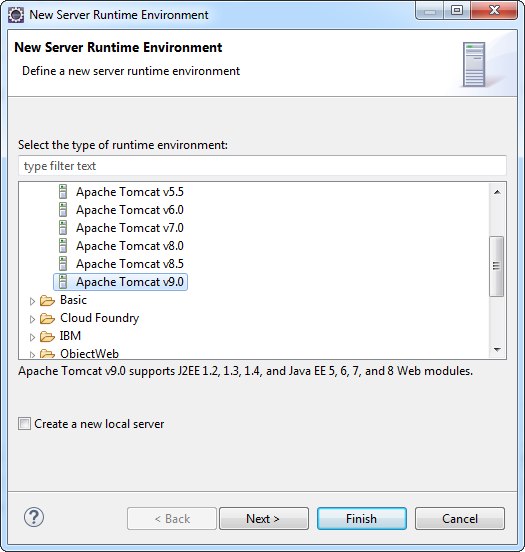
**Razvoj Web aplikacija(prejekata) u Eclipse okruženju**

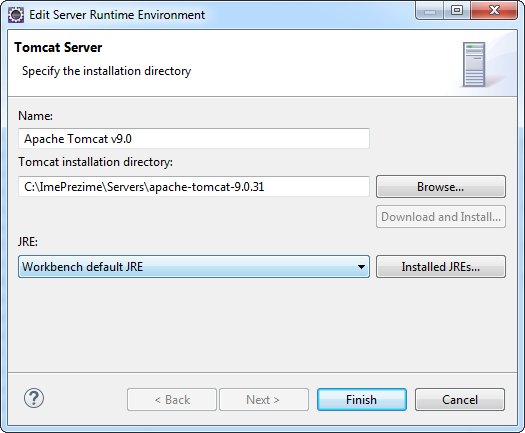
Pre nego što napravite bilo koji Dinamički web projekat uradite sledeće u Eclipsi:

Odaberite Window->Preferences->Server->RuntimeEnvironments, a zatim kliknite na dugme Add. Iz liste izaberite Apache Tomcat v9.0.



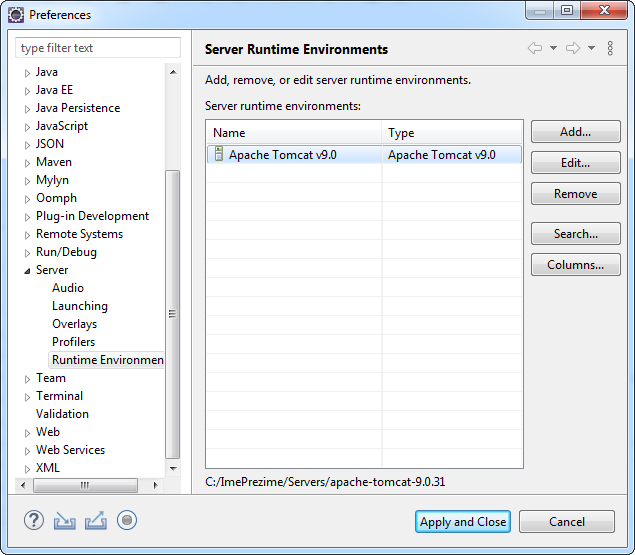
Slika 1 - Runtime Environment

Na sledećem dijalogu preko dugmeta Browse podesite putanju na vašem računaru do raspakovanog Tomcat web servera (Na slici 2 je primer putanje na mom hard disku).



Slika 2 - Dijalog za postavljanje putanje Servera

Kada odaberete dugme Finish imaćete sledeću situaciju:



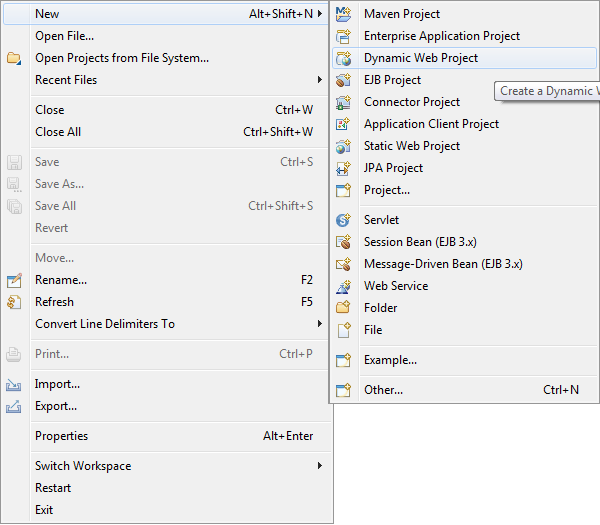
Slika 4 - Dodata veza između Tomcat i Eclipse

**Ovime smo obezbedili da Eclipse projekat bude svestan servelt-api.jar biblioteke iz Tomcat. Inače bi smo morali da stavimo ovu biblioteku u WEB-INF/lib folder u svaki novi dinamički web projekat. Ovo je mnogo bolja varijanta nego da kopiramo servelt-api.jar biblioteku.**

Postoje dva tipa web projekata: dinamički i statički. Dinamički web projekti mogu da sadrže servlete, jsp stranice i slične "dinamičke" sadržaje koje pozivaju korisnici slanjem odgovarajućih zahteva. Dok statički projekti sadrže samo HTML stranice, slike i druge "statičke" resurse koji se isporučuju klijentu u istom (nepromenjenom) obliku u kakvom se nalaze i na serveru.

