

# Cours IUT 2012

Introduction to Java

Getting Started in an Open Source Project

# How to Improve Your (Java) Development Skills?

Question: Read Or Write for Improving ?

Answer : Read !!

Monkey See – Monkey DO

$$1000 \times 1 \quad \neq \quad 1 \times 1000$$

- Writing 100 times an Hello world program  
!=  
Writing 1 time a 100 kloc programs
- There exists NO interesting program of 1 kloc
- Progress stops doing only easy things
- To Continue Progressing :  
read more and more complex things...

# But I was Told to write at school...

- At School => write, alone, exams, competition
- At Work => read, team, business, manage/execute
- In OpenSource Communities => read, communities, enjoy, contribute, collaborate

# How To Progress ?

- Ask / Answer / Help each-others
- Read Books
- Read Tutorial on the Web
- Watch Video Tutorial on the Web
- Read Source Code

# Sample Study : Logback

- What is LogBack ?

The successor of Log4J


- But What was Log4J ?  
As its name says: “Log For Java”
- The implementation of “SLF4J”  
“Simple Logging Facade For Java”

# Reasons of starting with LogBack

- Anyway... start with something!
- Logback is
  - open source => possible to read source
  - Simple => understandable by beginners
  - extremely well done => high quality reading
  - De-facto standard => worth read ... must-read

# First Step : Google It

I will use Google before asking dumb questions. I will use Google before asking dumb questions. I will use Google before asking dumb questions. I will use Google before asking dumb questions. I will use Google before asking dumb questions. I will use Google before asking dumb questions. I will use Google before asking dumb questions. I will use Google before asking dumb questions. I will use Google before asking dumb questions. I will use Google before asking dumb questions. I will use Google before asking dumb questions. I will use Google before asking dumb questions.

A cartoon illustration of Bart Simpson's head and shoulders. He has his signature yellow skin, spiky hair, and red shirt. He is looking slightly to the left with a neutral expression. The background behind him is dark green, matching the overall theme of the slide.



# Google Logback

+Vous Recherche Images Maps Play YouTube Actualités Gmail Documents Agenda Plus -



Recherche

Environ 3 770 000 résultats (0,18 secondes)

Web

Images

Maps

Vidéos

Actualités

Shopping

Blogs

Plus

Nanterre

Changer le lieu

Le Web

Pages en  
français

Pays : France

Pages en langue

[Logback Home](#)

[logback.qos.ch/](http://logback.qos.ch/) - Traduire cette page

**Logback** is intended as a successor to the popular log4j project, picking up where  
... **Logback's** basic architecture is sufficiently generic so as to apply under ...

Vous avez consulté cette page de nombreuses fois. Date de la dernière visite :  
09/02/12

[Logback Console Plugin for ...](#)

Logback Console Plugin for  
Eclipse. During the ...

[Logback Manual](#)

The logback manual. The complete  
logback manual ...

[Documentation](#)

Logback documentation. Below is a  
list of logback-related ...

[Configuration](#)

Assuming the configuration files  
logback-test.xml or logback.xml ...

[Download](#)

Download links. Logback modules  
are available as downloads ...

[Reasons to switch to logback ...](#)

They are too many to enumerate  
exhaustively. Nevertheless ...

[Autres résultats sur qos.ch »](#)

[Logback ou Log4J ?](#)

[blog.courtine.org/2010/09/13/logback-ou-log4j/](http://blog.courtine.org/2010/09/13/logback-ou-log4j/)

13 sept. 2010 – Log4J et **Logback** peuvent tous deux utiliser SLF4J , mais ne sont  
pas égaux : **Logback** est l'implémentation native de SLF4J , alors que son ...

# Logback Home Page



Logback project
<a href="#">Introduction</a>
<a href="#">Download</a>
<a href="#">Documentation</a>
<a href="#">License</a>
<a href="#">News</a>
Support
<a href="#">Mailing Lists</a>
<a href="#">Bug Report</a>
<a href="#">Source Repository</a>
<a href="#">Call for volunteers</a>
<a href="#">Support offerings</a>
Sister projects
<a href="#">Logback-audit</a>
Online Tools
<a href="#">log4j.properties Translator</a>
<a href="#">logback.XML to Groovy</a>

## Logback Project

Logback is intended as a successor to the popular log4j project, [picking up where log4j leaves off](#).

