A Journey in Software Development

An overview of methods and tools (part 2)

Mathieu Acher

Maître de Conférences

mathieu.acher@irisa.fr

UNIVERSITÉ DE RENNES 1

istic Informatique Électronique

Material

http://mathieuacher.com/teaching/PDL/

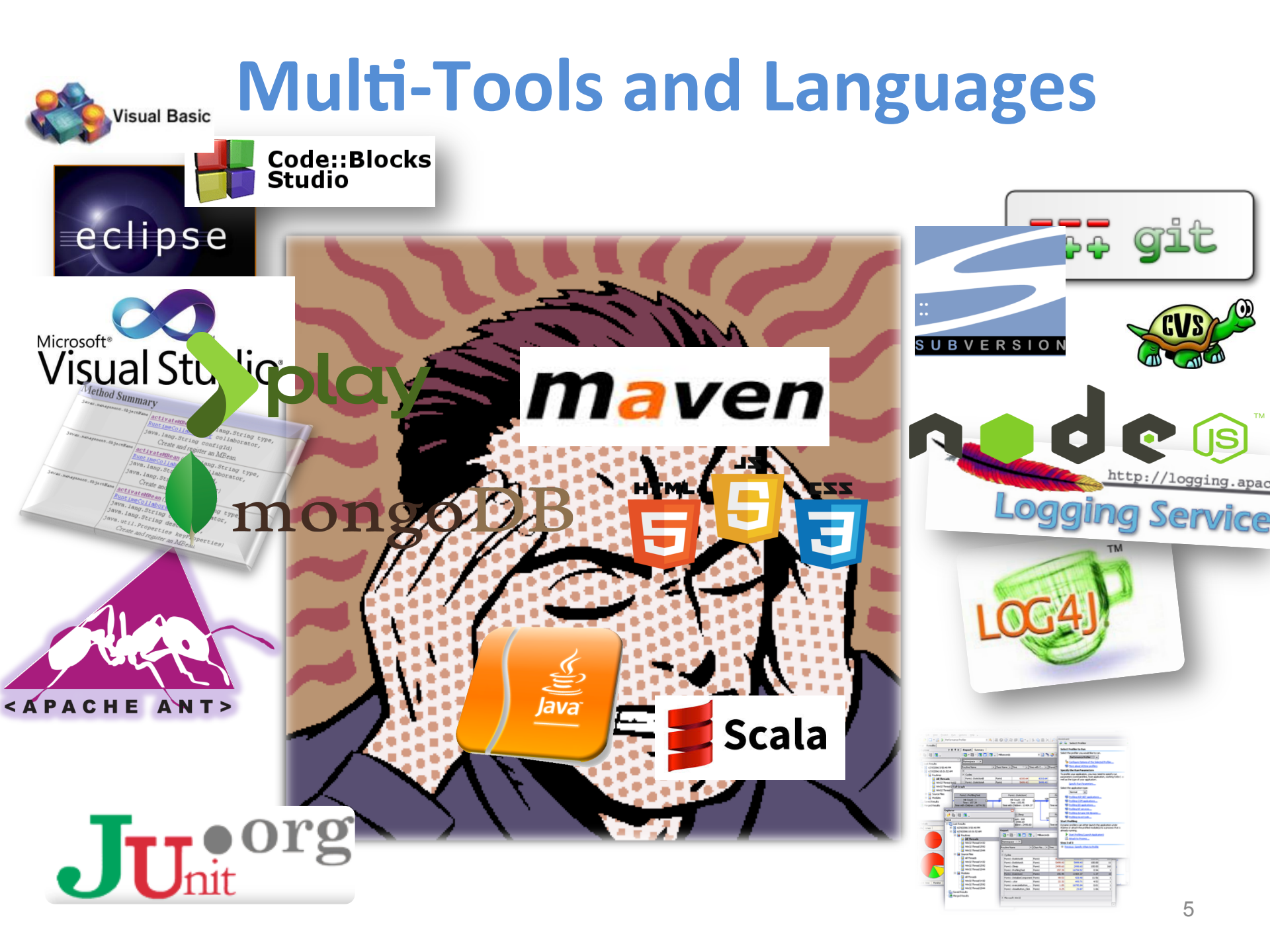
Agenda

Point projet

- Sprint 1: bientôt !
- Pas de CM la semaine prochaine

- Remplacé par un CM “soutenances des groupes” (mi-janvier) pour que tout un chacun assiste au travail des autres groupes et ainsi puisse:
  - Apprendre des technologies et méthodes utilisées
  - Retours d’expériences
  - Permet éventuellement de se comparer

Multi-Tools and Languages



Aujourd'hui

- Contenu du cours aujourd'hui: présentation d'outils et langages utiles pour le projet
  - pour une future utilisation concrète (sprints)
  - pour une compréhension du projet
- Logging (versus Debugging/Test)
  - Log4J
- Build System
  - Maven
- Web development
  - Play!

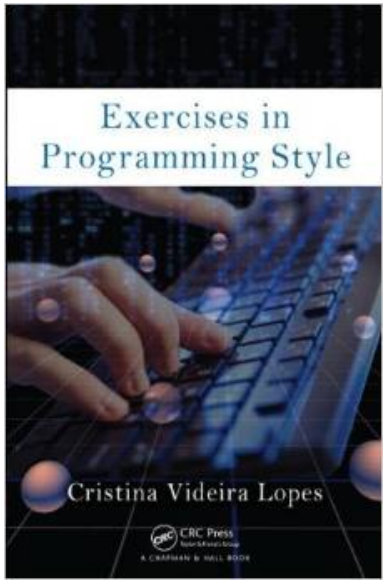
Aujourd'hui

- Contenu du cours aujourd'hui: présentation d'outils et langages utiles pour le projet
  - Logging, Build system, Web

- Principe général
  - Pourquoi ces outils/techniques existent?
  - Indépendamment de toute technologie dans un premier temps
- Illustration sur des technologies récentes et comparaison

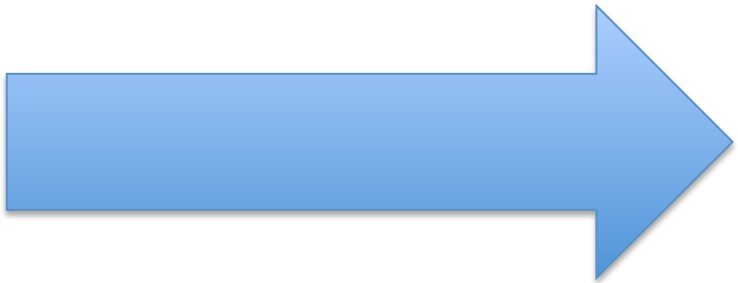
Logging, Debugging, Testing

Given a text file, output a list of the N most frequently-occurring non stop, words, ordered by decreasing frequency



N: integer

I2: File



O1: string

N=25

branch: master ▾ exercises-in-programming-style / input.txt

crista on 22 Sep 2013 Added input files

1 contributor

I2=

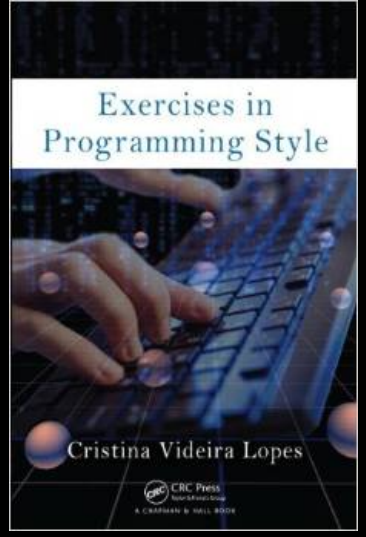
3 lines (2 sloc) | 0.067 kb

```
1  White tigers live mostly in India
2  Wild lions live mostly in Africa
```

« Given a text file, output a list of the N most frequently-occurring non stop, words, ordered by decreasing frequency »

O1=
« live - 2
mostly - 2
africa - 1
india - 1
lions - 1
tigers - 1
white - 1
wild - 1 »

```
import sys, string
# the global list of [word, frequency] pairs
word_freqs = []
# the list of stop words
with open('../stop_words.txt') as f:
    stop_words = f.read().split(',')
stop_words.extend(list(string.ascii_lowercase))

# iterate through the file one line at a time
for line in open(sys.argv[1]):
    start_char = None
    i = 0
    for c in line:
        if start_char == None:
            if c.isalnum():
                # We found the start of a word
                start_char = i
        else:
            if not c.isalnum():
                # We found the end of a word. Process it
                found = False
                word = line[start_char:i].lower()
                # Ignore stop words
                if word not in stop_words:
                    pair_index = 0
                    # Let's see if it already exists
                    for pair in word_freqs:
                        if word == pair[0]:
                            pair[1] += 1
                            found = True
                            found_at = pair_index
                            break
                        pair_index += 1
                    if not found:
                        word_freqs.append([word, 1])
                    elif len(word_freqs) > 1:
                        # We may need to reorder
                        for n in reversed(range(pair_index)):
                            if word_freqs[pair_index][1] >
                                word_freqs[n][1]:
                                # swap
                                word_freqs[n], word_freqs[
                                    pair_index] = word_freqs[
                                    pair_index], word_freqs[n]
                                pair_index = n
                # Let's reset
                start_char = None
        i += 1

for tf in word_freqs[0:25]:
    print tf[0], ' - ', tf[1]
```



```
# the global list of [word, frequency] pairs
word_freqs = []
# the list of stop words
with open('../stop_words.txt') as f:
    stop_words = f.read().split(',')
stop_words.extend(list(string.ascii_lowercase))
```

```
    for c in line:
        if start_char == None:
            if c.isalnum():
                # We found the start of a word
                start_char = i
        else:
            if not c.isalnum():
                # We found the end of a word. Process it
                found = False
                word = line[start_char:i].lower()
                # Ignore stop words
                if word not in stop_words:
                    pair_index = 0
                    # Let's see if it already exists
                    for pair in word_freqs:
                        if word == pair[0]:
                            pair[1] += 1
                            found = True
                            found_at = pair_index
                            break
                        pair_index += 1
                    if not found:
                        word_freqs.append([word, 1])
                    elif len(word_freqs) > 1:
                        # We may need to reorder
                        for n in reversed(range(pair_index)):
                            if word_freqs[pair_index][1] >
                                word_freqs[n][1]:
                                # swap
                                word_freqs[n], word_freqs[
                                    pair_index] = word_freqs[
                                    pair_index], word_freqs[n]
                                pair_index = n
                # Let's reset
                start_char = None
        i += 1

for tf in word_freqs[0:25]:
    print tf[0], ' - ', tf[1]
```

```
import sys, string
# the global list of [word, frequency] pairs
word_freqs = []
# the list of stop words
with open('../stop_words.txt') as f:
    stop_words = f.read().split(',')
```

```
for line in open(sys.argv[1]):
```

```
    for c in line:
```

```
            if not c.isalnum():
                # We found the end of a word. Process it
                found = False
                word = line[start_char:i].lower()
                # Ignore stop words
                if word not in stop_words:
                    pair_index = 0
                    # Let's see if it already exists
                    for pair in word_freqs:
                        if word == pair[0]:
                            pair[1] += 1
                            found = True
                            found_at = pair_index
                            break
                        pair_index += 1
                    if not found:
                        word_freqs.append([word, 1])
                    elif len(word_freqs) > 1:
                        # We may need to reorder
                        for n in reversed(range(pair_index)):
                            if word_freqs[pair_index][1] >
                                word_freqs[n][1]:
                                # swap
                                word_freqs[n], word_freqs[
                                    pair_index] = word_freqs[
                                    pair_index], word_freqs[n]
                                pair_index = n
                # Let's reset
                start_char = None
        i += 1

for tf in word_freqs[0:25]:
    print tf[0], ' - ', tf[1]
```

N=25

branch: master ▾ exercises-in-programming-style / input.txt

 **crista** on 22 Sep 2013 Added input files

1 contributor

I2=

3 lines (2 sloc) | 0.067 kb

```
1  White tigers live mostly in India
2  Wild lions live mostly in Africa
```

« Given a text file, output a list of the N most frequently-occurring non stop, words, ordered by decreasing frequency »

O1=
« live - 2
mostly - 2
africa - 1
india - 1
lions - 1
tigers - 3
white - 1
wild - 1 »

Let say…

N=25

branch: master ▾ exercises-in-programming-style / input.txt

crista on 22 Sep 2013 Added input files

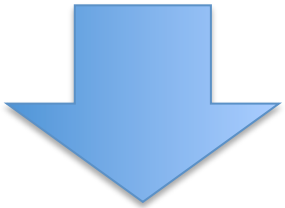
1 contributor

I2=

3 lines (2 sloc) | 0.067 kb

```
1  White tigers live mostly in India
2  Wild lions live mostly in Africa
```

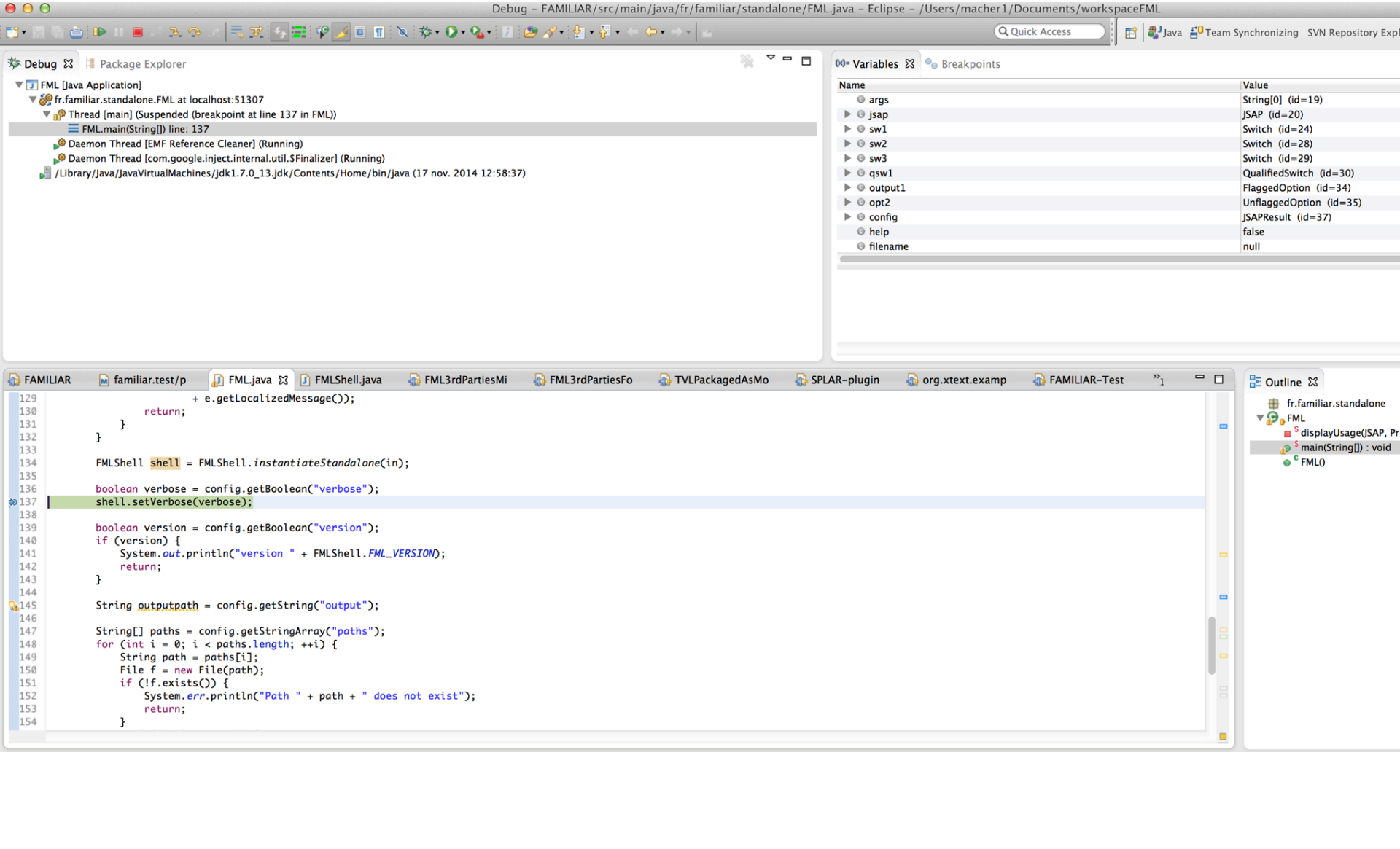
« Given a text file, output a list of the N most frequently-occurring non stop, words, ordered by decreasing frequency »

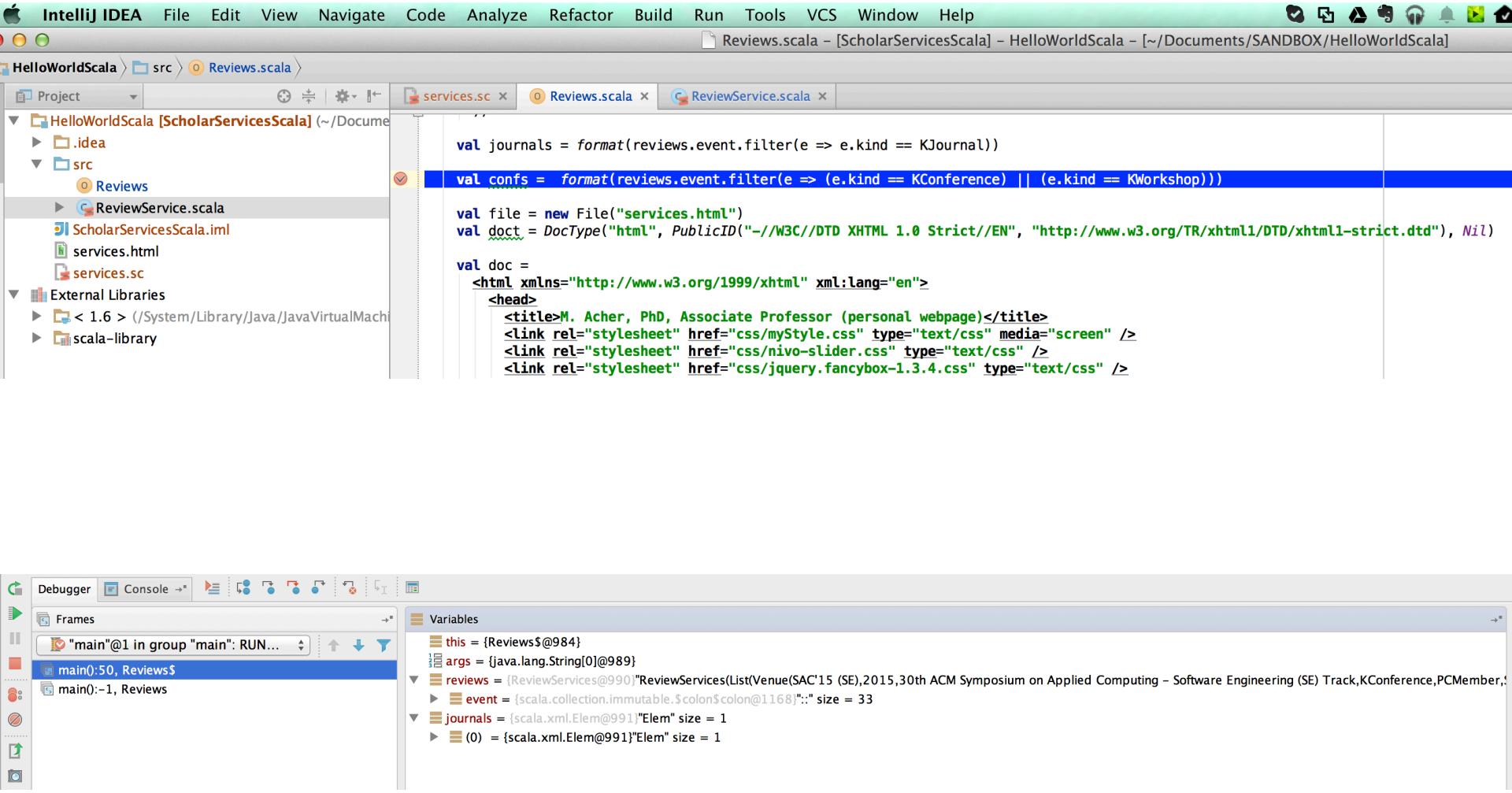


O1=
« live - 2
mostly - 2
africa - 1
india - 1
lions - 1
tigers - 3
white - 1
wild - 1 »

System.out.println (Logging)?
Debugging?
Profiling?
Testing?