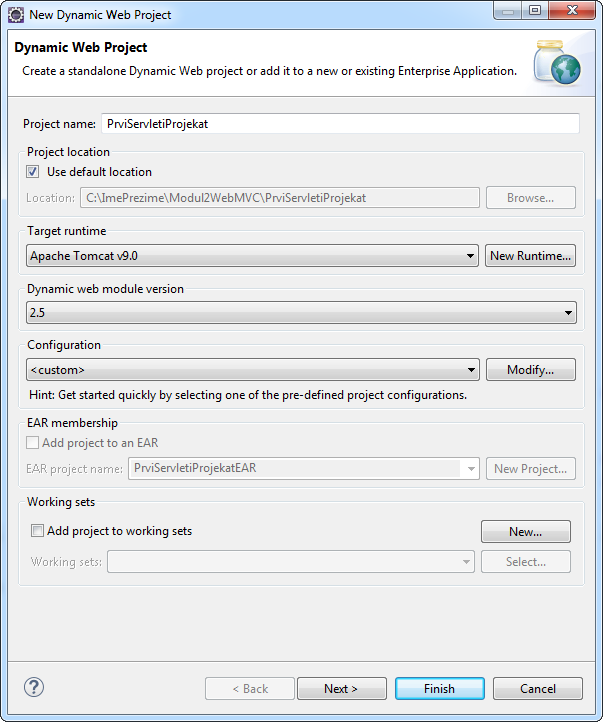
**Kreiranje dinamičkog web projekta**

Novi dinamički projekat se kreira u Eclipsu kao na slici 5. Eclipse *New-*>*Dynamic Web Project*

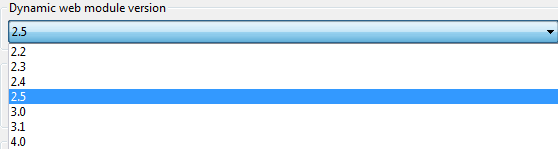


Slika 5 - Dinamički Web projekat

Na sledećem dijalogu odabiremo ime projekta i Runtime Environment Apache Tomcat v9.0 (to je ono što smo prethodno podesili) (slika 6) i verzija Dynamic Web Module (slika 7). Nama su interesantne dve verzije. Starija verzija *Dynamic web module version 2.5* i najnovija *Dynamic web module version 4.0.* Kod starije verzije 2.5 sva neophodna mapiraja se navode u web.xml kod novije verije 4.0 se mapiranja mogu navesti u web.xml ali se mogu navesti i anotacijama u samom kodu java klase. Radi detaljnijeg savladavanja tehnologije odabraće se verzija 2.5.

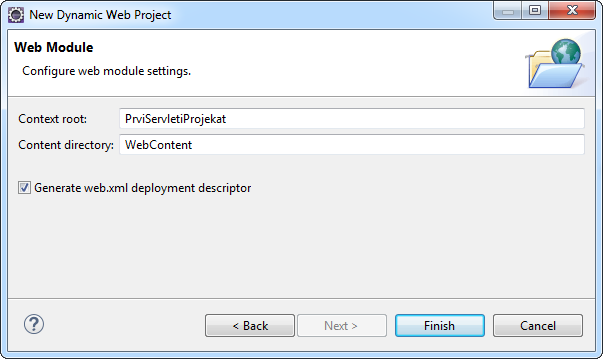


Slika 6 - Podešavanje Dynamic Web project



Slika 7 - Dynamic Web Module

Kliknemo na Next dugme dok ne dođemo do sledećeg dijaloga:



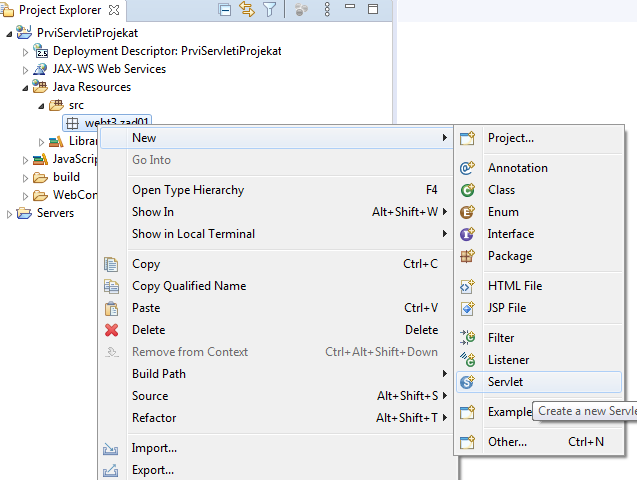
Slika 8 - Automatsko generisanje Deployment descriptora

Na njemu obavezo naznačiti da se prilikom formiranja projekta automatski generiše i web.xml (fajl u kome mapiramo Servlete).

