Code inspection

Figure 1: Politecnico di Milano

Version 1.0

- Claudio Cardinale (mat. 849760)
- Gilles Dejaegere (mat. 853950)
- Massimo Dragano (mat. 775392)

Contents

3. Massimo Dragano

Introduction

Purpose

The purpose of this document is to show all the problems found during the inspection of a small amount of code of a specific version of glashfish. The procces of inspecting source code has two main purposes. The first and most obvious one is to enhance to quality of the code and evantually identify remaining bugs. The second purpose is to improve the coding skills of the team. The inspectors improve themselves by analysing code made by others and eventually discovering coding methods that they did not know. The original authors of the code receive a list of eventuals mistakes that they could have done, wich of course help them improving themselfs.

Each group of the project has different methods assigned of a specif version of glashfish. We have to analyze these methods by checking that theyare in agreement with every point of a given checklist. We also have to find other problems, then we have to report the problems found in this document.

WRITE MORE

Scope

Glashfish is the official implementation of JEE. It is an open source project that uses svn as version system, in fact we used it to retrieve a specif version of glashfish: 64219 (of 16 Oct 2015 05:11).
This version is the version required by the assignment since we have been assigned some methods of this version to check.
Glashfish is a maven project, in fact we imported the pom file into intellij IDEA and we used it and sonar to test some check of the checklist. **KEEP OR REMOVE?**

WRITE MORE

Definitions, acronyms, abbreviations

- JEE: Java enterprise edition
- SVN: apache subversion, it is a version controller system, the successor of CVS **OR VCS?**
- CVS: Concurrent versions system, the first(/older/former/. . . ?) version controller system **gilles : I don't think it's the firts, it's a successor of SCCS**
- Context: Contextual Information, it's a design pattern where the the main information are stored inside one object and this object is used to pass everything

- Apache tomacat catalalina: It's an opensource web server developed by apache foundation (not oracle) for and only for servlets. **WRITE acronyms find in the code**

Reference documents

- Assignment document: Code inspection.pdf
- Glashfish javadoc of this version: http://glassfish.pompel.me/
- Methods assigned to each group: http://assignment.pompel.me/

Document structure

- **Introduction:** this section introduces the inspection document. It contains a justification of his utility and indications on which parts are covered in this document.
- **Classes:** this section describes the classes and the methods that have been inspected
- **Functional role:** this section describes the functional role of the class from which the methods assigned belong to. **TODO write role of each method?**
- **Issues list found by applying the checklist:** this section describes the issues found applying the checklist given.
- **Other problems:** this section describes other problems found that are not strictly related to the checklist.

Classes

All methods assigned to us belong to the same class.

StandardContext

Namespace: org.apache.catalina.core
Extends: ContainerBase
Implements: Context, ServletContext

Methods

```
Name:
    contextListenerStop( )
Start Line:
    5457
Name:
    eventListenerStop( )
Start Line:
    5509
Name:
    mergeParameters( )
Start Line:
    5537
Name:
    resourcesStart( )
Start Line:
    5564
Name:
    alternateResourcesStart( )
Start Line:
    5597
Name:
    resourcesStop( )
Start Line:
    5635
Name:
    alternateResourcesStop( )
Start Line:
    5662
Name:
    loadOnStartup( Container children [ ] )
Start Line:
    5708
```

[Gilles : is that better ?] [Claudio: I think so]

- Line 5457 : contextListenerStop( )
- Line 5509 : eventListenerStop( )
- Line 5537 : mergeParameters( )
- Line 5564 : resourcesStart( )
- Line 5597 : alternateResourcesStart( )
- Line 5635 : resourcesStop( )
- Line 5662 : alternateResourcesStop( )
- Line 5708 : loadOnStartup(Container children [ ])

WRITE IN A BETTER WAY? [Gilles : i think we have to exmplain the role of the methods (see in functionnal role) so maybe we can say the starting lines there and here just put alist of the methods] [Claudio: the assignment document says we have only to show the fuctional role of the classes not of the methods]

