Atelier de Développement: Configuration et Exécution du Build avec Maven

Nicolas HERBAUT

Année 2018-2019

UNIVERSITÉ PARIS 1
PANTHÉON SORBONNE

Qu'est-ce que le Build d'un projet?

Idée reçue n°1 ce n'est pas uniquement le code compilé!

UNIVERSITÉ PARIS 1
PANTHÉON SORBONNE

Idée reçue n°1 ce n'est pas uniquement le code compilé!

- Il existe d'autre nombreux artefacts:

Idée reçue n°1 ce n'est pas uniquement le code compilé!

- Il existe d'autre nombreux artefacts:
- *Documentation* utilisateur et d'API (Javadoc)

UNIVERSITÉ PARIS 1
PANTHÉON SORBONNE

Idée reçue n°1 ce n'est pas uniquement le code compilé!

- Il existe d'autre nombreux artefacts:
- *Documentation* utilisateur et d'API (Javadoc)
- Mesures de la *qualité du code*
  - est-ce que tous les tests unitaires passent?
  - est-ce que le style de codage est conforme aux guide de l'équipe?
  - est-ce que tout le code est testé?

UNIVERSITÉ PARIS 1
PANTHÉON SORBONNE

Idée reçue n°1 ce n'est pas uniquement le code compilé!

- Il existe d'autre nombreux artefacts:
- *Documentation* utilisateur et d'API (Javadoc)
- Mesures de la *qualité du code*
  - est-ce que tous les tests unitaires passent?
  - est-ce que le style de codage est conforme aux guide de l'équipe?
  - est-ce que tout le code est testé?
- *Code source* (pour les projets open-source)

Idée reçue n°1 ce n'est pas uniquement le code compilé!

- Il existe d'autre nombreux artefacts:
- *Documentation* utilisateur et d'API (Javadoc)
- Mesures de la *qualité du code*
  - est-ce que tous les tests unitaires passent?
  - est-ce que le style de codage est conforme aux guide de l'équipe?
  - est-ce que tout le code est testé?
- *Code source* (pour les projets open-source)
- *Licence*
- Tout ce qui est nécessaire pour faire tourner le programme:
  - Fichiers binaires (images, pdf...)
  - Les dépendances externes (libraries, framework)

UNIVERSITÉ PARIS 1
PANTHÉON SORBONNE

Idée reçue n°1 ce n'est pas uniquement le code compilé!

- Il existe d'autre nombreux artefacts:
- *Documentation* utilisateur et d'API (Javadoc)
- Mesures de la *qualité du code*
  - est-ce que tous les tests unitaires passent?
  - est-ce que le style de codage est conforme aux guide de l'équipe?
  - est-ce que tout le code est testé?
- *Code source* (pour les projets open-source)
- *Licence*
- Tout ce qui est nécessaire pour faire tourner le programme:
  - Fichiers binaires (images, pdf...)
  - Les dépendances externes (libraries, framework)

Build Automatisé

Le build est une étape complexe, qui doit être effectuée automatiquement

Idée reçue n°2: Si ça compile, c'est que ça marche

UNIVERSITÉ PARIS 1
PANTHÉON SORBONNE

Idée reçue n°2: Si ça compile, c'est que ça marche

- Le but de build est de s'assurer de la production de nos artefacts MAIS avec une qualité suffisante.

Idée reçue n°2: Si ça compile, c'est que ça marche

- Le but de build est de s'assurer de la production de nos artefacts MAIS avec une qualité suffisante.
- Mais doit-on considérer un build réussi si les tests échouent? Et si la doc est manquante?

Idée reçue n°2: Si ça compile, c'est que ça marche

- Le but de build est de s'assurer de la production de nos artefacts MAIS avec une qualité suffisante.
- Mais doit-on considérer un build réussi si les tests échouent? Et si la doc est manquante?
- En général non, pour cette raison le build est mis en échec lorsque les critères qualités ne sont pas au RDV

Idée reçue n°2: Si ça compile, c'est que ça marche

- Le but de build est de s'assurer de la production de nos artefacts MAIS avec une qualité suffisante.
- Mais doit-on considérer un build réussi si les tests échouent? Et si la doc est manquante?
- En général non, pour cette raison le build est mis en échec lorsque les critères qualités ne sont pas au RDV * manque de Documentation
  - utilisation d'une dépendance trop vielle
  - couverture de code insuffisante

UNIVERSITÉ PARIS 1
PANTHÉON SORBONNE

Idée reçue n°2: Si ça compile, c'est que ça marche

- Le but de build est de s'assurer de la production de nos artefacts MAIS avec une qualité suffisante.
- Mais doit-on considérer un build réussi si les tests échouent? Et si la doc est manquante?
- En général non, pour cette raison le build est mis en échec lorsque les critères qualités ne sont pas au RDV * manque de Documentation
  - utilisation d'une dépendance trop vielle
  - couverture de code insuffisante

Qualité insuffisante = Build en échec

Le build est un garde-fou contre une mauvaise qualité logicielle.