Kreirani Dinamički Web projekat treba da ima sledeću strukturu foldera:

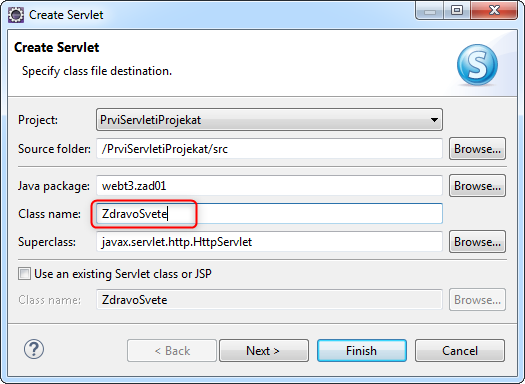
* Java ReSources - U njega ćemo smeštati java izvorne datoteke (.java), binove i servlete. Oni se u Eclipsi automatski kompajliraju i njihovi class fajlovi su smešteni u WEB-INF/classes direktorijumu ili u build/classes.
* WebContent - Folder u kome lokaciju nalaze Web resursi. Tako se u njemu mogu naći dinamički resusrsi (JSP stranice) ili statički (slike, fajlovi, HTML stranice, CSS stilovi ...). Kada se aplikacija prebaci na server, on neće moći da pristupi pomenutim web resursima ako se oni ne nalaze u ovom folderu ili nekom od njegovih podfoldera.
  + WEB-INF - Podfolder u WebContent. sadrži relevantne web resurse za funkcionisanje web aplikacije. Sadrži i fajl web.xml o kome će biti reči kasnije.
    - classes - Podfolder WEB-INF je output direktorijum prekomajliranih Java klasa (one iz JavaResources). Podfolderi u ovom folderu čuvaju strukuru paketa iz Java ReSources
    - lib - Podfolder u WebContent. Nalaze se sve eksterne biblioteke (jar fajlovi) koji su neophodni za funkcionisanje Web aplikacije.

Novi Servlet kreiramo tako što u okviru Java Resources folder src napravimo paket *webt3.zad01* i u njemu odaberemo kreiranje novog servleta kao na slici 9.



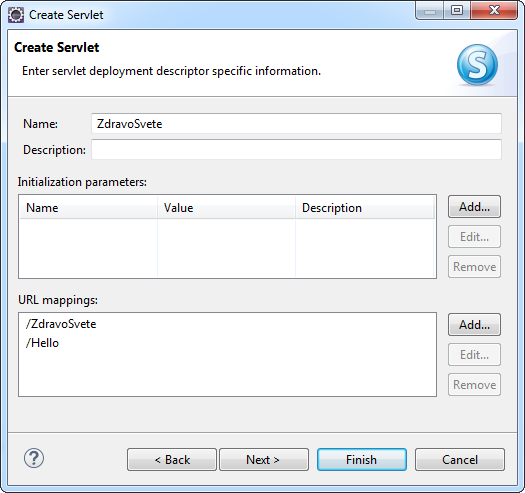
Slika 9 - Kreiranje novog Servleta

Na sledećem dijalogu odabiremo ime klase



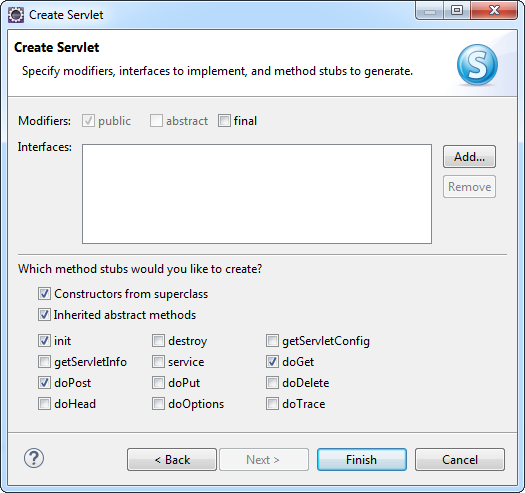
Slika 10 - Definisanje imena klase Servlet

U sledećem dijalogu moguće je definisati URL mapiranje za kreirani Servlet tj. pod kojim URL će se pristupiti Sevletu i pozvati njegove metode. Moguće je višestruko URL mapiranje za isti Servlet.



