and manage dependencies.

```
<dependencies>
   <dependency>
     <groupId>com.aerospike</groupId>
     <artifactId>aerospike-client</artifactId>
     <version>3.X.XX</version>
   </dependency>
 </dependencies>
```

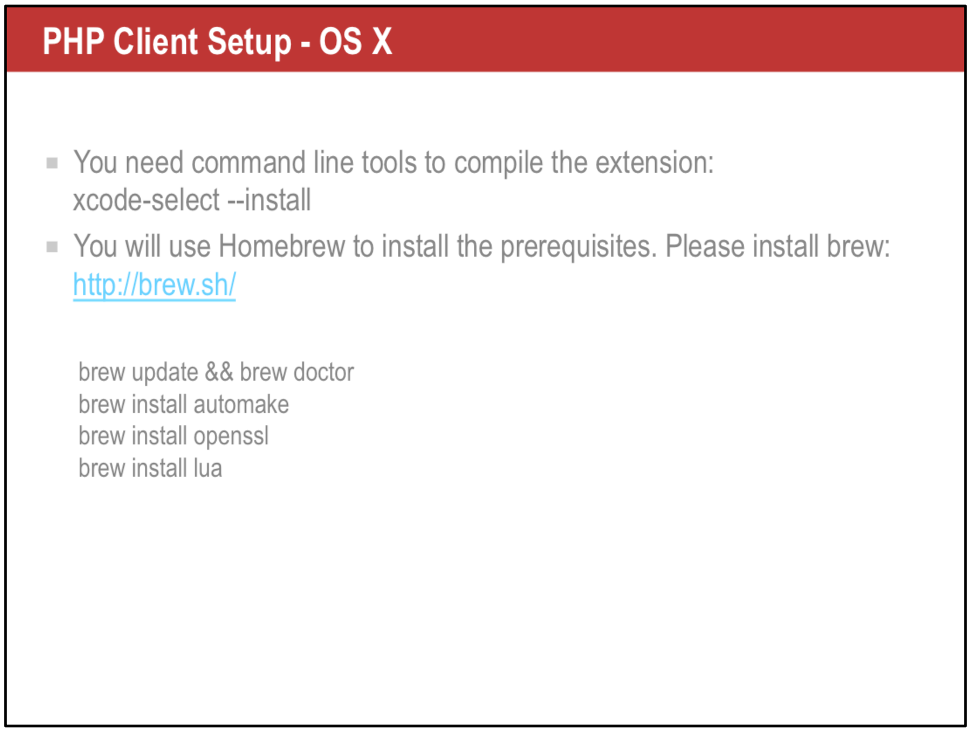


PHP Client Setup



PHP Client Setup - Download

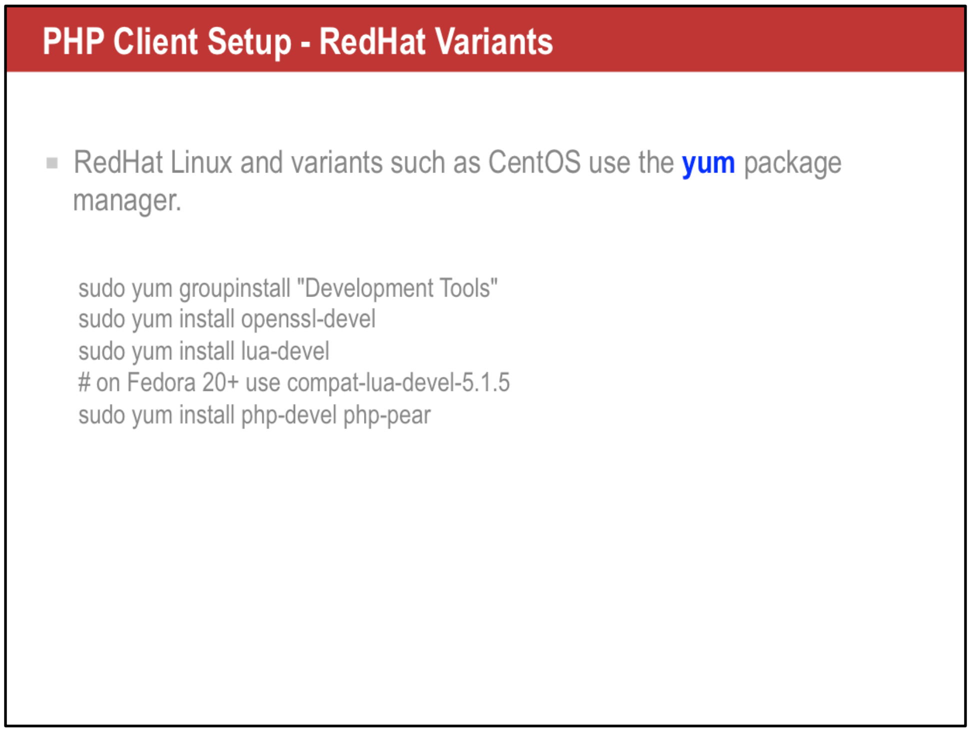
- **Download** the latest source from the GitHub repository: https://github.com/aerospike/aerospike-client-php/releases/latest
- Unzip the files to a local folder. The package contains the **source code** for building the PHP extension.
- Alternatively you can **use Composer** (https://getcomposer.org/): composer require aerospike/aerospike-client-php "*"