Logback's basic architecture is sufficiently generic so as to apply under different circumstances. At present time, logback is divided into three modules, logback-core, logback-classic and logback-access.

The logback-core module lays the groundwork for the other two modules. The logback-classic module can be assimilated to a significantly improved version of log4j. Moreover, logback-classic natively implements the [SLF4J API](#) so that you can readily switch back and forth between logback and other logging frameworks such as log4j or java.util.logging (JUL).

The logback-access module integrates with Servlet containers, such as Tomcat and Jetty, to provide HTTP-access log functionality. Note that you could easily build your own module on top of logback-core.

## Sister projects

# First Link : Documentation



## Chapter 1: Introduction

*The morale effects are startling. Enthusiasm jumps even a simple one. Efforts redouble when the first system appears on the screen, even if it is only a stage in the process, a working system. I find that*

—FREDERICK P. BR

**Authors: Ceki Gülcü, Sébastien Pennec**  
**Copyright © 2000-2012, QOS.ch**

This document is licensed under a [Creative Commons Attribution-NonCommercial-ShareAlike 2.5 License](#)

### Chapter Index

- [Ch1: Introduction to logback](#)
- [Ch2: Architecture](#)
- [Ch3: Configuration](#)
- [Ch4: Appenders](#)
- [Ch5: Encoders](#)
- [Ch6: Layouts](#)
- [Ch7: Filters](#)
- [Ch8: Mapped Diagnostic Contexts](#)
- [Ch9: Logging Separation](#)
- [Ch10: JMX Configurator](#)
- [Ch11: Joran](#)
- [Ch12: Groovy Configuration](#)
- [Ch13: Migration from log4j](#)



# Scroll 10 lines ... LogBack Sample

*Example 1.1: Basic template for logging ([logback-examples/src/main/java/chapters/introduction/HelloWorld1.java](#))*

```
package chapters.introduction;

import org.slf4j.Logger;
import org.slf4j.LoggerFactory;

public class HelloWorld1 {

    public static void main(String[] args) {

        Logger logger = LoggerFactory.getLogger("chapters.introduction.H
        logger.debug("Hello world.");

    }
}
```

# The Proof Is In The Pooding

- Step 1: Create project
  - Step 2: Add Dependency
  - Step 3: Import in Eclipse
  - Step 4: Write Code
  - Step 5: Run
- 
- Who bets to do it in less than 3 minutes ?
  - Who bets I can not do it in less than 3 minutes ?  
(at home, first try in 2mn 40s )

# LogBack Demo

```
<project xmlns="http://maven.apache.org/POM/4.0.0" xmlns:xsi="http://maven.apache.org/POM/4.0.0" xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 http://maven.apache.org/xsd/maven-4.0.0.xsd" >  
  <modelVersion>4.0.0</modelVersion>
```

```
  <groupId>fr.an.tests.testlogback</groupId>  
  <artifactId>testlogback</artifactId>  
  <version>1.0-SNAPSHOT</version>  
  <packaging>jar</packaging>
```

```
  <name>testlogback</name>  
  <url>http://maven.apache.org</url>
```

```
  <properties>  
    <project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>  
  </properties>
```

```
  <dependencies>  
    <dependency>  
      <groupId>ch.qos.logback</groupId>  
      <artifactId>logback-classic</artifactId>  
      <version>1.0.7-SNAPSHOT</version>  
    </dependency>
```

## testlogback

src/main/java

src/test/java

JRE System Library [J2SE-1.5]