Functional role

Javadoc

This class is the standard implementation of the *Context* interface. According to the javadoc it is:

> A **Context** is a Container that represents a servlet context, and therefore an individual web application, in the Catalina servlet engine. It is therefore useful in almost every deployment of Catalina (even if a Connector attached to a web server (such as Apache) uses the web server's facilities to identify the appropriate Wrapper to handle this request. It also provides a convenient mechanism to use Interceptors that see every request processed by this particular web application.
> The parent Container attached to a Context is generally a Host, but may be some other implementation, or may be omitted if it is not necessary.
> The child containers attached to a Context are generally implementations of Wrapper (representing individual servlet definitions).

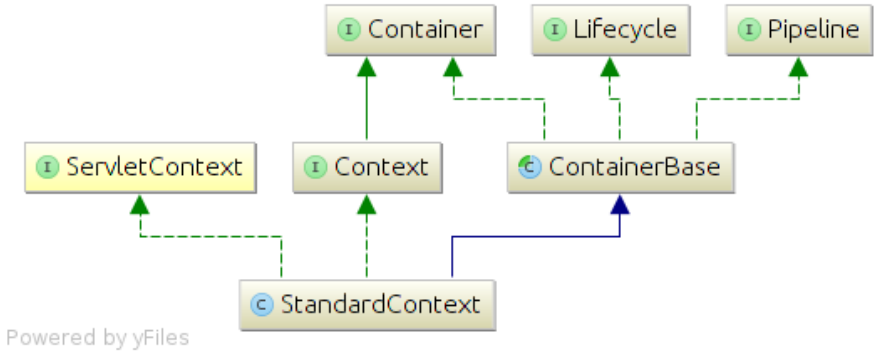

Figure 2: inheritance diagram

It extends *ContainerBase* that according to javadoc is:

> A **Container** is an object that can execute requests received from a client, and return responses based on those requests. A Container may optionally support a pipeline of Valves that process the request in a norder configured at runtime, by implementing the **Pipeline** interface as well

And implements *ServletContext* that is s standard *javax* interface that defines the basic methods to build a context for Servlet such as *addServlet*, *createListener* and so on

Usages

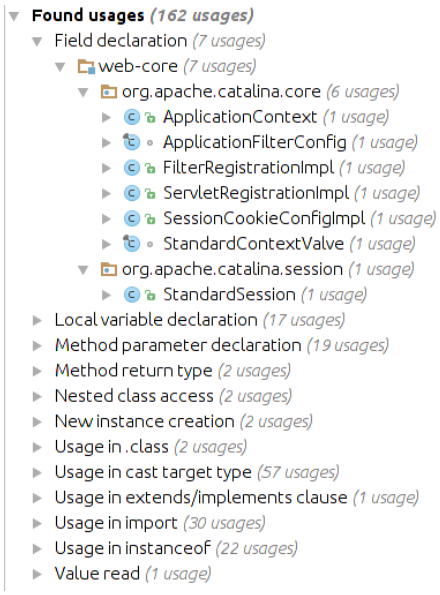

Figure 3: Usages

It's used in a lot of classes. In particularly that it is used as private property in catalina core classes.
For example we see that it is used by *ApplicationContext* that uses it to add everything, such as new servlet (*addServlet* on line *672* of *ApplicationContext*).

Role

Usages and javadoc suggest us that this class is very important because it is like the standard "manager" of apache tomacat catalina (that is a servlets server), in fact this class belongs to an host implementation (that uses it to manage all features inserted at high level) and it contains the servlets.
In fact in the *context pattern* (Contextual Information) we have a main class context that contains the main information, in this case contains the servlet refers or it allows to modify the request or responses via interceptor (in fact it extends *ContainerBase*).
The *context pattern* is very useful where there are a lot of data, for example it is

used in android applications to interact with the user. With this pattern you can manage a lot of features dynamically inserted via a single object from the usage side, from the creation side you can chose where use this features simply choosing the context. All data must pass via the context. **TODO improve**

TODO class diagram automatically generated? TODO write also our interpretation?