PHP Client Setup - OS X

- You need command line tools to compile the extension:
  xcode-select --install
- You will use Homebrew to install the prerequisites. Please install brew:
  http://brew.sh/

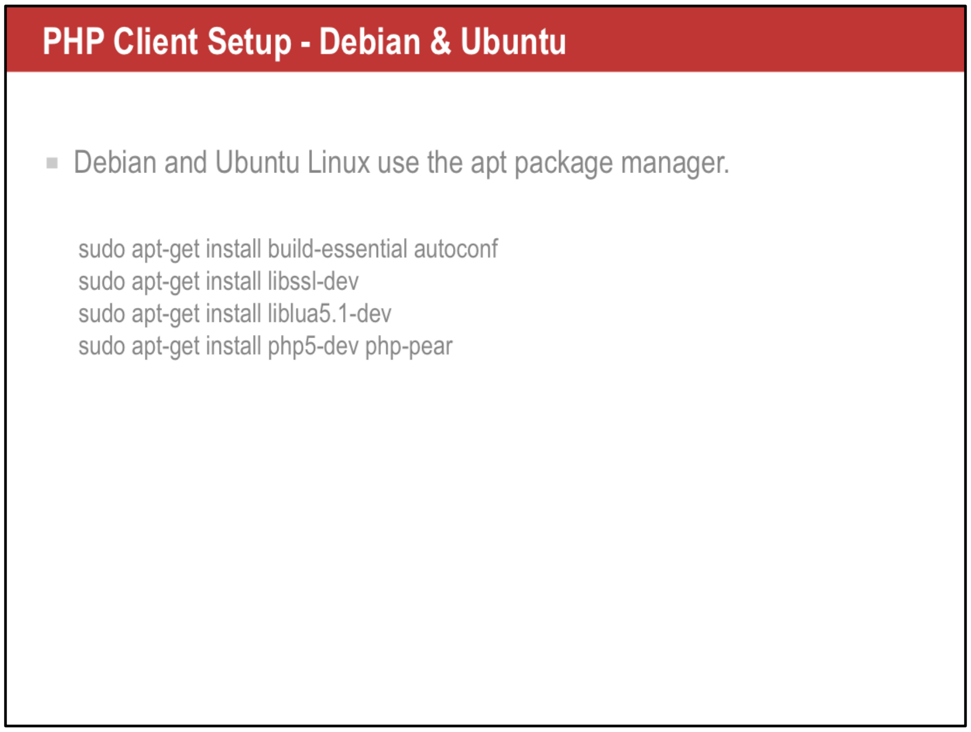
```
brew update && brew doctor
brew install automake
brew install openssl
brew install lua
```



PHP Client Setup - RedHat Variants

- RedHat Linux and variants such as CentOS use the **yum** package manager.