Slika 11 - Definisanje imena klase Servlet

Kada se stigne do poslednjeg dijaloga neophodno je naznačiti koje metode će Servlet incijlano generisati. Nama su potrebne doGet,doPost i init metoda,



Slika 12 - Metode Servleta

Init metoda servleta služi za inicijalizaciju Servleta od strane Servlet kontejnera i ona se radi samo jednom po kreiranju objekta koji predstavlja posmatrani Servlet.

**public** **void** init(ServletConfig config) **throws** ServletException {

**super**.init(config);

}

Da bi servlet vraćao neki dinamički kreiran resurs neophodno je u njegovor doGet i doPost metodi napisati neki kod koji bi vratio resurs.

**protected** **void** doGet(HttpServletRequest request, HttpServletResponse response) **throws** ServletException, IOException {

PrintWriter pw = response.getWriter();

response.setContentType("text/html");

pw.println("<html>");

pw.println("<body>");

pw.println("<h1>Zdravo svete</h1>");

pw.println("<p>"+Math.*random*()+"</p>");

pw.println("</body>");

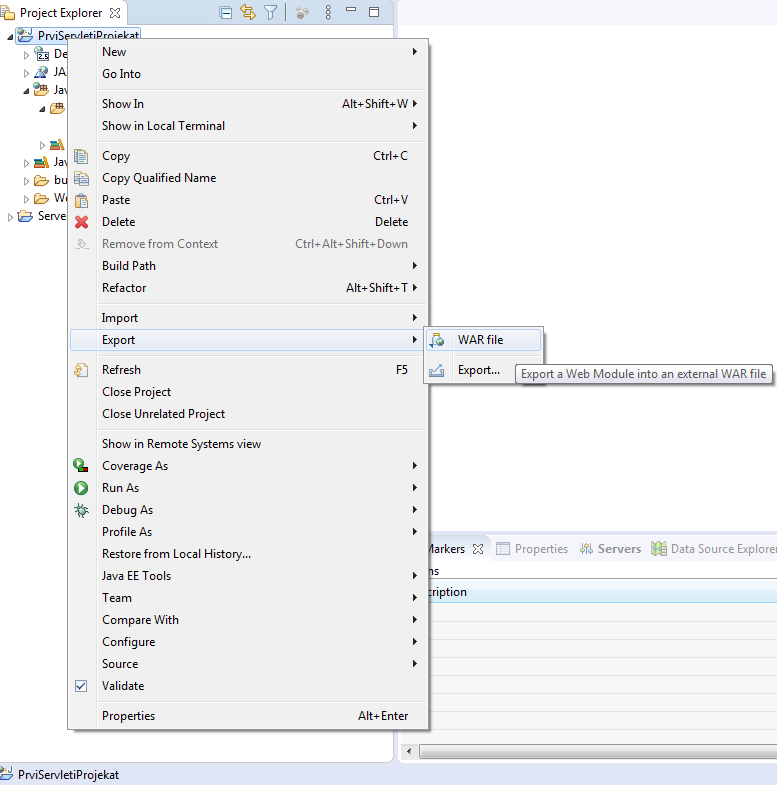
pw.println("</html>");

pw.flush();

pw.close();

}

Da bi se dinamička aplikacija pokrenula na Web serveru neophodno je da se aplikacija prebaci (eksportuje) u oblik čitljiv za Web server. Kako ćemo mi našu dinamičku web aplikaciju pokretati pod Apache Tomcat web serverom neophodno je da od aplikacije napravimo war arhivu (slika 13). War arhiva je packed arhiva slična zip ili jar arhivi.



Slika 13 - Eksport u war

U pomenutu war arhivu ide sadržaj WebContent foldera i podfoldera, iliti sve prekompajlirane java klase i web resursi. War arhiva se mora smestiti u webapps folder Tomcata.

**Web Deployment Descriptor osnove - web.xml**

Fajl u kome se opisuje naša web aplikacija za potrebe servera. U web terminologiji naziva se “web application's deployment descriptor”. U ovom fajlu se opisuju klase, resursi i konfiguracije, kao i način kako će ih web server koristiti kada odgovara na zahteve klijenata.

Ovaj fajl se mora nalaziti u WEB-INF folderu. Prilikom formiranja war fajla i ovaj fajl će se naći u arhivi u prethodno navedenom folderu. U nastavku će biti dat primer web.xml fajla i biće objašnjeni njegovi najznačajniji (najkorišćeniji) elementi.

