Politecnico di Milano
A.A. 2015-2016
Software Engineering 2: “MyTaxi”
Code inspection

Manzi Giuseppe (mat. 854470) &
Nicolini Alessandro (mat. 858858)

CONTENTS

1. Assigned classes

All the code that was assigned to us belongs to the same class, that is the class **ExtensionValidator**, located in:
appserver/web/web-core/src/main/java/org/apache/catalina/util/ExtensionValidator.java

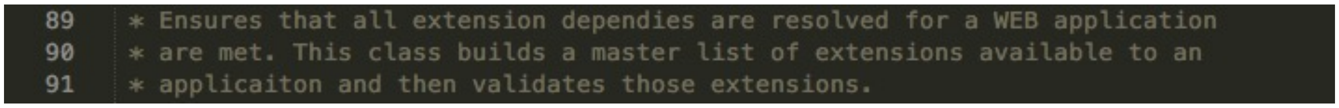
The code lines we had to check are the following ones:

- **Section 1** (lines **169 to 190**): *if* statement;
- **Section 2** (lines **215 to 292**): *validateApplication* method;
- **Section 3** (lines **317 to 385**): *validateManifestResources* method.

2. Functional role

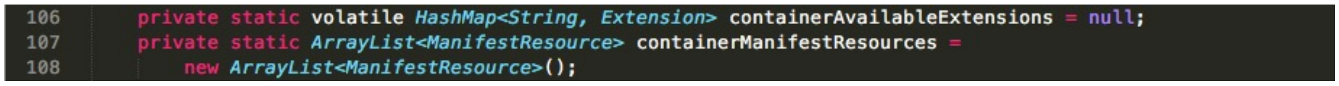
2.1 Functional role of the whole class

The class has to check the resources extentions, to ensures that all the needed resources are available.



```
 * Ensures that all extension dependies are resolved for a WEB application
 * are met. This class builds a master list of extensions available to an
 * applicaiton and then validates those extensions.
```

The class uses two lists, that contains the manifest resources, from which we can extract the extensions of the needed resources, and the available extentions. These lists are declared as attribiutes in the following code:



```
    private static volatile HashMap<String, Extension> containerAvailableExtensions = null;
    private static ArrayList<ManifestResource> containerManifestResources =
        new ArrayList<ManifestResource>();
```

2.2 Functional role of the sections

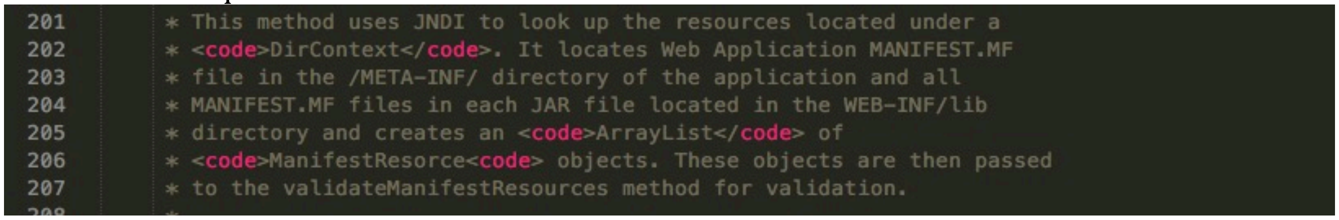
2.2.1 Section 1

Section 1 is part of the static initializer. It contains an *if* statement that get the files of the extension directory. For every directory listed in system properties it checks if it actually is a directory and add the manifest resources of the contained jar files in *containerManifestResources* using *addSystemResource* private static method.

```
        // get the files in the extensions directory
```

2.2.2 Section 2

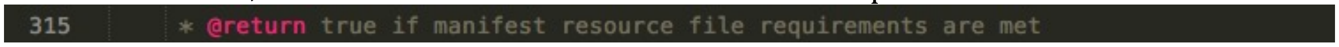
Section 2 is the *ValidateAppication* method. It validates at runtime the application and it returns true if all the required extensions are satisfied.



```
     * This method uses JNDI to look up the resources located under a
     * <code>DirContext</code>. It locates Web Application MANIFEST.MF
     * file in the /META-INF/ directory of the application and all
     * MANIFEST.MF files in each JAR file located in the WEB-INF/lib
     * directory and creates an <code>ArrayList</code> of
     * <code>ManifestResorce<code> objects. These objects are then passed
     * to the validateManifestResources method for validation.
```