Debugging


Debug - FAMILIAR/src/main/java/fr/familiar/standalone/FML.java - Eclipse - /Users/macher1/Documents/workspaceFML
Quick Access
Java
Team Synchronizing
SVN Repository Expl
Debug
Package Explorer
FML [Java Application]
fr.familiar.standalone.FML at localhost:51307
Thread [main] (Suspended (breakpoint at line 137 in FML))
FML.main(String[]) line: 137
Daemon Thread [EMF Reference Cleaner] (Running)
Daemon Thread [com.google.inject.internal.util.$Finalizer] (Running)
/Library/Java/JavaVirtualMachines/jdk1.7.0_13.jdk/Contents/Home/bin/java (17 nov. 2014 12:58:37)
Variables
Breakpoints
Name
Value
args
String[0] (id=19)
jsap
JSAP (id=20)
sw1
Switch (id=24)
sw2
Switch (id=28)
sw3
Switch (id=29)
qsw1
QualifiedSwitch (id=30)
output1
FlaggedOption (id=34)
opt2
UnflaggedOption (id=35)
config
JSAPResult (id=37)
help
false
filename
null
FAMILIAR
familiar.test/p
FML.java
FMLShell.java
FML3rdPartiesMi
FML3rdPartiesFo
TVLPackagedAsMo
SPLAR-plugin
org.xtext.examp
FAMILIAR-Test
Outline
fr.familiar.standalone
FML
displayUsage(JSAP, Pr
main(String[]) : void
FML()
129 + e.getLocalizedMessage());
130 return;
131 }
132 }
133
134 FMLShell shell = FMLShell.instantiateStandalone(in);
135
136 boolean verbose = config.getBoolean("verbose");
137 shell.setVerbose(verbose);
138
139 boolean version = config.getBoolean("version");
140 if (version) {
141 System.out.println("version " + FMLShell.FML_VERSION);
142 return;
143 }
144
145 String outputpath = config.getString("output");
146
147 String[] paths = config.getStringArray("paths");
148 for (int i = 0; i < paths.length; ++i) {
149 String path = paths[i];
150 File f = new File(path);
151 if (!f.exists()) {
152 System.err.println("Path " + path + " does not exist");
153 return;
154 }


IntelliJ IDEA File Edit View Navigate Code Analyze Refactor Build Run Tools VCS Window Help
Reviews.scala – [ScholarServicesScala] – HelloWorldScala – [~/Documents/SANDBOX/HelloWorldScala]
HelloWorldScala src Reviews.scala
Project
services.sc
Reviews.scala
ReviewService.scala
HelloWorldScala [ScholarServicesScala] (~/Docume
.idea
src
Reviews
ReviewService.scala
ScholarServicesScala.iml
services.html
services.sc
External Libraries
< 1.6 > (/System/Library/Java/JavaVirtualMachi
scala-library
val journals = format(reviews.event.filter(e => e.kind == KJournal))
val confs = format(reviews.event.filter(e => (e.kind == KConference) || (e.kind == KWorkshop)))
val file = new File("services.html")
val doct = DocType("html", PublicID("-//W3C//DTD XHTML 1.0 Strict//EN", "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd"), Nil)
val doc =
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="en">
<head>
<title>M. Acher, PhD, Associate Professor (personal webpage)</title>
<link rel="stylesheet" href="css/myStyle.css" type="text/css" media="screen" />
<link rel="stylesheet" href="css/nivo-slider.css" type="text/css" />
<link rel="stylesheet" href="css/jquery.fancybox-1.3.4.css" type="text/css" />
Debugger
Console
Frames
"main"@1 in group "main": RUN...
main():50, Reviews$
main():-1, Reviews
Variables
this = {Reviews$@984}
args = {java.lang.String[0]@989}
reviews = {ReviewServices@990}"ReviewServices(List(Venue(SAC'15 (SE),2015,30th ACM Symposium on Applied Computing – Software Engineering (SE) Track,KConference,PCMember,
event = {scala.collection.immutable.$colon$colon@1168}"::" size = 33
journals = {scala.xml.Elem@991}"Elem" size = 1
(0) = {scala.xml.Elem@991}"Elem" size = 1

ETAPE 2 : CHOISIS 3 BONS SOUVENIRS

a97d2829.main.js (index) a97d2829.main.js:formatted ×

```
5059        },pause: function() {
5060            this.element[this.subwidget]("pause")
5061        },resume: function() {
5062            this.element[this.subwidget]("resume")
5063        },getPosition: function() {
5064            return this.element[this.subwidget]("getPosition")
5065        },getDuration: function() {
5066            return this.element[this.subwidget]("getDuration")
5067        },playM3u8: function(a, c, d, e) {
5068            var f = null;
5069            c && (f = this.element[this.subwidget]("getPosition")), b.debug("play keeping position",
5070        },preloadHLS: function(a) {
5071            this._hlsArray = a.slice(0), this.element[this.subwidget]("supportDataUri") && (this._pre
5072        },playHLS: function(a, b, c, d) {
5073            h("wm.send", "event", this._analytics, "navigation", "playVideo"), 1 == d && h("wm.send",
5074            var e = this;
5075            this._preloading ? this._preloading.done(function(a) {
```

Watch Expressions

Call Stack Async

Not Paused

Scope Variables

Not Paused

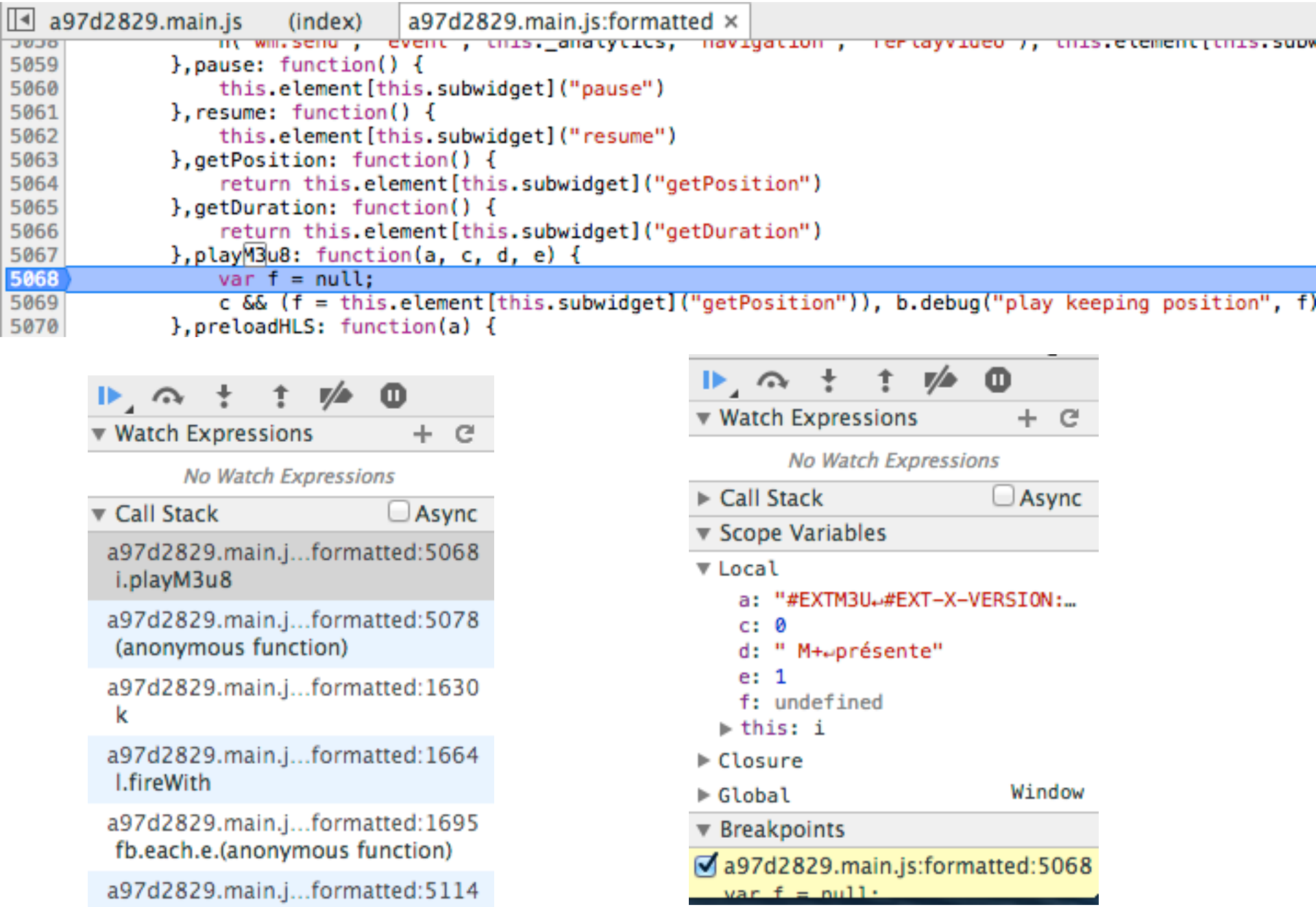
Breakpoints

a97d2829.main.js:formatted:5068
var f = null;

DOM Breakpoints

XHR Breakpoints

Event Listener Breakpoints


a97d2829.main.js
(index)
a97d2829.main.js:formatted
},pause: function() {
this.element[this.subwidget]("pause")
},resume: function() {
this.element[this.subwidget]("resume")
},getPosition: function() {
return this.element[this.subwidget]("getPosition")
},getDuration: function() {
return this.element[this.subwidget]("getDuration")
},playM3u8: function(a, c, d, e) {
var f = null;
c && (f = this.element[this.subwidget]("getPosition")), b.debug("play keeping position", f)
},preloadHLS: function(a) {
Watch Expressions
No Watch Expressions
Call Stack
Async
a97d2829.main.j...formatted:5068
i.playM3u8
a97d2829.main.j...formatted:5078
(anonymous function)
a97d2829.main.j...formatted:1630
k
a97d2829.main.j...formatted:1664
l.fireWith
a97d2829.main.j...formatted:1695
fb.each.e.(anonymous function)
a97d2829.main.j...formatted:5114
Watch Expressions
No Watch Expressions
Call Stack
Async
Scope Variables
Local
a: "#EXTM3U↵#EXT-X-VERSION:…
c: 0
d: " M+↵présente"
e: 1
f: undefined
this: i
Closure
Global
Window
Breakpoints
a97d2829.main.js:formatted:5068

Profiling

- ~**Debugging** the, e.g., performance
  - Say the computation of term frequency takes 5 minutes (instead of a few seconds)

N: integer

I2: File

O1: string

- Performance can be **tested** as well!

Logging

Logging, why? (claims)

- Logging is easier than debugging
- Logging is faster than debugging
- Logging can work in environments where debugging is not supported
- Can work in production environments
- Logs can be referenced anytime in future as the data is stored

Logging

- Logging is chronological and systematic record of data processing events in a program
  - e.g. the Windows Event Log
- Logs can be saved to a persistent medium to be studied at a later time
- Use logging in the development phase:
  - Logging can help you **debug** the code
- Use logging in the production environment:
  - Helps you **troubleshoot problems**

Logging Methods, How?

- The evil System.out.println()
- Custom Solution to Log to various datastores, eg text files, db, etc…
- Use Standard APIs
  - Don't reinvent the wheel

Log4J

- Popular logging frameworks for Java
- Designed to be reliable, fast and extensible
- Simple to understand and to use API
- Allows the developer to control which log statements are output with arbitrary granularity
- Fully configurable at runtime using external configuration files

Log4J Architecture

- Log4J has three main components: loggers, appenders and layouts
  - Loggers
    - Channels for printing logging information
  - Appenders
    - Output destinations (console, File, Database, Email/ SMS Notifications, Log to a socket, and many others…)
  - Layouts
    - Formats that appenders use to write their output
- Priorities

Logger

- Responsible for Logging
- Accessed through java code
- Configured Externally
- Every Logger has a name
- Prioritize messages based on level
  - TRACE, DEBUG, INFO, WARN, ERROR & FATAL
- Usually named following dot convention like java classes do.
  - Eg com.foo.bar.ClassName
- Follows inheritance based on name

Logger API

- Factory methods to get Logger
  - `Logger.getLogger(Class c)`
  - `Logger.getLogger(String s)`
- Method used to log message
  - `trace(), debug(), info(), warn(), error(), fatal()`
  - Details
    - `void debug(java.lang.Object message)`
    - `void debug(java.lang.Object message, java.lang.Throwable t)`
  - Generic Log method
    - `void log(Priority priority, java.lang.Object message)`
    - `void log(Priority priority, java.lang.Object message, java.lang.Throwable t)`

Root Logger

- The root logger resides at the top of the logger hierarchy. It is exceptional in two ways:
  1. it always exists,
  2. it cannot be retrieved by name.

- Logger.getRootLogger()

Appender

- Appenders put the log messages to their actual destinations.
- No programatic change is require to configure appenders
- Can add multiple appenders to a Logger.
- Each appender has its Layout.
- ConsoleAppender, DailyRollingFileAppender, FileAppender, JDBCAppender, JMSAppender, NTEventLogAppender, RollingFileAppender, SMTPAppender, SocketAppender, SyslogAppender, TelnetAppender

Layout

- Used to customize the format of log output.
- Eg. HTMLLayout, PatternLayout, SimpleLayout, XMLLayout
- Most commonly used is PatternLayout
  - Uses C-like syntax to format.
    - `Eg. "%-5p [%t]: %m%n`
    - `DEBUG [main]: Message 1 WARN [main]: Message 2`

Log4j Basics

- Who will log the messages?
  - The Loggers
- What decides the priority of a message?
  - Level
- Where will it be logged?
  - Decided by Appender
- In what format will it be logged?
  - Decided by Layout

Log4j in Action

```
// get a logger instance named "com.foo"
Logger  logger = Logger.getLogger("com.foo");

// Now set its level. Normally you do not need to set the
// level of a logger programmatically. This is usually done
// in configuration files.
logger.setLevel(Level.INFO);

Logger barlogger = Logger.getLogger("com.foo.Bar");

// This request is enabled, because WARN >= INFO.
logger.warn("Low fuel level.");

// This request is disabled, because DEBUG < INFO.
logger.debug("Starting search for nearest gas station.");

// The logger instance barlogger, named "com.foo.Bar",
// will inherit its level from the logger named
// "com.foo" Thus, the following request is enabled
// because INFO >= INFO.
barlogger.info("Located nearest gas station.");

// This request is disabled, because DEBUG < INFO.
barlogger.debug("Exiting gas station search");
```

Layouts bis (eg colorizing Logs)

- http://logging.apache.org/log4j/2.x/manual/layouts.html
- http://jeanchristophegay.com/de-la-couleur-dans-les-logs/

```xml
<?xml version="1.0" encoding="UTF-8"?>
<configuration status="OFF">
    <appenders>
        <Console name="Console" target="SYSTEM_OUT">
            <PatternLayout pattern="%d{HH:mm:ss.SSS} [%t] %highlight{%-5level} %logger{36} - %msg%n"/>
        </Console>
    </appenders>
    <loggers>
        <root level="trace">
            <appender-ref ref="Console"/>
        </root>
    </loggers>
</configuration>
```

```
19:54:42.838 [main] TRACE com.github.jcgay.example.log.log4j2.Main - a trace message.
19:54:42.841 [main] DEBUG com.github.jcgay.example.log.log4j2.Main - a debug message.
19:54:42.842 [main] INFO  com.github.jcgay.example.log.log4j2.Main - an info message.
19:54:42.842 [main] WARN  com.github.jcgay.example.log.log4j2.Main - a warn message.
19:54:42.843 [main] ERROR com.github.jcgay.example.log.log4j2.Main - a error message.
19:54:42.845 [main] FATAL com.github.jcgay.example.log.log4j2.Main - a fatal message.
```

Log4j Optimization & Best Practises

- User logger as private static variable
- Only one instance per class
- Name logger after class name
- Don’t use too many appenders
- Don’t use time-consuming conversion patterns (see javadoc)
- Use Logger.isDebugEnabled() if need be
- Prioritize messages with proper levels

Logging in JavaScript / NodeJS

- Not only in Java!
- Alternatives also in other languages…
- https://github.com/flatiron/winston

```
var winston = require('winston');

winston.log('info', 'Hello distributed log files!');
winston.info('Hello again distributed logs');
```

Using Logging Levels

Setting the level for your logging message can be accomplished in one of two ways. You can pass a string representing the logging level to the log() method or use the level specified methods defined on every winston Logger.

```
//
// Any logger instance
//
logger.log('silly', "127.0.0.1 - there's no place like home");
logger.log('debug', "127.0.0.1 - there's no place like home");
logger.log('verbose', "127.0.0.1 - there's no place like home");
logger.log('info', "127.0.0.1 - there's no place like home");
logger.log('warn', "127.0.0.1 - there's no place like home");
logger.log('error', "127.0.0.1 - there's no place like home");
logger.info("127.0.0.1 - there's no place like home");
logger.warn("127.0.0.1 - there's no place like home");
logger.error("127.0.0.1 - there's no place like home");
```

Colorization (back)

```
var myCustomLevels = {
  levels: {
    foo: 0,
    bar: 1,
    baz: 2,
    foobar: 3
  },
  colors: {
    foo: 'blue',
    bar: 'green',
    baz: 'yellow',
    foobar: 'red'
  }
};

var customLevelLogger = new (winston.Logger)({ levels: myCustomLevels.levels });
customLevelLogger.foobar('some foobar level-ed message');
```

Logging in Play!


```
MacBook-Pro-de-Mathieu-3:webfml macher1$ ~/Downloads/play-2.2.0/play
[info] Loading project definition from /Users/macher1/git/webfml/project

This project uses Play 2.2.4!
Update the Play sbt-plugin version to 2.2.0 (usually in project/plugins.sbt)

[info] Set current project to FMLApp (in build file:/Users/macher1/git/webfml/)

      _
 _ __ | | __ _ _  _
| '_ \| |/ _' | || |
|  __/|_|\____|\__ /
|_|            |__/

play 2.2.4 built with Scala 2.10.3 (running Java 1.7.0_13), http://www.playframework.com

> Type "help play" or "license" for more information.
> Type "exit" or use Ctrl+D to leave this console.

[FMLApp] $ run 8080

--- (Running the application from SBT, auto-reloading is enabled) ---

SLF4J: Class path contains multiple SLF4J bindings.
SLF4J: Found binding in [jar:file:/Users/macher1/git/webfml/lib/KSynthesis-0.0.1-SNAPSHOT-jar-with-dependencies.ja
SLF4J: Found binding in [jar:file:/Users/macher1/Downloads/play-2.2.0/repository/local/ch.qos.logback/logback-clas
SLF4J: See http://www.slf4j.org/codes.html#multiple_bindings for an explanation.
SLF4J: Actual binding is of type [org.slf4j.impl.Log4jLoggerFactory]
0    [pool-4-thread-2] INFO                                    play  - Listening for HTTP on /0:0:0:0:0:0:0:0:8080

(Server started, use Ctrl+D to stop and go back to the console...)
```

```
MacBook-Pro-de-Mathieu-3:webfml macher1$ ~/Downloads/play-2.2.0/play
[info] Loading project definition from /Users/macher1/git/webfml/project

This project uses Play 2.2.4!
Update the Play sbt-plugin version to 2.2.0 (usually in project/plugins.sbt)

[info] Set current project to FMLApp (in build file:/Users/macher1/git/webfml/)
      _
 _ __ | | __ _ _  _
| '_ \| |/ _' | || |
|  __/|_|\____|\__ /
|_|            |__/

play 2.2.4 built with Scala 2.10.3 (running Java 1.7.0_13), http://www.playframework.com

> Type "help play" or "license" for more information.
> Type "exit" or use Ctrl+D to leave this console.

[FMLApp] $ run 8080

--- (Running the application from SBT, auto-reloading is enabled) ---

SLF4J: Class path contains multiple SLF4J bindings.
SLF4J: Found binding in [jar:file:/Users/macher1/git/webfml/lib/KSynthesis-0.0.1-SNAPSHOT-jar-with-dependencies.ja
SLF4J: Found binding in [jar:file:/Users/macher1/Downloads/play-2.2.0/repository/local/ch.qos.logback/logback-clas
SLF4J: See http://www.slf4j.org/codes.html#multiple_bindings for an explanation.
SLF4J: Actual binding is of type [org.slf4j.impl.Log4jLoggerFactory]
0    [pool-4-thread-2] INFO                                play  - Listening for HTTP on /0:0:0:0:0:0:0:0:8080

(Server started, use Ctrl+D to stop and go back to the console...)

[info] Compiling 1 Scala source to /Users/macher1/git/webfml/target/scala-2.10/classes...
15409 [play-internal-execution-context-1] INFO                                play  - Application started (Dev)
15918 [play-akka.actor.default-dispatcher-4] INFO        akka.event.slf4j.Slf4jLogger  - Slf4jLogger started
not a USER:
I create a new Interpreter
I'm creating a new session
Session Created: 63
redirect to the ide
USER: m
I create a new Interpreter
I'm creating a new session
Session Created: 31
redirect to the ide
USER: m
I create a new Interpreter
I'm creating a new session
Session Created: 2
redirect to the ide
USER: m
I create a new Interpreter
I'm creating a new session
Session Created: 11
redirect to the ide
USER: ml
I create a new Interpreter
I'm creating a new session
Session Created: 14
redirect to the ide
```

This is bad!

This is bad!

tHub, Inc. [US] https://github.com/FAMILIAR-project/webfml/blob/master/app/controllers/FamiliarIDEController.scala

```
16    /**
17          * Function which receive information from the login
18          * page and create an interpreter and a session
19          */
20         def receiveInformations(login:String, password:String,language:String)
21
22                 var familiarInstance : FamiliarIDEFactory = new ConcreteFamili
23                 //an instane object of FamiliarInterpreter
24                 var inter = familiarInstance.createInterpreter()
25                 //a session object
26                 var ses = familiarInstance.createSession()
27                 //create a session
28                 ses.create(login,password,sessions.toList)
29                 //store
30                 sessions+=ses
                   storeFamiliar(ses, inter)
32                 println("redirect to the ide")
33                 //check the user
34
35                 //try to redirect to the page of the IDE
36                 //Redirect("/ide/familiar",MOVED_PERMANENTLY)
37                 Ok("")
38    }
```

Application.scala | loginPage.scala.html | FamiliarIDEController.scala

```
//list which contains all the the sessions
var sessions = new ListBuffer[Session]()
/**
 * Function which receive information from the login
 * page and create an interpreter and a session
 */
def receiveInformations (login : String, password : String, language : String) = A
   request.session.get("connected").map { user =>
    println("USER: " + user)
    Ok("Hello " + user)
  }.getOrElse {
    println("not a USER: ")
    Ok("Welcome!").withSession("connected" -> login)
    //Unauthorized("Oops, you are not connected")
  }



  var familiarInstance : FamiliarIDEFactory = new ConcreteFamiliarIDEFactory
  //an instane object of FamiliarInterpreter
  var inter = familiarInstance.createInterpreter()
  //a session object
  var ses = familiarInstance.createSession()
  //create a session
  ses.create(login,password,sessions.toList)
  //store
  sessions+=ses
  storeFamiliar(ses, inter)
  println("redirect to the ide")
  //check the user

  //try to redirect to the page of the IDE
  //Redirect("/ide/familiar", MOVED_PERMANENTLY)
  //Ok("")
  Ok("Welcome!").withSession("connected" -> login)
}
```

This is bad!

