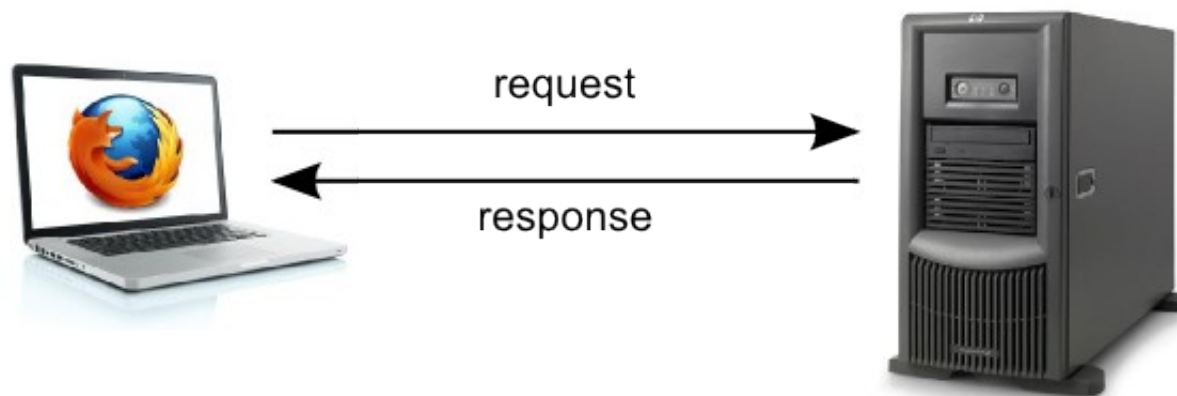


Úvod do Java EE

Klient – Server mechanismus

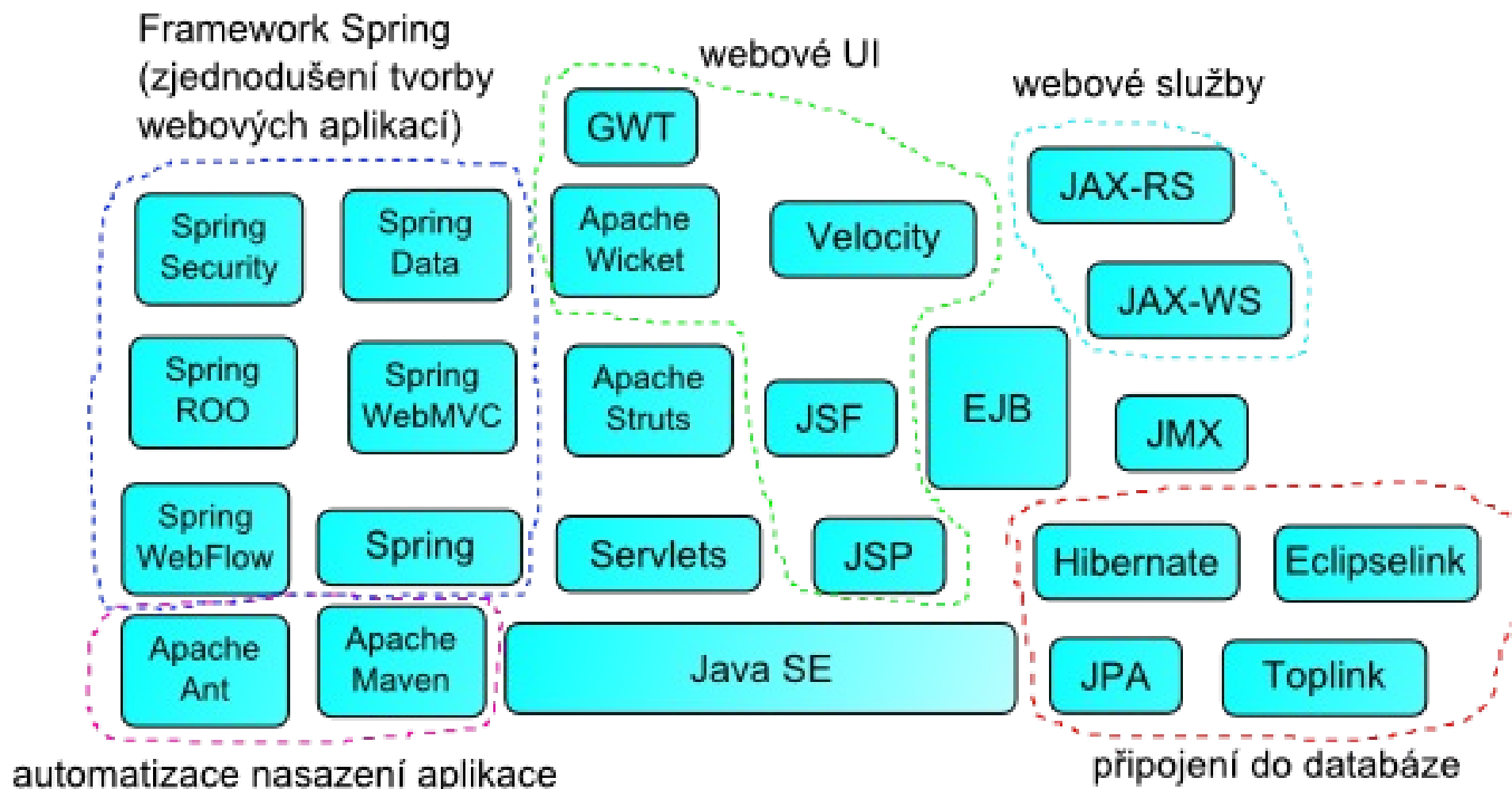
- V každé webové aplikaci komunikuje klient (obvykle webový prohlížeč, ale může to být i jiná aplikace) se serverem
 - Komunikují prostřednictvím request / response mechanismu.
 - Pro fungování tohoto mechanismu musí komunikovat „společnou řečí“. Pro přenos dat se používá HTTP protokol a pro zobrazení dat obvykle jazyk (X)HTML nebo XML.