## Maven Dependencies

logback-classic-1.0.7-SNAPSHOT.jar

logback-core-1.0.7-SNAPSHOT.jar - /

slf4j-api-1.6.6.jar - /home/arnaud/.m2/repo

junit-4.10.jar - /home/arnaud/.m2/repo

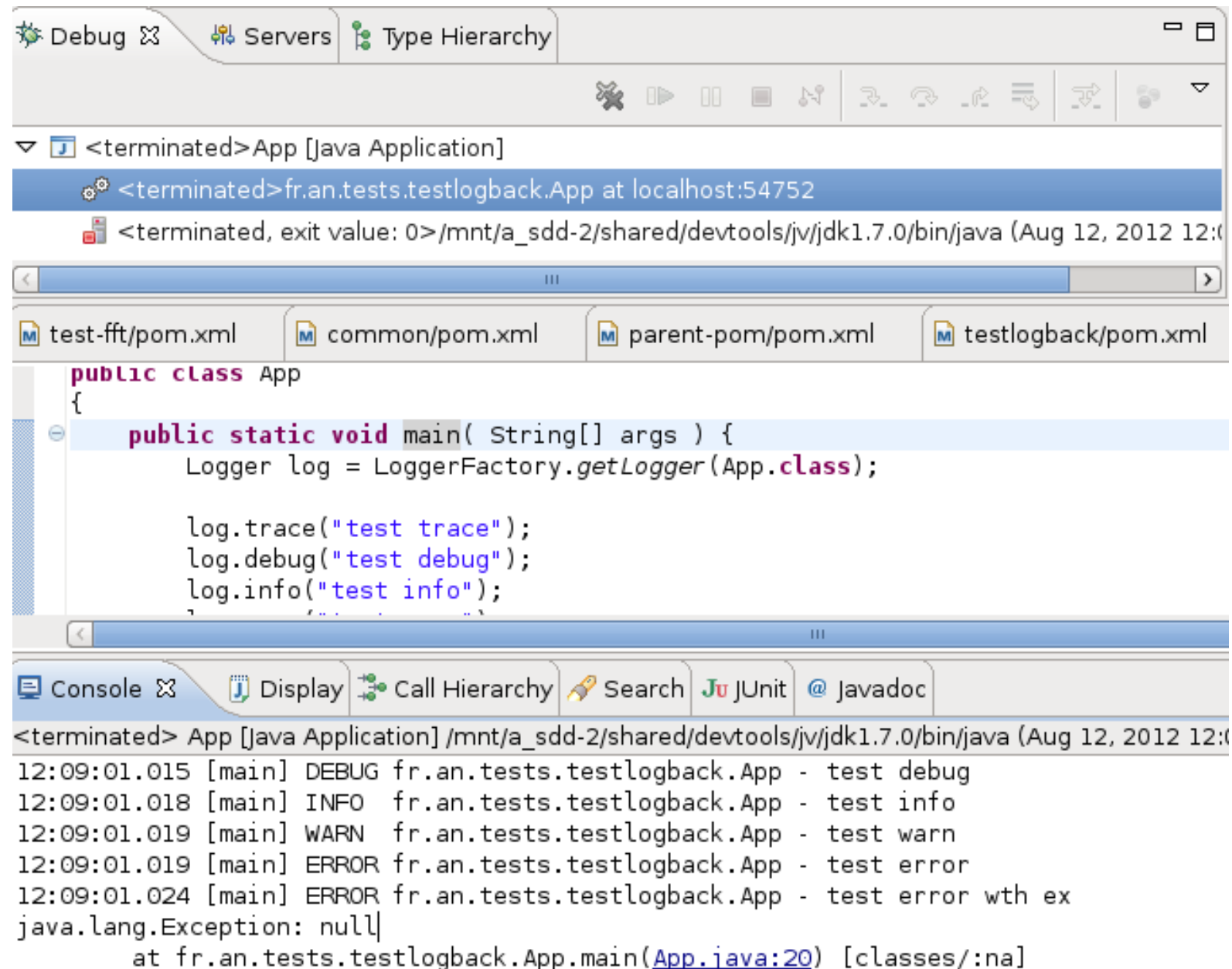
hamcrest-core-1.1.jar - /home/arnaud/.m2/repo

src

target

pom.xml

# Eclipse, Sample Code & Run



```
public class App
{
    public static void main( String[] args ) {
        Logger log = LoggerFactory.getLogger(App.class);

        log.trace("test trace");
        log.debug("test debug");
        log.info("test info");
        log.warn("test warn");
        log.error("test error");
        log.error("test error with exception", new java.lang.Exception());
    }
}
```

```
<terminated> App [Java Application] /mnt/a_sdd-2/shared/devtools/jv/jdk1.7.0/bin/java (Aug 12, 2012 12:09:01.015)
12:09:01.015 [main] DEBUG fr.an.tests.testlogback.App - test debug
12:09:01.018 [main] INFO fr.an.tests.testlogback.App - test info
12:09:01.019 [main] WARN fr.an.tests.testlogback.App - test warn
12:09:01.019 [main] ERROR fr.an.tests.testlogback.App - test error
12:09:01.024 [main] ERROR fr.an.tests.testlogback.App - test error with exception
java.lang.Exception: null
    at fr.an.tests.testlogback.App.main(App.java:20) [classes/:na]
```

