
LOG240 - Rapport de Lab 1

Asma Alibert
Quentin Marques

Table des matières

1. Connexion à la machine virtuelle VMware	2
2. Gestionnaire de sources	3
2.1. Configuration de SVN	3
2.2. Import/Export du code source vers SVN	4
3. Système de suivi de projet	7
3.1. Configuration de Trac	7
3.2. Gestion des permissions de Trac	7
3.3. Fonctionnalités de Trac	9
3.3.1. Tickets	9
3.3.2. Composants	10
3.3.3. Milestones	10
3.3.4. Versions	10
4. Gestionnaire de projet	11
4.1. Configuration Maven	11
4.2. Analyses des rapports	13
4.2.1. CheckStyle	14
4.2.2. PMD	16
4.3. QALab	17

1. Connexion à la machine virtuelle VMware

On se connecte à la machine virtuelle VMware pour s'assurer de son bon fonctionnement. Le laboratoire est réalisé sous environnement Windows et nous utilisons le terminal PuTTY pour nous connecter avec les paramètres suivants:

adresse IP: Log240-20133-14.logti.etsmtl.ca

port: 22

protocole: SSH

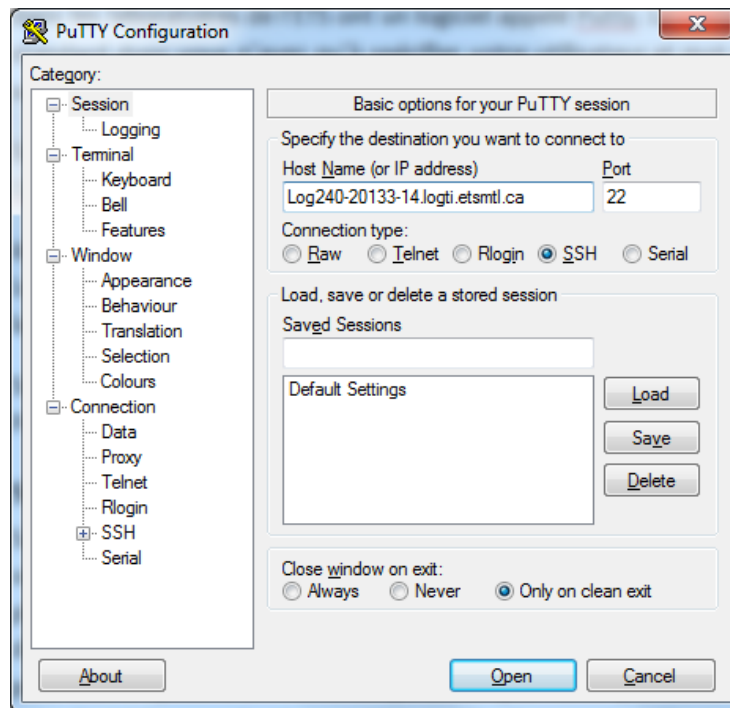


Figure 1. Connexion SSH au serveur Ubuntu en utilisant PuTTY

Un terminal s'ouvre alors et nous propose de nous connecter. Nous utilisons les identifiants suivants:

login: system

password: SystemPass14



Note

Sur un terminal UNIX, les mots de passe ne s'affichent pas lors de la saisie par soucis de sécurité, c'est pourquoi la sortie console suivante en est dépourvue.

```
login as: system
system@Log240-20133-14.logti.etsmtl.ca's password:
Linux log240-20133-14 2.6.32-51-generic-pae
#113-Ubuntu SMP Wed Aug 21 20:02:16 UTC 2013 i686 GNU/Linux
Ubuntu 10.04.4 LTS
```

```
Welcome to Ubuntu!
* Documentation:  https://help.ubuntu.com/

System information as of Mon Sep 16 15:48:01 EDT 2013

System load: 0.53           Memory usage: 18%   Processes:          75
Usage of /:  76.1% of 4.92GB Swap usage:   0%     Users logged in:  0

Graph this data and manage this system at https://landscape.canonical.com/

6 packages can be updated.
3 updates are security updates.

New release 'precise' available.
Run 'do-release-upgrade' to upgrade to it.

Last login: Mon Sep 16 15:45:09 2013 from logti-a3324-19.ens.ad.etsmtl.ca
system@log240-20133-14:~$
```



Note

Par soucis de clarté, l'invite de commande du shell (prompt) sera abrégée \$ par la suite.

On constate que la machine virtuelle est correctement lancée et en état de fonctionner.

2. Gestionnaire de sources

2.1. Configuration de SVN

Pour assurer le suivi du code source, nous utilisons le gestionnaire de sources Subversion (SVN). Nous commençons par changer les droits sur le dossier /opt qui sera notre dossier de travail pour lancer SVN en tant que daemon.

```
$ cd /opt
$ sudo /bin/chown -R www-data:svnusers svn
[sudo] password for system:
$ sudo /bin/chmod -R u+wx,g+wx,o-wrx svn
$ sudo /usr/bin/svnserve -d
```

On ajoute ensuite les différents utilisateurs:

client: L'utilisateur que notre client utilisera pour accéder au code

```
$ sudo /usr/sbin/useradd client -G svnusers
-c "Utilisateur client (BudgetPersonnel2000)" -m
$ sudo /usr/bin/passwd client
Enter new UNIX password:
Retype new UNIX password:
passwd: password updated successfully
```

asma: L'utilisateur du développeur 1

```
$ sudo /usr/sbin/useradd asma -G svnusers  
-c "Utilisateur developpeur (MaintenancePlus)" -m  
$ sudo /usr/bin/passwd asma  
Enter new UNIX password:  
Retype new UNIX password:  
passwd: password updated successfully
```

quentin: L'utilisateur du développeur 2

```
$ sudo /usr/sbin/useradd quentin -G svnusers  
-c "Utilisateur developpeur (MaintenancePlus)" -m  
$ sudo /usr/bin/passwd quentin  
Enter new UNIX password:  
Retype new UNIX password:  
passwd: password updated successfully
```



Important

Tout les mots de passes utilisés sont identiques aux noms des utilisateurs. Il est évident que ceci ne constitue pas une bonne pratique de sécurité mais, dans un soucis de simplification du rapport, nous avons décidé de faire ainsi.



