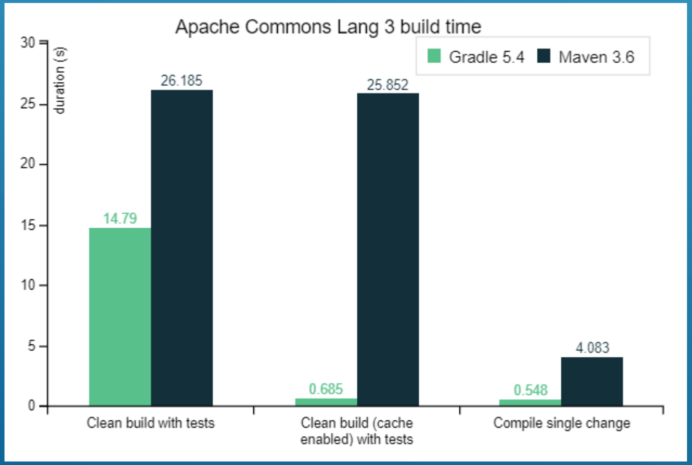
GRADLE

Breve introduzione

- Build Automation
  - gestione dipendenze (= maven, ivy, ant)
    - dinamiche: 3.+
  - compilazione, test, esecuzione (= maven, ant)
  - distribuzione / rilascio (= maven)

- Punti di forza
  - performance (build incrementale, cache, daemon)
  - flessibilità (DSL, moduli, plugin)
  - popolarità (IDE, docs, plugins.gradle.org)
  - maven repo



- Punti deboli
  - complessità: learning curve (API, lifecycle)
  - immaturità: API instabile, plugin obsolete, upgrade problematico
  - documentazione: confusa/inadeguata
  - pesantezza: RAM, processi daemon
  - cache
  - “magia”

ESEMPI

in groovy

BUILD MINIMALE

```
build.gradle:

plugins {
    id 'java'
    id 'war'
}

group = 'com.example'
version = '1.0-SNAPSHOT'

sourceCompatibility = '17'

repositories {
    mavenCentral()
}

dependencies {
    implementation 'javax.servlet:javax.servlet-api:4.+'
    testImplementation 'junit:junit:4.13.2'
}

war {
    archiveFileName = 'example.war'
}
```

```
C:\test> gradle build
```

Anche:

- dipendenze tra task
- minificazione js
- conversione SASS
- creazione war da più fonti
- exclude su dipendenze transitive
- build incrementale con “inputs” e “outputs”
- autoBuildTasks

DEPLOY 1/2

```
deploy.config:

server {
    tomcatuser = "tomcat8"
}

environments {
    prod {
        server {
          user = "produser"
          identity = new File("C:\\prod.pem")
          host = "www.example.com"
          tomcatbase = "/srv/myprod/tomcat1"
        }
    }
    col {
        server {
          user = "coluser"
          identity = new File("C:\\col.pem")
          host = "col.example.com"
          tomcatbase = "/srv/mycol/tomcat1"
        }
    }
}
```

```
build.gradle:

plugins {
    id 'org.hidetake.ssh' version '2.11.2'
}

def config = new ConfigSlurper(env).parse(new
File("$projectDir/deploy.config").toURI().toURL())

remotes {
    deployHost {
      host = config.server.host
      user = config.server.user
      identity = config.server.identity
      knownHosts = addHostKey(file("known_hosts"))
      jschLog=true
    }
}
```

DEPLOY 2/2

```
build.gradle:

task deployWar(dependsOn: [makeWar]) {
    doLast {
        ssh.run {
            session(remotes.deployHost) {
                put from: "$projectDir/env/${env}/bin/deploy.sh", into: "/srv/${acroenv}/bin"
                put from: warPath, into: "/srv/${acroenv}/deploy/$remoteWarName"
            }
        }
        ssh.run {
            session(remotes.deployHost) {
                execute """chmod a+x /srv/${acroenv}/bin/deploy.sh && ...""«
        ...
```

SUBPROJECTS

```
settings.gradle:

rootProject.name = ‘ExampleSite'
include 'YadaWeb'
project(':YadaWeb').projectDir = "../../yadaframework/YadaWeb" as File
include 'YadaWebCMS'
project(':YadaWebCMS').projectDir = "../../yadaframework/YadaWebCMS" as File
```

```
build.gradle:

dependencies {
    implementation   project(':YadaWeb')
    implementation    project(':YadaWebSecurity')
    implementation    project(':YadaWebCMS')
    implementation    project(':ArtemideCommon')
    implementation ...
```

FLEXIBLE SUBPROJECTS

```
gradle.local.properties:

yadaSourceRepoPath = /../../yadaframework
yadaProjects = YadaWeb,YadaWebCMS
```

```
settings.gradle:
Properties localProps = new Properties()
File localPropsFile = file('gradle.local.properties')
if(localPropsFile.exists()) {
    localPropsFile.withInputStream {
        localProps.load(it)
    }
...
```

```
build.gradle:
repositories {
    mavenCentral()
   maven {
       url "file:$projectDir/yadarepo"
   }
}
dependencies {
   if (yadaSourceRepoPath==null) {
```

PLUGIN

```
YadaCreateDbSchemaTask.groovy:

class YadaCreateDbSchemaTask extends DefaultTask {
    @OutputFile
    def outputfilename = "generated.sql";
    @Internal
    def update = false;

    @TaskAction
    def createDbSchema() {
        File fromFile =
project.sourceSets.main.resources.files.find({it.name=='persistence.xml'})
        File toFolder = new File("$project.buildDir/classes/java/main/META-INF");
...

build.gradle:

task schema(type: net.yadaframework.tools.YadaCreateDbSchemaTask) {
    inputs.files configurations.hibtools
    outputfilename = "V1__yadatest.sql"
    }
```