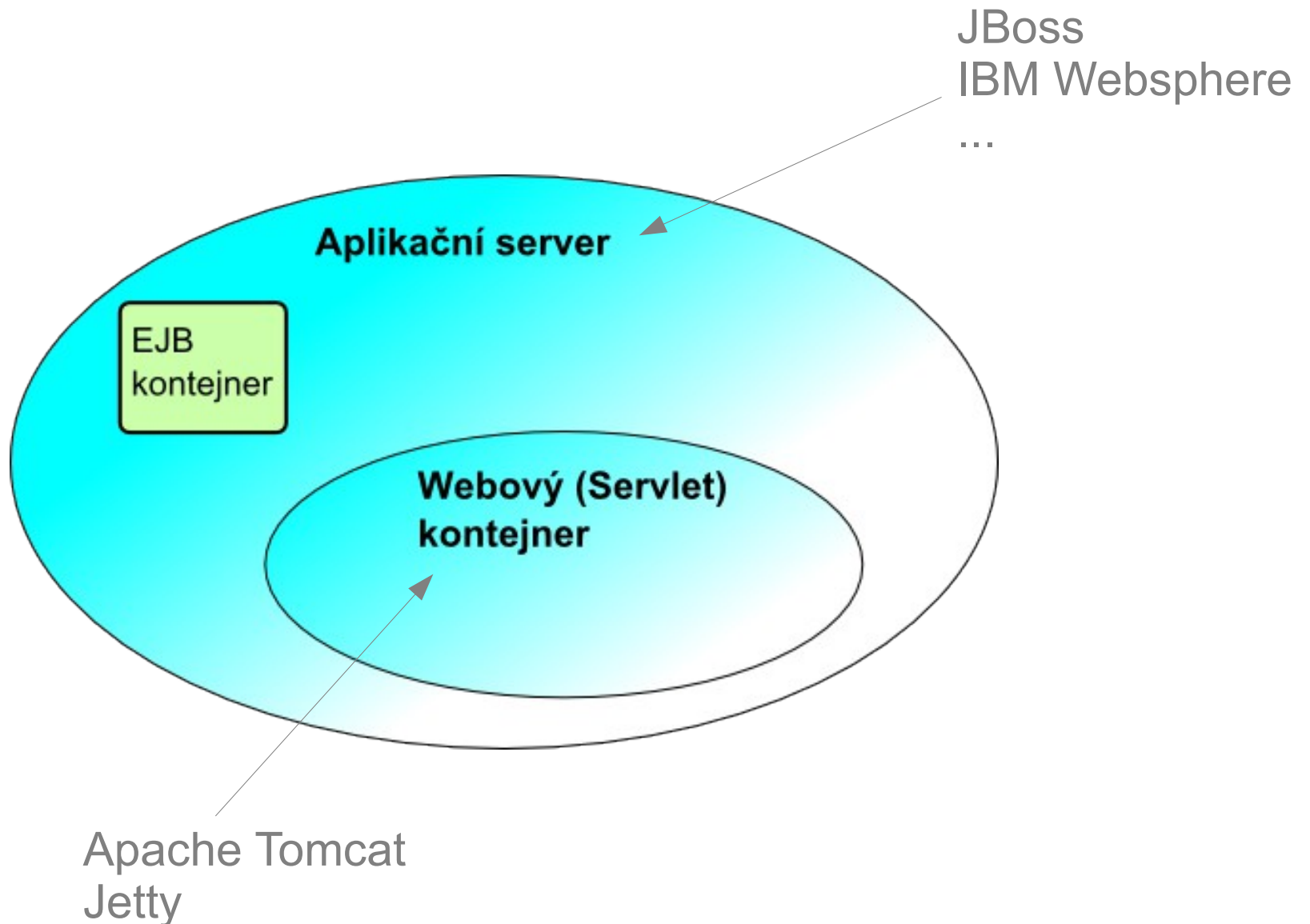
Přehled Java EE serverů

- Na trhu existuje celá řada Java EE serverů:
 - Apache Tomcat
 - Jetty
 - JBoss
 - GlassFish
 - IBM Websphere
 - Oracle (dříve BEA) WebLogic
 - Oracle Application Server
 - a další ...
- Některé jsou open source (Apache Tomcat, Jetty), jiné jsou placené.
- Všechny Java EE servery implementují standardní Java EE specifikace, tudíž je obvykle možné naprogramovat aplikaci na jednom serveru a spustit ji na jiném.