Astuce

Nous pouvons vérifier que ces utilisateurs ont bien été créés sur la VMware de la manière suivante:

```
$ cat /etc/passwd | grep "Utilisateur" | cut -d: -f1  
client  
asma  
quentin
```

2.2. Import/Export du code source vers SVN

Nous envoyons le code source de FinanceJ sur le dépôt SVN nouvellement configuré. Nous utilisons le logiciel TortoiseSVN pour réaliser cette opération. Dans un premier temps, nous créons un dossier Code contenant les dossiers branches, tags et trunk. Nous avons ajouté à ce dernier dossier les codes sources du projet FinanceJ.



Attention

Le vocabulaire de TortoiseSVN diffère de celui couramment utilisé: L'importation SVN est un envoi depuis un poste local vers le serveur et inversement concernant l'exportation. Afin de lever l'ambiguïté, nous utiliserons les termes classiques *import/export*.

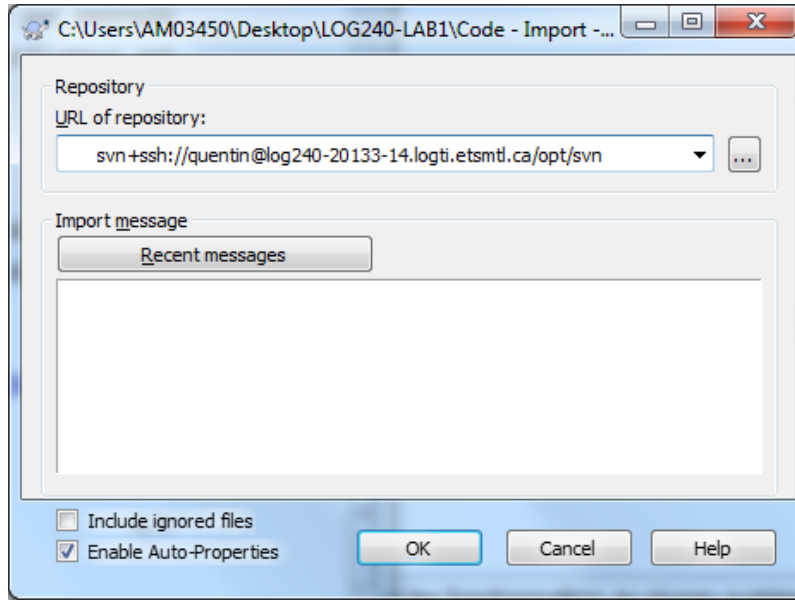


Figure 2. Exportation du dépôt en utilisant TortoiseSVN

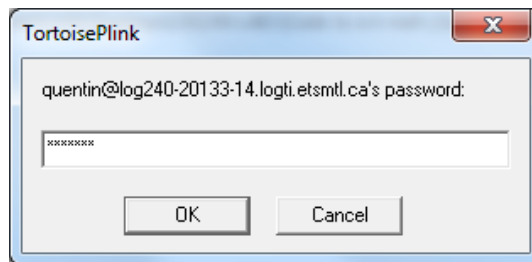


Figure 3. Confirmation du mot de passe SVN

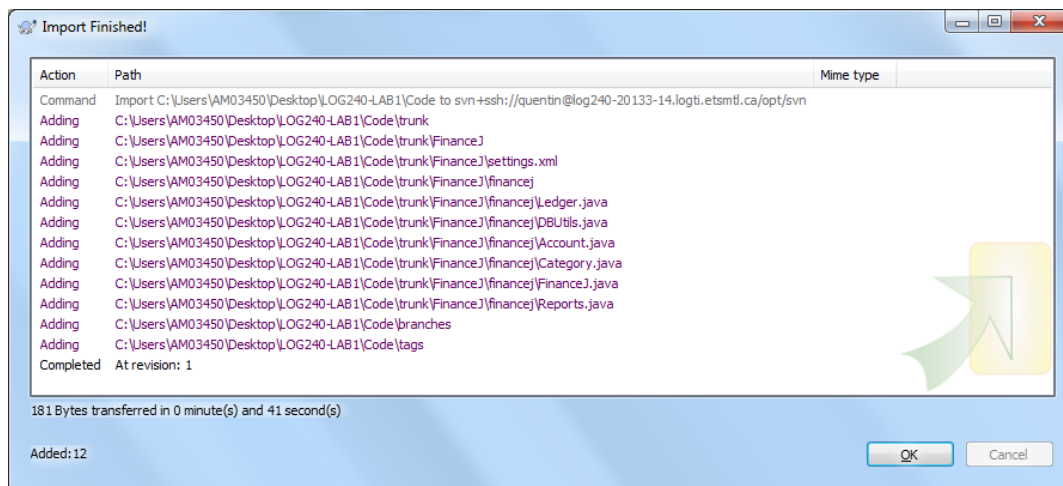


Figure 4. Résultat de l'exportation

Nous vérifions ensuite que l'envoi a correctement fonctionné en essayant d'importer le code envoyé dans un second dossier en local.



Note

Nous aurions également pu effectuer un **ls** directement sur le serveur via PuTTY mais des restrictions sur les VMwares nous en empêchent.

