Koontityökalut

Apache Maven™

Äiti, mitä on koonti?

- Build ~ koonti, valmiste, kokoonpano, pystytys, paketointi

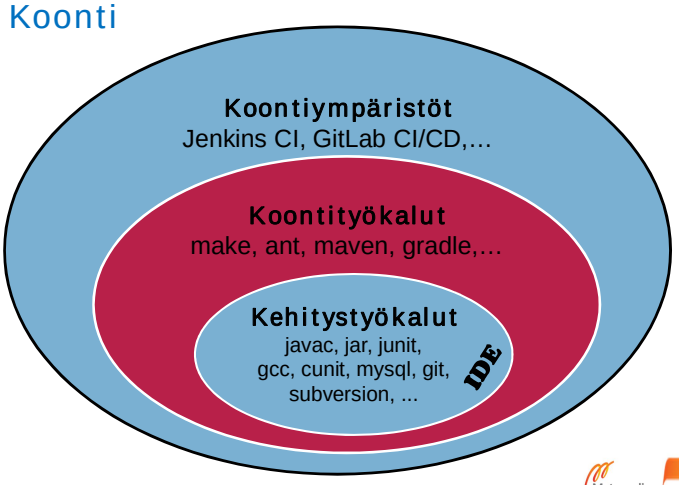
1) Normaalien kääntämistä vaativien ohjelmointikielten (kuten C, C++, Java ja C#) kääntämistä konekielelle
2) Tulkattavien sovellusten (kuten Perl ja Python) pakkaamista ja testausta
3) Web-sovellusten kääntämistä ja pakkaamista. Näissä on staattisia ja dynaamisia HTML-sivuja, lähdekielistä ohjelmaa Javalla tai C#:lla, Javascriptiä, PHP-koodia ja konfigurointitiedostoja
4) Yksikkötestien ajamista koodin testaamiseksi
5) Staattisen analyysin työkalujen ajamista: näistä tuloksena on yleensä raportti jossain muodossa
6) PDF- or HTML-dokumentaation generointia
7) ...

Miksi koontityökaluja?

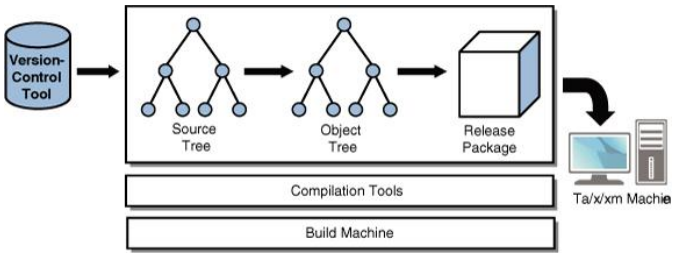
- Mahdollistavat koontivaiheiden yhtenäistämisen
  - yhdellä komennolla kaikki tehdyksi
- Projektin voi viedä IDE:n ulkopuolelle ja koostaa komentoriviltä muulla koneella
  - esim. Jenkins-palvelimella
- Tai projekti voidaan viedä toiseen IDE:een ja koonti toimii

- Tällä kurssilla suositellaan tekemään suoraan Maven-projekti
  - Helpottaa hengitystä, kun myöhemmin mukaan otetaan jatkuva koonti (continuous integration, CI)
  - Koonti Jenkins-palvelimella

Koonti



Perinteinen käännettävä kieli



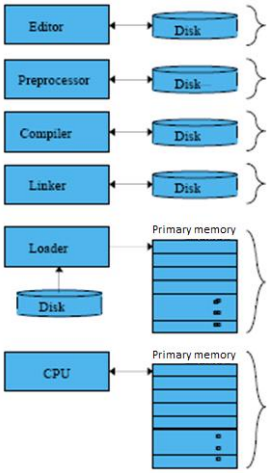
Source Tree =
Tiedosto- ja hakemistorakenne lähdekielisille tiedostoille
- Javassa esim. src-hakemisto ja test-hakemisto

Release Package =
Jakelupakkaus, Javassa jar-pakkaus

Object Tree =
Yleensä erillinen tiedosto- ja hakemistorakenne objektitiedostoille
- Eclipse: Javalle bin-hakemisto
- NetBeans: Javalle build/classes