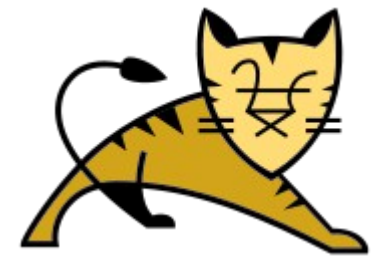
Přehled Java EE platformy



webový kontejner vs. aplikační server



Apache Tomcat 101



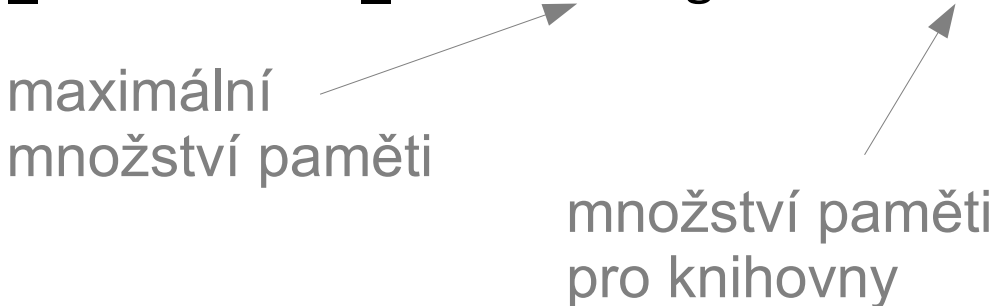
- Nejpoužívanější Java EE webový server. Jedná se o open source.
- **Spuštění Apache Tomcat:**
 - 1) Stáhnout ZIPku s Tomcatem
 - 2) Nastavit proměnnou JAVA_HOME na umístění JDK
 - 3) Nebo nastavit proměnnou JRE_HOME na umístění JRE
 - 4) Spustit [apache-tomcat]/bin/startup.bat
 - 5) Přejít v prohlížeči na webovou stránku <http://localhost:8080/>
- **Zastavení Apache Tomcat:**
 - Spustit [apache-tomcat]/bin/shutdown.bat

Apache Tomcat – konfigurace

- V [apache-tomcat]/bin/catalina.bat můžete nastavit cestu k JAVA_HOME nebo JRE_HOME a celou řadu dalších proměnných prostředí. Velice důležité je nastavit **maximální množství používané paměti**:
 - **Windows:**

```
set JAVA_OPTS=%JAVA_OPTS% -Xmx1g -XX:MaxPermSize=256m
```
 - **Linux:**