Issues list found by applying the checklist

NamingConventions

- from class `RestrictedServletContextListener`
  - method `contextInitialized` should start with a verb (hint: `onContextInitialized`)
  - method `contextDestroyed` should start with a verb (hint: `onContextDestroyed`)
- method `backgroundProcess` should start with a verb (hint: `runBackgroundProcess`)
- field `count` is not meaningful (hint: `backgroundProcessCounter`)
- method `contextListenerStart` should start with a verb (hint: `notifyContextStarted`)
- method `contextListenerStop` should start with a verb (hint: `stopContextListening`)
- return value of method `contextListenerStop` is never used (hint: change to `void`)
- method `create` is not clear and it looks like a simple alias of the `init` method
- method `create` is not used (hint: delete it)
- method `engineBase` should start with a verb (hint: `getEngineBase`)
- method `eventListenerStop` should start with a verb (hint: `stopEventListening`)
- method `eventListenerStop` always return true (hint: change to `void`)
- method `filterStart` should start with a verb (hint: `startFilters`)
- method `filterStop` should start with a verb (hint: `stopFilters`)
- method `managerStart` should start with a verb (hint: `startManager`)
- method `managerStop` should start with a verb (hint: `stopManager`)
- method `resourcesStart` should start with a verb (hint: `allocateResources`)
- method `resourcesStop` should start with a verb (hint: `freeResources`)
- method `restrictedSetPipeline` should start with a verb (hint: `setPipeline`)
- method `restrictedSetPipeline` should be made accessible only to certain packages (hint: declare it as `protected` and give a `friendly` accessor from the child class) **Gilles: I see no verb, shouldn't 'notifySessionCreated' be better ? Same thing for the other here under [Claudio: I think that on... is the standard for events]**
- method `sessionCreatedEvent` should start with a verb (hint: `onSessionCreatedEvent`)

- method `sessionDestroyedEvent` should start with a verb (hint: `onSessionDestroyedEvent`)
- method `sessionRejectedEvent` should start with a verb (hint: `onSessionRejectedEvent`)
- method `sessionExpiredEvent` should start with a verb (hint: `onSessionExpiredEvent`)
- method `sessionPersistedStartEvent` should start with a verb (hint: `onSessionPersistedStartEvent`)
- method `sessionPersistedEndEvent` should start with a verb (hint: `onSessionPersistedEndEvent`)
- method `sessionActivatedStartEvent` should start with a verb (hint: `onSessionActivatedStartEvent`)
- method `sessionActivatedEndEvent` should start with a verb (hint: `onSessionActivatedEndEvent`)
- method `sessionPassivatedStartEvent` should start with a verb (hint: `onSessionPassivatedStartEvent`)
- method `sessionPassivatedEndEvent` should start with a verb (hint: `onSessionPassivatedEndEvent`)
- method `sessionListenerStop` shlould start with a verb (hint: `stopSessionListening`)

Indention

- line `5479` start with a mismatching number of spaces
- line `5482` start with a mismatching number of spaces
- line `5486` start with a mismatching number of spaces
- line `5488` start with a mismatching number of spaces
- line `5625` start with a mismatching number of spaces

Braces

- single statement `if` without braces at line `5546`

N.B. K&R style is used

File Organization

- line `5487` can be easily rewritten to not exceed 80 columns.
- line `5574` can be easily rewritten to not exceed 80 columns.
- line `5576` can be easily rewritten to not exceed 80 columns.
- line `5582` can be easily rewritten to not exceed 80 columns.