# Logback is an implementation

## ... see Doc also in API part : SLF4J



SLF4J Project

[Introduction](#)

[Download](#)

[Documentation](#)

[License](#)

[News](#)

Support

[Mailing Lists](#)

[Bug Reporting](#)

[Source Repository](#)

[Support offerings](#)

[Training](#)

Native implementations

[Logback](#)

Wrapped implementations

[AVSL](#)

[JDK14](#)

[Log4j](#)

[Simple](#)

Sub-projects

## SLF4J user manual

The Simple Logging Facade for Java or (SLF4J) serves as a simple facade or abstraction for v and logback. SLF4J allows the end-user to plug in the desired logging framework at *deployme* implies the addition of only a single mandatory dependency, namely *slf4j-api-1.6.6.jar*.

**SINCE 1.6.0** If no binding is found on the class path, then SLF4J will default to a no-operation i

### Hello World

As customary in programming tradition, here is an example illustrating the simplest way to ou with the name "HelloWorld". This logger is in turn used to log the message "Hello World".

```
import org.slf4j.Logger;
import org.slf4j.LoggerFactory;

public class HelloWorld {
    public static void main(String[] args) {
        Logger logger = LoggerFactory.getLogger(HelloWorld.class);
        logger.info("Hello World");
    }
}
```

To run this example, you first need to [download the slf4j distribution](#), and then to unpack it. O



# Tutorial, Blogs and Presentations

- Thousands of Presentations (including this !)

Source code related documentation:

- [Javadoc](#)
- [Source code](#)
- [Test classes source code](#)

## Articles and Presentations

- [Enterprise Spring Best Practices - Part 1 - Project Config](#) by G
- [Application errors notification with Logback](#) by Maciej Walkowi
- [Sifting Logs in Jetty with Logback](#) by Joakim Erdfelt
- [Logging tests to separate files](#) by Nalin Makar
- [Logback project](#), by Ceki Gülcü and Sébastien Penneç.
- [Devoxx-2009 video presentation](#), by C. Gülcü
- [Logback: Evolving Java Logging](#) by Geoffrey Wiseman
- [Logging in OSGI Enterprise Applications](#), by Ekkehard Gentz
- [Understanding the OSGI logging service](#) by Nogunner
- [Logback in RCP application](#) by David Virdefors

## In french

- [SLF4J & LOGBack : simplifiez-vous les logs](#) by Ludovic Meurillon
- [Java en Production - Les fichiers de logs](#) by Cyrille Le Clerc
- [Logback ou Log4J?](#) by Benoit Courtine

<http://logback.qos.ch/logback.ppt>

## Questions?

- read the docs at <http://logback.qos.ch/>
- study the code at <http://svn.qos.ch>
- write to us at [logback-user@qos.ch](mailto:logback-user@qos.ch)
- file a bug report at <http://jira.qos.ch/>
- chat with us at [irc.freenode.net#qos](irc://freenode.net/qos)
- talk to us at +41 21 312 32 26

# Sample Video: Gülcü at Devovxx



# Source Code

Support

Mailing Lists

Bug Report

Source Repository

Call for volunteers

Support offerings

HTTP

Git Read-Only

`https://github.com/qos-ch/logback.git`

HTTP

Git Read-Only

`git://github.com/qos-ch/logback.git`

The screenshot shows the GitHub interface for the repository `qos-ch / logback`. The repository is public and has 27 pull requests. The description states: "The reliable, generic, fast and flexible logging framework for Java. — Read more <http://logback.qos.ch>". Below the description, there are buttons for ZIP, HTTP, and Git Read-Only access. The Git Read-Only access URL is `https://github.com/qos-ch/logback.git`. The repository has 7 branches, and the current branch is `master`. The latest commit to the `master` branch is titled "Added missing LoggerNameUtil, moved ScanException to spi package, min..." and was authored by `ceki` 8 days ago. Below the commit list, there is a table with columns `name`, `age`, and `message`.