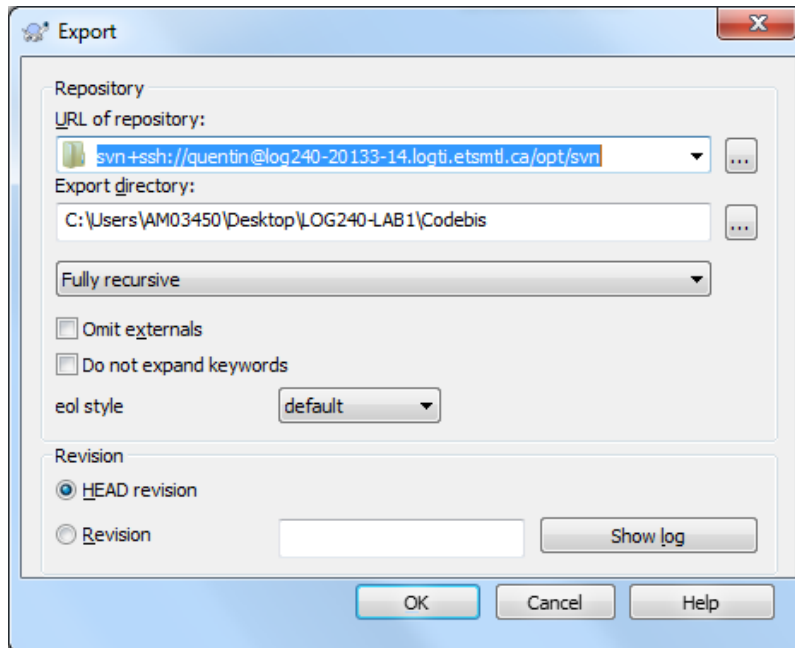


Figure 5. Vérification de l'importation dans le dossier Codebis

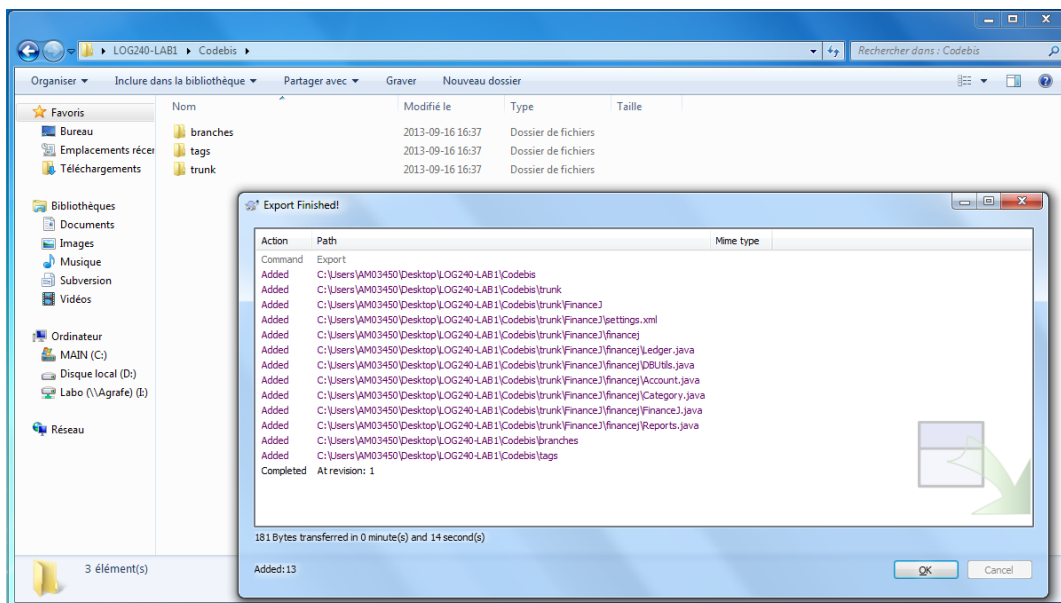


Figure 6. Succès du test d'importation dans le dossier Codebis

3. Système de suivi de projet

3.1. Configuration de Trac

Nous avons utilisé Trac, un système minimaliste de bug tracking et de wiki permettant la suivi de projet de développement. Tout d'abord, nous avons vérifié que Trac était correctement installé avec la bonne version:

```
$ tracd --version
tracd 0.11.7
```

Nous créons ensuite les 3 utilisateurs Trac de la manière suivante:

```
$ cd trac/conf/
$ sudo htpasswd trac.htpasswd client
New password:
Re-type new password:
Adding password for user client
```

```
$ sudo htpasswd trac.htpasswd asma
New password:
Re-type new password:
Adding password for user asma
```

```
$ sudo htpasswd trac.htpasswd quentin
New password:
Re-type new password:
Adding password for user quentin
```



Astuce

Nous pouvons vérifier que les utilisateurs ont correctement été rajoutés de la manière suivante:

```
$ cat trac.htpasswd | cut -d: -f1
admin
asma
quentin
client
```

3.2. Gestion des permissions de Trac

Nous nous sommes ensuite connectés à l'interface web de Trac à l'adresse `log240-20133-14.logti.etsmtl.ca/trac`. Nous nous sommes connectés en tant qu'administrateur avec les identifiants par défaut suivants:

login: admin

password: admin

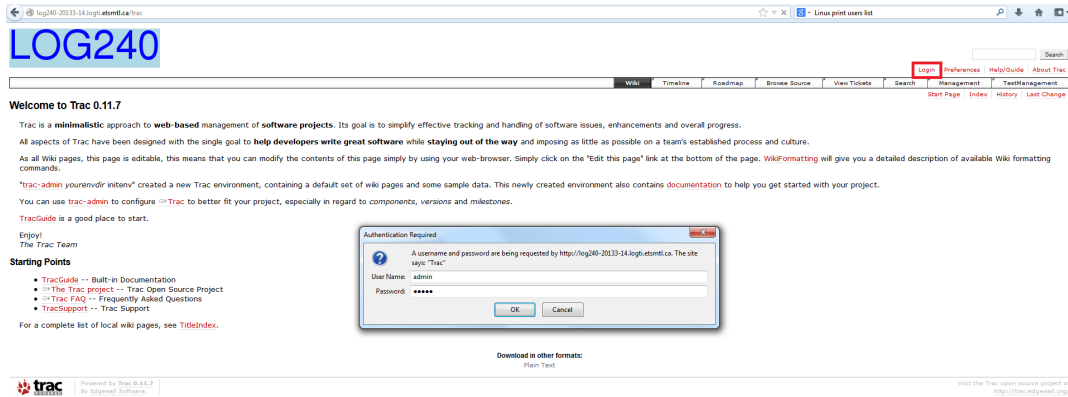


Figure 7. Connexion à l'interface web de Trac en admin

Nous avons ensuite défini les permissions de chaque utilisateur de Trac en suivant les contraintes suivantes:

- Il n'y a qu'un seul développeur ayant les droits d'administrateur (asma)
- Les développeurs doivent avoir accès à la plupart des fonctionnalités exceptés les tâches d'administration
- Le client peut uniquement consulter Trac et créer des tickets pour remonter des anomalies aux développeurs



Note

Afin d'éviter de redéfinir chaque permission pour chaque nouvel utilisateur éventuel, nous avons décidé de créer deux groupes de permissions:

- developpers
- clients

Ceci nous permet d'ajouter simplement des développeurs ou des comptes clients tout en conservant des permissions cohérentes.

Manage Permissions

Subject	Action					
admin	<input checked="" type="checkbox"/> TRAC_ADMIN					
anonymous	<input type="checkbox"/> BROWSER_VIEW	<input type="checkbox"/> CHANGESSET_VIEW	<input type="checkbox"/> FILE_VIEW	<input type="checkbox"/> LOG_VIEW	<input type="checkbox"/> MILESTONE_VIEW	
	<input type="checkbox"/> REPORT_SQL_VIEW	<input type="checkbox"/> REPORT_VIEW	<input type="checkbox"/> ROADMAP_VIEW	<input type="checkbox"/> SEARCH_VIEW	<input type="checkbox"/> TICKET_VIEW	
	<input type="checkbox"/> TIMELINE_VIEW	<input type="checkbox"/> WIKI_VIEW				
asma	<input checked="" type="checkbox"/> admin					
authenticated	<input type="checkbox"/> TICKET_CREATE	<input type="checkbox"/> TICKET_MODIFY	<input type="checkbox"/> WIKI_CREATE	<input type="checkbox"/> WIKI_MODIFY		
client	<input type="checkbox"/> clients					
clients	<input type="checkbox"/> CHANGESSET_VIEW	<input type="checkbox"/> MILESTONE_VIEW	<input type="checkbox"/> REPORT_VIEW	<input type="checkbox"/> ROADMAP_VIEW	<input type="checkbox"/> SEARCH_VIEW	
	<input type="checkbox"/> TICKET_CREATE	<input type="checkbox"/> TICKET_VIEW				
developers	<input type="checkbox"/> BROWSER_VIEW	<input type="checkbox"/> CHANGESSET_VIEW	<input type="checkbox"/> FILE_VIEW	<input type="checkbox"/> LOG_VIEW	<input type="checkbox"/> MILESTONE_VIEW	
	<input type="checkbox"/> REPORT_VIEW	<input type="checkbox"/> ROADMAP_VIEW	<input type="checkbox"/> SEARCH_VIEW	<input type="checkbox"/> TICKET_CREATE	<input type="checkbox"/> TICKET_MODIFY	
	<input type="checkbox"/> TICKET_VIEW	<input type="checkbox"/> TIMELINE_VIEW	<input type="checkbox"/> WIKI_ADMIN			
quentin	<input type="checkbox"/> developers					

Remove selected items

Figure 8. Modifications des permissions clients et développeurs

3.3. Fonctionnalités de Trac

3.3.1. Tickets

Les tickets dans Trac peuvent être utilisés pour assigner des tâches de projet, pour faire des requêtes, des reports de bugs et du support logiciel. Ici nous avons créé 2 requêtes / tickets / tâches. À l'aide de Trac, nous avons la possibilité de:

- Définir la priorité de la tâche
- Préciser le composant affecté
- Estimer le nombre d'heures pour la tâche
- Préciser la version
- Ajouter des commentaires
- Joindre des fichiers
- etc.

Ticket #1 (new task)

Vérifier le code		Opened 0 seconds ago	
Reported by:	asma	Owned by:	quentin
Priority:	major	Milestone:	
Component:	Code	Version:	Lab1
Keywords:		Cc:	
Estimated Number of Hours:	0.0		
Billable?:	yes	Total Hours:	0
Description			

Ticket #2 (new task)

S'assurer que le site web se lance		Opened 0 seconds ago	
Reported by:	asma	Owned by:	quentin
Priority:	major	Milestone:	
Component:	Site Web	Version:	Lab1
Keywords:		Cc:	
Estimated Number of Hours:	0.0		
Billable?:	yes	Total Hours:	0
Description			

Figure 9. Création de tickets Trac

3.3.2. Composants

Les composants sont les systèmes/modules qui peuvent être affectés par des problèmes et donc être concernés par les tickets. Nous avons créé 2 composants:

- code
- site web

Manage Components

	Name	Owner	Default
<input type="checkbox"/>	Code	asma	<input type="radio"/>
<input type="checkbox"/>	Site Web	asma	<input type="radio"/>

Figure 10. Création de composants Trac

3.3.3. Milestones

Les milestones sont les étapes importantes de la maintenance du projet.