Download | Documentation | Get Involved

Documentation

The Logging API

Using logging in your application can be useful for monitoring, debugging, error tracking, and business intelligence. Play provides an API for logging which is accessed through the `Logger` class and uses Logback as the logging engine.

Browse versions 2.3.x

Browse APIs Scala Java

Language English

Search 2.3.x docs

http://logback.qos.ch/
(successor of Log4J)

The Logging API

Using logging in your application can be useful for monitoring, debugging, error tracking, and business intelligence. Play provides an API for logging which is accessed through the `Logger` class and uses Logback as the logging engine.

Logging architecture

The logging API uses a set of components that help you to implement an effective logging strategy.

Logger

Your application can define loggers to send log message requests. Each logger has a name which will appear in log messages and is used for configuration.

Loggers follow a hierarchical inheritance structure based on their naming. A logger is said to be an ancestor of another logger if its name followed by a dot is the prefix of descendant logger name. For example, a logger named “com.foo” is the ancestor of a logger named “com.foo.bar.Baz.” All loggers inherit from a root logger. Logger inheritance allows you to configure a set of loggers by configuring a common ancestor.

Play applications are provided a default logger named “application” or you can create your own loggers. The Play libraries use a logger named “play”, and some third party libraries will have loggers with their own names.

Log levels

Log levels are used to classify the severity of log messages. When you write a log request statement you will specify the severity and this will appear in generated log messages.

This is the set of available log levels, in decreasing order of severity.

- `OFF` - Used to turn off logging, not as a message classification.
- `ERROR` - Runtime errors, or unexpected conditions.
- `WARN` - Use of deprecated APIs, poor use of API, ‘almost’ errors, other runtime situations that are undesirable or unexpected, but not necessarily “wrong”.
- `INFO` - Interesting runtime events such as application startup and shutdown.
- `DEBUG` - Detailed information on the flow through the system.
- `TRACE` - Most detailed information.

play

Class Logger

```
java.lang.Object
  └play.Logger
```

```
public class Logger
extends java.lang.Object
```

High level API for logging operations. Example, logging with the default application logger:

```
Logger.info("Hello!");
```

Example, logging with a custom logger:

```
Logger.of("my.logger").info("Hello!")
```

Nested Class Summary

static class	Logger.ALogger Typical logger interface

Constructor Summary

Logger()

Method Summary

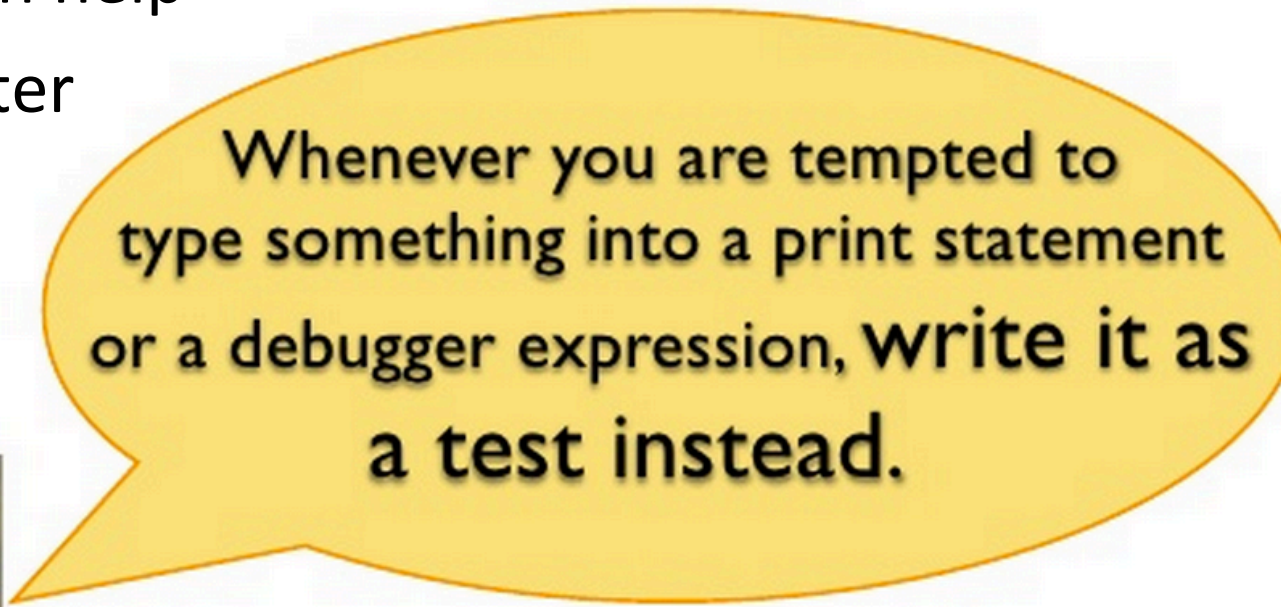
static void	debug(java.lang.String message) Log a message with the DEBUG level.
static void	debug(java.lang.String message, java.lang.Throwable error) Log a message with the DEBUG level.
static void	error(java.lang.String message) Log a message with the ERROR level.
static void	error(java.lang.String message, java.lang.Throwable error) Log a message with the ERROR level.
static void	info(java.lang.String message) Log a message with the INFO level.
static void	info(java.lang.String message, java.lang.Throwable error) Log a message with the INFO level.
static boolean	isDebugEnabled() Returns true if the logger instance enabled for the DEBUG level?
static boolean	isErrorEnabled() Returns true if the logger instance enabled for the ERROR level?
static boolean	isInfoEnabled() Returns true if the logger instance enabled for the INFO level?
static boolean	isTraceEnabled() Returns true if the logger instance enabled for the TRACE level?

You can’t test everything
(so one advice by Martin Fowler)

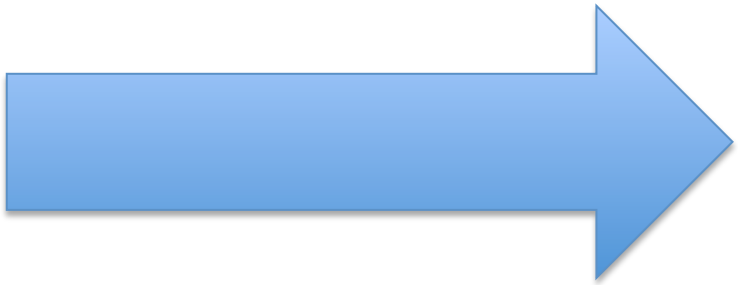
Whenever you are tempted to type something into a print statement or a debugger expression, write it as a test instead.

From Logging to Testing

- Testing: “the activity of finding out whether a piece of code produces the intended behavior”
  - Debugging can help
  - Testing is better than debugging



Inputs

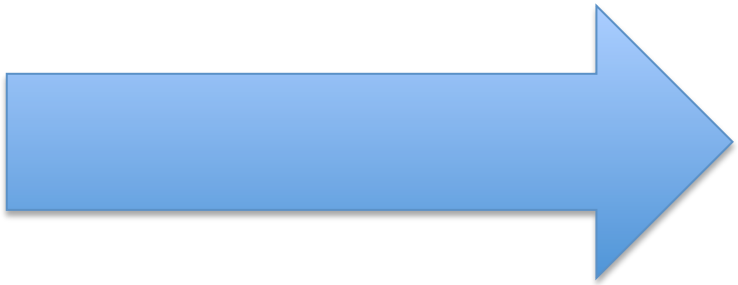


Outputs

Logging (manual inspection of some values at specific points)

N: integer

I2: File

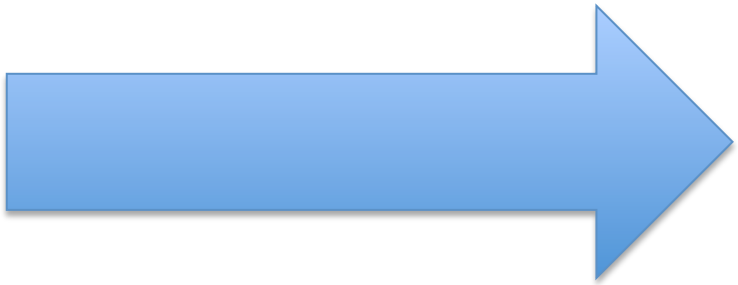


O1: integer

Debugging (manual inspection of values at some points)

N: integer

I2: File



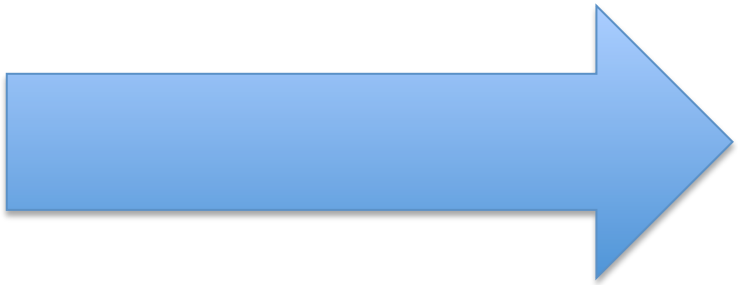
O1: integer

Testing

(automated verification)

N: integer

I2: File

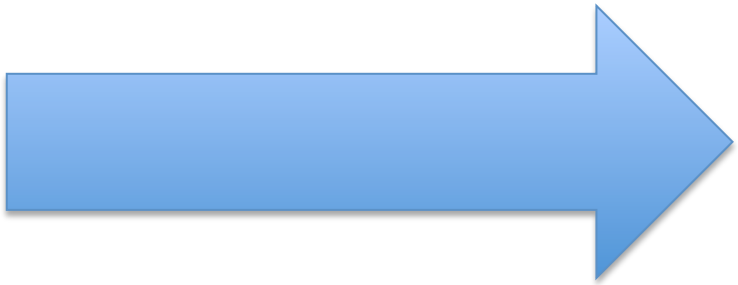


O1: integer

Testing

(automated verification)

Inputs



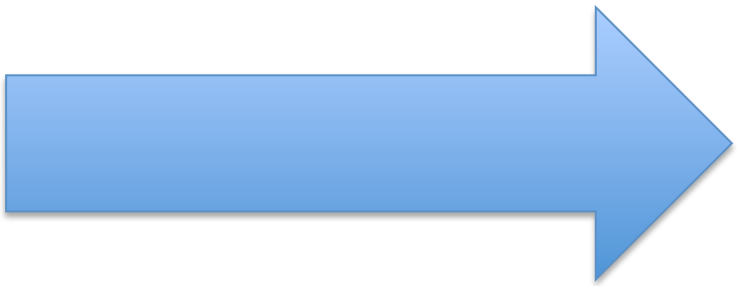
Outputs

CONTROLLABILITY

ability to manipulate the software's input as well as to place this software into a particular state

OBSERVABILITY

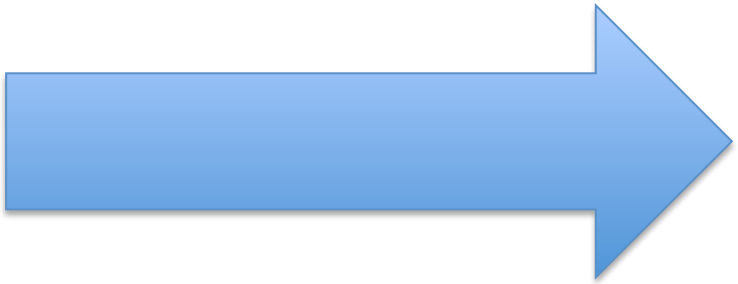
deals with the possibility to observe the outputs and state changes that

Inputs  Outputs

TESTABILITY

degree to which a system or component **facilitates the establishment** of test criteria and the performance of tests to determine whether those criteria have been met.

Controllability + Observability

Inputs  Outputs

Conclusion

- How to improve Testability?
  - Refactoring, Design patterns
  - Separation of concerns, Modularity, Abstractions

- Logging
- Debugging
- Testing

Whenever you are tempted to type something into a print statement or a debugger expression, write it as a test instead.

- Logging
  - Manual observation
  - (Usually) manual control on input values
  - Pre-defined exploration of values
- Debugging
  - Manual observation
  - (Usually) manual control on input values
  - Interactive, fine-grained exploration of values
- Testing
  - Automated control and observation (assertions)
  - More ameanable to re-executing on different inputs
  - Not to understand, but to verify some properties

Document, refactor… Execute your tests… Debug.. Write test..

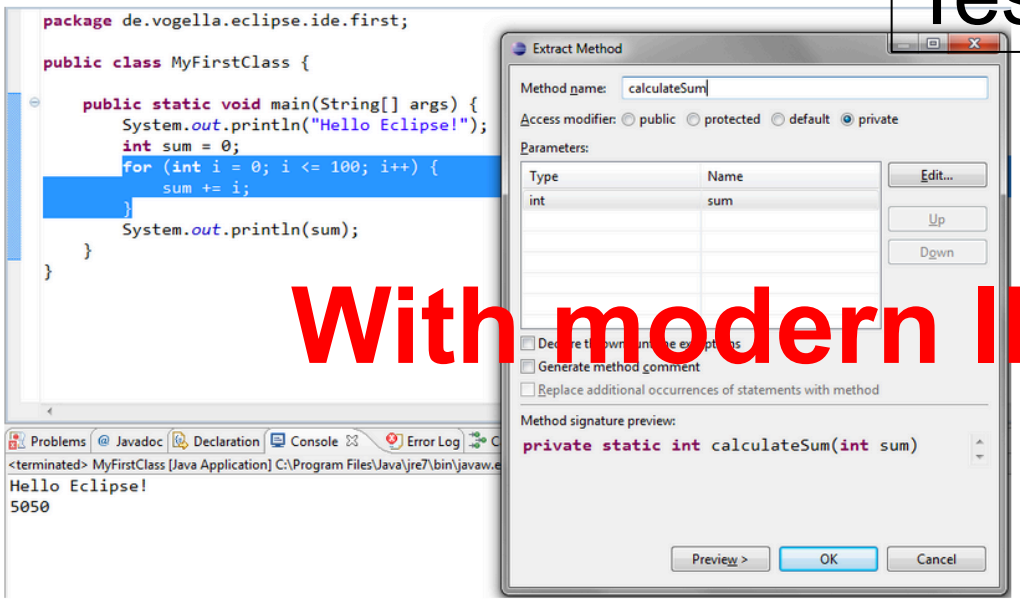
And so on!

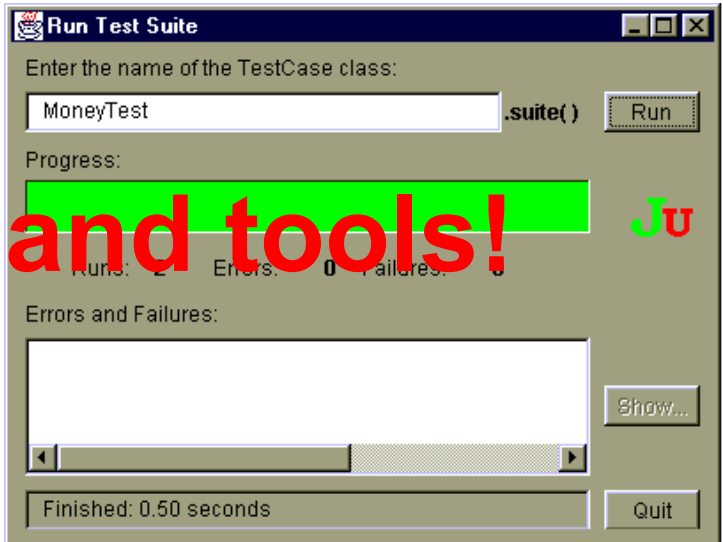
Documenting

Debugging

Refactoring

Testing





With modern IDE and tools!

Maven

```
PACKAGE      = package
VERSION      = `date "+%Y.%m%d%"`
RELEASE_DIR  = ..
RELEASE_FILE = $(PACKAGE)-$(VERSION)

# Notice that the variable LOGNAME comes from the environment in
# POSIX shells.
#
# target: all - Default target. Does nothing.
all:
        echo "Hello $(LOGNAME), nothing to do by default"
        # sometimes: echo "Hello ${LOGNAME}, nothing to do by default"
        echo "Try 'make help'"

# target: help - Display callable targets.
help:
        egrep "^# target:" [Mm]akefile

# target: list - List source files
list:
        # Won't work. Each command is in separate shell
        cd src
        ls

        # Correct, continuation of the same shell
        cd src; \
        ls
```

Make/Makefile

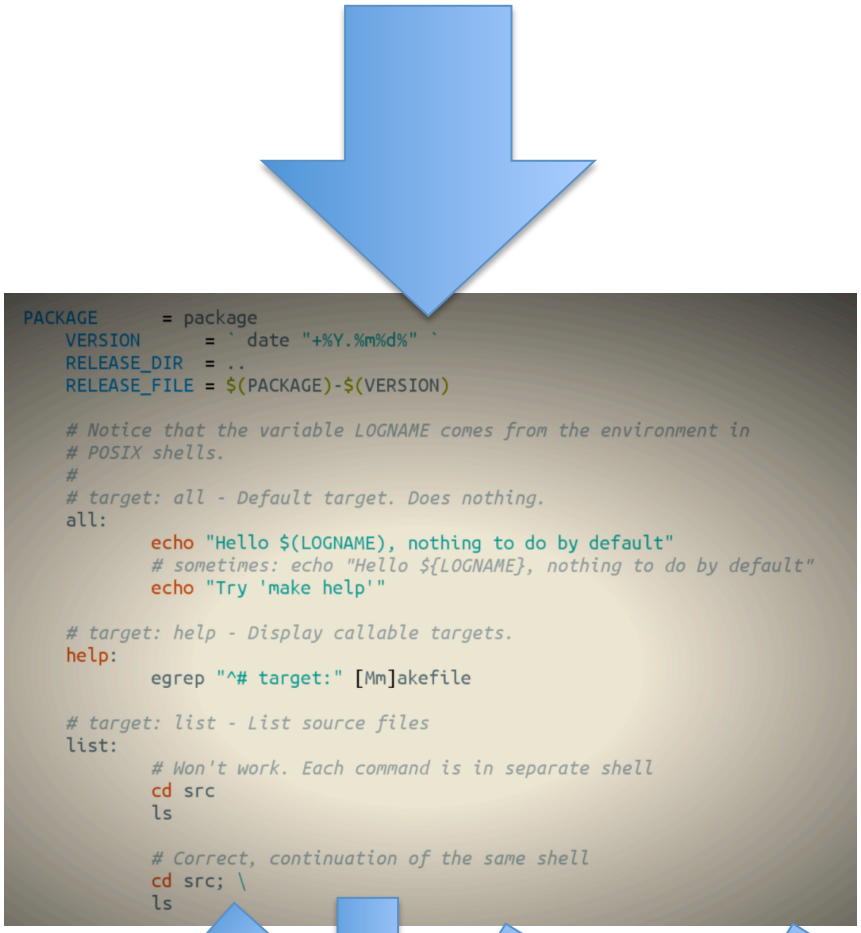
Original problem: compiling your source code files can be tedious!

- several source files
- type the compiling commands everytime

Make for increasing automation, avoiding accidental complexity, and have more flexibility when « compiling » projects

(initial release in 1977)

Make



```
PACKAGE      = package
    VERSION      = ` date "+%Y.%m%d%" `
    RELEASE_DIR  = ..
    RELEASE_FILE = $(PACKAGE)-$(VERSION)

    # Notice that the variable LOGNAME comes from the environment in
    # POSIX shells.
    #
    # target: all - Default target. Does nothing.
    all:
            echo "Hello $(LOGNAME), nothing to do by default"
            # sometimes: echo "Hello ${LOGNAME}, nothing to do by default"
            echo "Try 'make help'"

    # target: help - Display callable targets.
    help:
            egrep "^# target:" [Mm]akefile

    # target: list - List source files
    list:
            # Won't work. Each command is in separate shell
            cd src
            ls

            # Correct, continuation of the same shell
            cd src; \
            ls
```

Makefile

help

compile

gendoc

list

.....