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

<web-app xmlns:xsi=*"http://www.w3.org/2001/XMLSchema-instance"* xmlns=*"http://java.sun.com/xml/ns/javaee"* xsi:schemaLocation=*"http://java.sun.com/xml/ns/javaee http://java.sun.com/xml/ns/javaee/web-app\_2\_5.xsd"* id=*"WebApp\_ID"* version=*"2.5"*>

<display-name>PrviServletiProjekat</display-name>

<welcome-file-list>

<welcome-file>index.html</welcome-file>

<welcome-file>index.htm</welcome-file>

<welcome-file>index.jsp</welcome-file>

<welcome-file>default.html</welcome-file>

<welcome-file>default.htm</welcome-file>

<welcome-file>default.jsp</welcome-file>

</welcome-file-list>

<servlet>

<description></description>

<display-name>ZdravoSvete</display-name>

<servlet-name>ZdravoSvete</servlet-name>

<servlet-class>webt3.zad01.ZdravoSvete</servlet-class>

</servlet>

<servlet-mapping>

<servlet-name>ZdravoSvete</servlet-name>

<url-pattern>/ZdravoSvete</url-pattern>

<url-pattern>/Hello</url-pattern>

</servlet-mapping>

</web-app>

Element <servlet> definiše java web servlet klasu, podelement <servlet-name> predstavlja ime servleta, <servlet-class> predstavlja putanju (hijerarhiju paketa-direktorijuma) do konkretne klase u našem projeku. <servlet-mapping> ide u paru sa elementom <servlet>. Za svaki servlet moramo imati mapiranje koje koristi sam web server prilikom odgovra na zahteve korisnika. Pod element <servlet-name> mora da se poklapa sa istim podlementom iz <servlet> elementa. <url-pattern> je naziv koji će vam omogućiti da pristupite konkretnom resursu servletu. Na primer, ako se projekat zove **Examples** servlet HelloWorld a elementu <url-pattern> piše **/Hello**, mi ćemo pristupiti tom servletu kroz web čitač ako unesemo:

* *http://localhost:8080/PrviServletiProjekat/ZdravoSvete*
* *http://localhost:8080/PrviServletiProjekat/Hello*

U slučaju da unesemo samo *http://localhost:8080/PrviServletiProjekat*/

web server će pogledati da li postoji element <welcome-file> i odgovoriće sa prvim resusrom navedenim u ovoj listi (index.html).

**Uvlačenje postojećeg dinamičkog web projekta**

Za uvlačenje postojećeg projekta koristiće se projekat koji je prethodno bio kreiran sa *Dynamic web module version 4.0*.

*Dynamic web module version 4.0*  predstavlja ternutno aktuelnu verziju modula

* Za razliku od verzije 2.5 u verziji 3.0 ka na dalje nije potrebno ekspicitno definisati mapiranja u *web.xml* fajlu već se mapiranja mogu odraditi direktno iz java koda anotacijama u okviru Servlet klase. U *web.xml* se razlikuje *namespace*.
* Kreiranje novog projekta je isto kao što je ranije već bilo pokazano, samo se sada postavi *Dynamic web module version 4.0*

Opcija uvlačenja projekta je isto kao i običnog java projekta.

* U okviru workspace ImePrezime/Predmet2Web raspakovati zip arhivu DrugiServletiProjekat.zip
* Izvršite uvlačenje projekta DrugiServletiProjekat u Eclipse
* Izvršite postavljanje aplikacije na Tomcat
* Pristupiti aplikaciji preko brauzera. Kucate adresu
  + http://localhost:8080/DrugiServletiProjekat/HelloHello

U klasi ZdravoSvete anotacijama je definisano mapiranje za posmatrani Servlet. Pri čemu je izbegnuta potreba da se dato mapiranje eksplicitno definiše u web.xml.

@WebServlet(name = "Hello", value = "/HelloHello")

**public** **class** ZdravoSvete **extends** HttpServlet {

**Uvođenje anotacija u source kodu preko kojih bi se konfigurisao rad web aplikacije nezaobilazni je deo svakog novog radnog okvira (framework), bez obzira na programski jezik na koji se taj radni okvir oslanja.**

**Web Deployment Descriptor detaljno - web.xml**

Objašnjenja tagova data su u kodu

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

<web-app xmlns:xsi=*"http://www.w3.org/2001/XMLSchema-instance"*

xmlns=*"http://xmlns.jcp.org/xml/ns/javaee"*

xsi:schemaLocation=*"http://xmlns.jcp.org/xml/ns/javaee http://xmlns.jcp.org/xml/ns/javaee/web-app\_4\_0.xsd"*