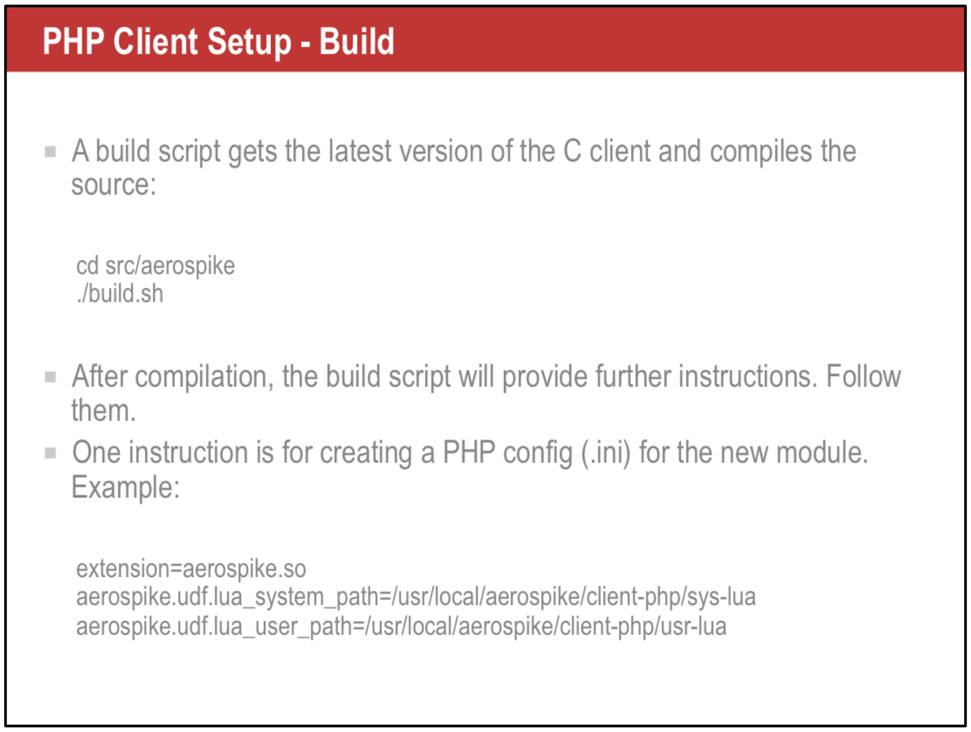
```
sudo yum groupinstall "Development Tools"
sudo yum install openssl-devel
sudo yum install lua-devel
# on Fedora 20+ use compat-lua-devel-5.1.5
sudo yum install php-devel php-pear
```



PHP Client Setup - Debian & Ubuntu

- Debian and Ubuntu Linux use the apt package manager.

```
sudo apt-get install build-essential autoconf
sudo apt-get install libssl-dev
sudo apt-get install liblua5.1-dev
sudo apt-get install php5-dev php-pear
```