Manage Milestones

	Name	Due	Completed	Default	Tickets
<input type="checkbox"/>	Assigner la requête à un membre de l'équipe			<input type="radio"/>	0
<input type="checkbox"/>	Catégoriser la requête			<input type="radio"/>	0
<input type="checkbox"/>	Effectuer l'analyse d'impact			<input type="radio"/>	0
<input type="checkbox"/>	Ouvrir un Billet			<input type="radio"/>	0
<input type="checkbox"/>	Programmation			<input type="radio"/>	0
<input type="checkbox"/>	Reproduire le problème			<input type="radio"/>	0
<input type="checkbox"/>	Surveillance de la production			<input type="radio"/>	0
<input type="checkbox"/>	Test d'acceptation avec le client			<input type="radio"/>	0
<input type="checkbox"/>	Test système et documenter			<input type="radio"/>	0
<input type="checkbox"/>	Vérifier la complétion des activités			<input type="radio"/>	0

Figure 11. Création de milestones Trac

3.3.4. Versions

Nous considérons que les versions représenteront les 5 laboratoires

Manage Versions

	Name	Released	Default
<input type="checkbox"/>	Lab5	12/10/13 16:05:56	<input type="radio"/>
<input type="checkbox"/>	Lab4	11/19/13 16:05:17	<input type="radio"/>
<input type="checkbox"/>	Lab3	10/30/13 16:04:14	<input type="radio"/>
<input type="checkbox"/>	Lab2	10/14/13 00:00:00	<input type="radio"/>
<input type="checkbox"/>	Lab1	09/30/13 00:00:00	<input type="radio"/>

Remove selected items

Apply changes

Figure 12. Création de versions Trac

4. Gestionnaire de projet

4.1. Configuration Maven

Nous avons déplacé le fichier `settings.xml` fourni dans le répertoire `C:\Documents and Settings\am03450\.m2\`. Nous avons configuré le fichier `pom.xml` puis nous l'avons exécuté. Maven a alors compilé et exécuté le projet. Maven pouvant générer un site web du projet à partir des sources, nous avons configuré le fichier `pom.xml` pour ajouter un plugin permettant la génération de rapports. Maven permet également de générer des rapports d'analyse de la qualité du code source :

- PMD qui analyse la qualité du code source Java
- CheckStyle permet de vérifier le style du code source
- QALab permet de récupérer les rapports de PMD et CheckStyle au cours du temps

Nous avons ensuite configuré Maven afin qu'il puisse ajouter au site web du projet les informations provenant de Trac et SVN. Pour faire cela, nous avons ajouté des plugins au fichier `settings.xml` ainsi qu'au `pom.xml`. Les rapports produits par QALab sont générés sur le site web du projet par Maven dans la section "Projects Reports".

```

C:\Windows\system32\cmd.exe - min_execjava
parent: 24 pom (23 KB at 49.1 KB/sec)
Downloaded: http://repo1.maven.org/maven2/org/codehaus/mojo/exec-naven-plugin/1.2/exec-naven-plugin-1.2.jar
Downloaded: http://repo1.maven.org/maven2/org/codehaus/mojo/exec-naven-plugin/1.2/exec-naven-plugin-1.2.jar (36 KB at 144.2 KB/sec)
[INFO]
[INFO] Building FinanceJ 1.0-SNAPSHOT
[INFO]
[INFO] >>> exec-naven-plugin:1.2:java (default-cli) @ financej-tean14 >>>
[INFO]
[INFO] <<< exec-naven-plugin:1.2:java (default-cli) @ financej-tean14 <<<
[INFO]
[INFO] --- exec-naven-plugin:1.2:java (default-cli) @ financej-tean14 ---
Downloaded: http://repo1.maven.org/maven2/org/apache/maven/maven-plugin-api/2.0/maven-plugin-api-2.0.pom
Downloaded: http://repo1.maven.org/maven2/org/apache/maven/maven-plugin-api/2.0/maven-plugin-api-2.0.pom (601 B at 9.3 KB/sec)
Downloaded: http://repo1.maven.org/maven2/org/apache/maven/maven/2.0/maven-2.0.pom
Downloaded: http://repo1.maven.org/maven2/org/apache/maven/maven/2.0/maven-2.0.pom (2 KB at 189.8 KB/sec)
Downloaded: http://repo1.maven.org/maven2/org/codehaus/plexus/plexus-container-default/1.0-alpha-9/plexus-container-default-1.0-alpha-9.pom
Downloaded: http://repo1.maven.org/maven2/org/codehaus/plexus/plexus-container-default/1.0-alpha-9/plexus-container-default-1.0-alpha-9.pom (2 KB at 15.5 KB/sec)
Downloaded: http://repo1.maven.org/maven2/org/apache/commons/commons-exec/1.0.1/commons-exec-1.0.1.pom
Downloaded: http://repo1.maven.org/maven2/org/apache/commons/commons-exec/1.0.1/commons-exec-1.0.1.pom (9 KB at 113.5 KB/sec)
Downloaded: http://repo1.maven.org/maven2/org/apache/commons/commons-parent/11/commons-parent-11.pom
Downloaded: http://repo1.maven.org/maven2/org/apache/commons/commons-parent/11/commons-parent-11.pom (25 KB at 256.1 KB/sec)
Downloaded: http://repo1.maven.org/maven2/org/apache/commons/commons-exec/1.0.1/commons-exec-1.0.1.jar (49 KB at 437.3 KB/sec)
org.apache.derby.jdbc.EmbeddedDriver loaded.
Connected to database FinanceJDB
. . . creating table account
. . . creating table category
. . . creating table ledger
Accounts Combo Box Performed: null