id=*"WebApp\_ID"* version=*"4.0"*>

<display-name>DrugiServletiProjekat</display-name>

<welcome-file-list>

<welcome-file>pocetna.html</welcome-file>

<welcome-file>index.html</welcome-file>

<welcome-file>index.htm</welcome-file>

<welcome-file>index.jsp</welcome-file>

<welcome-file>default.html</welcome-file>

<welcome-file>default.htm</welcome-file>

<welcome-file>default.jsp</welcome-file>

</welcome-file-list>

<!-- sets Servlet and its' mapping -->

<!-- sets init parameters for Servlet -->

<!-- sets initialisation of Servlet during startup of application -->

<servlet>

<servlet-name>nameOfServlet</servlet-name>

<servlet-class>packet.ServletClass</servlet-class>

<init-param>

<param-name>nameOfInitParam1</param-name>

<param-value>valueOfInitParam1</param-value>

</init-param>

<init-param>

<param-name>nameOfInitParam1</param-name>

<param-value>valueOfInitParam1</param-value>

</init-param>

<load-on-startup>1</load-on-startup>

</servlet>

<servlet-mapping>

<servlet-name>nameOfServlet</servlet-name>

<url-pattern>/URLforMapping</url-pattern>

</servlet-mapping>

<!-- sets init parameter for appliaction -->

<!-- parameter is available from ServletContext -->

<context-param>

<param-name>nameOfParamforServletContext</param-name>

<param-value>10</param-value>

</context-param>

<!-- sets timeout for inactive session -->

<session-config>

<session-timeout>5</session-timeout>

</session-config>

<!-- sets listener for aplication -->

<listener>

<listener-class>util.InitSessionListener</listener-class>

</listener>

</web-app>

**Apache Tomcat**

Apache Tomcat je web server otvorenog koda razvijen od strane Apache Software Fondation. Tomcat predstavlja web server koji u potpunosti podržava java web bazirane aplikacije. Kako bi mogli da upravljamo Tomcat serverom neophodno je da poznajemo i strukturu foldera istog. Tomcat sadrži sledeće foldere:

* bin - sadrži fajlove koji omogućavaju pokretanje i zaustavljanje samog servera.
* conf - čuva konfiguracione fajlove kojima se može menjati ponašanje servera.
* lib - ugrađene biblioteke koje koristi prilikom rada sa svojim aplikacijama. bibloteka koja se nalazi u ovom folderu dostupna je za sve aplikacije koje su podignute pod Tomacat serverom.
* webapps - najznačajniji folder. U njemu kopiramo war arhivu koja se raspakuje prilikom pokretanja tomcat servera. Nakon raspakovanja dobijamo folder sa istim imenom kao i war arhiva u kome se nalaze svi web resursi (sve iz WebContent iz Eclipse) koje je programer definisao prilikom konstruisanja web aplikacije.
* temp, work - tomcat koristi ove foldere za smeštanje privremenih resursa(fajlova) tokom svog rada.

**Pokretanje Apache Tomcat**

**Iz komandnog prozora (konzole)**

Tomcat se startuje pokretanjem određenog fajla (skripte) iz bin foldera:

Linux: sh catalina.sh run

Windows: startup.bat

Da bi ste proverili da li radi Tomcat u web browseru ćete ukucati localhost:8080

Pokretanje konkretne aplikacije koja radi pod Tomcat web serverom obavljamo tako što u browseru kucate sledeću adresu:

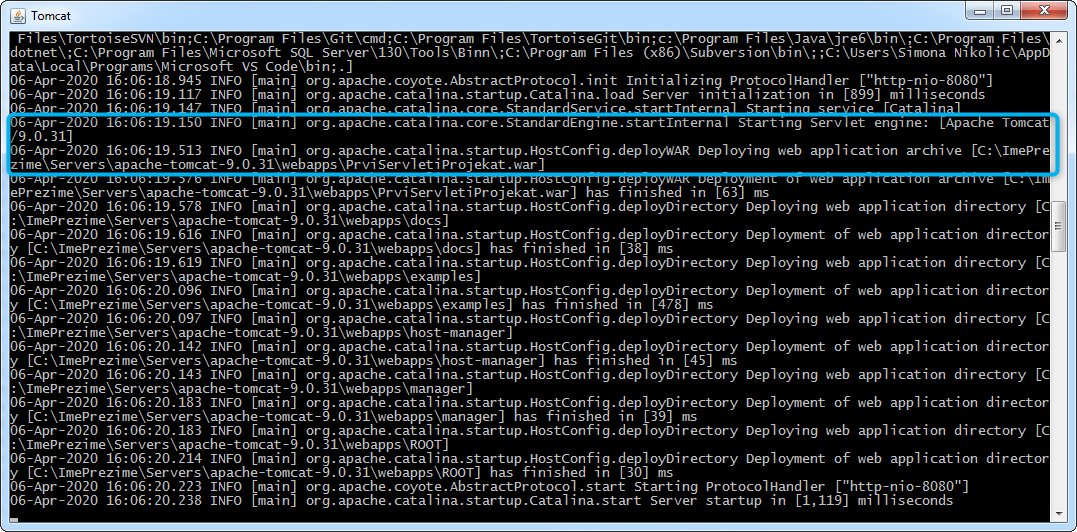
localhost:8080/Ime\_Aplikacije

localhost – je simboličko ime vaše lokalne mašine. Umesto ovoga može da stoji i IP adresa.