Idée reçue n°3: c'est le travail du développeur

Idée reçue n°3: c'est le travail du développeur

- Oui, mais pas que!

Idée reçue n°3: c'est le travail du développeur

- Oui, mais pas que!
- L'équipe qualité s'assure des rapports qualité des artefacts

UNIVERSITÉ PARIS 1
PANTHÉON SORBONNE

Idée reçue n°3: c'est le travail du développeur

- Oui, mais pas que!
- L'équipe qualité s'assure des rapports qualité des artefacts
- L'équipe de gestion de projet s'assure que toutes les fonctionnalités du projet sont présentes

Idée reçue n°3: c'est le travail du développeur

- Oui, mais pas que!
- L'équipe qualité s'assure des rapports qualité des artefacts
- L'équipe de gestion de projet s'assure que toutes les fonctionnalités du projet sont présentes
- L'architecte s'assure que les normes de développement sont respectées

Idée reçue n°3: c'est le travail du développeur

- Oui, mais pas que!
- L'équipe qualité s'assure des rapports qualité des artefacts
- L'équipe de gestion de projet s'assure que toutes les fonctionnalités du projet sont présentes
- L'architecte s'assure que les normes de développement sont respectées
- Le directeur technique doit imposer l'utilisation de mécanismes de production d'artefacts répétables.

UNIVERSITÉ PARIS 1
PANTHÉON SORBONNE

Idée reçue n°3: c'est le travail du développeur

- Oui, mais pas que!
- L'équipe qualité s'assure des rapports qualité des artefacts
- L'équipe de gestion de projet s'assure que toutes les fonctionnalités du projet sont présentes
- L'architecte s'assure que les normes de développement sont respectées
- Le directeur technique doit imposer l'utilisation de mécanismes de production d'artefacts répétables.
- En général le build est préparé par un Robot appartenant à l'usine logicielle (Jenkins, Travis...)

UNIVERSITÉ PARIS 1
PANTHÉON SORBONNE

Idée reçue n°3: c'est le travail du développeur

- Oui, mais pas que!
- L'équipe qualité s'assure des rapports qualité des artefacts
- L'équipe de gestion de projet s'assure que toutes les fonctionnalités du projet sont présentes
- L'architecte s'assure que les normes de développement sont respectées
- Le directeur technique doit imposer l'utilisation de mécanismes de production d'artefacts répétables.
- En général le build est préparé par un Robot appartenant à l'usine logicielle (Jenkins, Travis...)

Build Automatisé

Le build est le travail de tous, et concerne toutes les parties prenantes du logiciel.

Maven pour le build

Qu'est-ce que Maven

- C'est une outil d'automatisation du Build
- Un projet Apache
- Privilégie une approche **convention over configuration** (il faut respecter la maven-way si on souhaite avoir le moins de chose à configurer)
- Toute la configuration est basée sur le fichier pom.xml

```xml
<?xml version="1.0" encoding="UTF-8"?>

<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>

    <artifactId>diploma-generator</artifactId>

    <name>diploma-generator</name>
    <!-- FIXME change it to the project's website -->
    <url>http://www.example.com</url>

    <properties>
        <project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>
        <maven.compiler.source>1.8</maven.compiler.source>
        <maven.compiler.target>1.8</maven.compiler.target>
    </properties>

    <dependencies>
        <dependency>
            <groupId>org.junit.jupiter</groupId>
            <artifactId>junit-jupiter-api</artifactId>
            <scope>test</scope>
        </dependency>
        <!--   https://bugs.eclipse.org/bugs/show_bug.cgi?id=538956   -->
        <dependency>
            <groupId>junit</groupId>
            <artifactId>junit</artifactId>
            <version>4.12</version>
            <scope>test</scope>
        </dependency>
```

UNIVERSITÉ PARIS 1
PANTHÉON SORBONNE

Qu'est-ce que Maven (II)