```

Figure 13. Compilation du projet FinanceJ par Maven

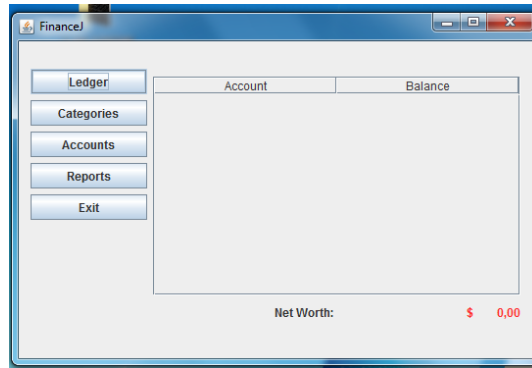


Figure 14. Code compilé et exécuté par Maven

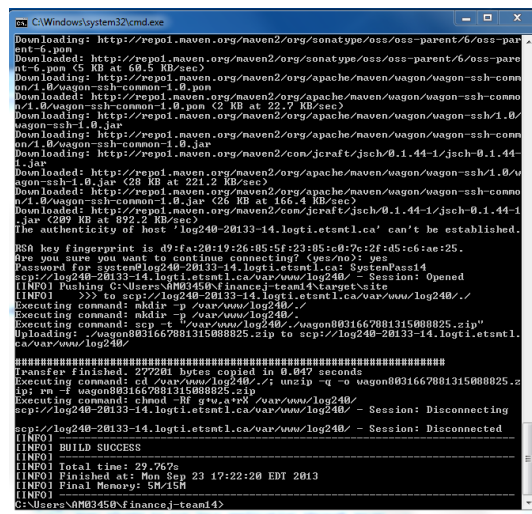


Figure 15. Compilation du site web par Maven

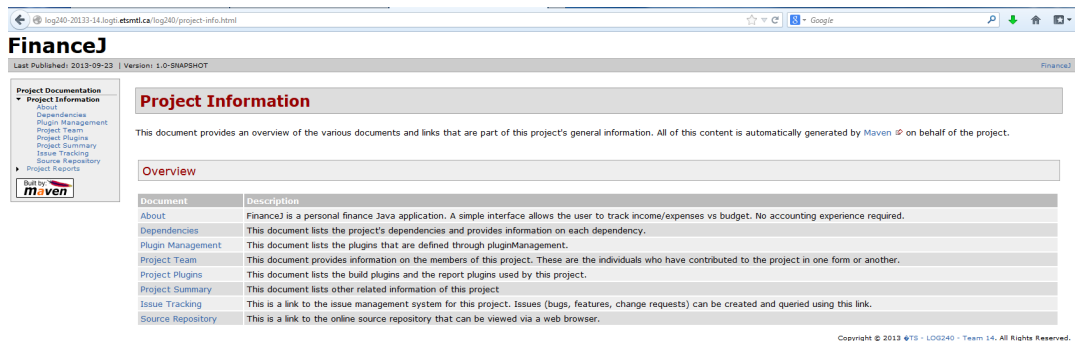


Figure 16. Site web générés par Maven

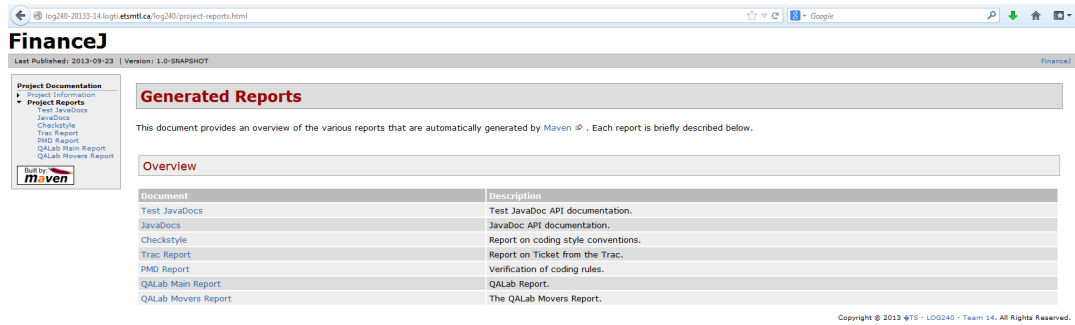



Figure 17. Rapports générés disponibles sur le site web

4.2. Analyses des rapports

Nous avons généré différents rapports pour auditer la qualité du code. Nous avons notamment utilisé CheckStyle et PMD dont les rapports ont été générés par Maven.

4.2.1. CheckStyle

Checkstyle Results

The following document contains the results of Checkstyle [🔗](#). 

Summary

Files	Infos 	Warnings 	Errors 
6	0	0	1515

Files

Files	I 	W 	E 
ca/etsmtl/log240/financej/Account.java	0	0	507
ca/etsmtl/log240/financej/Category.java	0	0	178
ca/etsmtl/log240/financej/DBUtils.java	0	0	45
ca/etsmtl/log240/financej/FinanceJ.java	0	0	236
ca/etsmtl/log240/financej/Ledger.java	0	0	349
ca/etsmtl/log240/financej/Reports.java	0	0	200

Rules







Rules	Violations	Severity
JavadocPackage <ul style="list-style-type: none"> allowLegacy: "true" 	1	 Error
NewlineAtEndOfFile	6	 Error
Translation	0	 Error
FileLength	0	 Error
FileTabCharacter <ul style="list-style-type: none"> eachLine: "true" 	326	 Error
RegexpSingleline <ul style="list-style-type: none"> message: "Line has trailing spaces." 	53	 Error

Figure 18. Rapports CheckStyle analysant les problèmes de styles (nommage, indentation, etc.)

CheckStyle identifie les types d'erreurs suivants:

- Javadoc: Absence de commentaires, manquement d'un paramètre, mauvais formatage etc.

Ceci a un impact très négatif sur la maintenabilité car cela empêche de futurs développeurs d'analyser le code pour le maintenir sans avoir à rentrer dans le détail du code.

- Respect des conventions de codage: Convention de nommage des méthodes, classes, attributs etc.

Il est beaucoup plus aisé de se repérer dans un code quand les conventions de nommages sont respectées. Une syntaxe particulière pour chaque développeur peut mener à des incompréhensions, surtout quand la casse a un sens pour différencier classe, objet, attributs, variables locales etc.

- Formatage du code: Indentation, nombre de caractères par ligne, espacement entre caractères etc.

Un code est plus lisible s'il n'est pas trop chargé. Idéalement en dessous de 80 caractères par ligne, un développeur peut même lire son code sur des terminaux étroits ou avec plusieurs pages côte à côte sur un écran large. De plus, les niveaux d'indentations permettent de repérer facilement les unités logiques, ce qui facilite la maintenance.