8080 – predefinisani port na kome je pokrenut Tomcat web server

Ime\_Aplikacije – Najčešće se poklapa sa imenom projekta(aplikacije) koju ste razvijali.

Indikator da je vaša aplikacija (war) smeštana i adekvatno pokrenuta pod Tomcat je natpis na slici:

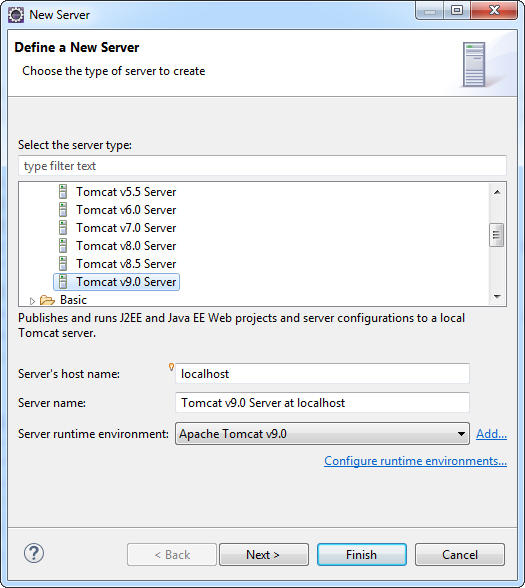


Slika 14 - Izgled konzole kada startovan Tomcat

Da bi ste zaustavili (ugasili) Tomcat morate da zatvorite konzolu u kojoj je on startovan ili istovremeno pritisnete tastere Ctr i c dok se nalazite u konzoli.

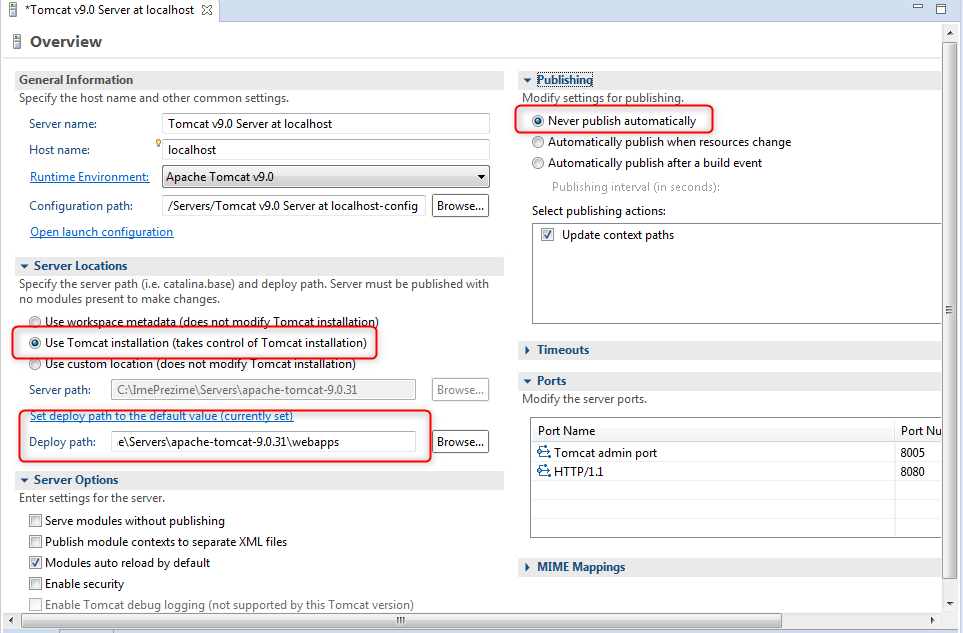
**Pokretanje Apache Tomcat iz Eclipse okruženja**

Ukoliko je nekome zgodnije može pozivati Tomcat direktno iz Eclipse. Za to je neophodno da povežemo na specifičan način raspakovani Tomacat server sa našom Eclipsom. U Eclipsi treba da se odabere Window->ShowView->Server. U Donjem delu Eclipse (tamo gde je i konzola) otvoriće se pogled za upravljanje serverima. Nakon toga treba kliknuti na link sa slike 15 i dijaloga odabrati **Tomcat v9.0**.