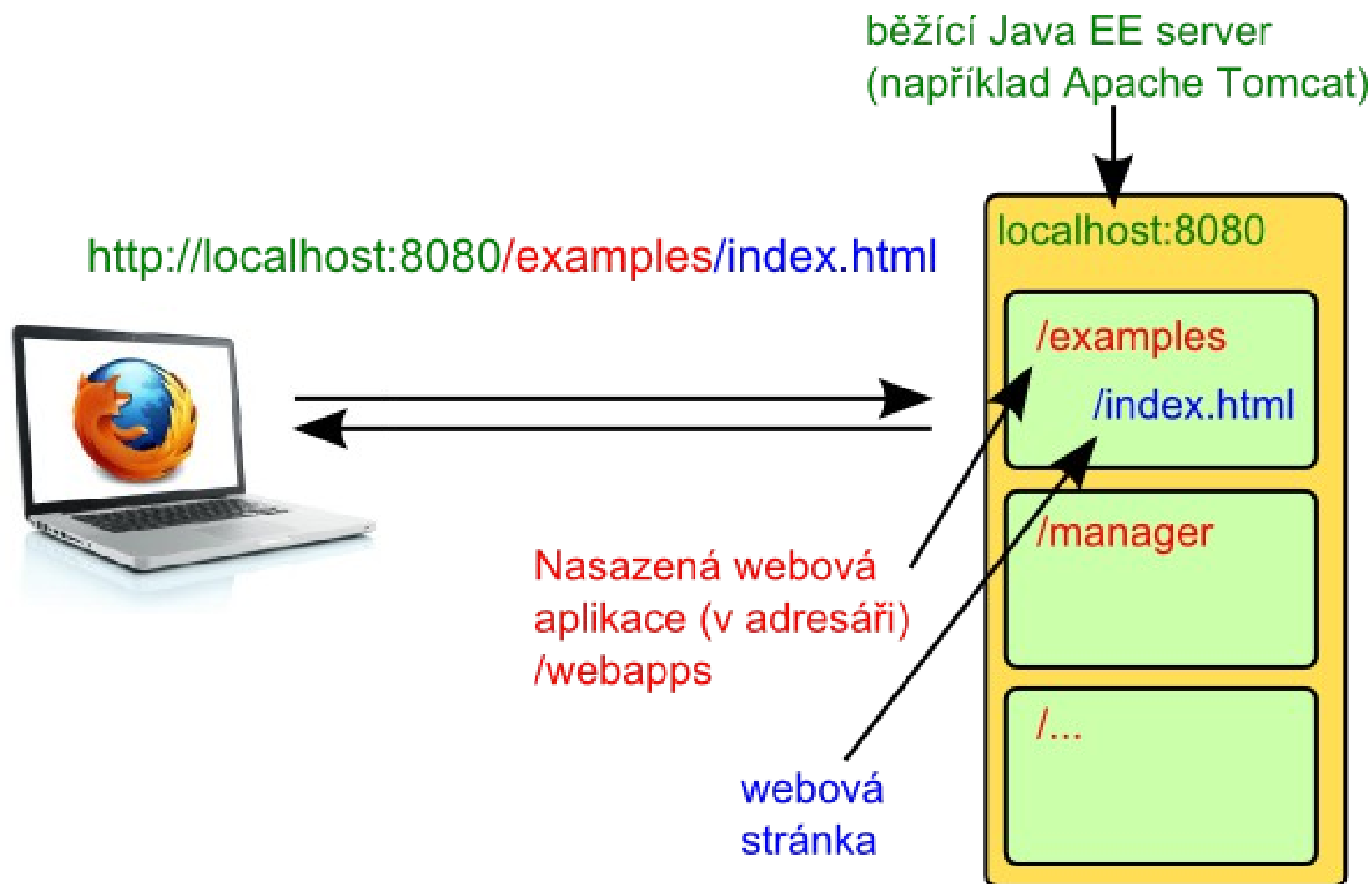
```
export JAVA_OPTS="$JAVA_OPTS -Xmx1g -XX:MaxPermSize=256m"
```



maximální
množství paměti

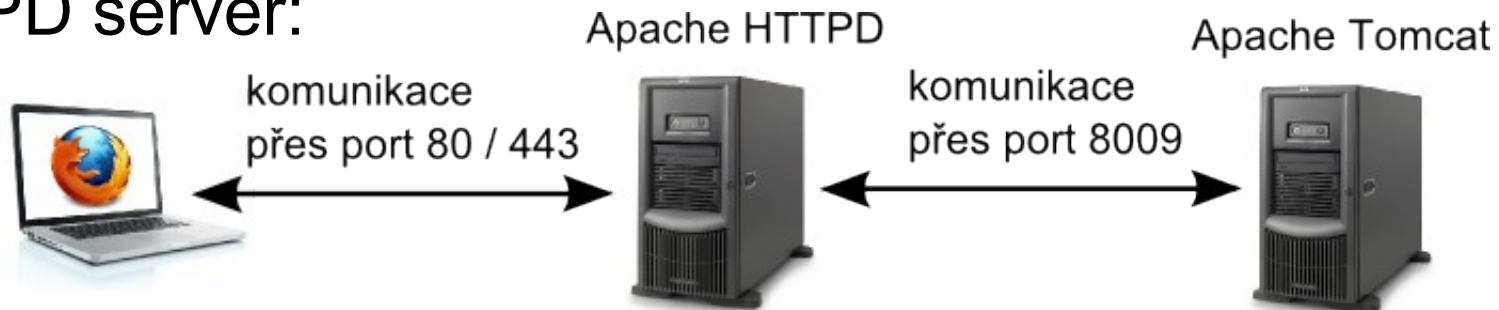
množství paměti
pro knihovny

Webová aplikace na serveru I.