github  [Explore](#) [Gist](#) [Blog](#) [Help](#)

PUBLIC `qos-ch / logback` [Watch](#)

**Code** Network Pull Requests 27

The reliable, generic, fast and flexible logging framework for Java. — [Read more](#)  
<http://logback.qos.ch>

[ZIP](#) **HTTP** Git Read-Only `https://github.com/qos-ch/logback.git` [Read-Only access](#)

[branch: master](#) **Files** Commits Branches 7

🕒 Latest commit to the **master** branch

Added missing LoggerNameUtil, moved ScanException to spi package, min... [...](#)

**ceki** authored 8 days ago

**logback** /

name	age	message
------	-----	---------

# Git Clone

```
$ git clone --depth 10 git://git.qos.ch/logback
Cloning into logback...
remote: Counting objects: 20507, done.
remote: Compressing objects: 100% (10061/10061), done.
remote: Total 20507 (delta 11796), reused 15120 (delta 7033)
Receiving objects: 100% (20507/20507), 16.72 MiB | 85 KiB/s, done.
Resolving deltas: 100% (11796/11796), done.
```

```
$ ls
logback
$ cd logback/
$ ls
codeStyle.xml  LICENSE.txt  logback-classic  logback-examples  pom.xml  release.sh  version.pl
goscip        logback-access  logback-core    logback-site      README.txt  src
```

# Discover Source

```
$ tree -L 2
```

```
.
|-- codeStyle.xml
|-- gosc
|-- LICENSE.txt
|-- logback-access
|   |-- build.xml
|   |-- keywords.html
|   |-- lib
|   |-- LICENSE.txt
|   |-- pom.xml
|   |-- src
|   `-- target
-- logback-classic
|   |-- bundle
|   |-- felix-cache
|   |-- integration.xml
|   |-- lib
|   |-- LICENSE.txt
|   |-- osgi-build.xml
|   |-- performance
|   |-- pom.xml
|   |-- src
|   `-- target
-- logback-core
|   |-- LICENSE.txt
|   |-- pom.xml
|   |-- src
|   `-- target
-- logback-examples
|   |-- lib
|   |-- pom.xml
|   |-- src
|   `-- target
```

```
$ cd logback-core/
```

```
$ tree -L 3
```

```
.
|-- pom.xml
|-- src
|   |-- main
|   |   |-- java
|   |   `-- test
|   |-- input
|   |-- java
|   |-- loopfs
|   |-- resource
|   |-- scala
|   `-- witness
-- target
```

10 directories, 2 files

```
$ cd src/main/java/
```

```
$ tree -L 5
```

```
.
|-- ch
|   |-- qos
|   |   |-- logback
|   |   |   |-- core
|   |   |   |   |-- AppenderBase.java
|   |   |   |   |-- Appender.java
|   |   |   |   |-- AsyncAppenderBase.java
|   |   |   |   |-- BasicStatusManager.java
|   |   |   |   |-- boolex
|   |   |   |   |-- ConsoleAppender.java
|   |   |   |   |-- ContextBase.java
|   |   |   |   |-- Context.java
|   |   |   |   |-- CoreConstants.java
|   |   |   |   |-- db
|   |   |   |   |-- encoder
|   |   |   |   |-- FileAppender.java
|   |   |   |   |-- filter
|   |   |   |   |-- helpers
|   |   |   |   |-- html
|   |   |   |   |-- joran
|   |   |   |   |-- layout
|   |   |   |   |-- LayoutBase.java
|   |   |   |   |-- Layout.java
|   |   |   |   |-- LogbackException.java
|   |   |   |   |-- net
```