C-ohjelman koonti ja suoritus

Vaiheet
1. editointi
2. esiprosessointi
3. kääntäminen
4. linkittäminen

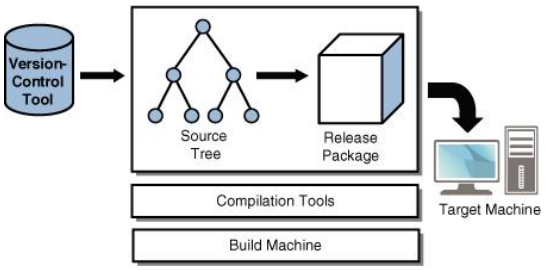


Program is created in the **editor** and stored on disk.
Preprocessor processes the code.
Compiler creates object code and stores it on disk.
Linker links the object code with the libraries, creates a.out and stores it on disk

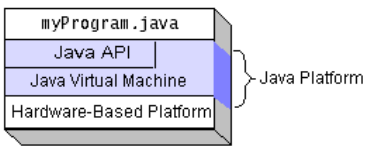
Loader puts program in memory

CPU takes each instruction and executes it, possibly storing new data values as the program executes

Tulkattava kieli

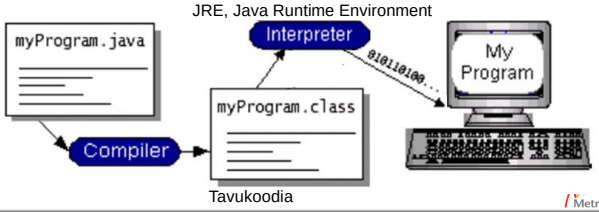


Java-ohjelman koonti




```
public Hello();
 Code:
   Stack=1, Locals=1, Args_size=1
   0:  aload_0
   1:  invokespecial   #8; //Method
java/lang/Object."<init>":()V
   4:  return

public static void main(java.lang.String[]);
 Code:
   Stack=2, Locals=1, Args_size=1
   0:  getstatic       #12; //Field
java/lang/System.out:Ljava/io/PrintStream;
   3:  ldc     #18; //String Hei Metropolia
   5:  invokevirtual   #20; //Method
java/io/PrintStream.println:(Ljava/lang/String;)
   8:  return
```

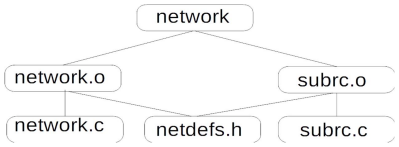


make

Makefile

n. 1977

- Aluksi
  - kaikki koonnissa tarvittavat komennot kirjoitettiin käsin komentokehotteeseen
  - samojen komentorimpsujen toistoa
- Makefile
  - vaiheiden komennot Makefile-nimiseen tiedostoon
  - kuvaa riippuvuuksia ja niihin liittyvät komennot
  - sekä implisiittisiä että eksplisiittisiä riippuvuuksia
  - make-komento käsittelee vaiheet
- make tutkii tiedostojen aikaleimoja
  - huomaa muuttuneet moduulit ja suorittaa tarvittavat minimitoiminnot
  - jos esim. network.c muuttuu, niin tuotettava uusi network.o
- Makefileen voi kerätä muitakin komentokokonaisuuksia



- network-nimisen sovelluksen riippuvuusverkko

kohde → network: network.o subrc.o ← riippuvuudet

komennot → gcc -o network network.o subrc.o

```
$cat Makefile
network: network.o subrc.o
	gcc -o network network.o subrc.o
network.o: network.c netdefs.h
	gcc -c network.c -o network.o
subrc.o: subrc.c netdefs.h
	gcc -c subrc.c -o subrc.o
clean:
    rm –f *.o core
$make clean
$make
```


ANT

Koontityökalu Ant ("Another Neat Tool")

n. 2000

- Java-projekteja varten kehitetty koontityökalu
- Koonti kuvataan xml-tiedostossa
  - vastaa Makefile-tiedostoa
  - `build.xml`
- Eclipsessä syntyy oletuksena ant-projekti
  - valmiina ant-plugin
- Koonnin voi käynnistää komentoriviltä tai IDE:n kautta
  - `ant build.xml`
- Kaikki project-elementin sisällä, kohteet ilmaistaan target-elementeillä

Esimerkki

kohde → target name="compile"

komennot → mkdir, javac

```
<!– build.xml Ant build file-->
<project>
    <description>Simple example build file</description>
    <target name="clean">
        <delete dir="build"/>
    </target>
    <target name="compile">
        <mkdir dir="build/classes"/>
        <javac srcdir="src" destdir="build/classes"/>
    </target>
    <target name="jar">
        <mkdir dir="build/jar"/>
        <jar destfile="build/jar/HelloWorld.jar" basedir="build/classes">
            <manifest>
                <attribute name="Main-Class" value="oata.HelloWorld"/>
            </manifest>
        </jar>
    </target>
    <target name="run">
        <java jar="build/jar/HelloWorld.jar" fork="true"/>
    </target>
</project>
```

```
$ ant –v –f build.xml
```