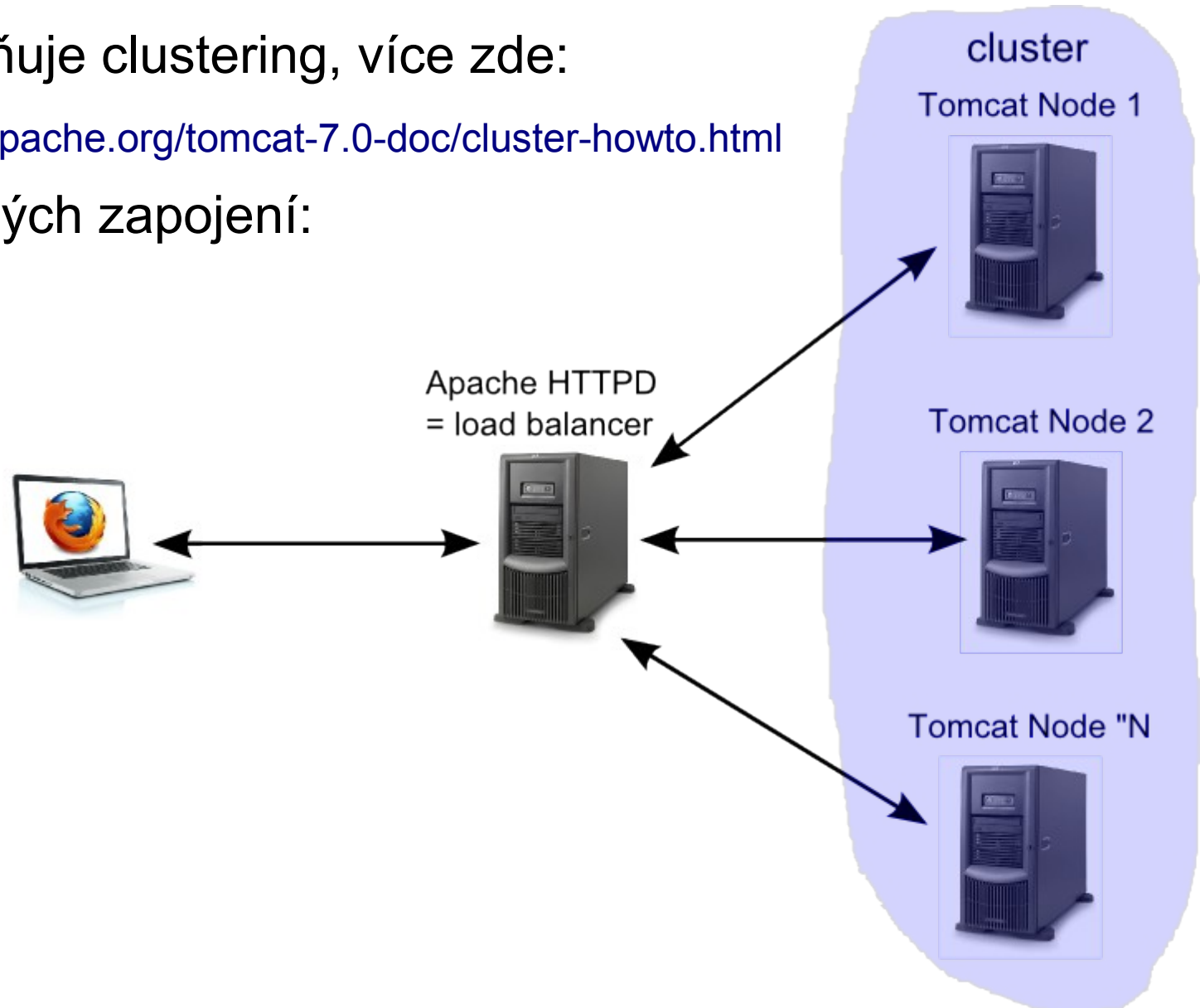
Webová aplikace na serveru II.

- Jedna z webových aplikací může mít název ROOT. Taková webová aplikace je pak přístupná přímo z URL adresy:
<http://localhost:8080/>
- Vše uvedené je možné změnit jinými způsoby nasazování aplikací na Apache Tomcat (viz. [Context](#))
- Tomcat nemusí běžet na portu 8080, ale na jakémkoli jiném portu (viz. [apache-tomcat]/conf/server.xml tag <Connector>)
- V produkčním prostředí se obvykle předřazuje Apache HTTPD server:



Tomcat Cluster

- Tomcat umožňuje clustering, více zde:
 - <http://tomcat.apache.org/tomcat-7.0-doc/cluster-howto.html>
- Jedno z možných zapojení:

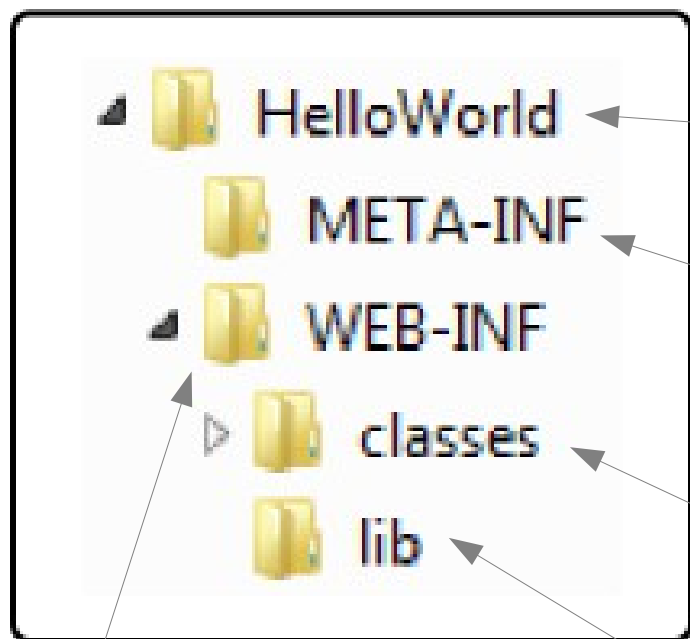


Tvorba webové aplikace bez Mavenu

- Webovou aplikaci můžete vytvořit bez Mavenu:
 - 1) Přidat do Eclipse Tomcat server runtime
 - 2) Vytvořit nový projekt typu Dynamic Web Project
 - 3) Spustit aplikaci pomocí Run As → Run on Server nebo je také možné provést deploy projektu pomocí drag and drop.
 - 4) Vytvoření WAR souboru: Export → WAR file

Struktura webové aplikace (WAR souboru)

HelloWorld.war:



V hlavním adresáři jsou veřejně přístupné soubory a adresáře

Tento adresář není zvenjšku přímo přístupný a standardně je v něm pouze MANIFEST.MF

Zde se nacházejí zkompilevané třídy, které aplikace používá

Zde se nacházejí JAR soubory, jejichž třídy aplikace používá

Tento adresář není zvenjšku přímo přístupný. Nachází se v něm deployment descriptor webové aplikace (od Java EE 6 nepovinný) a adresáře classes a lib

Deploy, undeploy, redeploy webové aplikace na Tomcat ručně

- Nasazování webových aplikací se provádí nejčastěji buď pomocí webového rozhraní, nebo ručním způsobem.
- Webová aplikace je obvykle ve formátu WAR souboru.
- **Deploy** = nasazení aplikace na server.
 - Zkopírování WAR souboru při běžícím Tomcatu do adresáře [apache-tomcat]/webapps/
- **Undeploy** = odebrání aplikace ze serveru.
 - Smáznutí WAR souboru při běžícím Tomcatu z adresáře [apache-tomcat]/webapps/
- **Redeploy** = kombinace undeploy a deploy.

Tvorba webové aplikace s Mavenem

- Webovou aplikaci můžete vytvořit s Mavenem:
 - Maven je nástroj pro správu, řízení a automatizaci buildů aplikací.
 - Pro práci s Mavenem v příkazové řádce je nutné mít v Path cestu do bin adresáře Mavenu a nastavenou proměnnou JAVA_HOME na adresář s JDK.
 - Pro práci s Mavenem v Eclipse je nutné:
 - Nainstalovat plugin m2e (Maven Integration for Eclipse).
 - Poznámka: STS (SpringSource Tool Suite) i Eclipse Juno již v základu obsahují tento plugin.
 - Provést integraci JDK s Eclipse. Maven je závislý na JDK, nikoli pouze na JRE! (Window → Preferences ... → Java → Installed JREs → Add ...)

Maven – Web app Hello world

- Tvorba webové aplikace postavené na Servletech a JSP, která bude běžet na Tomcatu pomocí Mavenu:
 - V konzoli napište:

```
mvn archetype:generate
```
 - Vyberte artefact s názvem (artefacty se dají filtrovat, zadejte webapp-javaee6 a poté zadejte číslo artefactu s požadovaným názvem):

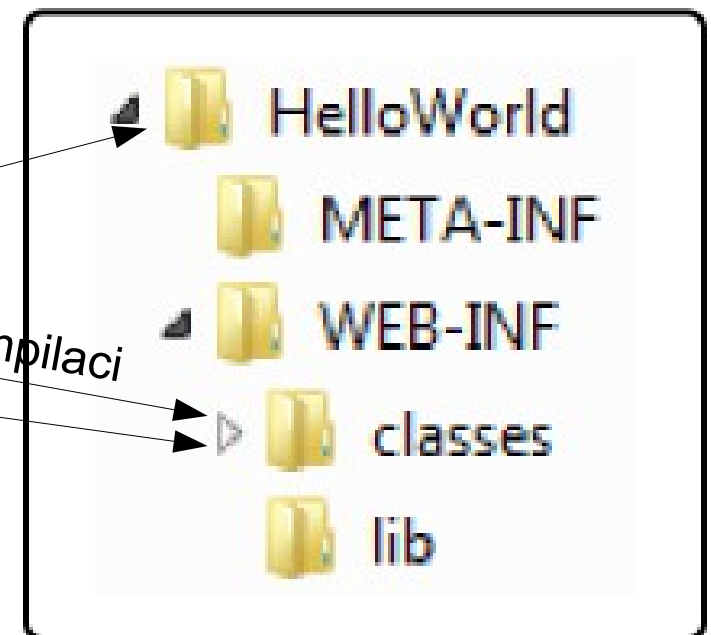
```
org.codehaus.mojo.archetypes:webapp-javaee6
```
 - Zadejte požadované informace, vytvoří se webová aplikace.
 - Naimportujte nově vytvořenou aplikaci do Eclipse pomocí File → Import ... → Existing Maven Projects.