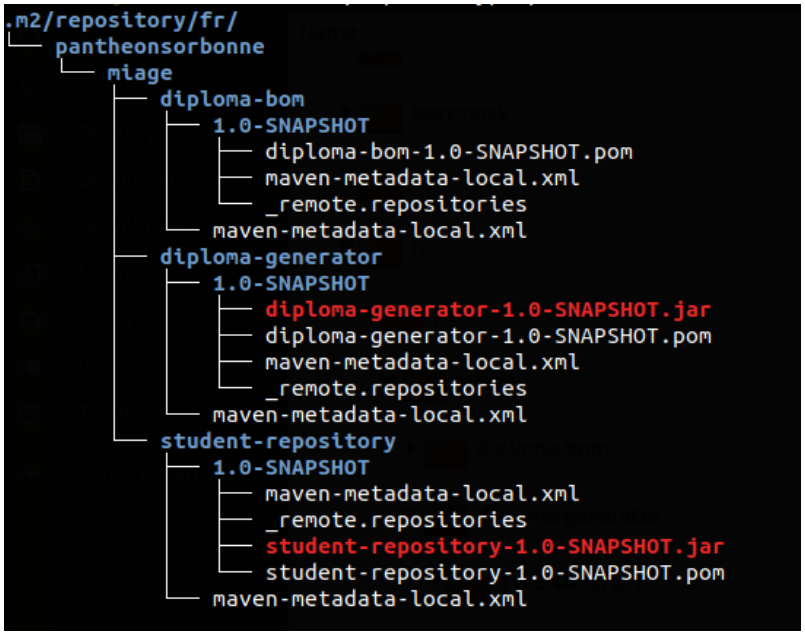
- 1 build = 1 ou plusieurs artefacts
- Les artefacts sont classifiés par

```
<groupId>fr.pantheonsorbonne.miage</groupId>
<artifactId>diploma-bom</artifactId>
<version>1.0-SNAPSHOT</version>
<packaging>pom</packaging>
```

UNIVERSITÉ PARIS 1
PANTHÉON SORBONNE

Qu'est-ce que Maven (III)

- Les artefacts, une fois installés sont stockés sur le disque sous forme hiérarchique /[groupId]/[artifactId]/[version]/[artifactId]-[version].[ext]
- Si notre projet dépend d'autres artefacts, ils seront récupérés dans le cache local (répertoire .m2) ou téléchargé automatiquement depuis le Maven Central Repository



UNIVERSITÉ PARIS 1
PANTHÉON SORBONNE

Le Build avec Maven

- 1 *goal* est une action
- 1 *phase* est une collection de *goals*
- 1 *lifecycle* est une séquence de *phases*
- De nombreux plugins fournissent de nouveaux *goals* et peuvent les associés aux phases existantes

UNIVERSITÉ PARIS 1
PANTHÉON SORBONNE

Lifecycle par défaut

- validate # *est-ce que tout est prêt pour le build?*
- compile # *compulation à proprement parler*
- test # *effectue les tests unitaires*
- package # *assemble les fichiers compilés (e.g. *.class => jar)*
- verify # *s'assure que les critères qualités sont satisfaits*
- install # *installe les artefacts dans le cache local*
- deploy # *déploie les artefact dans un dépôt distant*

UNIVERSITÉ PARIS 1
PANTHÉON SORBONNE

Nos premières commandes

- supprime tout ce qui a été fait précédemment, valide, compile, teste et génère les artefacts
- peut prendre du temps la première fois (téléchargement)

```
$ mvn clean install
```

Dépendances I

- Les dépendances sont téléchargées automatiquement du dépôt maven Central
- Toutes les installations de maven possèdent un cache des dépendances (répertoire .m2)
- Les dépendances sont gérées dans le pom.xml

```xml
<dependencies>
    <dependency>
        <groupId>org.junit.jupiter</groupId>
        <artifactId>junit-jupiter-api</artifactId>
        <scope>test</scope>
    </dependency>
    <!-- https://bugs.eclipse.org/bugs/show_bug.cgi?id=538956 -->
    <dependency>
        <groupId>junit</groupId>
        <artifactId>junit</artifactId>
        <version>4.12</version>
        <scope>test</scope>
    </dependency>
    <dependency>
        <groupId>org.junit.jupiter</groupId>
        <artifactId>junit-jupiter-engine</artifactId>
        <scope>test</scope>
        <version>5.1.0</version>
    </dependency>
    <dependency>
        <groupId>fr.pantheonsorbonne.miage</groupId>
        <artifactId>student-repository</artifactId>
        <version>1.0-SNAPSHOT</version>
    </dependency>
    <dependency>
        <groupId>com.itextpdf</groupId>
        <artifactId>itextpdf</artifactId>
        <version>5.5.10</version>
    </dependency>
    <dependency>
        <groupId>org.apache.pdfbox</groupId>
        <artifactId>pdfbox</artifactId>
        <version>2.0.12</version>
        <scope>test</scope>
    </dependency>
```