MAVEN

Koontityökalu Maven

n. 2004

- Entistä enemmän automatisoitu koonti
- Maven nojaa asioiden standardoimiseen
  - vakiohakemistorakenne
  - vakionimeämistapa moduuleille
  - tietyt perustoimet menevät samalla tavalla projektista riippumatta
  - yksi ja sama oletuskoontiskripti (XML:ää) sopii kaikkien projektien pohjaksi
- Projektinhallinnan yksityiskohtien määrittelyn voi sivuuttaa
  - kunhan noudattaa Mavenin tapaa tehdä asiat
  - Mavenille kuvataan/esitellään asioita ja toimintatapoja, ei konfiguraatiota
- Yhtenäinen ja selkeä riippuvuuksienhallinta
  - osaa hakea Mavenin omasta repositoriosta
  - osaa hakea julkisista repositorioista
  - osaa hakea paikallisesta järjestelmästä
- Laajennettavissa
  - voi täydentää liitännäisillä (plugins)

Apache Maven™

Maven-projektin luonti

- Riippuvuus (dependency)
  = projektissa tarvittava kirjastopakkaus
- Liitännäinen / lisäosa (plugin)
  = projektissa tarvittava koodia suorittava lisäosa
- Archetype (perikuva, perustyyppi, malli)
  - Maven tunnistaa valmiita riippuvuusmäärittelyjen perusmalleja erilaisille projekteille
    - Myös JavaFX-projekteille on omansa
  - Riippuvuudet voi myös kirjata käsin pom.xml:ään

- Eclipse
  `File | New | Project | Maven | Maven Project`
  - Laita ruksi ruutuun "Create a simple project (skip archetype selection)" ja merkitse tarvittavat riippuvuudet ja lisäosat käsin pom.xml:ään

GAV – GroupId, ArtifactId, Version

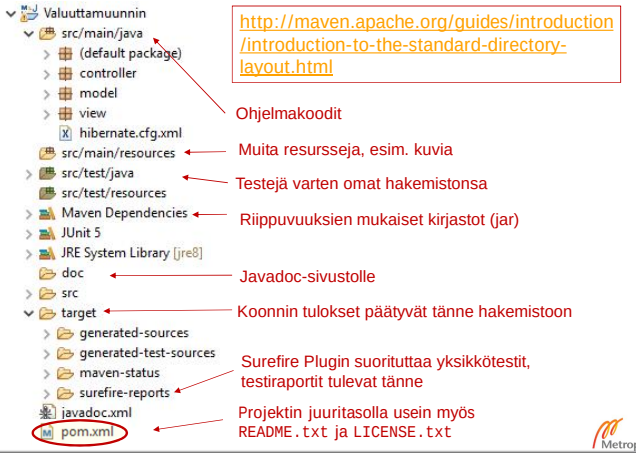
- groupId
  - yksilöi yrityksen tai ryhmän, jolle projekti (tuote) kuuluu
  - tavallisesti käänteinen domainnimi, esim. fi.metropolia.otp
  - määrää myös luotavan pakkausrakenteen
- artifactId
  - yksilöi tuotteen nimen
  - käytetään juurihakemiston nimenä

Näitä käytetään, kun riippuvuus kirjataan pom.xml:ään

- version
  - tuotteen versionumero
  - jos lopussa sana SNAPSHOT tarkoittaa as-yet-unreleased
  - ks. docs.oracle.com/middleware/1212/core/MAVEN/maven_version.htm

- Tuotoksesta käytetään termiä artefakti
  - ihmisen tekemä esine, rakennelma ym.
  - esim. suoritettava koodi, dokumentti, tms.

Vakioitu hakemistorakenne

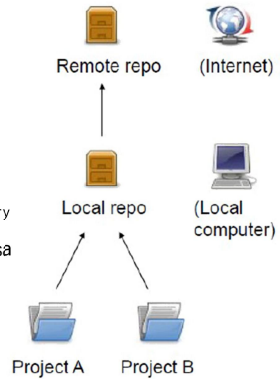


http://maven.apache.org/guides/introduction/introduction-to-the-standard-directory-layout.html

Ohjelmakoodit
Muita resursseja, esim. kuvia
Testejä varten omat hakemistonsa
Riippuvuuksien mukaiset kirjastot (jar)
Javadoc-sivustolle
Koonnin tulokset päätyvät tänne hakemistoon
Surefire Plugin suorituttaa yksikkötestit, testiraportit tulevat tänne
Projektin juuritasolla usein myös `README.txt` ja `LICENSE.txt`

