Code inspection

Figure 1: Politecnico di Milano

Version 1.0

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

Contents

Introduction

Purpose

The purpose of this document is to give the problems found during the inspection of a small amount of code of a specific version of glashfish.
Each group of the project has different methods assigned of a specif version of glashfish. We have to analyze them making all check of a checklist and finding other problems, then we have to report 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 required by the assignment text since we have some methods of this version assigned to us 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 version controller system **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 assigned
- **Functional role:** this section describes the functional role of the class of the methods assigned. **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
```

WRITE IN A BETTER WAY?

Functional role

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).

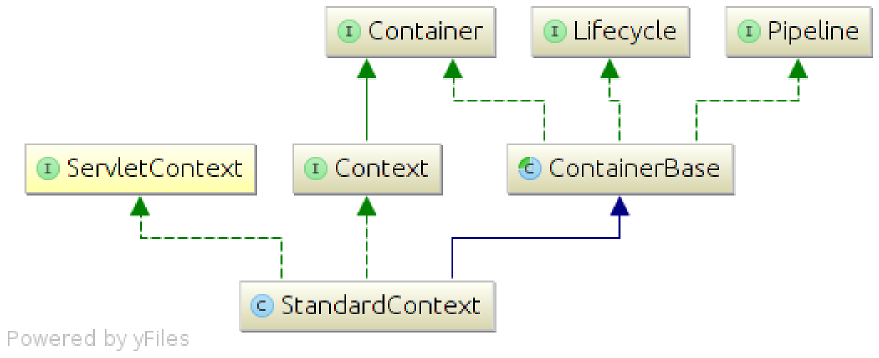

Figure 2: inheritance diagram

It extends *ContainerBase* that is **TODO**

And implements *ServletContext* that is a competitive interface. . . ..

TODO class diagram automatically generated? TODO write also our interpretation? TODO write role of each method? TODO remember that the class implements ServletContext that is compiled and extends ContianerBase

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)
- 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

All ok

Comments

TODO

Java Source Files

TODO

Package and Import Statements

TODO

Class and Interface Declarations

TODO

Initialization and Declarations

TODO

Method Calls

TODO

Arrays

All ok

Object Comparison

All ok

Output Format

All ok

Computation, Comparisons and Assignments

TODO

Exceptions

- Exception `5619`is not logged

Flow of Control

All ok (there are no switches)

Files

All ok, no files

TODO write task lists ??

Other problems

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