UNIVERSITÉ PARIS 1
PANTHÉON SORBONNE

Dépendances II

- Les dépendances peuvent être gérées directement dans le pom.xml de l'artefact ou dans son parent via la balise

Création de projet Maven

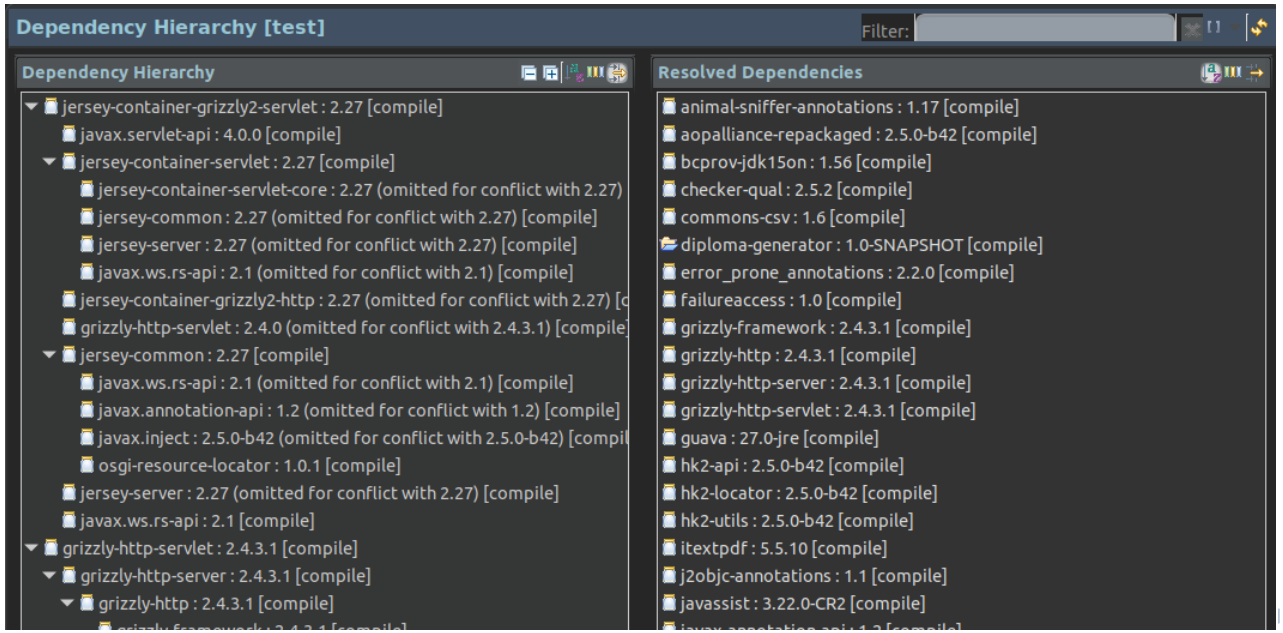
- utiliser “un exemple d’internet”
- ou utiliser maven directement et se laisser guider (pensez-y pour vos prochains projets!)

```
shell $ mvn archetype:generate
-DarchetypeGroupId=org.apache.maven.archetypes
-DarchetypeArtifactId=maven-archetype-quickstart
```

UNIVERSITÉ PARIS 1
PANTHÉON SORBONNE

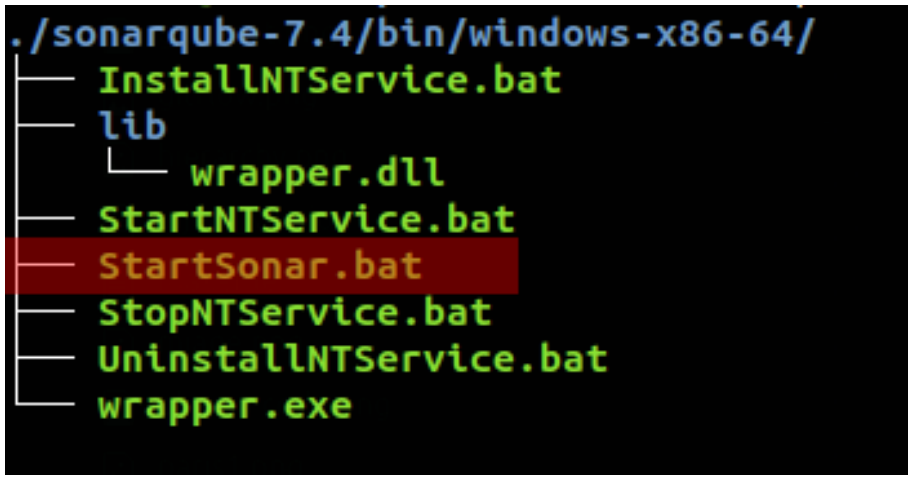
Intégration aux IDE (Eclipse, Netbeans, IntelliJ)