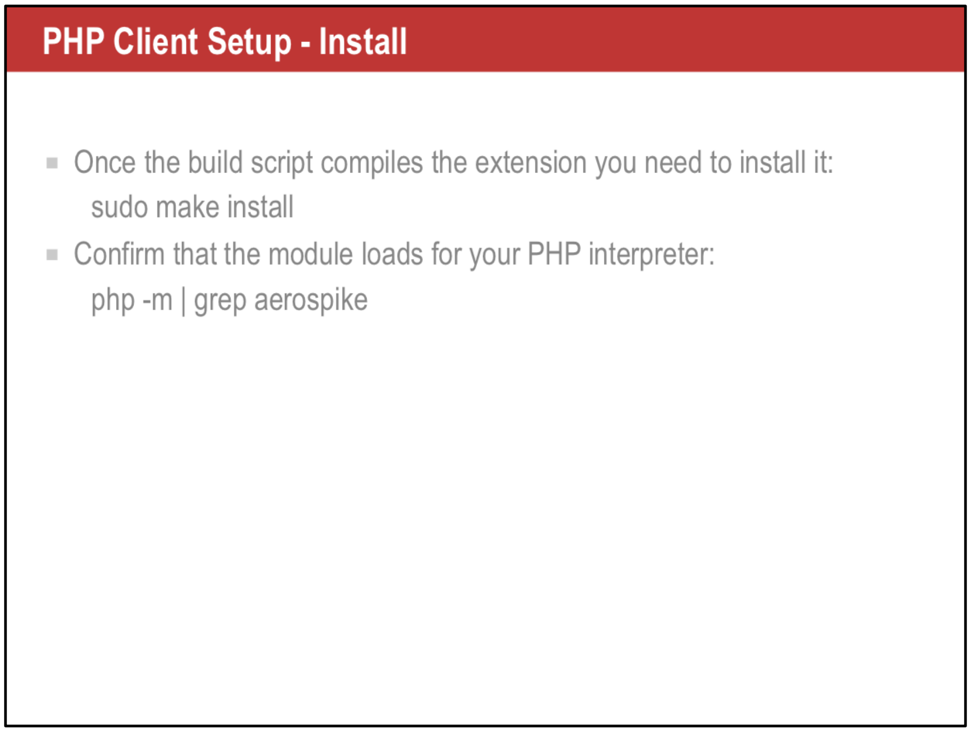
PHP Client Setup - Build

- A build script gets the latest version of the C client and compiles the source:

```
cd src/aerospike
./build.sh
```

- After compilation, the build script will provide further instructions. Follow them.
- One instruction is for creating a PHP config (.ini) for the new module. Example:

```
extension=aerospike.so
aerospike.udf.lua_system_path=/usr/local/aerospike/client-php/sys-lua
aerospike.udf.lua_user_path=/usr/local/aerospike/client-php/usr-lua
```



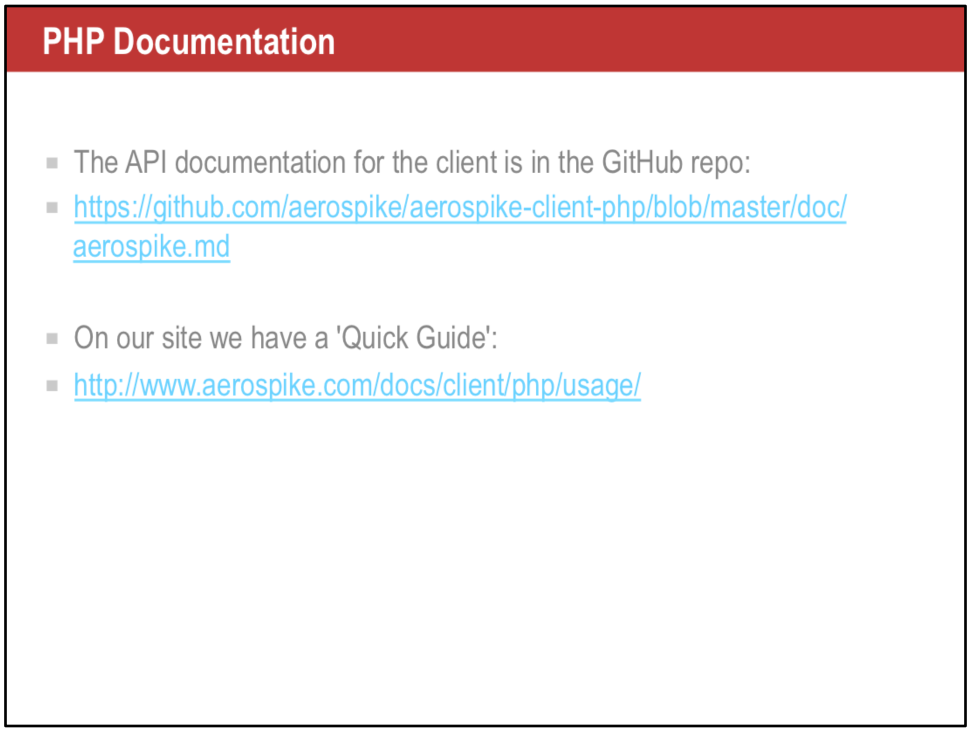
PHP Client Setup - Install

- Once the build script compiles the extension you need to install it:

```
sudo make install
```

- Confirm that the module loads for your PHP interpreter:

```
php -m | grep aerospike
```

