- versions , releases

- generate javadocs, other site information, sources jar file

- produkuje artifact w podanej wersji

- wszystkie akcje to pluginy

- ant to scripting tool, nie ma standard nazywania

- maven – build tool, sa standardy

- musi byc odpowiednia struktura projektu, np. Src/main/groovy dla groovy

- pluginy

<**plugin**>  
 <**groupId**>org.apache.maven.plugins</**groupId**>  
 <**artifactId**>maven-surefire-plugin</**artifactId**>  
 <**version**>2.19.1</**version**>  
 </**plugin**>  
 <**plugin**>  
 <**groupId**>org.sonarsource.scanner.maven</**groupId**>  
 <**artifactId**>sonar-maven-plugin</**artifactId**>  
 <**version**>3.0.1</**version**>  
 </**plugin**>  
</**plugins**>

- w target zbudowany kod, z tad testy startują

- pom z 4 części

1. info o projekcie

2. dependencies

3. build

-- plugins, directory structure

4. repositories

-- skąd pobieramy, default central maven repo, ale mozna swoje artifactory/nexus, spring…

-- jak potrzebne credentials/proxy to w settings.xml

<**distributionManagement**>  
 <**repository**>  
 <**id**>yy</**id**>  
 <**url**>http…/releases/</**url**>  
 </**repository**>  
 <**snapshotRepository**>  
 <**id**>xx</**id**>  
 <**url**>http://.../snapshots/</**url**>  
 </**snapshotRepository**>  
</**distributionManagement**>

- przy dependencji group/artifact I version

- install pakuje I wrzuca nowa wersja do local repository .m2

- deploy – może robić deploy na corpo repository jak nexus

- w sekcji build mozna nadpisywac default gole – np. Nadpisanie nazwy w build><finalName>

- jak zaczynamy kodować to mamy snapshot

- jak SNAPSHOT jest, to za kazdym razem jak kompiluje sprawdza, czy nie ma nowej wersji ,

Zmiany zawsze zaciągane,

- wersjonowanie mozna róznie, mozemy uzyc 4rech rodzajów liczb

- czasem są dodatkowe nazwy –MA milestone release, -RC1/RC2– release candidate, -RELEASE or FINAL – jako release

- rózne typy pakowane, zazwyczaj zipy inne

- może byc typ pom, jako typ dependencji – pobież dependencje z pom

- ustawiamy to w <packaging>

- TRANSITIVE DEPENDENCIES – zaciągane z zadeklarowanej

- 6 scopes – compile – default, available everywhere, provided – provided when deployed

- runtime – available during execution, not included in final artifact

- test – only for testing, not included

- system, specify path in file system, do not use

- import – with dependency management works, advanced

- z transitive wzięta nowsza – DOCZYTAĆ !!!

- pokazuje na drzewku która nadpisana przez którą

--

Repo to miejsce ggdzie trzymane biblioteki, może byc zdalnie lub lokalnie

- http accesible location

- default jest repo.maven.apache.org, ale mozna wiele , mozna też własne nexus/artifatory

- own repo



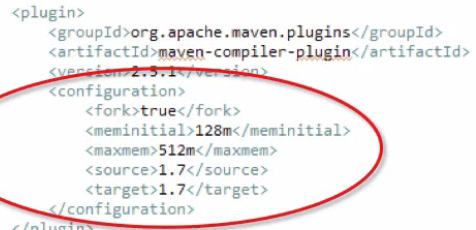
- plugins repository, to samo ale dla pluginów, bto samo, tylko inne tagi początkowe



- częto zworzy się osobne repo dla snapshot i dla release

--

Pluginy do clean budowania i całego phase, compiler wywołuje javac z classpath ustawione tam gdzie dependency ściągane



- jest plugin dla jar, można w nim manifest, mozna dodać sources