- Eclipse : File > Import > Maven > Existing Maven Project. Placez le Root directory dans le dossier contant votre pom



Utilisation avancée de maven: sonaar

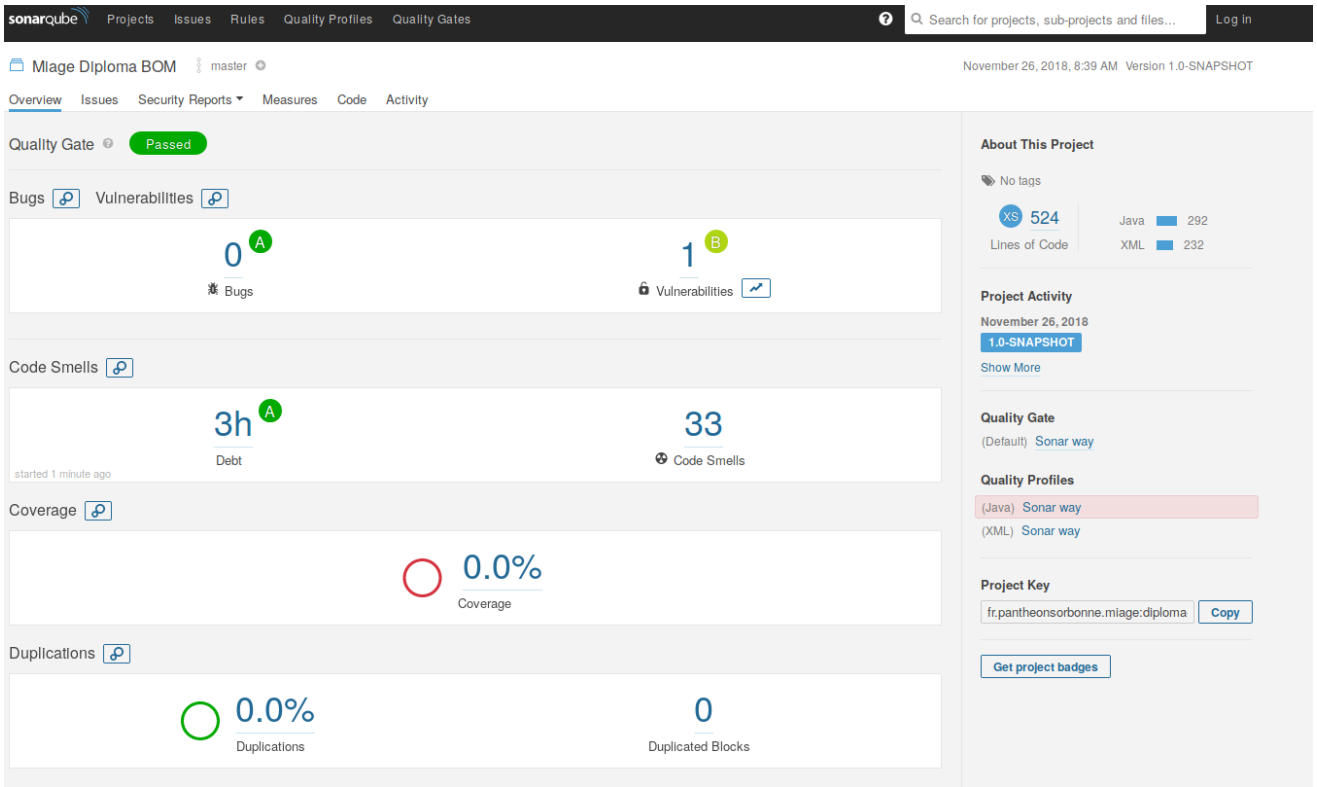
Après avoir lancé le server Sonar




```
$ mvn clean install
$ mvn sonar:sonar
```

UNIVERSITÉ PARIS 1
PANTHÉON SORBONNE

Interace Graphique sonaar



UNIVERSITÉ PARIS 1
PANTHÉON SORBONNE