2.2.3 Section 3

Section 3 is the *ValidateManifestResources* method. Using two nested while, it checks if, for every manifest resource, it exists an available resource that has a compatible extension.



```
     * @return true if manifest resource file requirements are met
```

3. Found issues (from checklist)

3.1 Section 0

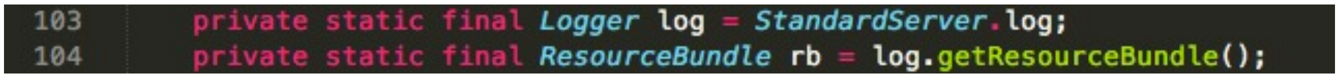
3.1.1 Description
Attributes and imports.

3.1.2 Lines
From 59 to 128.

3.1.3 Location
appserver/web/web-core/src/main/java/org/apache/catalina/util/ExtensionValidator.java

3.1.4 Issues

3.1.4.1
- **Lines:** 103 to 104
- **Rule (Kind of rule):** 7 (Naming conventions)
- **Description:** Static and final attributes should be all capitals
- **Code:**



```
103     private static final Logger log = StandardServer.log;
104     private static final ResourceBundle rb = log.getResourceBundle();
```

3.2 Section 1

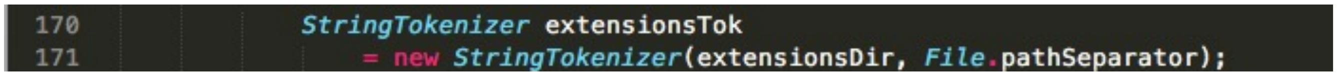
3.2.1 Description
if(extensionsDir != null) {---}

3.2.2 Lines
From 169 to 190.

3.2.3 Location
appserver/web/web-core/src/main/java/org/apache/catalina/util/ExtensionValidator.java

3.2.4 Issues

3.2.4.1
- **Lines:** 170 to 171
- **Rule (Kind of rule):** 15 (Line Break)
- **Description:** The break is before an operator (=) instead of being after it.
- **Code:**



```
170             StringTokenizer extensionsTok
171                 = new StringTokenizer(extensionsDir, File.pathSeparator);
```

3.3 Section 2

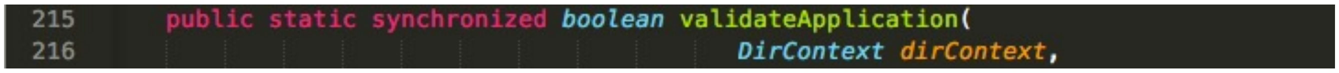
3.3.1 Description
ValidateApplication(DirContext dirContext, StandardContext context) method.

3.3.2 Lines
From 215 to 292.

3.3.3 Location
appserver/web/web-core/src/main/java/org/apache/catalina/util/ExtensionValidator.java

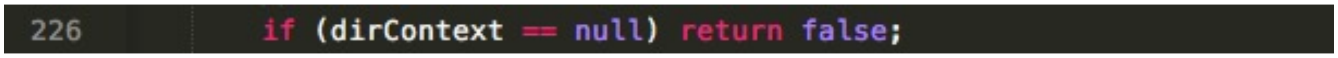
3.3.4 Issues

3.3.4.1
- **Lines:** 215 to 216
- **Rule (Kind of rule):** 15 (Line Break)
- **Description:** Break after an open parenthesis.
- **Code:**



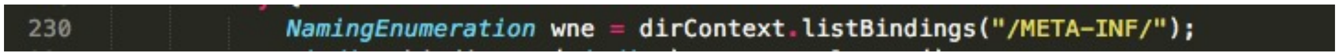
```
215     public static synchronized boolean validateApplication(
216                                                     DirContext dirContext,
```

3.3.4.2
- **Lines:** 226
- **Rule (Kind of rule):** 11 (Braces)
- **Description:** If without braces
- **Code:**