- Rigueur du code: Absence de mots clé `abstract`/`final` quand c'est possible, utilisation du `private` et des accesseurs, optimisations des imports etc.

La rigueur dans l'utilisation des mots clé de restrictions (`abstract`, `final`, `private` etc.) permet de limiter l'impact de comportement inattendu en spécifiant plus précisément la portée d'action de l'élément concerné. Sur le long terme, négliger d'utiliser ces mots clés peut créer des défauts difficiles à trouver.

Les erreurs les plus récurrentes dans ce code sont:

- `LineLength` (341 occurrences) : Trop de caractères sur une ligne nuisent à la lisibilité
- `FileTabCharacter` (326 occurrences) : Des tabulations sont présentes dans les lignes de code.
- `FinalParameters` (139 occurrences) : Des paramètres pourraient être constants et ne le sont pas.
- `MagicNumber` (130 occurrences) : Utilisation de constantes numériques non-nommée ou mal documentée.
- `JavadocMethod` (102 occurrences) : Manque d'un commentaire javadoc

Les mesures correctives sont les suivantes:

- Mettre en place une politique de commentaire systématique et rigoureuse pour les méthodes. Le format type d'un commentaire doit être suivi pour respecter les règles de `CheckStyle`.
- Imposer une convention de nommage des variables, notamment au niveau de la casse et des caractères spéciaux autorisés.
- L'utilisation d'un outil de formatage automatique peut être une bonne solution pour forcer une indentation correcte ainsi qu'un formatage régulier.
- Des revues de codes régulières permettent de détecter ce types d'erreurs et des les corriger rapidement avant qu'elles ne s'accumulent.

4.2.2. PMD

PMD Results

The following document contains the results of [PMD](#) 4.2.5.

Files

[ca/etsmtl/log240/financej/Account.java](#)

Violation	Line
Perhaps 'conn' could be replaced by a local variable.	19
Perhaps 'AddAccountButton' could be replaced by a local variable.	218
Perhaps 'CloseButton' could be replaced by a local variable.	219
Perhaps 'DeleteAccountButton' could be replaced by a local variable.	220
Perhaps 'jLabel1' could be replaced by a local variable.	223
Perhaps 'jLabel2' could be replaced by a local variable.	224
Perhaps 'jPanel1' could be replaced by a local variable.	225
Perhaps 'jScrollPane1' could be replaced by a local variable.	226
Private field 'columnNames' could be made final; it is only initialized in the declaration or constructor.	232
Private field 'conn' could be made final; it is only initialized in the declaration or constructor.	233
Ensure that resources like this ResultSet object are closed after use	244
Ensure that resources like this Statement object are closed after use	245
Ensure that resources like this ResultSet object are closed after use	268
Ensure that resources like this Statement object are closed after use	269
Avoid unnecessary if..then..else statements when returning a boolean	307 - 311
Ensure that resources like this Statement object are closed after use	320
Ensure that resources like this Statement object are closed after use	333

[ca/etsmtl/log240/financej/Category.java](#)

Violation	Line
Perhaps 'conn' could be replaced by a local variable.	19
Perhaps 'AddCategoryButton' could be replaced by a local variable.	224

Figure 19. Rapports PMD analysant les problèmes de codages (portée des variables, initialisations etc.)

PMD identifie les types d'erreur suivants:

- Les problèmes de portées de variables.

Cela a un impact négatif sur la maintenance car cela peut entraîner des comportements inattendus de la part d'un objet si son état interne devient incohérent.

- Les problèmes d'initialisation des attributs/variables.

Si l'état d'un objet n'est pas cohérent dès l'initialisation, cela peut entraîner facilement des erreurs où l'on s'attend à ce que l'objet soit utilisable alors qu'il doit encore être initialisé.

- La libération des ressources.

Sur le long terme, ce type d'erreurs peut entraîner des problèmes de performances très gênants pour l'utilisateur.

- Optimisations diverses (ex: refactor d'un if...else..., utilisation des valeurs littérales avant les chaînes pour les comparaisons etc.)

L'utilisation de ces techniques d'optimisations peut rendre le code plus clair, plus performant et plus maintenable.