Riippuvuuksien hallinta

- Sovelluksen tarvitsemat kirjastot määritellään pom.xml:ssä
- Maven osaa hakea kirjastot repostoriosta
  - Mavenin ylläpitämästä etärepositoriosta
    - `https://repo.maven.apache.org/maven2`
    - ks. myös `https:\\mvnrepositorio.com`
  - Paikallisella koneella olevasta repostitoriosta
    - Linux: `/home/tunnus/.m2/repository`
    - Windows: `C:\Users\tunnus\.m2\repository`
- Maven-repositorio on yhteinen kaikille samassa koneessa oleville projekteille



pom.xml (Project Object Model)

- Projektiin liittyvä metadata
- Määrittelee projektin rakentamiseen, kääntämiseen, testaamiseen, dokumentointiin, jakeluun jne. liittyvät vaiheet
- pom.xml:n perusteella Maven osaa mm.
  - hakea riippuvuuksissa kuvatut ohjelmistokirjastot repositorioista
  - kääntää
  - ajaa testit
  - paketoida projektin jakelukuntoon
- Mavenin voi konfiguroida myös hoitamaan projektin tuotoksen jakelemisen eteenpäin
  - ks. DevOps-sykli
- Verkosta saatavilla paljon ulkopuolisia liitännäisiä (plugins) erikseen yksilöitäviä tehtäviä varten ks. maven.apache.org/plugins/index.html

- Myös Maven käyttää JDK:ta
  - Tarvitsee mm. myös javac-kääntäjää

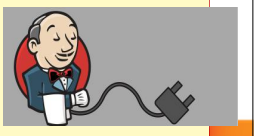
pom.xml: JavaFX

```
<project xmlns="http://maven.apache.org/POM/4.0.0" ...>
  <modelVersion>4.0.0</modelVersion>
  <groupId>fi.metropolia.OTP</groupId>
  <artifactId>OtpSovellus</artifactId>
  <version>0.0.1-SNAPSHOT</version>
  <description>JavaFX, MariaDB, JDBC, Hibernate, JUnit </description>

  <properties>
    <!-- Merkistö UTF-8, Java 11 kirjastojen mukainen source ja target -->
    <project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>
    <maven.compiler.release>11</maven.compiler.release>
  </properties>

  <dependencies>
    <!-- JavaFX-projekti -->
    <dependency>
        <groupId>org.openjfx</groupId>
        <artifactId>javafx-controls</artifactId>
        <version>11</version>
   </dependency>
```

Sopivia kikkareita löytyy netistä.



pom.xml: MariaDB+JDBC, Hibernate

```
    <!-- JavaFX-projekti, joka käyttää FXML:ää-->
    <dependency>
      <groupId>org.openjfx</groupId>
      <artifactId>javafx-fxml</artifactId>
      <version>11</version>
    </dependency>
    <!-- MariaDB tietokanta-Client ja Java Database Connector -->
    <dependency>
       <groupId>org.mariadb.jdbc</groupId>
       <artifactId>mariadb-java-client</artifactId>
       <version>2.7.4</version>
    </dependency>
    <!-- Hibernate JPA, automatisoi tietokannan käyttöä -->
    <dependency>
       <groupId>org.hibernate</groupId>
       <artifactId>hibernate-core</artifactId>
       <version>5.6.3.Final</version>
    </dependency>
```


pom.xml: JUnit Jupiter

```
    <!-- JUnit Jupiter, yksikkötestausta varten -->
    <dependency>
       <groupId>org.junit.jupiter</groupId>
       <artifactId>junit-jupiter-engine</artifactId>
       <version>5.7.2</version>
       <scope>test</scope>
    </dependency>
  </dependencies>
```

pom.xml: lisäosia

```
<build>  <!-- Koonnissa ja raporttien tuottamisessa tarvittavia lisäosia -->
  <plugins>
  <plugin>
      <!-- Java-koodin kääntäminen (kun ei IDE:n kautta) -->
      <groupId>org.apache.maven.plugins</groupId>
      <artifactId>maven-compiler-plugin</artifactId>
      <version>3.8.1</version>
      <configuration>
         <release>11</release>
      </configuration>
  </plugin>
  <!-- JUnit-testien käynnistys (kun ei IDE:n kautta) -->
  <plugin>
      <groupId>org.apache.maven.plugins</groupId>
      <artifactId>maven-surefire-plugin</artifactId>
      <version>3.0.0-M5</version>
  </plugin>
```

- Pluginin oletus: source 1.6 ja target 1.6
- Java 17 viimeisin LTS-versio (long-term-support) (edeltävä Java 11)