# Compile Source (Maven)

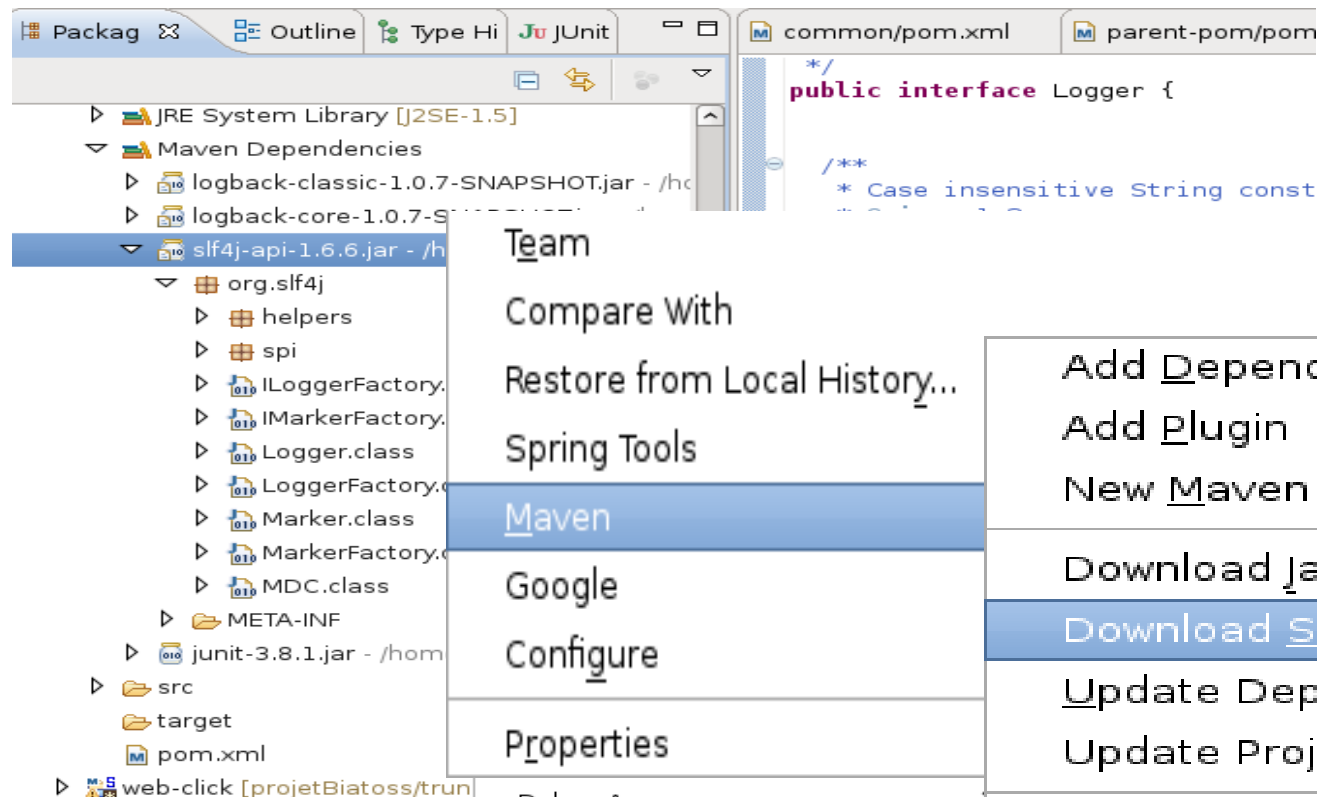
- file pom.xml : “Project Object Model”
- => descriptor for Maven
- Run “mvn install”

```
$ mvn install
[INFO] Scanning for projects...
[INFO] -----
[INFO] Reactor Build Order:
[INFO] -----
[INFO] Logback-Pare[INFO] Reactor Summary:
[INFO] Logback Core[INFO] Logback-Parent ..... SUCCESS [0.649s]
[INFO] Logback Clas[INFO] Logback Core Module ..... SUCCESS [4.582s]
[INFO] Logback Acce[INFO] Logback Classic Module ..... SUCCESS [7.969s]
[INFO] Logback Site[INFO] Logback Access Module ..... SUCCESS [0.957s]
[INFO] Logback Exan[INFO] Logback Site ..... SUCCESS [0.636s]
[INFO] [INFO] Logback Examples Module ..... SUCCESS [0.674s]
[INFO] [INFO] -----
[INFO] [INFO] BUILD SUCCESS
[INFO] [INFO] -----
[INFO] Building Log[INFO] Total time: 16.396s
[INFO] [INFO] Finished at: Sun Aug 12 11:46:07 CEST 2012
[INFO] [INFO] Final Memory: 30M/231M
[INFO] [INFO] -----
$
```