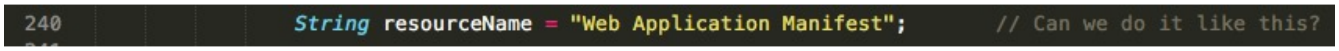
```
226             if (dirContext == null) return false;
```

3.3.4.3
- **Lines:** 230
- **Rule (Kind of rule):** 1 (Name)
- **Description:** Meaningless name
- **Code:**



```
230                 NamingEnumeration wne = dirContext.listBindings("/META-INF/");
```

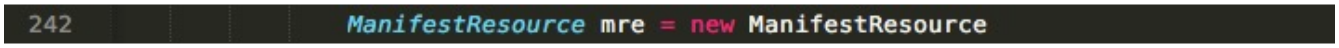
3.3.4.4
- **Lines:** 240
- **Rule (Kind of rule):** 18 (Comments)
- **Description:** Useless Comment
- **Code:**



```
240                 String resourceName = "Web Application Manifest";       // Can we do it like this?
```

3.3.4.5

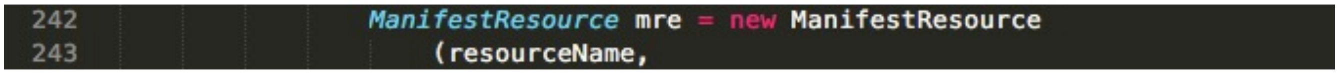
- **Lines:** 242
- **Rule (Kind of rule):** 1 (Name)
- **Description:** Meaningless name (mre -> manifestResource)
- **Code:**



```
242                     ManifestResource mre = new ManifestResource
```

3.3.4.6

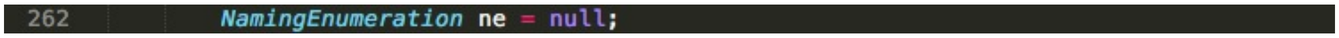
- **Lines:** 242 to 243
- **Rule (Kind of rule):** 15 (Line Break)
- **Description:** The code should be on the same line.
- **Code:**



```
242                     ManifestResource mre = new ManifestResource
243                         (resourceName,
```

3.3.4.7

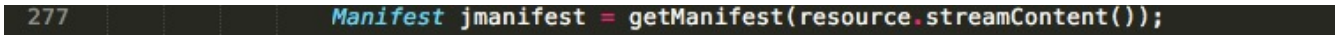
- **Lines:** 262
- **Rule (Kind of rule):** 1 (Name)
- **Description:** Meaningless name (ne-> namingEnumeration)
- **Code:**



```
262             NamingEnumeration ne = null;
```

3.3.4.8

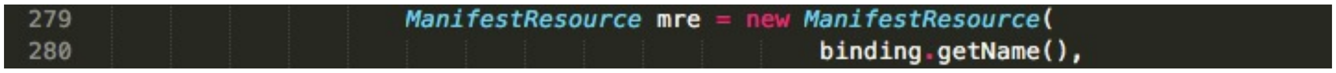
- **Lines:** 277
- **Rule (Kind of rule):** 1 (Name)
- **Description:** Meaningless name
- **Code:**



```
277                     Manifest jmanifest = getManifest(resource.streamContent());
```

3.3.4.9

- **Lines:** 279
- **Rule (Kind of rule):** 15 (Line Break)
- **Description:** The code should be on the same line
- **Code:**



```
279                     ManifestResource mre = new ManifestResource(
280                                             binding.getName(),
```

3.4 Section 3

3.4.1 Description

ValidateManifestResources(String appName, ArrayList<ManifestResource> resources) method.

3.4.2 Lines

From 317 to 385.