4.3. QALab

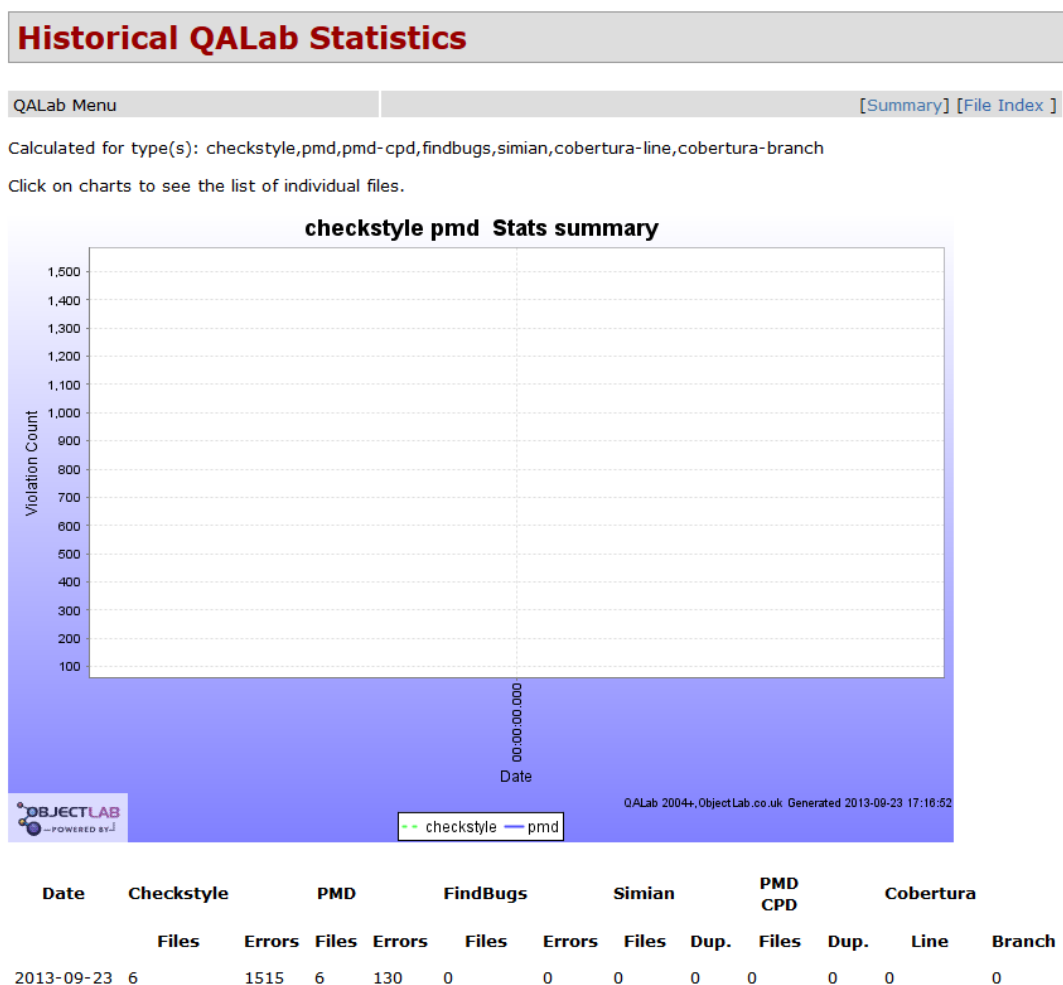


Figure 20. Rapport de synthèse QALab sans modification du code (pas de progression à afficher)

Details

ca/etsmtl/log240/financej/Account.java

Violation	Message	Line
✖	File does not end with a newline.	0
✖	Missing package-info.java file.	0
✖	Using the '*' form of import should be avoided - java.sql.*.	8
✖	Using the '*' form of import should be avoided - java.util.*.	9
✖	Using the '*' form of import should be avoided - javax.swing.table.*.	10
✖	Using the '*' form of import should be avoided - javax.swing.*.	11
✖	Line contains a tab character.	19
✖	Missing a Javadoc comment.	19
✖	Line contains a tab character.	20
✖	Missing a Javadoc comment.	20
✖	First sentence should end with a period.	22
✖	Line contains a tab character.	22
✖	Line contains a tab character.	23
✖	Parameter parent should be final.	23
✖	Expected @param tag for 'parent'.	23
✖	Parameter modal should be final.	23
✖	Expected @param tag for 'modal'.	23

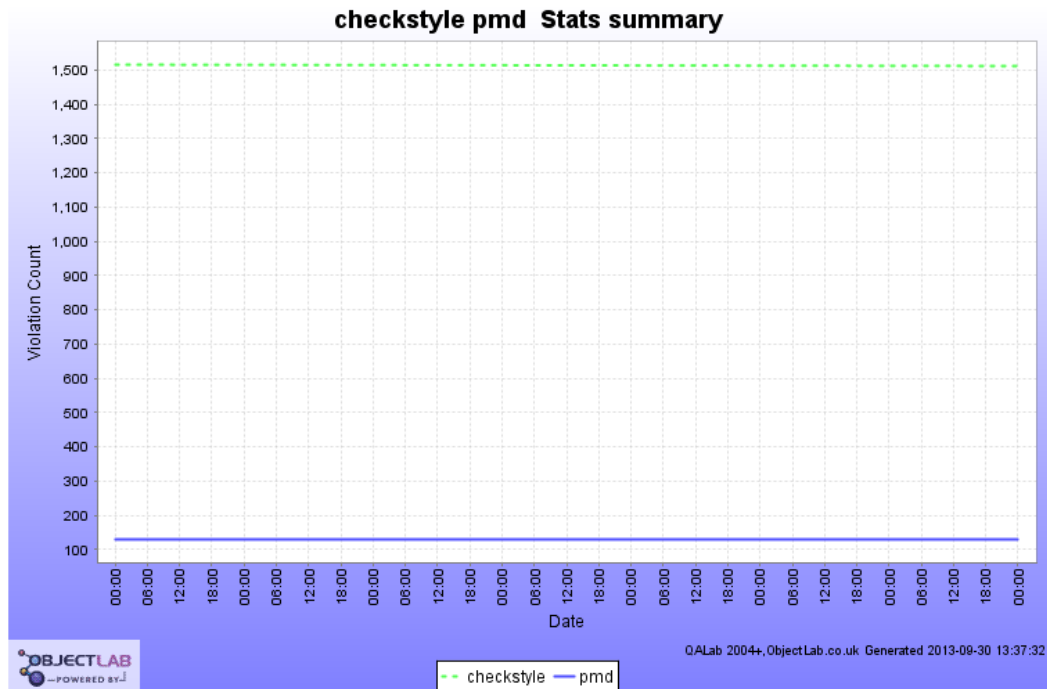
Figure 21. Erreur CheckStyle que nous avons corrigé

Historical QALab Statistics

QALab Menu

Calculated for type(s): checkstyle,pmd,pmd-cpd,findbugs,simian,cobertura-line,cobertura-branch

Click on charts to see the list of individual files.



Date	Checkstyle		PMD		FindBugs	
	Files	Errors	Files	Errors	Files	Err
2013-09-30	6	1514	6	130	0	0
2013-09-23	6	1515	6	130	0	0

Figure 22. Rapport de synthèse QALab avec modifications du code (progression de la qualité)

Nous avons effectué une modification trop mineure pour que la progression se voit sur la prise d'écran. Cependant, dans le détail en dessous du graphe, nous observons qu'une des erreurs a bien disparue depuis la précédente version.