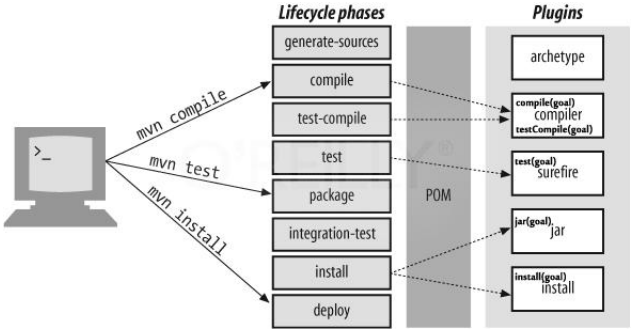
pom.xml: lisäosia

```
      <plugin>
          <!-- JavaFX-koodin GUI:n käynnistys IDE:stä -->
          <groupId>org.openjfx</groupId>
          <artifactId>javafx-maven-plugin</artifactId>
          <version>0.0.7</version>
          <configuration>
              <mainClass>com.example.App</mainClass>
          </configuration>
      </plugin>
    </plugins>
  </build>
</project>
```

Sen luokan polku, jossa main()

Oletuselinkaaret

- Kolme sisäänrakennettua elinkaarimallia
  - default koonnin perustoiminnot
  - clean koonnissa tuotettujen "asioiden" poisto
  - site sovelluksen käyttöönotto, dokumenttien generointi
- Elinkaari koostuu vaiheista, esim. default-elinkaaren vaiheet ovat
  - validate - validate the project is correct and all necessary information is available
  - compile - compile the source code of the project
  - test - test the compiled source using a suitable unit testing framework
  - package - take the compiled code and package it in its distributable format, such as a JAR
  - verify - run any checks on results of integration tests to ensure quality criteria are met
  - install - install the package into the local repository, for use as a dependency in other projects locally
  - deploy - done in the build environment, copies the final package to the remote repository for sharing with other developers and projects



- Tietyn elinkaarivaiheen voi käynnistää erikseen
  - tällöin Maven suorittaa kaikki sitä edeltävätkin elinkaarivaiheet
  - esim. komento `mvn clean deploy`
    - poistaa kaikki vanhaan koontiin liittyvän
    - tekee uuden käännöksen
    - suorituttaa kaikki testit
    - tekee jar-pakkauksen, siirtää sovelluksen palvelimelle käytettäväksi

Mavenin ja Antin eroja

Maven
- Tulospohjainen, esittelevä prosessikuvaus
- Projektin rakenteelle ja koonnille vakiomalli, joka sopii (tai ei sovi)
- Elinkaari ja vaiheet oletusarvot
- Ylläpitää automaattisesti riippuvuuksia (kirjastot)
- Paljon valmiita lisäosia

Ant
- Hyvä yksityiskohtainen ja vaiheittainen prosessikuvaus
- Vaiheet määritellään itse
- Projektin rakenteen ja koontiprosessin voi määrittää halutunlaisiksi
- Riippuvuuksien hallinta mahdollista ulkoisin työkaluin

Apache Maven™ <APACHE ANT>

GRADLE

Koontityökalu Gradle

n. 2007

- Yleistyi, kun 2013 Google alkoi käyttää Android-studiossa
- Pyrkii yhdistelemään Mavenin ja Antin hyviä puolia
  - tuloksena monipuolinen, moneen taipuva koontityökalu
  - nopeampi koonti
  - mutta samalla monimutkainen, vaikea omaksua ja ymmärtää
- Mavenin tapaan Gradlella vakiokäytäntöjä
- Käytännöistä poikkeaminen kerrottava koontitiedostosssa
- Koontitiedosto (build.gradle) Groovy-skriptikielellä
- Osaa hakea riippuvuuksia sekä Ivyn että Mavenin repositorioista

Gradle

Esimerkki

```
// build.gradle
// Top-level build file where you can add configuration options
// common to all sub-projects/modules.
buildscript {
    repositories {
        google()
        jcenter()
    }
    dependencies {
        classpath 'com.android.tools.build:gradle:3.3.2'
        // NOTE: Do not place your application dependencies here; they belong
        // in the individual module build.gradle files
    }
}
allprojects {
    repositories {
        google()
        jcenter()
    }
}
```

Vertailua

- Ant vs. Maven vs. Gradle
  - https://www.baeldung.com/ant-maven-gradle
- Apache Ant vs. Gradle vs. Apache Maven
  - https://stackshare.io/stackups/ant-vs-gradle-vs-maven