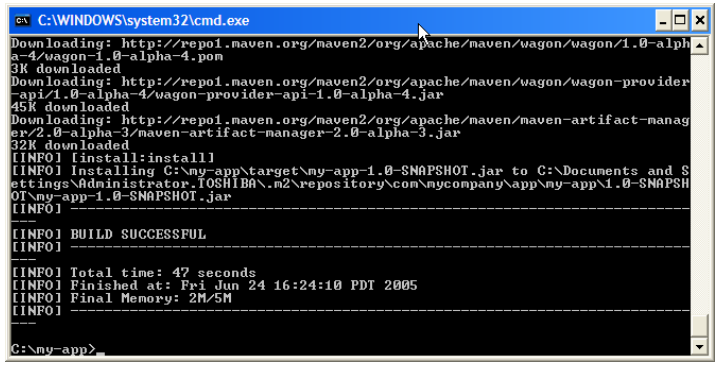
Compile chain

- Sometimes hidden in the IDE
  - But generally speaking, you need to master your “compile” chain
- Tools
  - make, gmake, nmake (Win),
  - Apache ANT, Apache **MAVEN**, Freshmeat 7Bee ...
- To **automate**:
  - pre-compilation, obfuscation, verification
  - generation of .class and .jar
    - normal, tracing, debug, …
  - documentation generation
  - « stubs » generation (rmic, idl2java, javacard …)
  - test
  - 3rd party libraries/dependencies

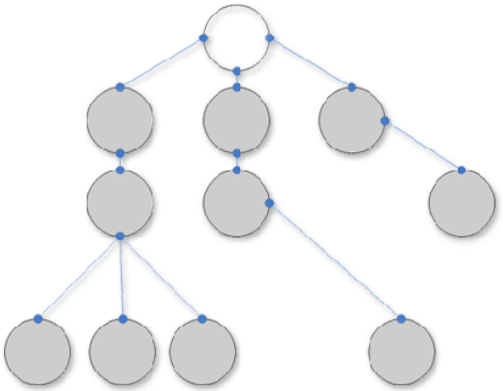
  … And a **combination** of all these tasks

What is Maven?

A build tool



A dependency management tool



A documentation tool

Apply patterns to project build infrastructure

Maven is really a process of applying **patterns** to a build infrastructure in order to provide a coherent view of software projects.

Provides a way to help with managing:

- Builds
- Documentation
- Reporting
- Dependencies
- Software Configuration Management
- Releases

Objectives

- Make the development process visible or transparent
- Provide an easy way to see the health and status of a project
- Decreasing training time for new developers
- Bringing together the tools required in a uniform way
- Preventing inconsistent setups
- Providing a standard development infrastructure across projects
- Focus energy on writing applications

Benefits

- Standardization
- Fast and easy to set up a powerful build process
- Greater momentum vs. Ant – it is now becoming legacy and not moving fast ahead.
- Dependency management (automatic downloads)
- Project website generation, Javadoc
- Repository management
- Extensible architecture

Maven and POM

aka project's configurations

```
<project xmlns="http://maven.apache.org/POM/4.0.0" xmlns:xsi="http://www.w3.org/2001/
XMLSchema-instance"
  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>com.mycompany.app</groupId>
  <artifactId>my-app</artifactId>
  <version>1.0-SNAPSHOT</version>
  <packaging>jar</packaging>

  <name>Maven Quick Start Archetype</name>
  <url>http://maven.apache.org</url>

  <dependencies>
    <dependency>
      <groupId>junit</groupId>
      <artifactId>junit</artifactId>
      <version>4.8.2</version>
      <scope>test</scope>
    </dependency>
  </dependencies>
</project>
```

Kind of packaging

Maven facilities and lifecycle

validate: validate the project is correct and all necessary information is available
compile: compile the source code of the project
test: test the compiled source code using a suitable unit testing framework. These tests should not require the code be packaged or deployed
package: take the compiled code and package it in its distributable format, such as a JAR.
integration-test: process and deploy the package if necessary into an environment where integration tests can be run
verify: run any checks to verify the package is valid and meets quality criteria
install: install the package into the local repository, for use as a dependency in other projects locally
deploy: done in an integration or release environment, copies the final package to the remote repository for sharing with other developers and projects.

clean: cleans up artifacts created by prior builds

site: generates site documentation for this project

Build the Project

```
mvn package
```

Generating the Site

```
mvn site
```

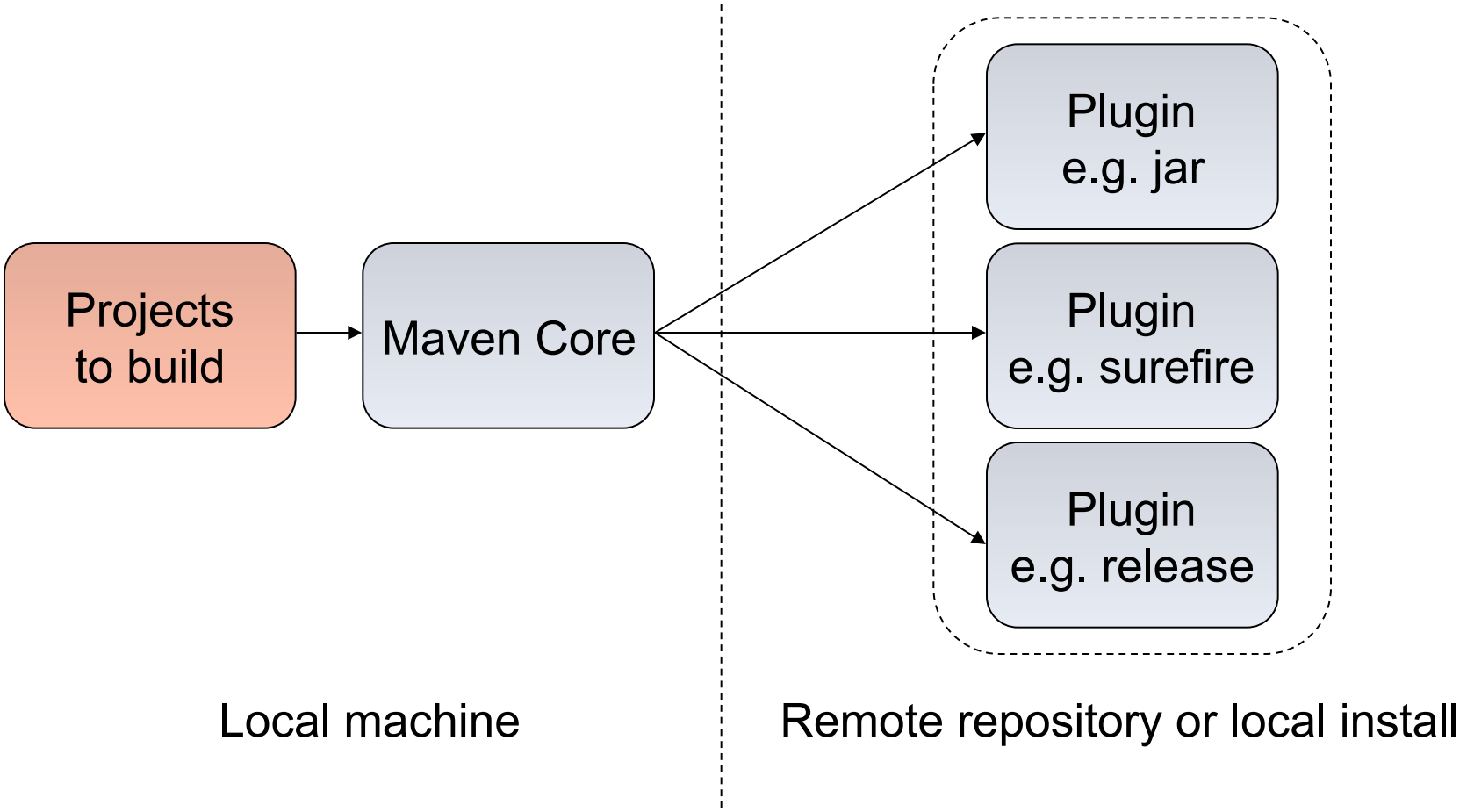
Maven

- Abstract project model (POM)
  - Object oriented, inheritance
  - Separation of concerns
- Default lifecycle
  - Default state (goals) sequence
    - plugins depend on states
- Give a project « standard » structure
  - Standard naming conventions
  - Standard lifecycle
- Automatic handling of dependencies between projects
  - Including updates
- Project repositories
  - public or private, local or remotes
  - caching and proxy
- Extensible via external plugins

Maven plugins

- Core
  - clean, compiler, deploy, install, resources, site, surefire, verifier
- Packaging
  - ear, ejb, jar, rar, war, bundle (OSGi)
- Reporting
  - changelog, changes, checkstyle, clover, doap, docck, javadoc, jxr, pmd, project-info-reports, surefire-report
- Tools
  - ant, antrun, archetype, assembly, dependency, enforcer, gpg, help, invoker, one (interop Maven 1), patch, plugin, release, remote-resource, repository, scm
- IDEs
  - eclipse, netbeans, idea
- Others
  - exec, jdepend, castor, cargo, jetty, native, sql, taglist, javacc, obr …

Maven Architecture



Common project metadata format

- POM = Project Object Model = pom.xml
- Contains metadata about the project
  - Location of directories, Developers/Contributors, Issue tracking system, Dependencies, Repositories to use, etc
- Example:

```
<project>
  <modelVersion>4.0.0</modelVersion>
  <groupId>org.codehaus.cargo</groupId>
  <artifactId>cargo-core-api-container</artifactId>
  <name>Cargo Core Container API</name>
  <version>0.7-SNAPSHOT</version>
  <packaging>jar</packaging>
  <dependencies/>
  <build/>
[…]
```

Minimal POM

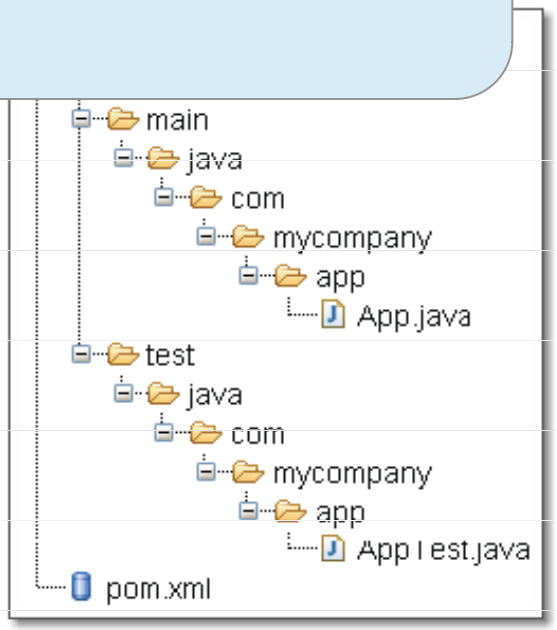
Use Inheritance

Standard directory organization

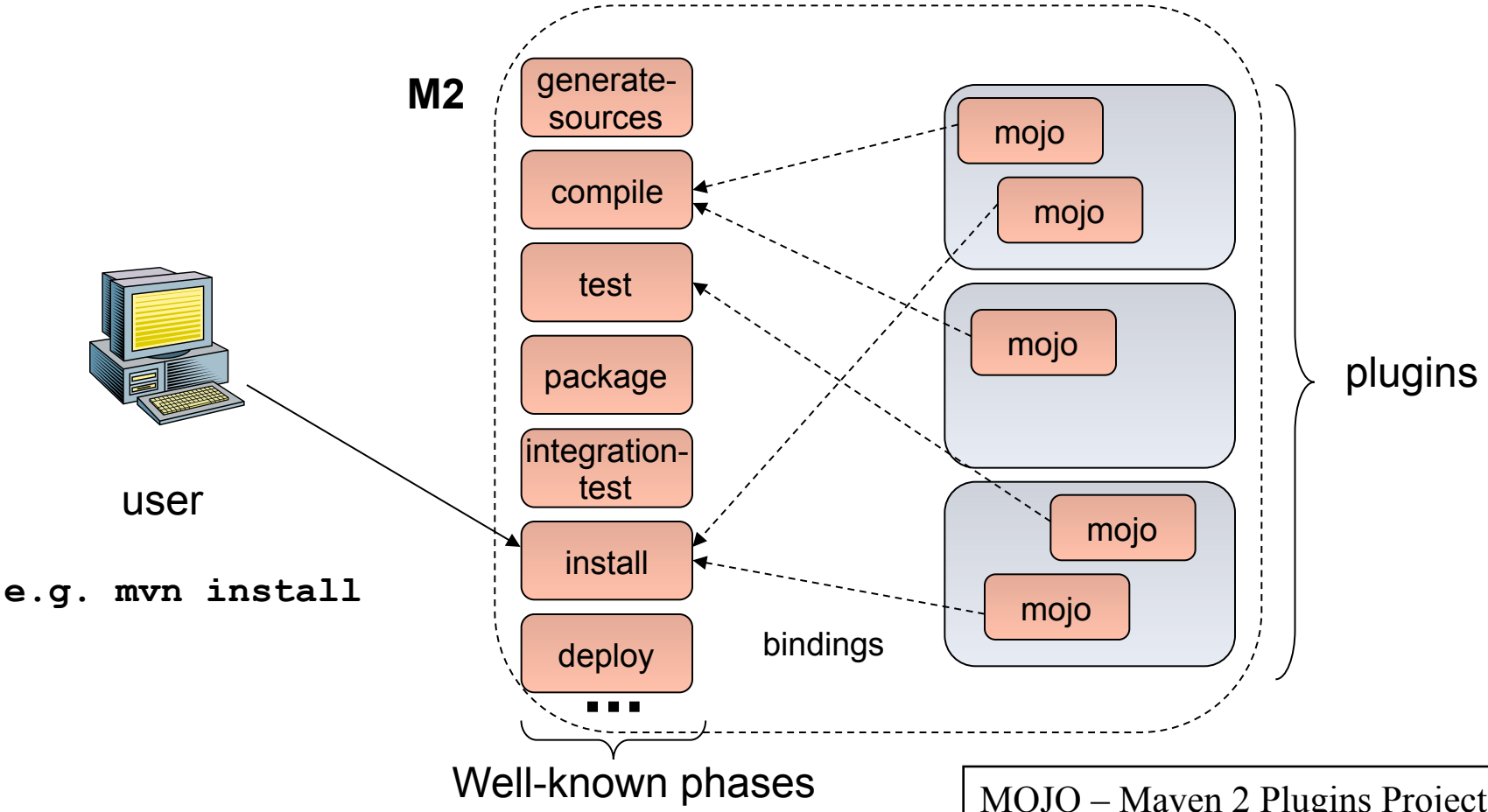
- Having a common directory layout would allow for users familiar with one Mav ther Maven projec

Convention over configuration

src/main/java	
src/main/resources	
src/main/filters	
src/main/assembly	Assembly des
src/main/config	Configurati
src/main/webapp	Web appl es
src/test/java	Test so
src/test/resources	Test r ources
src/test/filters	Test resource filter files
src/site	Site
LICENSE.txt	Project's license
README.txt	Project's readme



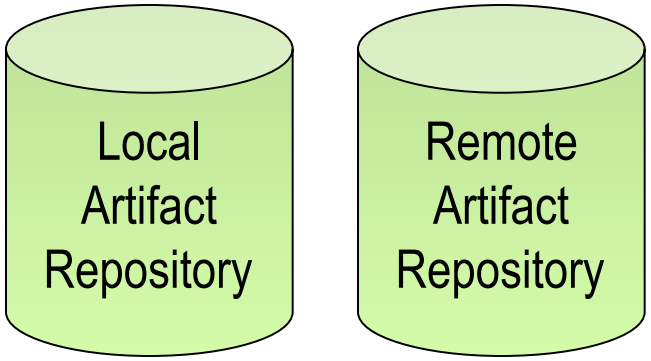
Common way to build applications



MOJO – Maven 2 Plugins Project
http://mojo.codehaus.org/

Artifact repositories (1/3)

- Used to store all kind of artifacts
  - JARs, EARs, WARs, NBMs, EJBs, ZIPs, plugins, …
- All project interactions go through the repository
  - No more relative paths!
  - Easy to share between team

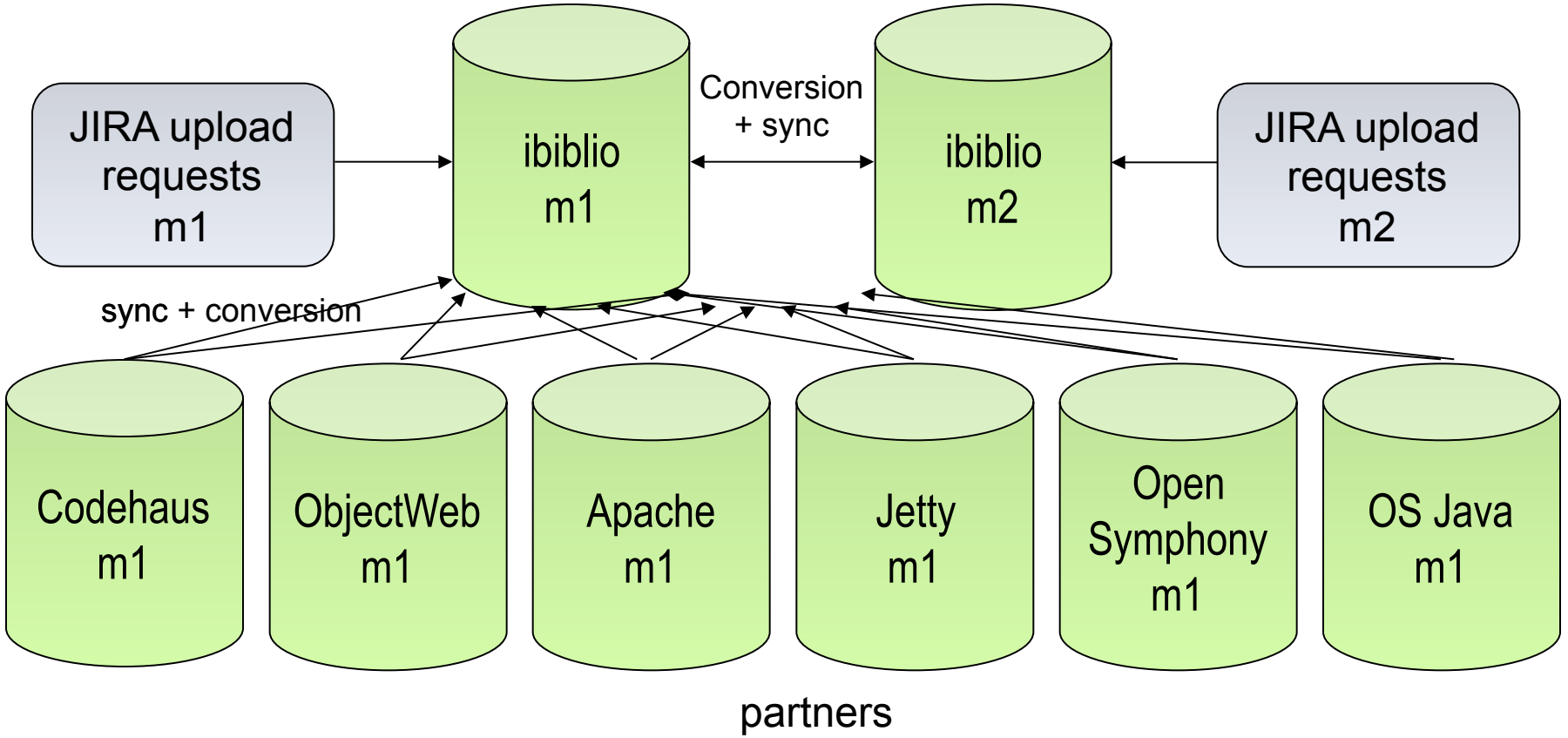


e.g. `http://ibiblio.org/maven2`

```
<repositories>
  <repository>
    <id>maven2-snapshot</id>
    <releases>
      <enabled>true</enabled>
    </releases>
    <name>Maven Central Development Repository</name>
    <url>http://snapshots.maven.codehaus.org/maven2</url>
    <layout>legacy|default</layout>
  </repository>
</repositories>
```

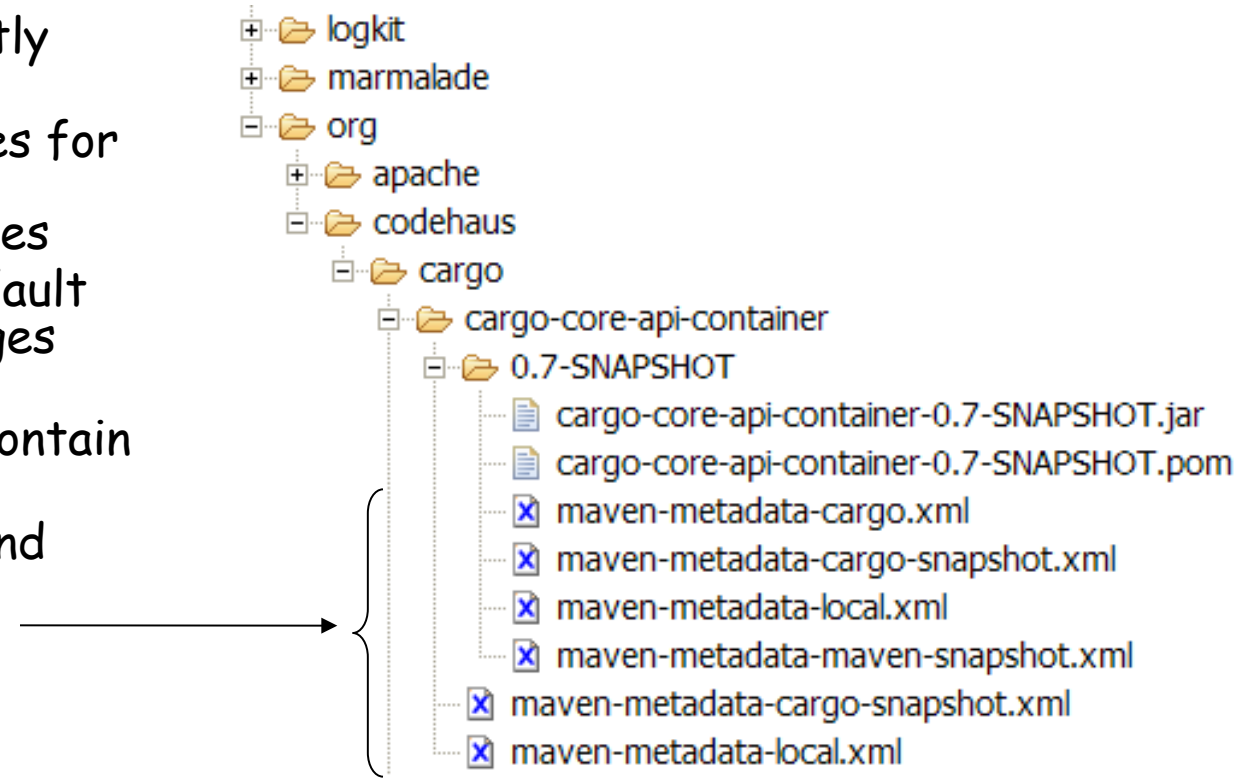
Artifact repositories (2/3)