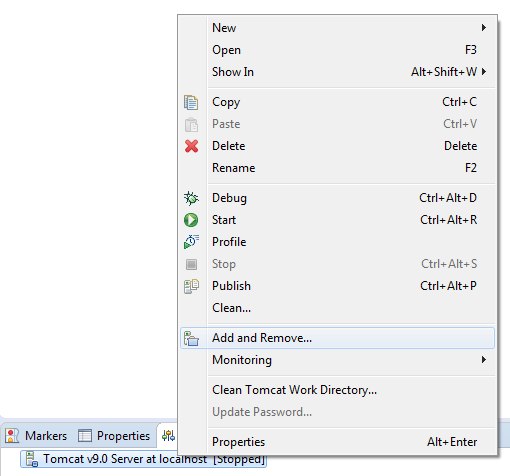
Slika 15 - Kreiranje Tomcat servera u Eclipsi

Nakon toga kliknuti na dugme Finish. Posle uradite dvoklik na novonastali server pa će se pojaviti dijalog kao na slici 16



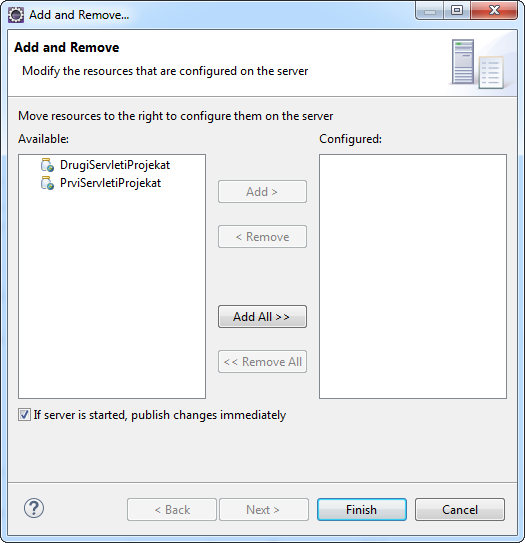
Slika 16 - Podešavanje Tomcat servera u Eclipsi

U Server Location sekciji postavite drugu opciju i kao deploy path upišite putanju do webapps foldera u Tomcat web serveru, dok u sekciji Publising odabarite prvu opciju kao na slici 16. Na kraju sa File->Save ili Ctrl+s snimite ove izmene. Kada završite sa podešavanjima kliknite desnim tasterom miša na server i dobićete dijalog kao na slici 17.



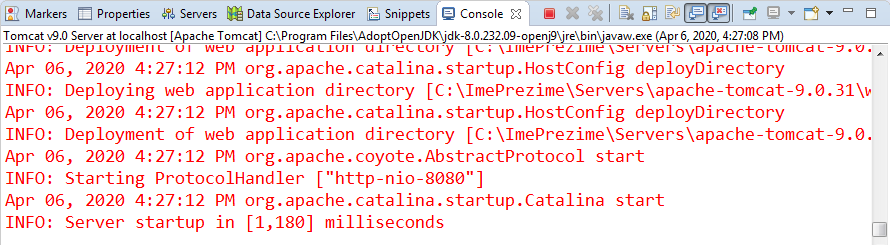
Slika 17 - Upravljanje Tomcat serverom u Eclipsi

Neophodne opcije su vam Start i Stop. Međutim pre pokretanja servera neophodno je da **dodate aplikaciju na Server** klikom na Add and Remove. Nakon toga kao na slici 18 odabir željenu aplikaciju i dugmetom Add je dodajem na Server.



Slika 18 - Dodavanje aplikacije na Tomcat server u Eclipsi

Kada završim sa dodavanjem neophodno je samo da pokrenemo server. Informacije o radu Servera možemo pratiti u konzoli (slika 19)



Slika 19 - Izgled Tomcat konzole u Eclipse

Redosled akcija koje trebate da radite kada koristite ugrađeni Tomcat su sledeće

1. Programirajte u Eclipsi dok je server zaustavljen (stop)
2. Samo prvi put je neophodno da dodate aplikaciju, nakon toga će Tomcat biti svestan promena u vašoj aplikaciji samo njegovim ponovnim startovanjem
3. Bilo bi dobro da pre svakog pokretanja (start) pozovete Clean opciju

**Dodatno**

**Napomena:** Da bi ste pokrenuli Tomcat neophodno je da imate podešenu sistemsku promenljivu

JAVA\_HOME !!! Ona se podešava na nivou operativnog sistema tako je za:

Windows

Desno dugme na Computer->Properties->AdvanceSystemSetings->EnviromentVariable

U dijalogu treba da se napravi nova promenljiva pod imenom JAVA\_HOME čija vrednost mora biti putanja do instalacije Java (SDK)

**Opšta preporuka prilikom razvoja web aplikacije**

1. Dok razvijate aplikaciju bilo bi dobro da Tomcat bude ugašen.
2. Kada završite sa određenim delom i želite da to i proverite kako izgleda pod Tomcat uradite *Deploy To War* pa onda startujte Tomcat.
3. Kada završite sa proverom onoga što ste uradili ugasite Tomcat i vratite se u Eclipse kako bi ste nastavili razvoj svoje aplikacije .