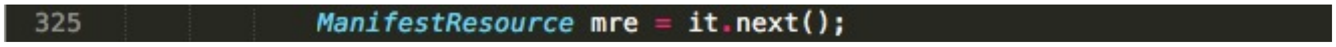
3.4.3 Location

appserver/web/web-core/src/main/java/org/apache/catalina/util/ExtensionValidator.java

3.4.4 Issues

3.4.4.1

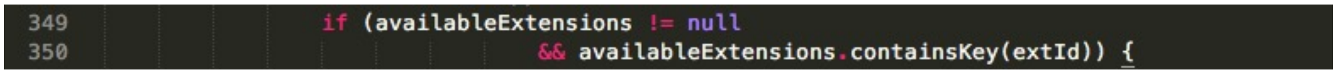
- **Lines:** 325
- **Rule (Kind of rule):** 1 (Name)
- **Description:** Meaningless name (mre -> manifestResource)
- **Code:**



```
325                 ManifestResource mre = it.next();
```

3.4.4.2

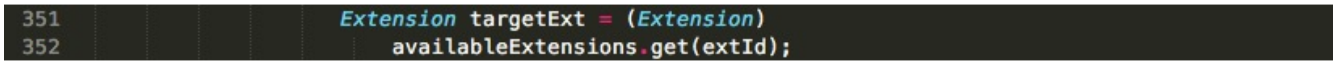
- **Lines:** 349 to 350
- **Rule (Kind of rule):** 15 (Wrapping lines)
- **Description:** The break is before an operator (&&) instead of being after it.
- **Code:**



```
349                 if (availableExtensions != null
350                         && availableExtensions.containsKey(extId)) {
```

3.4.4.3

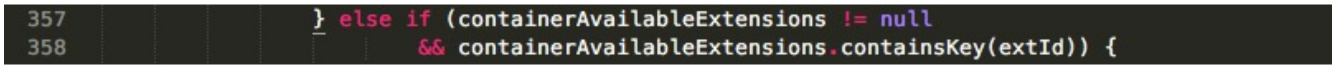
- **Lines:** 351 to 352
- **Rule (Kind of rule):** 15 (Wrapping lines)
- **Description:** The break is after a casting instead of being between the operator (=) and the casting.
- **Code:**



```
351                 Extension targetExt = (Extension)
352                     availableExtensions.get(extId);
```

3.4.4.4

- **Lines:** 357 to 358
- **Rule (Kind of rule):** 15 (Wrapping lines)
- **Description:** The break is before an operator (&&) instead of being after it.
- **Code:**



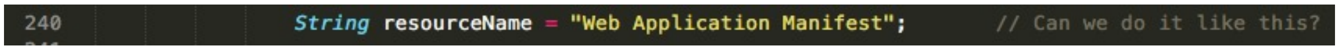
```
357             } else if (containerAvailableExtensions != null
358                     && containerAvailableExtensions.containsKey(extId)) {
```

4. Other problems

4.1 Section 2

4.1.1

- **Lines:** 240
- **Description:** Initialization should be done using binding.getname()
- **Code:**

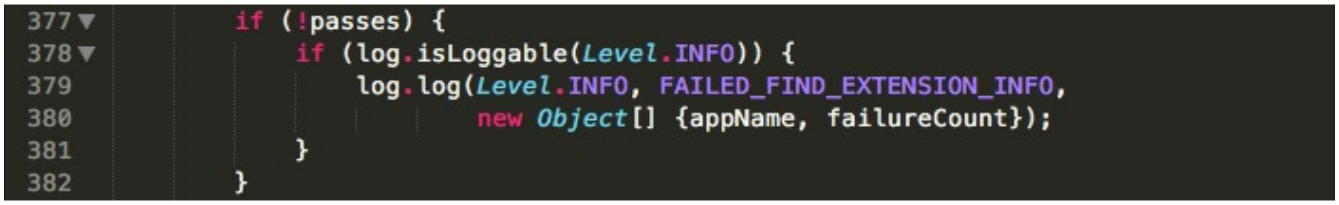


```
240             String resourceName = "Web Application Manifest";        // Can we do it like this?
```

4.1 Section 2

4.1.1

- **Lines:** 378 to
- **Description:** It is possible to combine the if statements
- **Code:**



```
377         if (!passes) {
378             if (log.isLoggable(Level.INFO)) {
379                 log.log(Level.INFO, FAILED_FIND_EXTENSION_INFO,
380                         new Object[] {appName, failureCount});
381             }
382         }
```