- Some public remote repositories



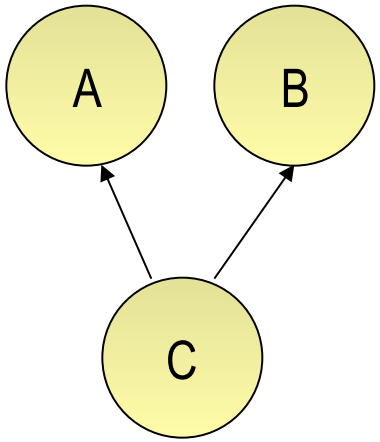
Artifact repositories (3/3)

- Hierarchical structure
- Automatic plugin download
- Plugins are read directly from the repository
- Configurable strategies for checking the remote repositories for updates
  - Daily check by default for plugin and ranges updates
- Remote repositories contain Metadata information
  - Releases, latest, and more to come



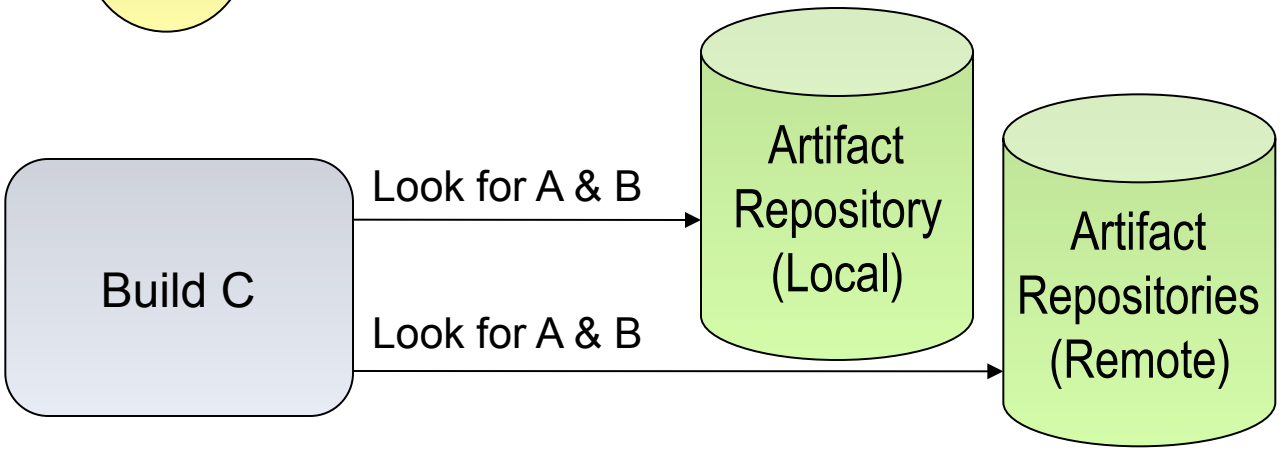
Dependency management (1/2)

- Maven uses binary dependencie



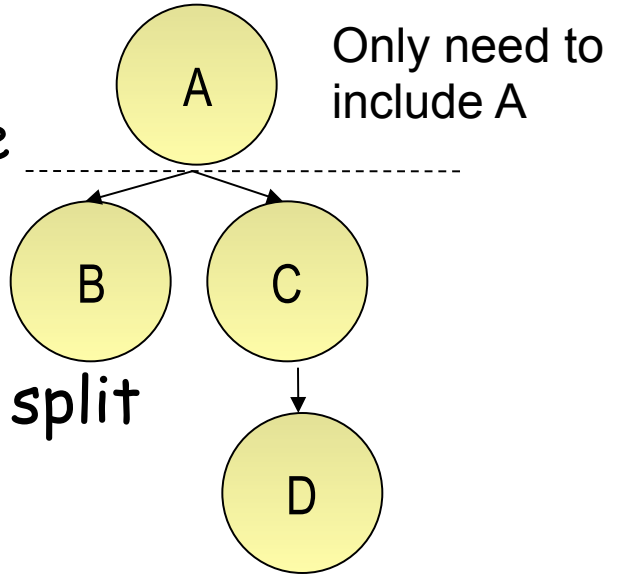
« any version after 1.0 »

```
<dependencies>
  <dependency>
    <groupId>com.acme</groupId>
    <artifactId>B</artifactId>
    <version>[1.0,)</version>
    <scope>compile</scope>
  </dependency>
</dependencies>
```



Dependency management (2/2)

- Transitive dependencies
  - Possibility to exclude some depndencies
  - Need good metadata
  - Ideally projects should be split
- SNAPSHOT handling
  - Always get latest
- Automatic dependency updates
  - By default every day



Installation and Setup

- **http://maven.apache.org/**
- Add Maven's bin directory to PATH
- Ensure JAVA_HOME is set to SDK
- Run **mvn -version** to test install

```
C:\Documents and Settings\alina>mvn -version
Maven version: 3.0.4
Java version: 1.6.0_30
```

Maven plugin for JAVA IDE

- Maven plugins exists for
  - Eclipse
  - Intellij
  - NetBeans
  - …

Installing JARs to local repository

- Sometimes you need to put some specific JARs in your local repository for use in your builds
- The JARs must be placed in the correct place in order for it to be correctly picked up by Maven
- To install a JAR in the local repository use the following command:

```
mvn install:install-file -Dfile=<path-to-file> -DgroupId=<group-id> \
-DartifactId=<artifact-id> -Dversion=<version> -Dpackaging=jar
```

- Now can include dependency in pom.xml:

```
<dependency>
    <groupId><group-id></groupId>
    <artifactId><artifact-id></artifactId>
    <version><version></version>
</dependency>
```

Overview of common Goals

- **clean** – clean the current project
- **validate** - validate the project is correct and all necessary information is available
- **compile** - compile the source code of the project
- **test** - test the compiled source code using a suitable unit testing framework. These tests should not require the code be packaged or deployed
- **package** - take the compiled code and package it in its distributable format, such as a JAR
- **integration-test** - process and deploy the package if necessary into an environment where integration tests can be run
- **install** - install the package into the local repository, for use as a dependency in other projects locally
- **deploy** - done in an integration or release environment, copies the final package to the remote repository for sharing with other developers and projects

More stuff

- Automatically generate reports, diagrams, and so on through Maven / the project site
- Internationalization – create different language project websites
- Create projects within projects (more pom.xml files inside sub dirs\jars), with different build stats and so on
- Maven can make .war files, EJBs, etc.

Using Maven Plugins

- Whenever you want to customise the build for a Maven project, this is done by adding or reconfiguring plugins
- For example, configure the Java compiler to allow JDK 5.0 sources
- Plugins in Maven 3.0 look much like a dependency

```
...
<build>
  <plugins>
    <plugin>
      <groupId>org.apache.maven.plugins</groupId>
      <artifactId>maven-compiler-plugin</artifactId>
      <configuration>
        <source>1.5</source>
        <target>1.5</target>
      </configuration>
    </plugin>
  </plugins>
</build>
...
```

Maven Plugins

- **AlmostPlainText**
- **Maven Axis**
- **Maven Cobertura**
- **Maven DB2**
- **Dbunit**
- **Debian Package**
- **Maven DotUml**
- **Doxygen**
- **Maven Files**
- **FindBugs**
- **Maven flash**
- **Help**
- **Maven IzPack**
- **Java Application**
- **Maven JAVANCSS**
- **Maven JAXB**
- **JUNITPP**
- **Kodo**
- **Maven Macker**
- **SDocBook**
- **Sourceforge**
- **Maven SpringGraph**
- **RPM Plugin**
- **Runtime Builder**
- **Strutsdoc**
- **Tasks**
- **Maven Transform**
- **Maven UberDist**
- **Maven Vignette**
- **WebSphere 4.0**
- **WebSphere 5 (5.0/5.1)**
- **Maven WebLogic**
- **Canoo WebTest**
- **Wiki**
- **Word to HTML**
- **XML Resume**
- **Maven DotUml**
- **Middlegen**
- **Maven News**

Archetypes

- For reuse, create archetypes that work as project templates with build settings, etc
- An archetype is a project, with its own pom.xml
- An archetype has a descriptor called archetype.xml
- Allows easy generation of Maven projects

Good things about Maven

- Standardization
- Reuse
- Dependency management
- Build lifecycle management
- Large existing repository
- IDE aware
- One directory layout
- A single way to define dependencies
- Setting up a project is really fast
- Transitive dependencies
- Common build structure
- Use of remote repository
- Web site generation
- Build best practices enforcement
- Automated build of application
- Works well with distributed teams
- All artifacts are versioned and are stored in a repository
- Build process is standardized for all projects
- A lot of goals are available
- It provides quality project information with generated site
- Easy to learn and use
- Makes the build process much easier at the project level
- Promotes modular design of code

References

- Maven Home
  http://maven.apache.org/
- Maven Getting Started Guide
  http://maven.apache.org/guides/getting-started/index.html
- Steps for creating a Maven-based Website
  http://www.javaworld.com/javaworld/jw-02-2006/jw-0227-maven_p.html
/
- Maven Integration for Eclipse
  http://m2eclipse.codehaus.org/

Relationship with PDL (your project)

Impacts

- Use/experiment with a subset of these tools
  - IDE in general (Eclipse, IntelliJ, etc.) and all services…
  - Debugging
  - Refactoring
  - Testing
  - Documentation
  - Maven
  - Versioning systems
- You will have to in your professional career!

Web development


https://www.google.fr
+Mathieu
Recherche
Images
Maps
Play
YouTube
Actualités
Gmail
Drive
Agenda
Plus
Google
France
Recherche Google
J'ai de la chance

Google web

Web Images Maps Shopping Plus Outils de recherche

Environ 14 990 000 000 résultats (0,20 secondes)

World Wide Web - Wikipédia
fr.wikipedia.org/wiki/World_Wide_Web
Le World Wide **Web** (WWW), littéralement la « toile (d'araignée) mondiale », communément appelé le **web**, le **Web**, et parfois la toile, est un système hypertexte ...
Terminologie - Architecture - Types de ressource - Conception

WEB.DE - E-Mail-Adresse kostenlos, FreeMail, Nachrichten & Services
web.de/ - Traduire cette page
Das beliebteste Internetportal Deutschlands mit Angeboten rund um Suche, Kommunikation (E-Mail, De-Mail & mehr), Information und Services.

Définition > Internet - Web
www.futura-sciences.com/fr/definition/t/.../internet_3983/
Définition : Internet - Internet est un réseau informatique mondial constitué d'un ensemble de réseaux nationaux, régionaux et privés. L'ensemble utilise un ...

Web - 20 Minutes
www.20minutes.fr/**web**/
15 éléments – Sauter aux éditions locales; Sauter à la navigation; Sauter au ...
Google lance son «graph du savoir» en France Hier à 7h11 Facebook ...
Facebook: Avoir beaucoup d'amis génère du stress Mardi à 15h46 le 26 ...

WebGirondins.com - le site des supporters des Girondins de ...
www.**web**girondins.com/
Site entièrement consacré au club du FC Girondins de Bordeaux. Toute l'actualité par et pour les supporters.

ARTE Live Web

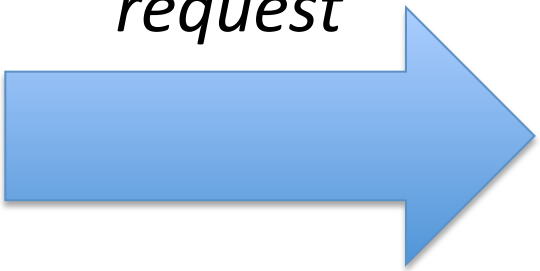
liveweb.arte.tv/fr
29 mai 2009
Présenté à la Biennale de la Danse de Lyon, au Musée du Quai Branly, et sur ARTE Live **Web**, le Lac des Cygnes ...

Autres vidéos pour **web** »

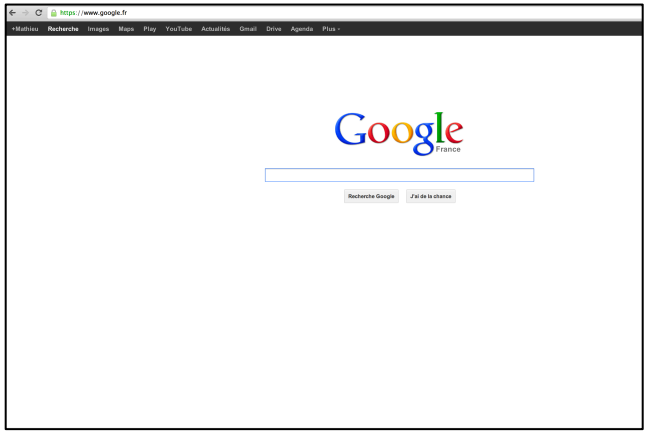
Paris Web – Accueil
www.paris-**web**.fr/
Paris **Web**, la conférence francophone des gens qui font le **web**, explore les thèmes de l'accessibilité **Web**, du design numérique et des standards ouverts.

Web browser

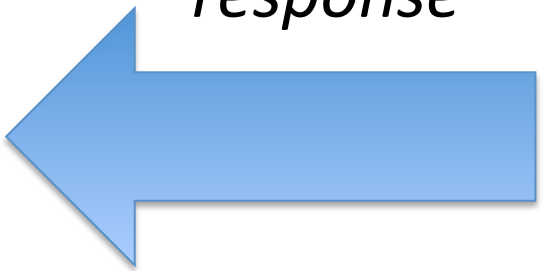
request



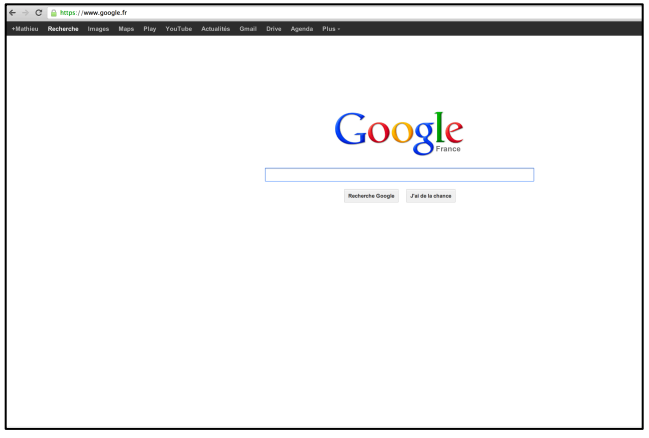
HTTP server



response



Web browser



request

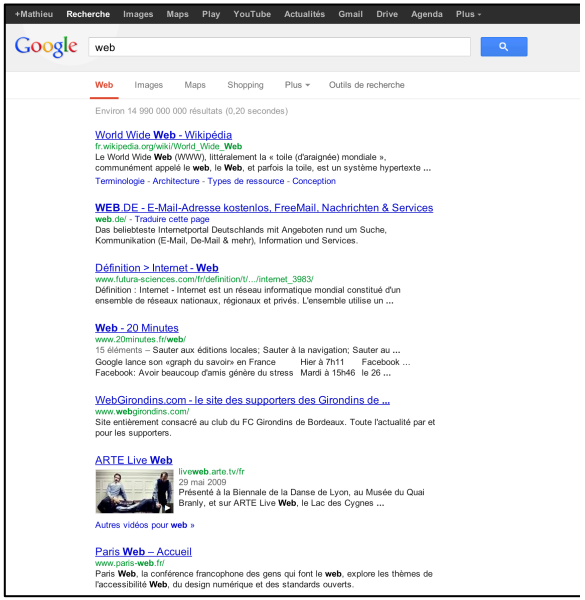
response

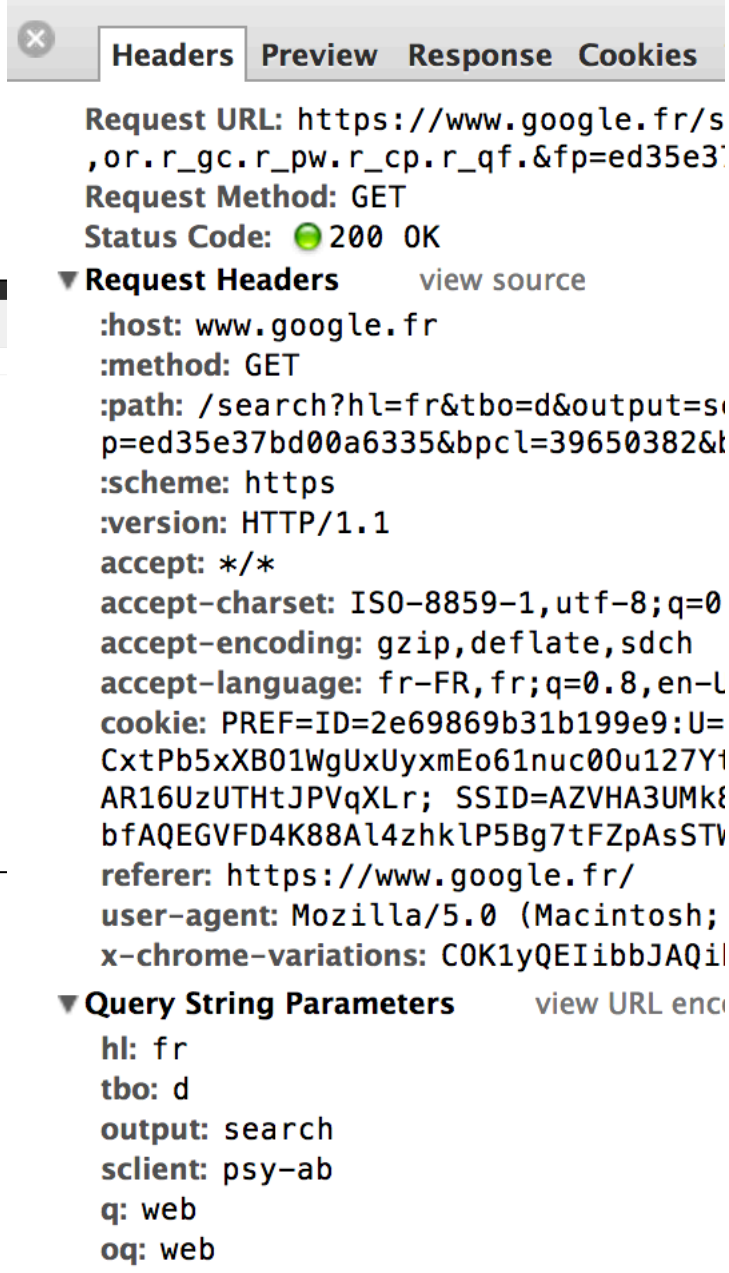
Request URL: https://www.google.fr/
Request Method: GET
Status Code: 200 OK
Request Headers view source
:host: www.google.fr
:method: GET
:path: /
:scheme: https
:version: HTTP/1.1
accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
accept-charset: ISO-8859-1,utf-8;q=0.7,*;q=0.3
accept-encoding: gzip,deflate,sdch
accept-language: fr-FR,fr;q=0.8,en-US;q=0.6,en;q=0.4
cookie: PREF=ID=2e69869b31b199e9:U=b1fcad55eed1904b:FF=0:LD=fr:TM=1339164992:
CxtPb5xXBO1WgUxUyxmEo61nuc0Ou127YtQK7YYOPh-qm62ys5084UXT7NGSUy3xO_EWnF0LH9w
AR16UzUTHtJPVqXLr; SSID=AZVHA3UMk87RYchN6; APISID=1iFaN2NJgQOzCeZL/AOPySPpw
bfAQEGVFD4K88Al4zhklP5Bg7tFZpAsSTW-jY4SpVzy90DJnrzUQKCyMB-7FIqHzxDILxgOl_iQ
user-agent: Mozilla/5.0 (Macintosh; Intel Mac OS X 10_8_2) AppleWebKit/537.11
x-chrome-variations: COK1yQEIibbJAQibtskBCKW2yQEIp7bJAQiqtskBCLe2yQEIu4PKAQ==

Response Headers view source
cache-control: private, max-age=0
content-encoding: gzip
content-type: text/html; charset=UTF-8
date: Thu, 06 Dec 2012 04:55:00 GMT
expires: -1
server: gws
status: 200 OK
version: HTTP/1.1
x-frame-options: SAMEORIGIN
x-xss-protection: 1; mode=block