Maven I.

- Maven usnadňuje:
 - Správu závislostí jednotlivých Java knihoven (**dependencies**)
 - Buildování aplikací (**plugins**)
- Plugins i dependencies jsou definovány v souboru `pom.xml`.
- Maven web projekt má následující adresářovou strukturu:

adresář	obsah adresáře
kořenový adresář aplikace	soubor <code>pom.xml</code> a ostatní adresáře
<code>src/main/webapp</code>	veřejné webové stránky a adresář <code>WEB-INF</code>
<code>src/main/java</code>	.java soubory
<code>src/main/resources</code>	konfigurační soubory
<code>src/test/java</code>	třídy testů
<code>src/test/resources</code>	konfigurační soubory testů

HelloWorld.war:



Maven II.

- Při použití Mavenu má každý projekt tyto základní fáze buildování:

1) process-resources

2) compile

3) process-test-resources

4) test-compile

5) test

6) package

7) install

8) deploy

Bez úspěšného spuštění
testů se nevytvoří WARko!

do adresáře „target“
vytvoří WAR soubor

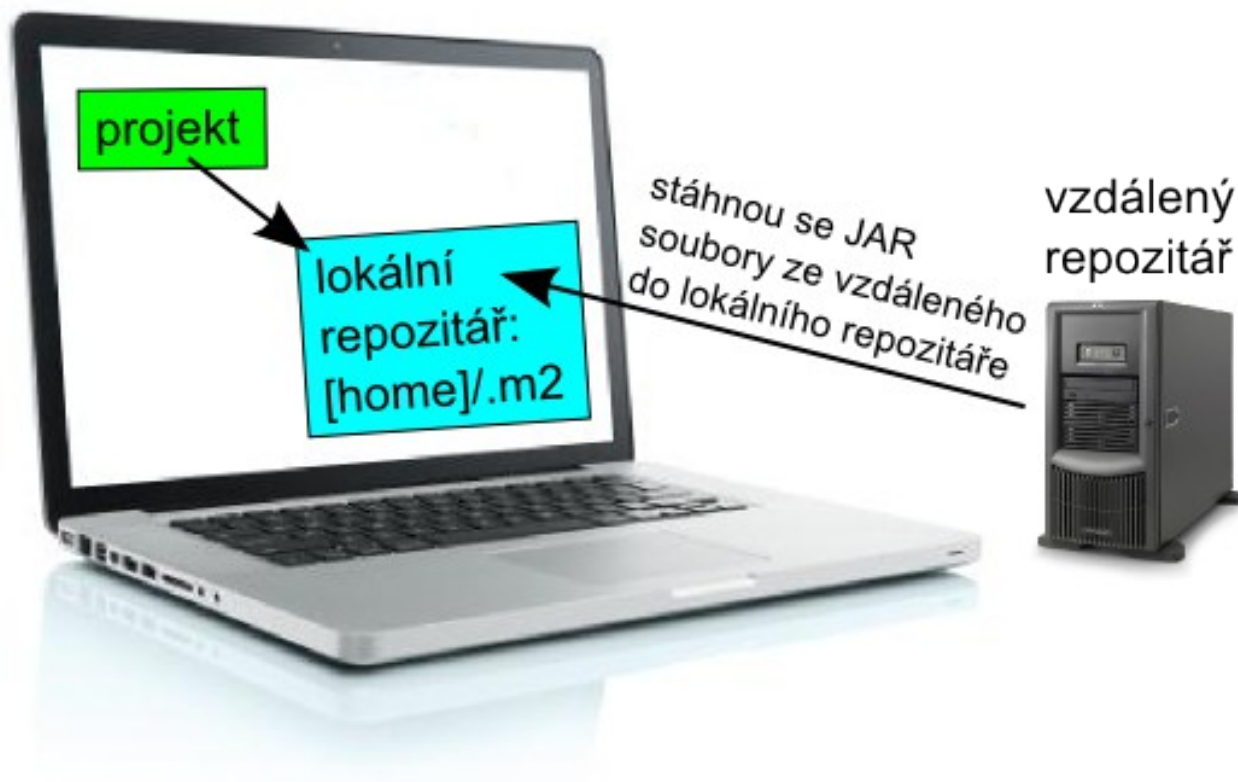
Instalace WAR souboru
do lokálního repozitáře

Instalace WAR souboru
do vzdáleného repozitáře

směr provádění jednotlivých fází

Maven III.

- Maven usnadňuje správu závislostí Java knihoven (JAR) souborů. K tomu musí JAR knihovny stáhnout z nějakého vzdáleného repozitáře do lokálního repozitáře a poté je může propojit s Maven projektem.
- Lokální repozitář je standardně v `[home]/.m2`
- V některých společnostech se také můžete setkat s vnitrofiremními repozitáři.