- line 5613 can be easily rewritten to not exceed 80 columns.
- line 5618 can be easily rewritten to not exceed 80 columns.
- line 5621 can be easily rewritten to not exceed 80 columns.
- line 5624 can be easily rewritten to not exceed 80 columns.
- line 5680 can be easily rewritten to not exceed 80 columns.
- line 5734 can be easily rewritten to not exceed 80 columns.
- line 5735 can be easily rewritten to not exceed 80 columns.

Wrapping Lines

Everything ok

Comments

- commented code without any reason from line 5704 to 5706

Java Source Files

Everything ok

Package and Import Statements

Everything ok

Class and Interface Declarations

Everything ok

Initialization and Declarations

- can be private at line 5564
- can be private at line 5597
- can be private at line 5635
- can be private at line 5662
- can be private at line 5708
- `event` not declared at beginning of the block at line 5465
- `len` not declared at beginning of the block at line 5467
- `msg` not declared at beginning of the block at line 5487
- `iter` not declared at beginning of the block at line 5514

- `sc` not declared at beginning of the block at line `5553`
- `env` not declared at beginning of the block at line `5603`
- `alternateWebappResources` not declared at beginning of the block at line `5680`
- `key` not declared at beginning of the block at line `5719`
- `list` not declared at beginning of the block at line `5720`

Method Calls

TODO

Arrays

Everything ok

Object Comparison

Everything ok

Output Format

Everything ok

Computation, Comparisons and Assignments

TODO

Exceptions

- Exception `5619` is not logged

Flow of Control

Everything ok (there are no switches)

Files

Everything ok, no files

Other problems

- eventListenerStop: while can be replaced with foreach.
- alternateResourcesStart: unnecessary array for vararg `5625`
- loadOnStartup: parameter “children” declared in C-style array `5708`
- contextListenerStop: it’s more readable to use an iterator or to build a new list and reverse it.
- addEnvironment: never used
- addResource: never used
- addResourceLink: never used
- getStartupTime: never used
- getTldScanTime: never used
- setTldScanTime: never used
- redundant assignment at line `7498`
- redundant assignment at line `7545`
- redundant assignment at line `7559`
- getState: should return an enum or a class
- field `tldScanTime` is not used
- setCompilerClasspath: never used
- getOriginalDocBase: never used
- isReplaceWelcomeFiles: never used
- setUnloadDelay: never used
- setUnpackWAR: never used
- getCharsetMapperClass: never used
- setCharsetMapperClass: never used
- addResourceParams: never used
- addServletMapping(ServletMap): never used
- findMappingObject: never used
- findMessageDestination: never used
- findMessageDestinations: never used
- findMessageDestinationRef: never used
- findMessageDestinationRefs: never used
- removeMessageDestination: never used
- removeMessageDestinationRef: never used
- managerStart: never used
- managerStop: never used
- getDefaultConfigFile: never used
- getResourceNames: never used
- getResourceLinks: never used
- addEnvironment: never used
- addResource: never used
- addResourceLink: never used

- getStaticResources: never used
- startRecursive: never used
- getStartTimeMillis: never used
- isEventProvider: never used
- isStatisticsProvider: never used
- setCachingAllowed: never used
- setCaseSensitive: never used
- setCaseSensitiveMapping: never used
- setCacheTTL: never used
- setCacheMaxSize: never used
- getAntiJARLocking: never used
- setAntiJARLocking: never used
- MessageFormat is often called with an Object array intead of using vararg.
- some 'if' have a space before '(', but other not, the same thing for other construct like 'catch'
- } catch(Throwable t) { **TODO explain**
  - 5619
  - 5648
  - 5483
  - 5580
  - 5675
  - 5685
- two occurrences of new Object[], this is not a good way **KEEP OR REMOVE?**

Used tools

- intellij IDEA: JAVA EE IDE
- sonar: useful tools to analyze code from style point of view
- Github: for version controller
- Gedit and ReText: to write MarkDown with spell check

Hours of work

Claudio Cardinale

Gilles Dejaegere

Massimo Dragano