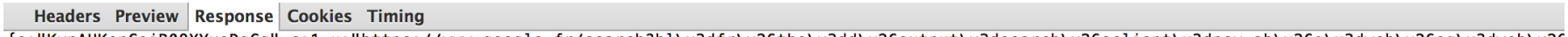
HTTP server

Web browser





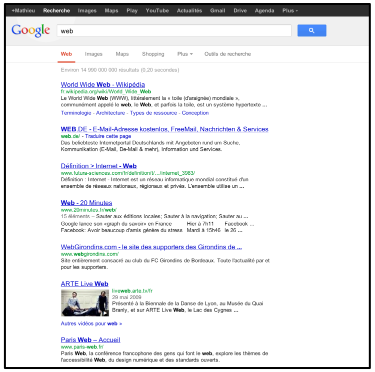
HTTP server



request

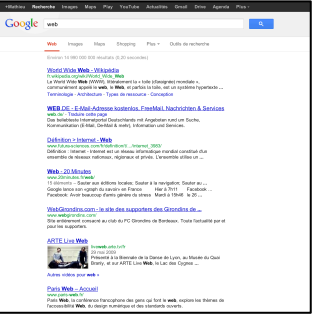
HTTP server

Web browser



Name Path	Method	Status Text	Type	Initiator	Size Content	Time Latency
s	GET	200 OK	application/json	/xjs/_/js/s/c,sb,wta,cr,cdos,j Script	0B 2.12KB	65ms 65ms
s	GET	200 OK	application/json	/xjs/_/js/s/c,sb,wta,cr,cdos,j Script	0B 121.77KB	355ms 63ms
gen_204	GET	204 No Content	text/html	/:3 Script	241B 0B	58ms 57ms
s	GET	200 OK	application/json	/xjs/_/js/s/c,sb,wta,cr,cdos,j Script	0B 1.36KB	63ms 62ms
data:image/gif;base...	GET	Success	image/gif	/xjs/_/js/s/c,sb,wta,cr,cdos,j Script	(from cache)	0ms 0ms
data:image/gif;base...	GET	Success	image/gif	/xjs/_/js/s/c,sb,wta,cr,cdos,j Script	(from cache)	0ms 0ms
data:image/gif;base...	GET	Success	image/gif	/xjs/_/js/s/c,sb,wta,cr,cdos,j Script	(from cache)	0ms 0ms
data:image/gif;base...	GET	Success	image/gif	/xjs/_/js/s/c,sb,wta,cr,cdos,j Script	(from cache)	0ms 0ms
data:image/gif;base...	GET	Success	image/gif	/xjs/_/js/s/c,sb,wta,cr,cdos,j Script	(from cache)	0ms 0ms
data:image/gif;base...	GET	Success	image/gif	/xjs/_/js/s/c,sb,wta,cr,cdos,j Script	(from cache)	0ms 0ms
data:image/jpeg;bas...	GET	Success	image/jpeg	/#hl=fr&tbo=d&output=searc Script	0B 1.67KB	0ms 0
data:image/jpeg;bas...	GET	Success	image/jpeg	/#hl=fr&tbo=d&output=searc Script	0B 1.94KB	0ms 0
data:image/jpeg;bas...	GET	Success	image/jpeg	/#hl=fr&tbo=d&output=searc Script	0B 2.20KB	0ms 0
data:image/jpeg;bas...	GET	Success	image/jpeg	/#hl=fr&tbo=d&output=searc Script	0B 2.88KB	0ms 0
data:image/jpeg;bas...	GET	Success	image/jpeg	/#hl=fr&tbo=d&output=searc Script	0B 2.05KB	0ms 0

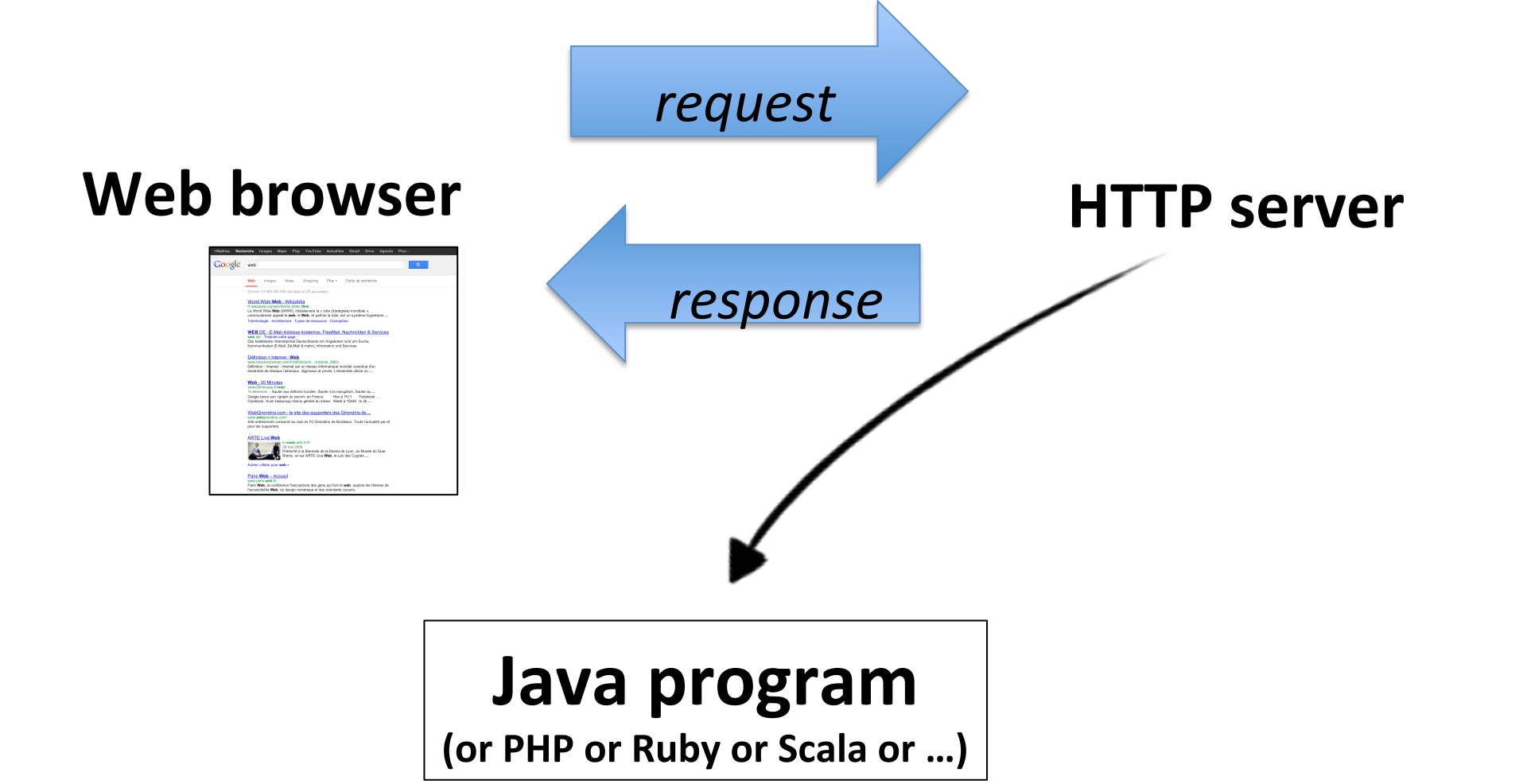
response



```
<!DOCTYPE html>
<html itemscope="itemscope" itemtype="http://schema.org/WebPage">
  <head>…</head>
  <body onload="try{if(!google.j.b){document.f&&document.f.q.focus();document.gbqf&&document.gbqf.q.focus();}}catch(e){}if(document.images)new Image().src='/images/nav_logo114.png'" dir="ltr" alink="#dd4b39" bgcolor="#fff" id="gsr" link="#12c" text="#222" vlink="#61c" style class="tbo vsh">
    <div id="pocs" style="position: absolute; z-index: 986; left: 126px; top: 79px; display: none;" class="sft">…</div>
    <div id="cst" style="display: none;">…</div>
    <a href="/setprefs?prev=https://www.google.fr/&sig=0_6x3nmUugcPvrsYlGwHkIz4c1HUY%3D&suggon=2" style="left:-1000em;position:absolute">…</a>
    <textarea name="csi" id="csi" style="display:none"></textarea>
    <script>if(google.j.b)document.body.style.visibility='hidden';</script>
    <div id="mngb">…</div>
    <iframe src="/blank.html" onload="google.j.l()" onerror="google.j.e()" name="wgjf" style="display:none">…</iframe>
    <textarea name="wgjc" id="wgjc" style="display:none"></textarea>
    <textarea name="wwcache" id="wwcache" style="display:none"></textarea>
    <textarea name="hcache" id="hcache" style="display:none"></textarea>
    <div id="main" style>
      <div id="ignore" style></div>
      <div>
        <div id="cnt" class>
          <script>…</script>
          <div class="mw">…</div>
          <div id="bst" style="display: none;">…</div>
          <div id="top_nav" style>…</div>
          <div id="appbar" style>…</div>
          <div class="mw" id="ucs" style></div>
          <div class="mw">
            <div id="rcnt" style="clear:both;position:relative;zoom:1">…</div>
            <div class="tsf-p" id="foot" role="contentinfo" style>…</div>
            <div id="bfoot" style>…</div>
          </div>
          <div id="xfoot" style>…</div>
          <div id="lfoot" style></div>
        </div>
      </div>
    </div>
    <script>…</script>
    <script data-url="/extern_chrome/ed35e37bd00a6335.js" id="ecs">…</script>
    <div id="xjsd">…</div>
    <div id="xjsi">…</div>
    <script>…</script>
    <script src="/xjs/_/js/s/sy8,qf,lor/rt=j/ver=ml8H-8_903g.en_US./d=0/sv=1/rs=AItRST0536Ewq15GFZZEVzACJOaXm-0N9w"></script>
    <table cellspacing="0" cellpadding="0" style="width: 572px; top: 78px; position: absolute; text-align: left; display: none; left: 126px;" class="gstl_0 gssb_c" dir="ltr">…</table>
    <script src="//ssl.gstatic.com/gb/js/smm_a79c05aaa286c9a4782660d65a6dd540.js" async></script>
    <script src="//ssl.gstatic.com/gb/js/abc/gci_91f30755d6a6b787dcc2a4062e6e9824.js" async gapi_processed="true"></script>
    <style type="text/css">…</style>
```

```
<head>
  <meta itemprop="image" content="/images/google_favicon_128.png">
  <title>web 2.0 - Recherche Google</title>
  <script src="https://apis.google.com/_/abc-static/_/js/gapi/googleapis_client,plusone/rt=j/ver=_CZX0pAF71I.en./sv=1/am=!7O-CuSNjNcIpPOou/d=1/cb=gapi.loaded_0" async></script>
  <script>
    (function(){
    window.google={kEI:"TCrAULeJLYi40QXqjYG4Cg",getEI:function(a){for(var b;a&&(!a.getAttribute||!(b=a.getAttribute("eid")));)a=a.parentNode;return b||google.kEI},https:function(){return"h
    b,c,j){var d=new Image,f=google.lc,e=google.li,g="";d.onerror=d.onload=d.onabort=function(){delete f[e];f[e]=!c&&-1==b.search("&ei=")&&(g="&ei="+google.getEI(j));c=c||"/gen_204?atyp=
    pm:"p",pl:[],mc:0,sc:0.5,u:"1cfeb9e1"},Toolbelt:{},y:{},x:function(a,b){google.y[a.id]=[a,b];return!1}};
    window.onpopstate=function(){google.j.psc=1};for(var h="ad api bc is p pa ac pc pah ph sa sifp slp spf spn x z zz".split(" "),i=0,k;k=h[i++];)(function(a){google.j[a]=function(){goog
    window.chrome||(window.chrome={}),window.chrome.sv=2.00,window.chrome.searchBox||(window.chrome.searchBox=[]),window.chrome.searchBox.onsubmit=function(){google.x({id:"psyapi"},functio
    window.google.sn="webhp";window.google.timers={};window.google.startTick=function(a,b){window.google.time[a]={t:{start:(new Date).getTime()},bfr:!(!b)}};window.google.tick=function(a
    (function(){'use strict';var d=null,j=this;var l="undefined"!=typeof navigator&&/Macintosh/.test(navigator.userAgent);var q="undefined"!=typeof navigator&&/iPhone|iPad|iPod/.test(naviga
  </script>
  <style>…</style>
  <style id="gstyle">…</style>
  <style>…</style>
  <script>…</script>
  <style type="text/css">…</style>
</head>
```

HTML + Javascript + CSS



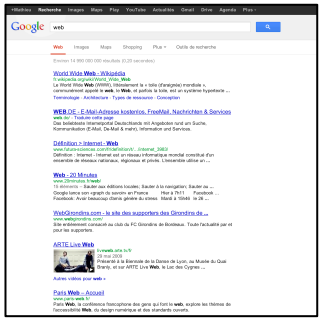
~ treat requests and user inputs (e.g., input forms)
~ generate HTML/JS/CSS stuff accordingly

Can be highly complex and require a combination of technology (user session, data storage, business logic, user interface concerns, performance, reliability, etc.)

request

response

Web browser



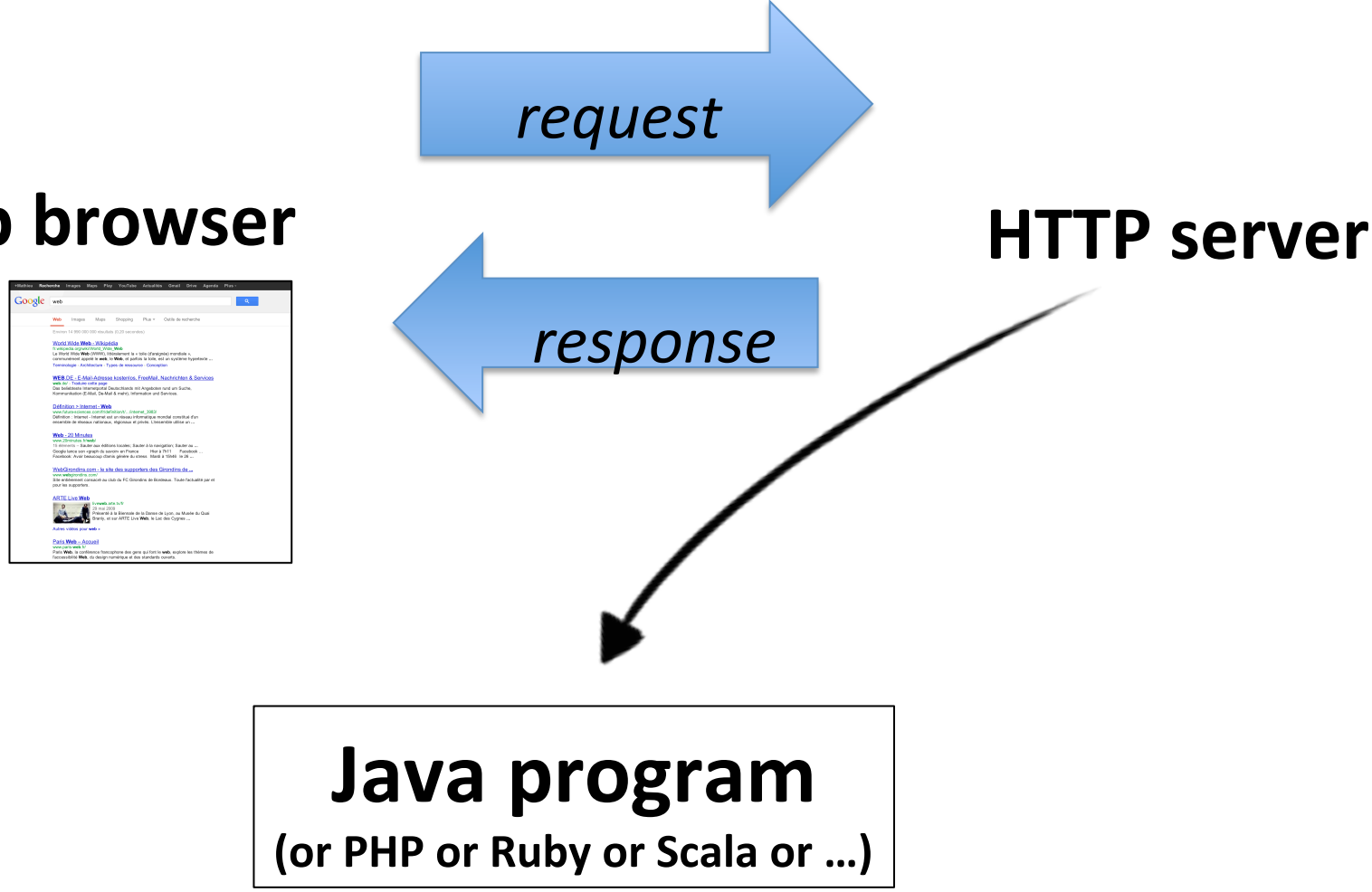
HTTP server

Java program

(or PHP or Ruby or Scala or …)

JEE (Servlets+JSP)

Play! framework



Disclaimer: this is just an overview with a quick focus on two technologies

#1 You can consider other technologies as well
(the principles are likely to be the same)

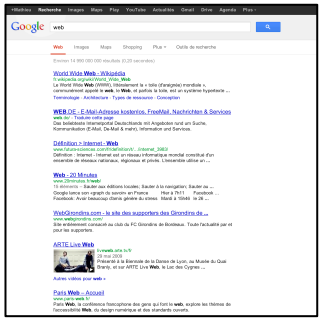
#2 You will have to learn and practice more by yourself
(available for any questions)

JEE (Servlets + JSP)

request

response

Web browser



HTTP server

Java servlet

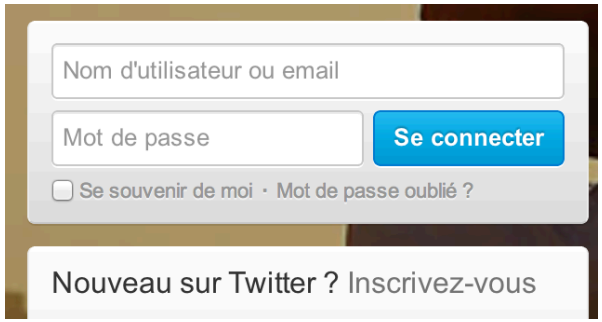
(and Java ecosystem actually)

Java Servlet

- Java program
  - rely on Java ecosystem
- Receive and treat HTTP requests
  - inputs (e.g., input forms)
  - resources (e.g., images)
- Generate web pages “on the fly”
  - Response to “clients” (browser)
  - HTML code + (JavaScript + CSS)

Web browser

HTTP server



request

response

Java servlet

```
<form action="https://twitter.com/sessions" class="signin" method="post">
  <div class="placeholding-input username">
    <input type="text" id="signin-email" class="text-input email-input" name="session[username_or_email]" title="Nom d'utilisateur ou email" autocomplete="on" tabindex="1">
    <label for="signin-email" class="placeholder">Nom d'utilisateur ou email</label>
  </div>
  <table class="flex-table password-signin">…</table>
  <div class="remember-forgot">…</div>
  <input type="hidden" name="return_to_ssl" value="true">
  <input type="hidden" name="scribe_log">
  <input type="hidden" name="redirect_after_login" value="/">
  <input type="hidden" value="6af33df72c59cf3ace4375a2dc7d2016cb96d726" name="authenticity_token">
</form>
```

Treat input forms

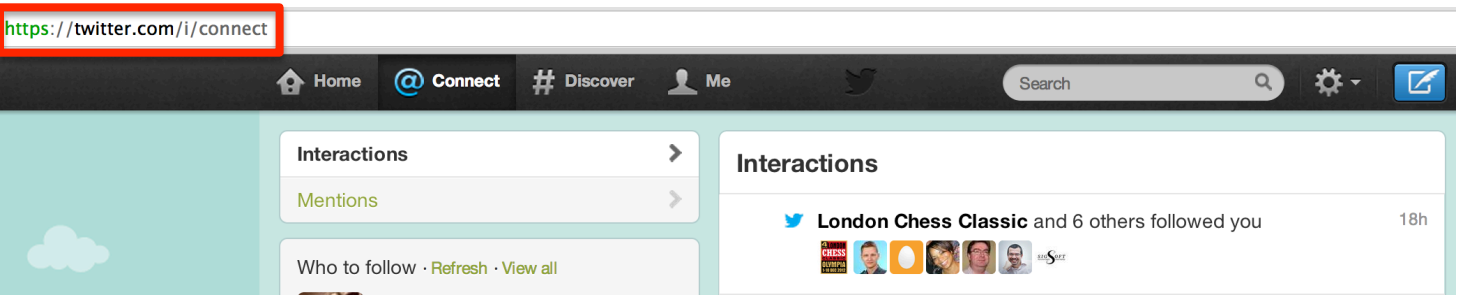
Identify/Retrieve User (if everything goes fine)