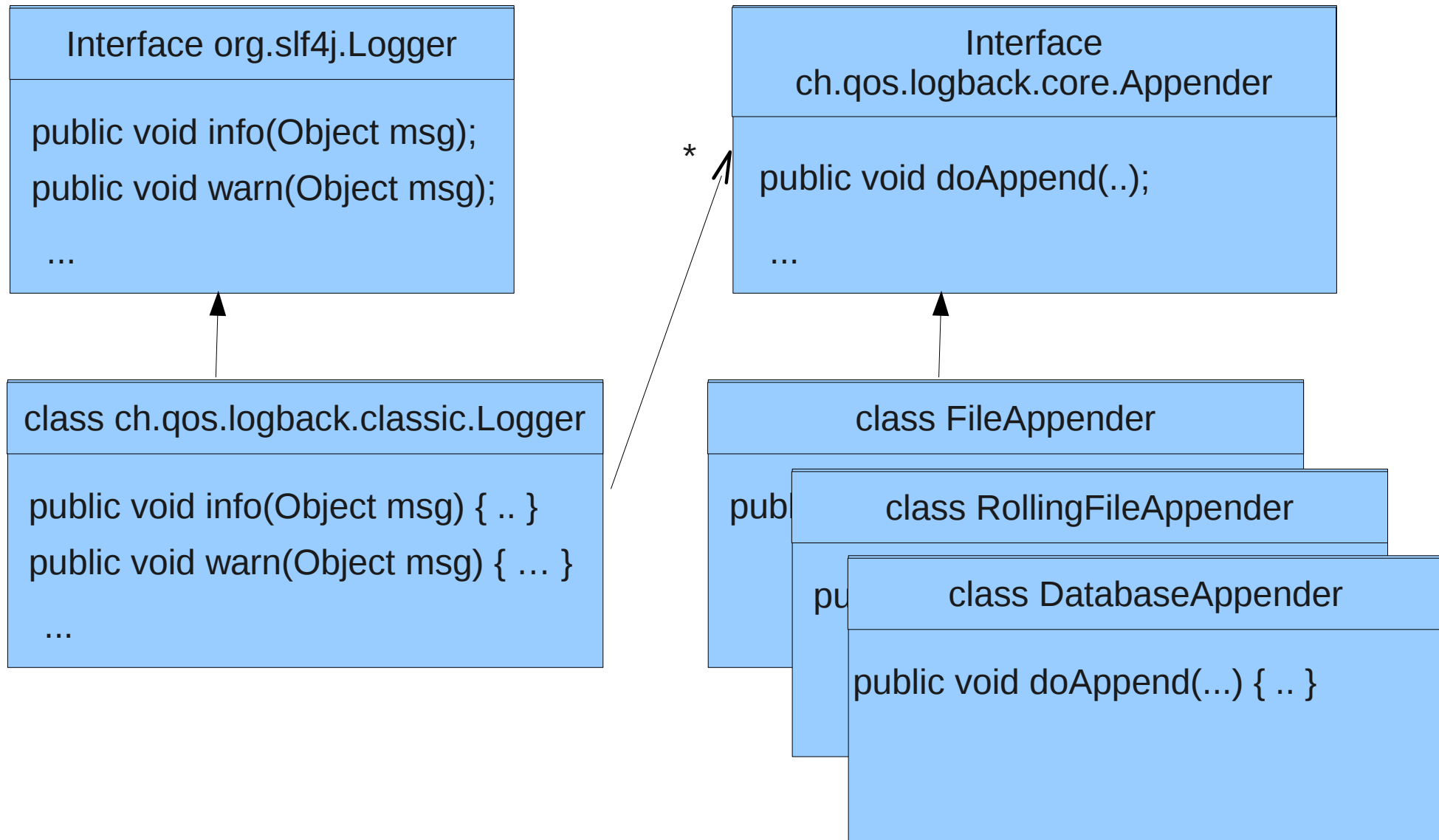
# Download Other Sources in Maven-Eclipse!