Maven IV.

při spouštění
Mavenu z konzole

- Build fáze je možné volat následujícím způsobem:
 - `mvn [název build fáze]`
- Build fáze, která smaže obsah adresáře target:
 - `mvn clean`
- Jednotlivé build příkazy je možné řetězit:
 - `mvn clean package`
- Maven je rozšiřitelný pomocí pluginů, požadované pluginy se definují v `pom.xml` souboru.
- Plugin se spustí zavoláním:
 - `mvn [název pluginu]:[goal]`

co bude
plugin dělat

Embedded Jetty server I.

- Na školení budeme pro vývoj používat primárně embedded Jetty server.
- Embedded servery jsou velice vhodné pro rychlý vývoj webových aplikací. U každého projektu je README, ve kterém je postup rozchození příslušných projektů.
- Pokud používáte Java 7, pak přidejte do pom.xml tento plugin:

```
<plugin>  
  <groupId>org.eclipse.jetty</groupId>  
  <artifactId>jetty-maven-plugin</artifactId>  
  <version>9.1.0.v20131115</version>  
  <configuration>  
    <scanIntervalSeconds>1</scanIntervalSeconds>  
  </configuration>  
</plugin>
```



Nastavení automatického redeploy

Embedded Jetty server II.

- Pokud používáte Java 6 nebo nižší, pak použijte tento plugin:

```
<plugin>
  <groupId>org.mortbay.jetty</groupId>
  <artifactId>jetty-maven-plugin</artifactId>
  <version>8.1.13.v20130916</version>
  <configuration>
    <scanIntervalSeconds>1</scanIntervalSeconds>
  </configuration>
</plugin>
```

Nastavení automatického redeploy



- Spuštění Jetty serveru:

```
mvn jetty:run
```

Embedded Tomcat

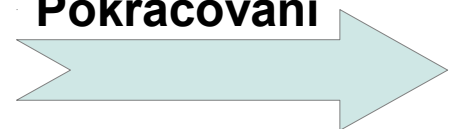
- Také můžete používat embedded Apache Tomcat.
- Do pom.xml přidejte dovnitř tagu <build> následující kus XML kódu:

```
<plugin>
  <groupId>org.apache.tomcat.maven</groupId>
  <artifactId>tomcat7-maven-plugin</artifactId>
  <version>2.2</version>
  <configuration>
    <contextFile>src/main/webapp/WEB-INF/context.xml</contextFile>
  </configuration>
</plugin>
```

- Spuštění Tomcat serveru:

```
mvn tomcat7:run
```

Pokračování



Tip: Automatický redeploy

- V současnosti když se změní JSP soubor, tak se změna ihned promítne v nasazené webové aplikaci, není k tomu třeba redeploy web. aplikace.
- Při změně JAVA souboru se ale taková změna ve webové aplikaci neprojeví, v současnosti je nutné restartovat celý server. To ale není nutné když se provede redeploy, který se navíc může provést automaticky.
- Vytvořte soubor `src/main/webapp/WEB-INF/context.xml` s následujícím obsahem:

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

```
<Context reloadable="true" backgroundProcessorDelay="1">
```

```
</Context>
```



Nastavení automatického redeploy

Další způsoby deploy web. aplikací

- Další aplikační servery mají své Maven pluginy.
- Můžete také pomocí Mavenu provádět deploy na externí Tomcat.
- Můžete také v Eclipse integrovat libovolný aplikační server (ve View Servers) a spustit na něm aplikaci (na aplikaci Run As → Run on Server ...).

EAR soubor I.

- Pokud používáte aplikační server (JBoss, GlassFish, WebLogic, WebSphere, ...), pak se můžete také setkat s EAR souborem.
- Prakticky se jedná o ZIP soubor, který sdružuje WAR, JAR a EJB soubory:
 - WAR soubor = webová aplikace
 - JAR soubor = Java knihovna
 - EJB soubor = soubor obsahující EJB třídy
- Uvnitř EAR souboru jsou výše uvedené typy souborů a soubor `META-INF/application.xml`, ve kterém jsou uvedeny jejich názvy.
- Poznámka: Pokud chcete používat EJB, pak není nutné je mít v EJB souboru (a ten pak zabalit do EAR souboru). Můžete je také mít ve WAR nebo JAR souboru.

EAR soubor II.

- EAR soubor má oproti WAR souboru tu výhodu, že může obsahovat více webových aplikací (více WAR souborů).
- Nevýhodou je, že je možné ho nasadit pouze na aplikační server, nikoli webový kontejner (čili ho nenasadíte na Tomcat / Jetty).
- EAR vytvoříte jednoduše pomocí tohoto Maven archetype:
 - `jboss-javaee6-webapp-ear-archetype-blank`