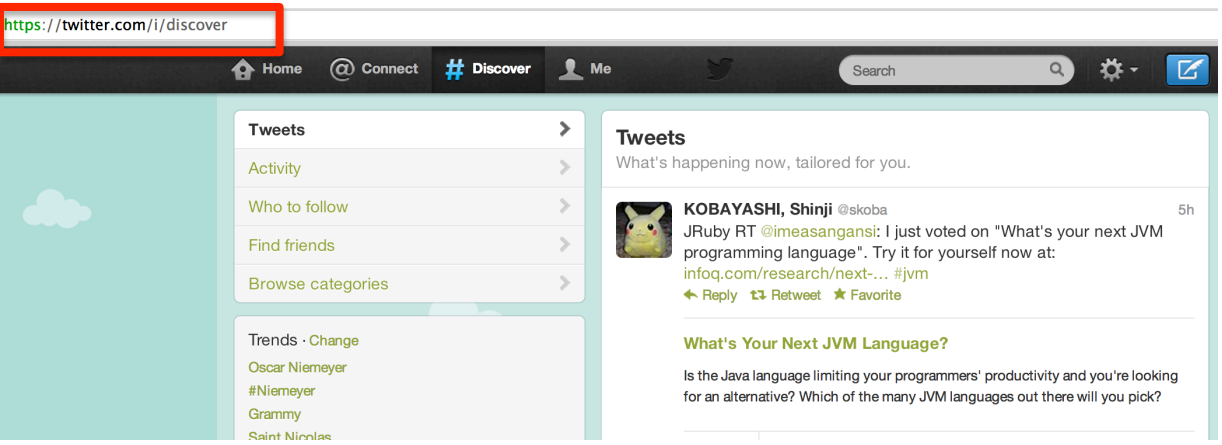
Find associated Followers/tweets/ads

Generate HTML stuff

The server responds according to the **URL** of the request (« **context** »)

Different applications/processings according to different contexts

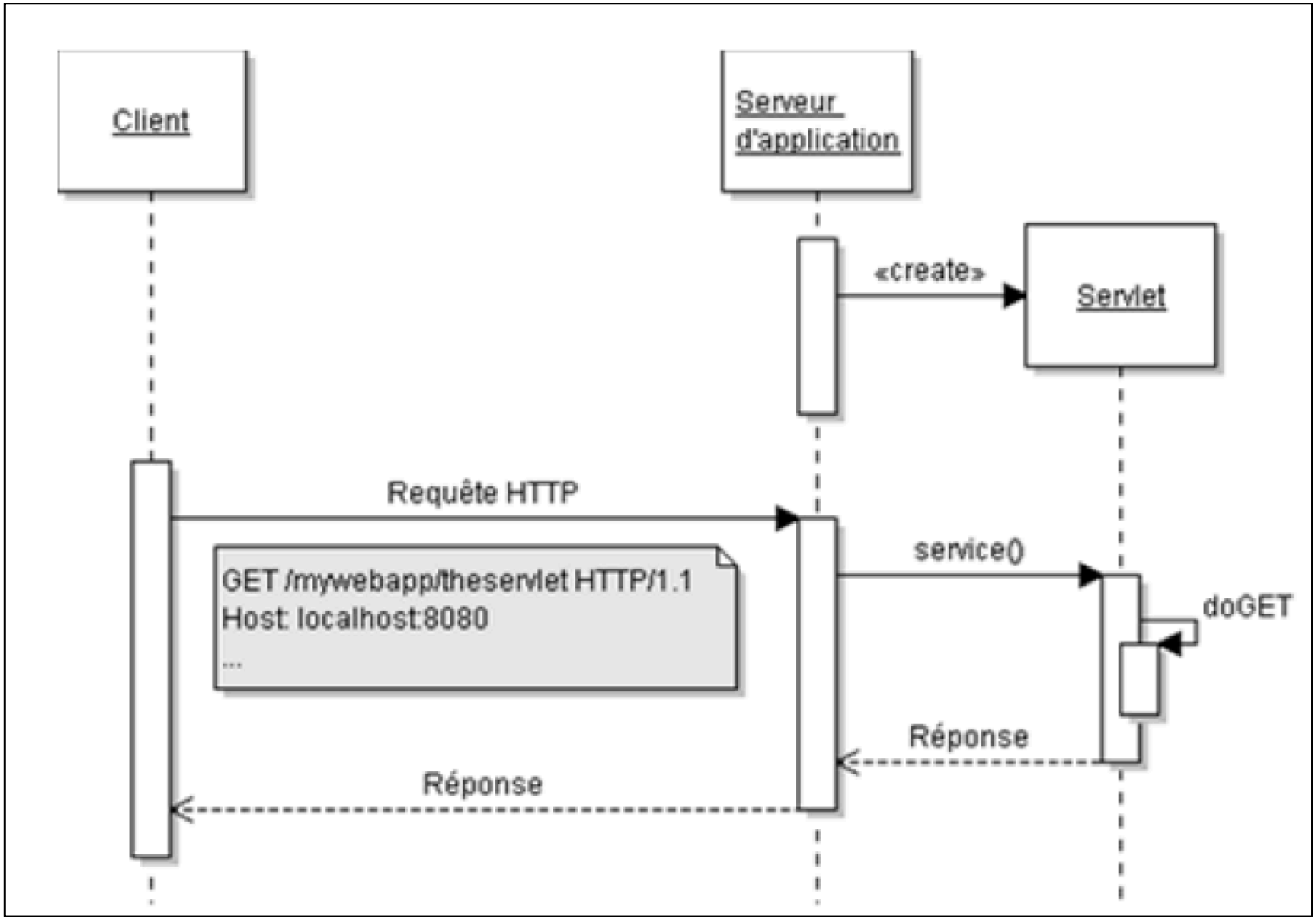




HTTP server

Java servlets

Client, Server and Java Servlet



Java Servlet (Hello World)

```
1  import java.io.*;
2  import javax.servlet.*;
3  import javax.servlet.http.*;
4
5  public class HelloWorld extends HttpServlet {
6
7    public void doGet (HttpServletRequest request,
8    HttpServletResponse response)
9    throws ServletException, IOException {
10
11   response.setContentType("text/html");
12   PrintWriter pw = response.getWriter();
13   pw.println("<html>");
14   pw.println("<head><title>Hello World</title></title>");
15   pw.println("<body>");
16   pw.println("<h1>Hello World</h1>");
17   pw.println("</body></html>");
18
19   }
20 }
```

Java Servlet (zoom on “doGet”)

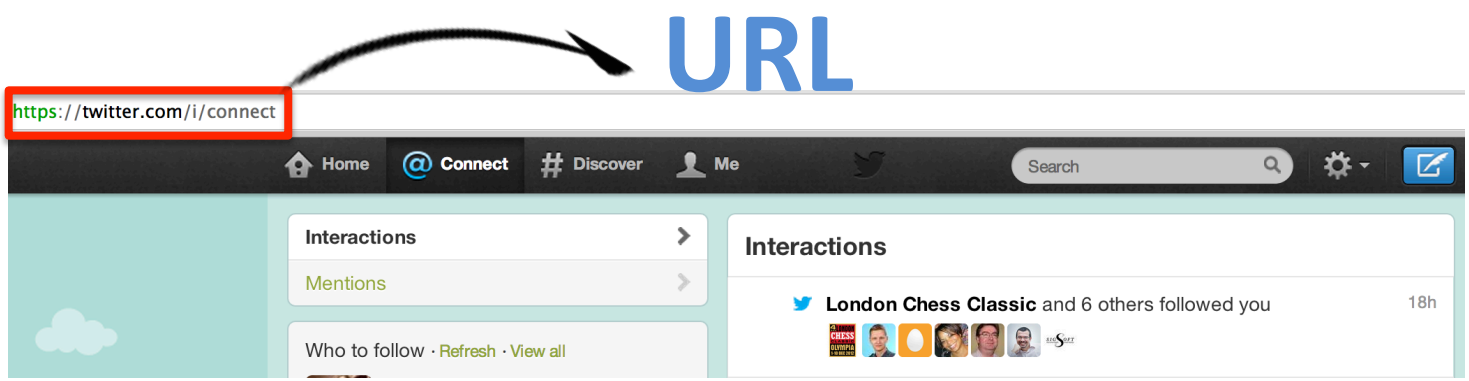
```
1  public void doGet(HttpServletRequest request,
2  HttpServletResponse response)
3  throws ServletException, IOException {
4  // use "request" for reading parameters
5  // and headers of HTTP request
6  ...
7  // treat the request
8  ...
9  // use response to specify the response status
10 // (including headers of the response)
11 ...
12 PrintWriter out = response.getWriter();
13 // send the content of the response
14 ...
15 }
```

Servlet Container

- HTTP server does not know how to execute the Java code of a servlet (obvious)
  - HTTP requests, that's it
- A servlet container is needed and manages a set of servlets
  - Management of servlet names (class names)
  - Creation and initialization of servlets
  - Deletion of servlets
- HTTP server delegates the requests to the container

HTTP server

Java
servlets



- Client (browser) won't specify a direct reference to a Servlet but rather an URL
- The web application should establish a correspondence between an URL and a servlet
  - **Mapping** URL-Servlet
- The corresponding container of the servlet will execute the servlet

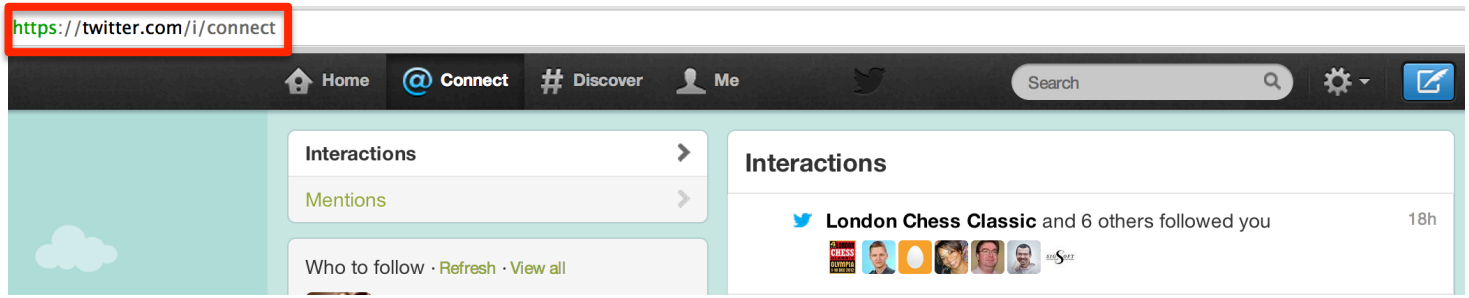
Mapping URL-Servlet

- Two solutions
  - Java annotations
  - web.xml (configuration file)

Mapping URL-Servlet

- **Java annotations** (since Servlet 3.0)

```
1  @WebServlet(urlPatterns={"/connect"})
2
3  public class Servlet1 extends HttpServlet {
4  ...
5  }
```



Mapping URL-Servlet

- **web.xml** (configuration file, since the beginning)
  - Included in the WAR archive that packages all code and resources of the applications

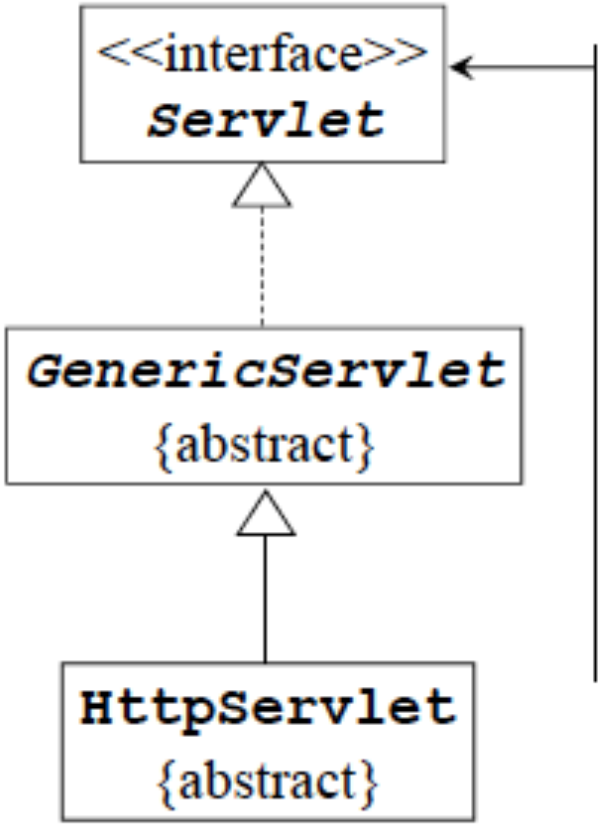
```xml
<?xml version="1.0" encoding="ISO-8859-1" ?>

<web-app xmlns="http://java.sun.com/xml/ns/j2ee"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation="http://java.sun.com/xml/ns/j2ee http://java.sun.com/xml/ns/j2ee/web-app_2_4.xsd"
    version="2.4">

    <display-name>HelloWorld Application</display-name>
    <description>
        This is a simple web application with a source code organization
        based on the recommendations of the Application Developer's Guide.
    </description>

    <servlet>
        <servlet-name>HelloServlet</servlet-name>
        <servlet-class>examples.Hello</servlet-class>
    </servlet>

    <servlet-mapping>
        <servlet-name>HelloServlet</servlet-name>
        <url-pattern>/hello</url-pattern>
    </servlet-mapping>

</web-app>
```

Mapping URL-Servlet

- **web.xml** (configuration file, since the beginning)
- Joker can be used
  - <url-pattern>/users/*</url-pattern>

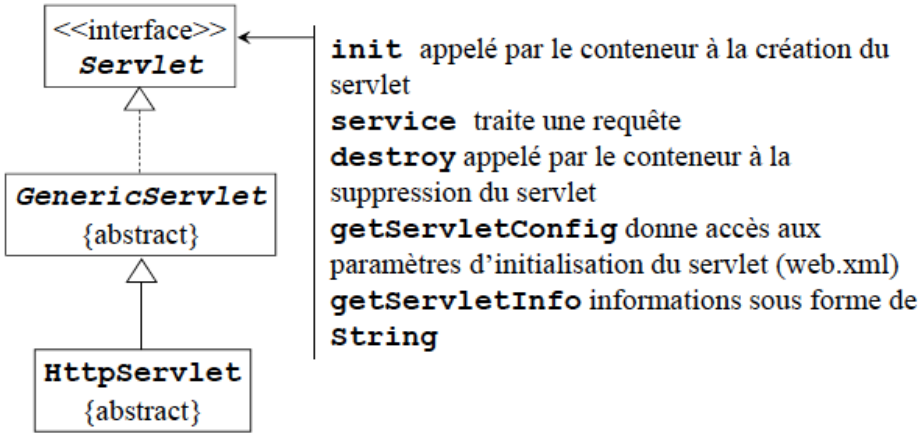
```xml
<?xml version="1.0" encoding="ISO-8859-1" ?>

<web-app xmlns="http://java.sun.com/xml/ns/j2ee"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation="http://java.sun.com/xml/ns/j2ee http://java.sun.com/xml/ns/j2ee/web-app_2_4.xsd"
    version="2.4">

    <display-name>HelloWorld Application</display-name>
    <description>
        This is a simple web application with a source code organization
        based on the recommendations of the Application Developer's Guide.
    </description>

    <servlet>
        <servlet-name>HelloServlet</servlet-name>
        <servlet-class>examples.Hello</servlet-class>
    </servlet>

    <servlet-mapping>
        <servlet-name>HelloServlet</servlet-name>
        <url-pattern>/hello</url-pattern>
    </servlet-mapping>

</web-app>
```

HttpServlet (1)



init appelé par le conteneur à la création du servlet
service traite une requête
destroy appelé par le conteneur à la suppression du servlet
getServletConfig donne accès aux paramètres d'initialisation du servlet (web.xml)
getServletInfo informations sous forme de **String**

HttpServlet (2)



- service() method of HttpServlet delegates the treatment to another method, depending on the kinf of HTTP request sent by the client
  - doGet(), doPost(), doPut(), doDelete(), doHead(), doOptions(), doTrace()

Writing a Servlet

- Boil down to…
- Writing a class that inherits from HttpServlet
- Override at least one method (doGet, doPost, etc.)
- Eventually override init or destroy if the servlet manages a set of resources that need to be initialized or removed

```
1  import java.io.*;
2  import javax.servlet.*;
3  import javax.servlet.http.*;
4
5  public class HelloWorld extends HttpServlet {
6
7    public void doGet (HttpServletRequest request,
8    HttpServletResponse response)
9    throws ServletException, IOException {
10
11   response.setContentType("text/html");
12   PrintWriter pw = response.getWriter();
13   pw.println("<html>");
14   pw.println("<head><title>Hello World</title></title>");
15   pw.println("<body>");
16   pw.println("<h1>Hello World</h1>");
17   pw.println("</body></html>");
18
19   }
20 }
```

Writing a Servlet

- Boil down to…
- Writing a class that inherits from HttpServlet
- Override at least one method (doGet, doPost, etc.)
- Eventually override init or destroy if the servlet manages a set of resources that need to be initialized or removed

```
1  import java.io.*;
2  import javax.servlet.*;
3  import javax.servlet.http.*;
4
5  public class HelloWorld extends HttpServlet {
6
7    public void doGet (HttpServletRequest request,
8    HttpServletResponse response)
9    throws ServletException, IOException {
10
11   response.setContentType("text/html");
12   PrintWriter pw = response.getWriter();
13   pw.println("<html>");
14   pw.println("<head><title>Hello World</title></title>");
15   pw.println("<body>");
16   pw.println("<h1>Hello World</h1>");
17   pw.println("</body></html>");
18
19   }
20 }
```

Treating input forms (1)

Servlet in action

- HTTP recalls
  - With GET (within the URL):
    - http://www.truc.com/abonnement?id=23378&nom=Toto&op=d&v=56
  - With POST
    - Parameters are directly in the HTTP request

```
▼<form action="https://twitter.com/sessions" class="signin" method="post">
  ▼<div class="placeholding-input username">
      <input type="text" id="signin-email" class="text-input email-input" name="session[username_or_email]" title="Nom d'utilisateur ou email" autocomplete="on" tabindex="1">
      <label for="signin-email" class="placeholder">Nom d'utilisateur ou email</label>
    </div>
  ▶<table class="flex-table password-signin">…</table>
  ▶<div class="remember-forgot">…</div>
    <input type="hidden" name="return_to_ssl" value="true">
    <input type="hidden" name="scribe_log">
    <input type="hidden" name="redirect_after_login" value="/">
    <input type="hidden" value="6af33df72c59cf3ace4375a2dc7d2016cb96d726" name="authenticity_token">
  </form>
```

- Methods to get the parameters are independent of the kind of request (POST or GET)
  - As a result doGet can call doPost (or the other way)

Treating input forms (2)

Servlet in action

```
1  public void doGet(HttpServletRequest request,
2  HttpServletResponse response)
3  throws ServletException, IOException {
4  // use "request" for reading parameters
5  // and headers of HTTP request
6  ...
7  // treat the request
8  ...
9  // use response to specify the response status
10 // (including headers of the response)
11 ...
12 PrintWriter out = response.getWriter();
13 // send the content of the response
14 ...
15 }
```

- **HttpServletRequest**
  - Interface, represents a request of the client
- Several methods are provided
  - String getParameter (String nomParam)
  - String[] getParameterValues()
    - e.g. multivalues
  - Enumeration<String> getParameterNames()

http://www.truc.com/abonnement?id=23378&nom=Toto&op=d&v=56

```
1 <input type="text" name="nom">
2 <input type="checkbox" name="ville"
3 value="Nice">Nice<br>
4 <input type="checkbox" name="ville"
5 value="Paris">Paris<br>
```

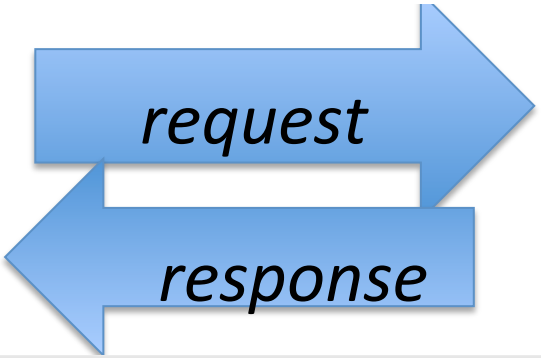
```
1 String nom = req.getParameter("nom");
2 String[] villes =
3 req.getParameterValues("ville");
```

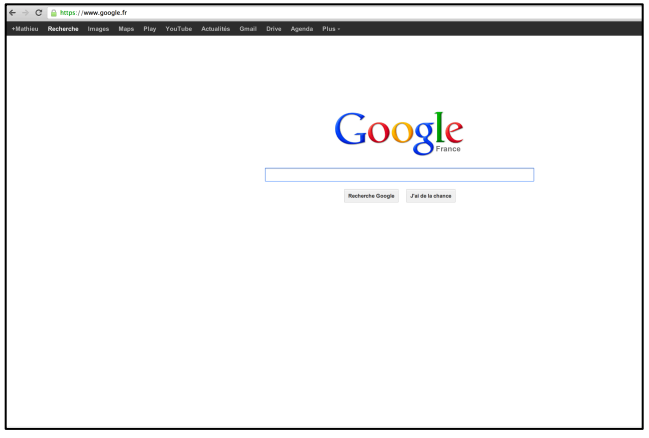
Play!