Overlay decorator means  
Eclipse HAS sources

➤  slf4j-api-1.6.6.jar  
➤  junit-3.8.1.jar - /h



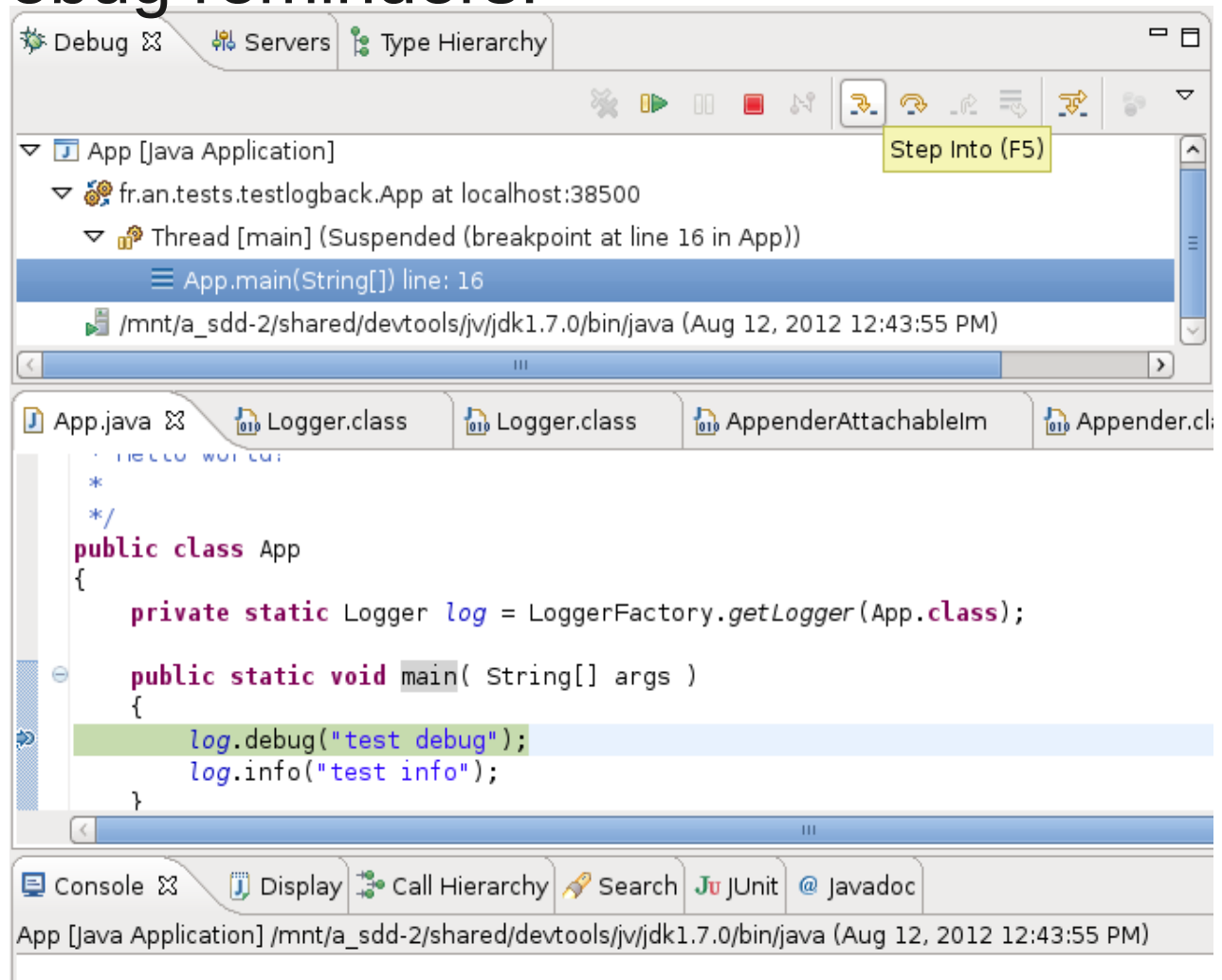
# Read = Understand as UML + Draw





# Test = Debug It

- Run Eclipse, Put Breakpoints...
- Eclipse Shortcuts Debug reminders:
- F11 = launch
- F5 < F6 < F7 < F8  
(Step Into  
< Step Over  
< Step Out  
< Resume )



# Conclusion

Logback is a great project  
High Quality, Open Source, Standard

Easy and Worth Reading It

Perfect for Starting Reading