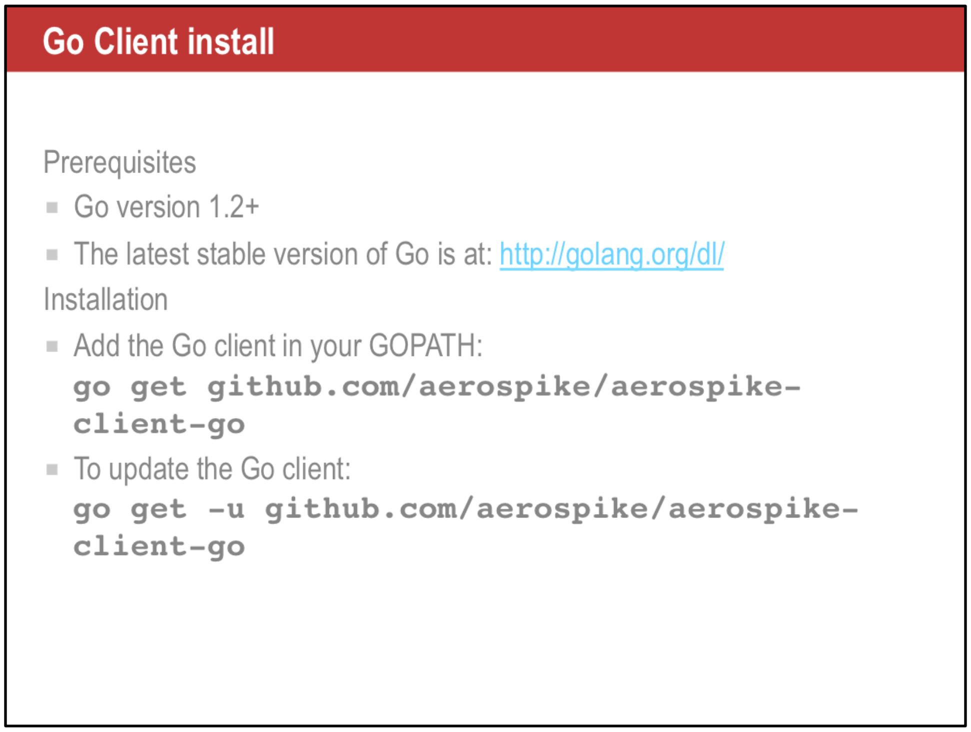


PHP Documentation

- The API documentation for the client is in the GitHub repo:
- https://github.com/aerospike/aerospike-client-php/blob/master/doc/aerospike.md

- On our site we have a 'Quick Guide':
- http://www.aerospike.com/docs/client/php/usage/

Go Client Setup



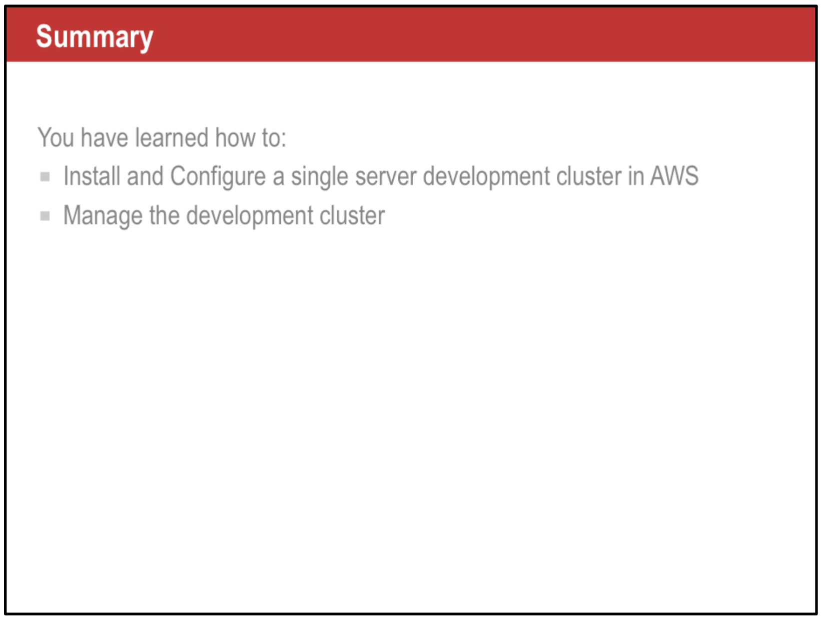
Go Client install

Prerequisites

- Go version 1.2+
- The latest stable version of Go is at: http://golang.org/dl/

Installation

- Add the Go client in your GOPATH:
  ```
  go get github.com/aerospike/aerospike-
  client-go
  ```
- To update the Go client:
  ```
  go get -u github.com/aerospike/aerospike-
  client-go
  ```



Summary

You have learned how to:

- Install and Configure a single server development cluster in AWS
- Manage the development cluster

AEROSPIKE