First, let us talk about REST architecture (revisiting HTTP)

(revisiting)

Web browser





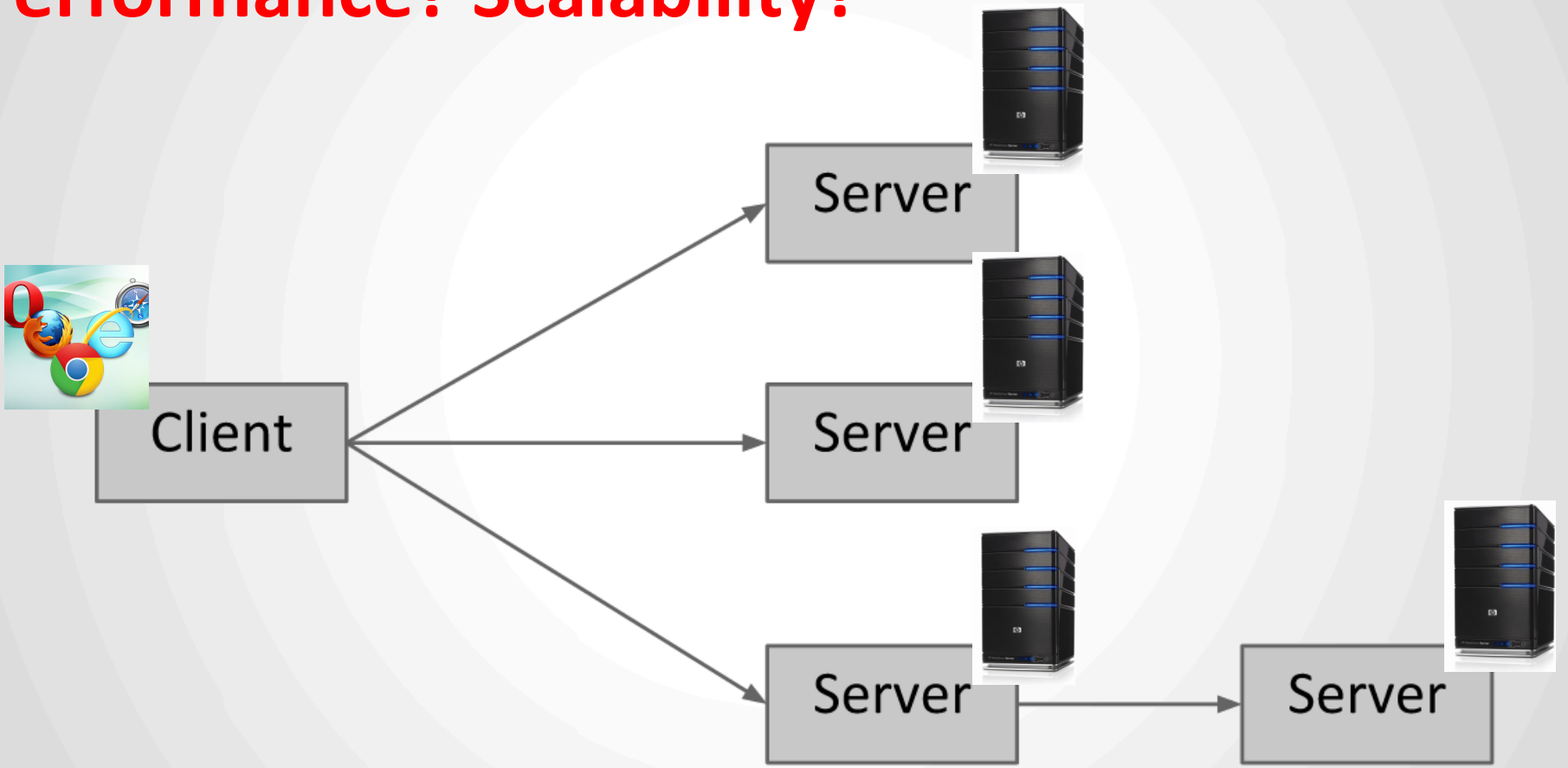
Request URL: https://www.google.fr/
Request Method: GET
Status Code: 200 OK
Request Headers view source
:host: www.google.fr
:method: GET
:path: /
:scheme: https
:version: HTTP/1.1
accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
accept-charset: ISO-8859-1,utf-8;q=0.7,*;q=0.3
accept-encoding: gzip,deflate,sdch
accept-language: fr-FR,fr;q=0.8,en-US;q=0.6,en;q=0.4
cookie: PREF=ID=2e69869b31b199e9:U=b1fcad55eed1904b:FF=0:LD=fr:TM=1339164992:
CxtPb5xXBO1WgUxUyxmEo61nuc0Ou127YtQK7YYOPh-qm62ys5084UXT7NGSUy3xO_EWnF0LH9w
AR16UzUTHtJPVqXLr; SSID=AZVHA3UMk87RYchN6; APISID=1iFaN2NJgQOzCeZL/AOPySPpw
bfAQEGVFD4K88Al4zhklP5Bg7tFZpAsSTW-jY4SpVzy90DJnrzUQKCyMB-7FIqHzxDILxgOl_iQ
user-agent: Mozilla/5.0 (Macintosh; Intel Mac OS X 10_8_2) AppleWebKit/537.11
x-chrome-variations: COK1yQEIibbJAQibtskBCKW2yQEIp7bJAQiqtskBCLe2yQEIu4PKAQ==

Response Headers view source
cache-control: private, max-age=0
content-encoding: gzip
content-type: text/html; charset=UTF-8
date: Thu, 06 Dec 2012 04:55:00 GMT
expires: -1
server: gws
status: 200 OK
version: HTTP/1.1
x-frame-options: SAMEORIGIN
x-xss-protection: 1; mode=block

HTTP server

Web Architecture

Performance? Scalability?



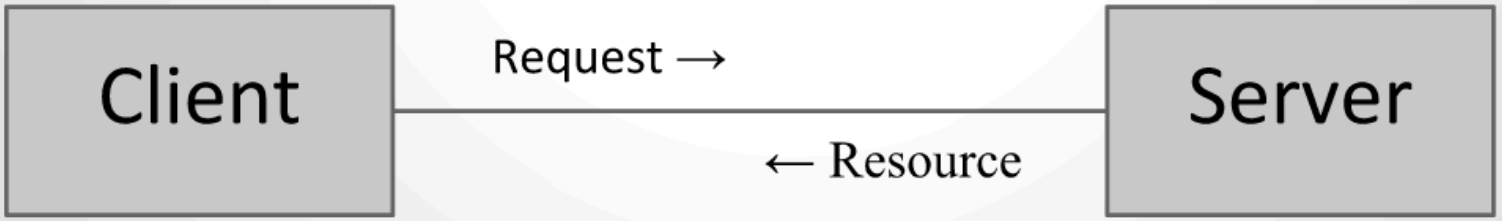
REST

Representational State Transfer

- Roy Fielding, 2000
- Architectural style for distributed systems
- Decrease coupling between Web Services
- Improve scalability
- **Stateless**

REST & HTTP (basic)

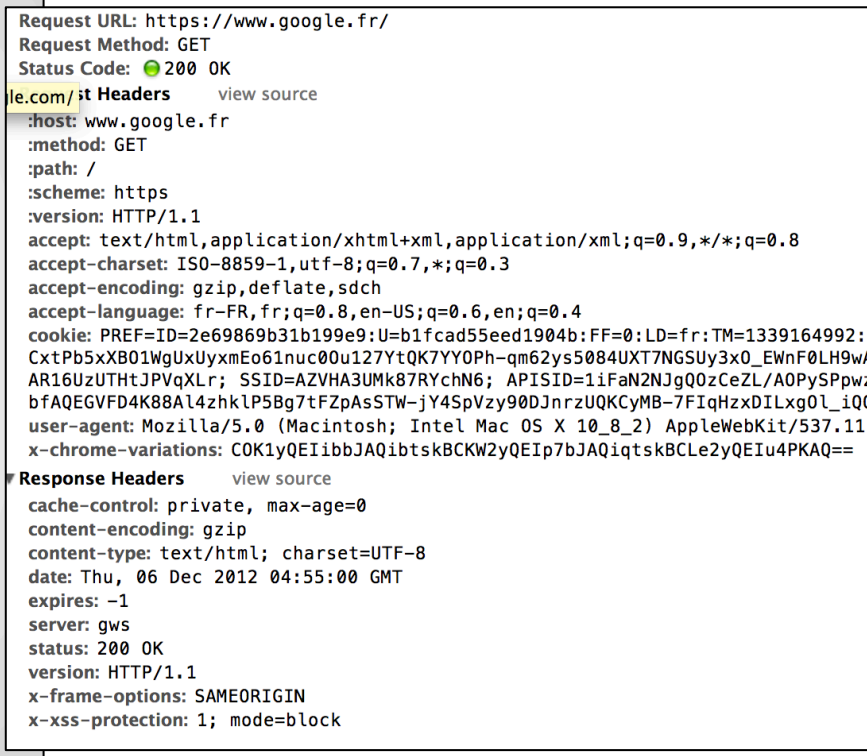
- An HTTP request is a self-descriptive message
- Resources are identified by an URL and manipulated through their representations



HTTP request (basics)

HTTP Request

- URL: `http://myapp.com/foo/bar?baz=bah`
- Verb
  - GET, get the current resource state, nullipotent
  - POST, create a new resource
  - PUT, update an existing resource, idempotent
  - DELETE, delete a resource, idempotent
- Headers
  - `Accept`, `Accept-Language`, `Accept-Encoding`
  - `Content-Type`
  - `Cookie`
  - `If-Modified-Since`
  - `User-Agent`
- Body



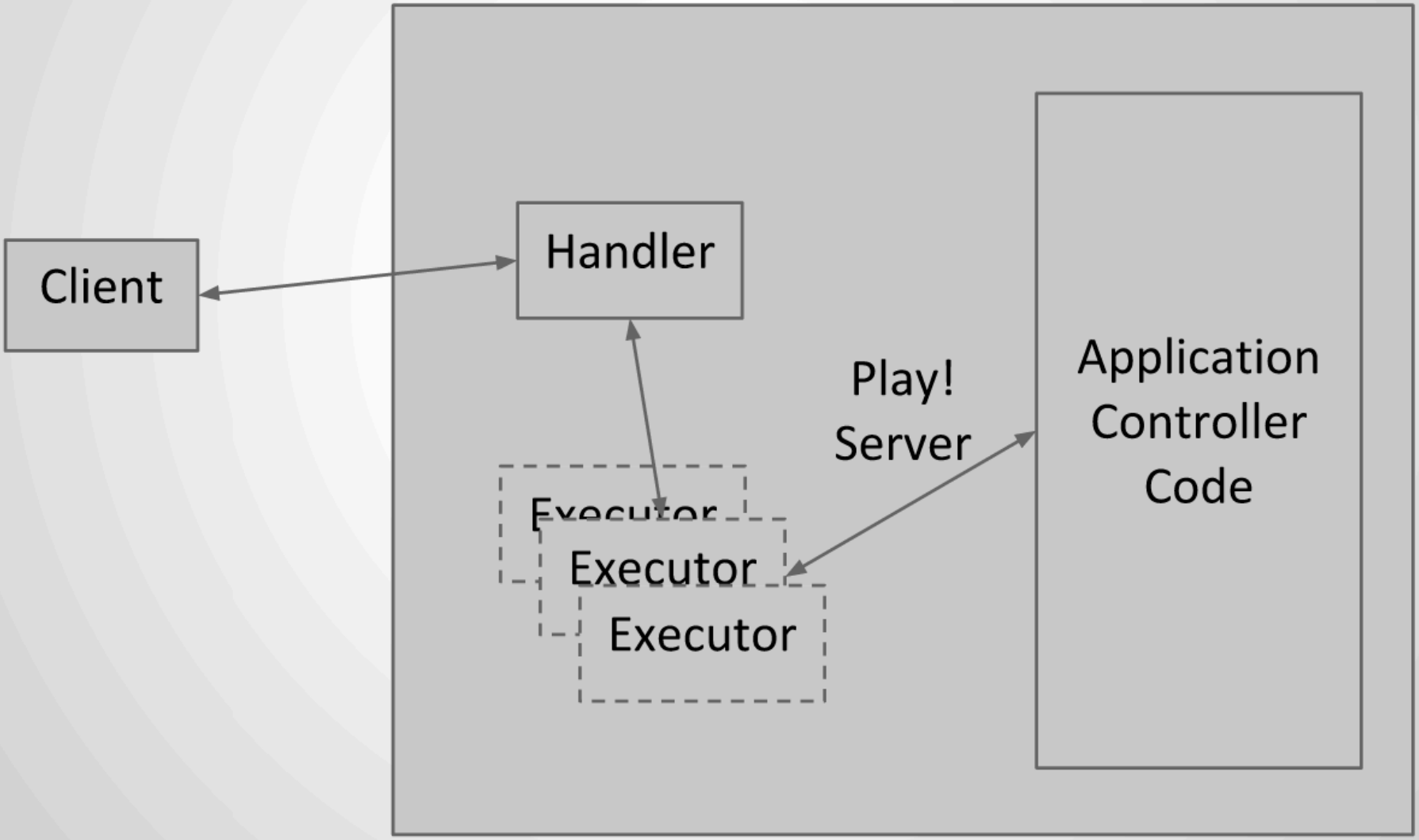
HTTP Response

- Status
  - 2xx, success (e.g. 200 OK, 206 Partial Content)
  - 3xx, redirection (e.g. 303 See Other, 304 Not Modified)
  - 4xx, client error (e.g. 404 Not Found)
  - 5xx, server error (e.g. 500 Internal Server Error, 503 Service Unavailable)
- Headers
  - Content-Type, Content-Length
  - Expires, ETag, Last-Modified, Cache-Control
  - Set-Cookie
  - Location
- Body

Play!

- **Web framework** for Java and Scala
- **Stateless** (client session stored in cookies) **REST principles**
- **Asynchronous** programming model based on Futures
- **Reactive** programming model for stream processing
- **Modern** HTTP support (WebSockets, Server-Sent Events, Chunked transfer encoding)
- Version 2.0 released in March 2012
- More than 2,000 followers and 500 forks

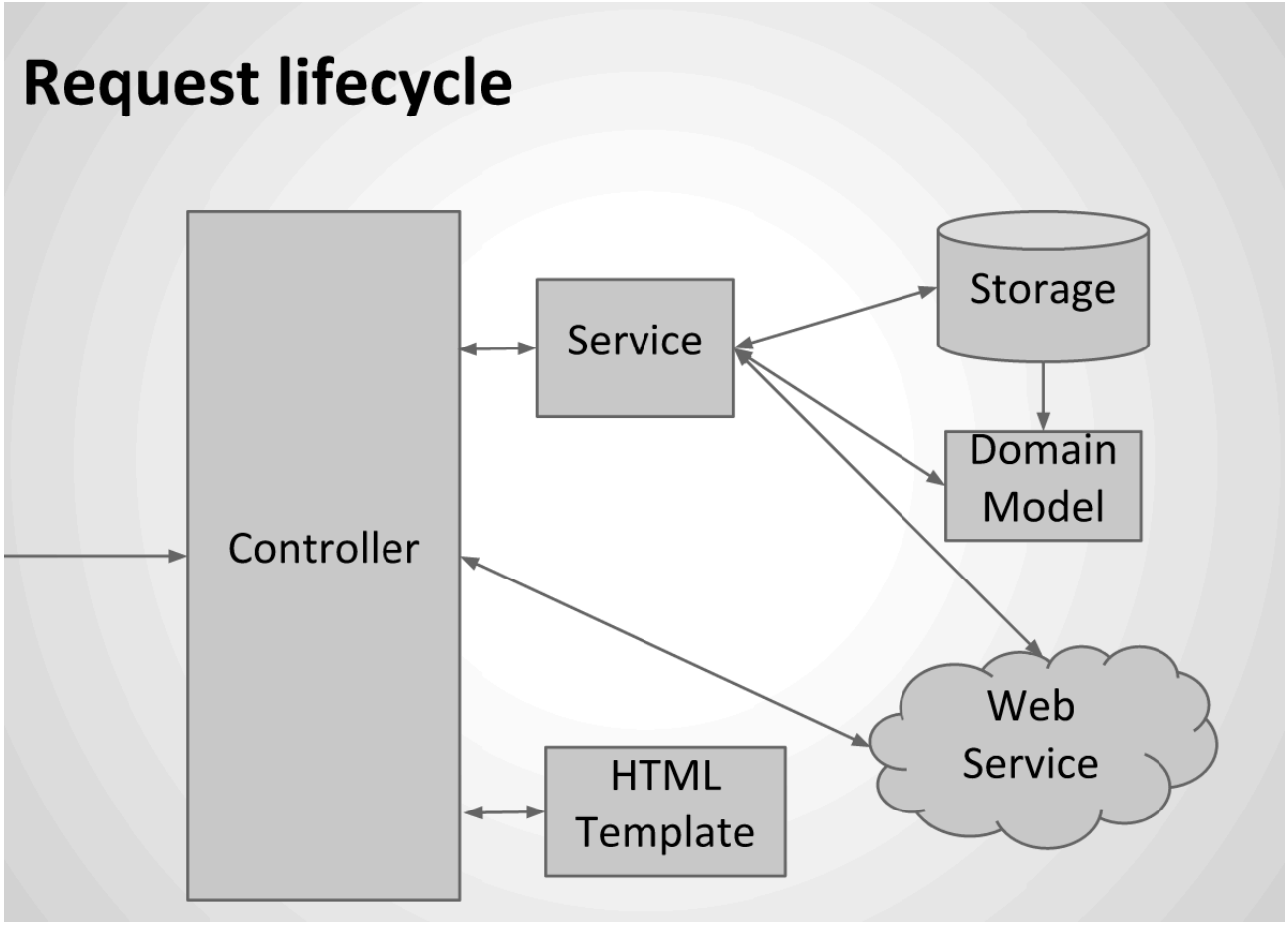
Request lifecycle



Play! framework

offers an integrated solution

- controller facilities for treating HTTP requests (controllers)
- template language/engine (views)
- specification/serialization of domain model (models)



Organization of a Play! Project (main excerpt)

```
app                   → Application sources
 └ assets             → Compiled asset sources
    └ stylesheets     → Typically LESS CSS sources
    └ javascripts     → Typically CoffeeScript sources
 └ controllers        → Application controllers
 └ models             → Application business layer
 └ views              → Templates
conf                  → Configurations files and other non-compiled resources
 └ application.conf   → Main configuration file
 └ routes             → Routes definition
public                → Public assets
 └ stylesheets        → CSS files
 └ javascripts        → Javascript files
 └ images             → Image files
```

Model
View
Controller

~ mapping URL – controller action

Example: routes definition

mapping URL – controller action

routes (file)

```
# Hello action
GET    /hello/:name          controllers.Application.hello(name)
```

controller

```
package controllers;

import play.*;
import play.mvc.*;

public class Application extends Controller {

  public static Result hello(String name) {
      return ok("Hello " + name + "!");
  }

}
```

http://localhost:9000/hello?name=World

Example: routes definition

mapping URL – controller action

```
# Home page
GET     /                           controllers.Application.index()

# Tasks
GET     /tasks                      controllers.Application.tasks()
POST    /tasks                      controllers.Application.newTask()
POST    /tasks/:id/delete           controllers.Application.deleteTask(id: Long)
```

```
public class Application extends Controller {

  public static Result index() {
    return ok(index.render("Your new application is ready."));
  }

  public static Result tasks() {
    return TODO;
  }

  public static Result newTask() {
    return TODO;
  }

  public static Result deleteTask(Long id) {
    return TODO;
  }

}
```

Controller action and result

```
package controllers;

import play.*;
import play.mvc.*;

public class Application extends Controller {

  public static Result hello(String name) {
      return ok("Hello " + name + "!");
  }

}
```

```
Result ok = ok("Hello world!");
Result notFound = notFound();
Result pageNotFound = notFound("<h1>Page not found</h1>").as("text/html");
Result badRequest = badRequest(views.html.form.render(formWithErrors));
Result oops = internalServerError("Oops");
Result anyStatus = status(488, "Strange response type");
```

Model

```
package models;

import java.util.*;
import javax.persistence.*;

import play.db.ebean.*;
import play.data.format.*;
import play.data.validation.*;

@Entity
public class Task extends Model {

  @Id
  @Constraints.Min(10)
  public Long id;

  @Constraints.Required
  public String name;

  public boolean done;

  @Formats.DateTime(pattern="dd/MM/yyyy")
  public Date dueDate = new Date();

  public static Finder<Long,Task> find = new Finder<Long,Task>(
    Long.class, Task.class
  );

}
```

Persistance
and Querying for free

Annotations define the constraints on the model data
Models can be serialized in a database (eg SQL)

Model and Controller

```java
package models;

import java.util.*;
import javax.persistence.*;

import play.db.ebean.*;
import play.data.format.*;
import play.data.validation.*;

@Entity
public class Task extends Model {

  @Id
  @Constraints.Min(10)
  public Long id;

  @Constraints.Required
  public String name;

  public boolean done;

  @Formats.DateTime(pattern="dd/MM/yyyy")
  public Date dueDate = new Date();

  public static Finder<Long,Task> find = new Finder<Long,Task>(
    Long.class, Task.class
  );

}
```

code that can be used in the controller actions

```java
// Find all tasks
List<Task> tasks = Task.find.all();

// Find a task by ID
Task anyTask = Task.find.byId(34L);

// Delete a task by ID
Task.find.ref(34L).delete();

// More complex task query
List<Task> tasks = find.where()
    .ilike("name", "%coco%")
    .orderBy("dueDate asc")
    .findPagingList(25)
    .getPage(1);
```

Template Language/Engine

Mix of HTML (+ Javascript + CSS)

and

Application code (models)

```
<div class="article">
  <span>@article.name</span>
  <span>@article.price €</span>
</div>
<ul>
  @for(article <- articles) {
    <li>@show(article)</li>
  }
</ul>
@if(isAdmin) {
  <button>Delete</button>
}
```

Much more here:

http://www.playframework.org/documentation/

start with a simple tutorial
don't try to master everything about the framework
get simple results ASAP