

---

# Bab 1

---

## Pengenalan JSP

---

### POKOK BAHASAN:

- Dasar JSP
- Daur hidup JSP]
- Web Container
- Jakarta Tomcat
- Contex
- Java Virtual Machine
- Yang diperlukan untuk menjalankan JSP
- Membuat dan mendeploy halaman JSP

### TUJUAN BELAJAR:

Setelah mempelajari bab ini, mahasiswa diharapkan mampu:

1. Mengetahui JSP
2. Mengetahui web server
3. Mengetahui daur hidup JSP
4. Mengetahui yang diperlukan untuk menjalankan JSP
5. Membuat dan mendeploy halaman JSP

#### 1.1 Dasar JSP

JSP adalah suatu teknologi web berbasis bahasa pemrograman Java dan berjalan di Platform Java, serta merupakan bagian teknologi J2EE (Java 2 Enterprise Edition). JSP

sangat sesuai dan tangguh untuk menangani presentasi di web. Sedangkan J2EE merupakan platform Java untuk pengembangan sistem aplikasi enterprise dengan dukungan API (Application Programming Interface) yang lengkap dan portabilitas serta memberikan sarana untuk membuat suatu aplikasi yang memisahkan antara business logic (sistem), presentasi dan data.

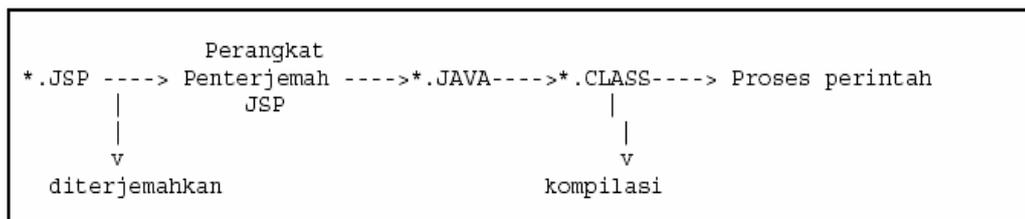
JSP merupakan bagian dari J2EE dan khususnya merupakan komponen web dari aplikasi J2EE secara keseluruhan. JSP juga memerlukan JVM (Java Virtual Machine) supaya dapat berjalan, yang berarti juga mengisyaratkan keharusan menginstal Java Virtual Machine di server, dimana JSP akan dijalankan. Selain JVM, JSP juga memerlukan server yang disebut dengan Web Container.

Teknologi JSP menyediakan cara yang lebih mudah dan cepat untuk membuat halaman-halaman web yang menampilkan isi secara dinamik. Teknologi JSP didesain untuk membuat lebih mudah dan cepat dalam membuat aplikasi berbasis web yang bekerja dengan berbagai macam web server, application server, browser dan development tool.

Java Server Pages (JSP) adalah bahasa scripting untuk web programming yang bersifat *server side* seperti halnya PHP dan ASP. JSP dapat berupa gabungan antara baris HTML dan fungsi-fungsi dari JSP itu sendiri. Berbeda dengan Servlet yang harus dikompilasi oleh USER menjadi class sebelum dijalankan, JSP tidak perlu dikompilasi oleh USER tapi SERVER yang akan melakukan tugas tersebut. Makanya pada saat user membuat pertama kali atau melakukan modifikasi halaman dan mengeksekusinya pada web browser akan memakan sedikit waktu sebelum ditampilkan.

## 1.2 Daur Hidup JSP

Sebagai gambaran bagaimana JSP melalui masa hidupnya bisa dilihat pada gambar berikut :



Gambar 1.1 Daur hidup JSP

Seperti tipe aplikasi java lainnya (Servlet, Applet, Midlet dll), JSP juga bertipe strong Type artinya penggunaan variable pada halaman tersebut harus dideklarasikan terlebih dahulu. Misalnya pada sintaks pengulangan berikut:

```
for (int i=1; i<13; i++)
{
// statement
}
```

Seperti halnya skrip-skrip server side yang lain, JSP pun memerlukan Web server. Skrip ASP memerlukan IIS sebagai web server, PHP memerlukan IIS atau Apache, sedangkan JSP bisa menggunakan Apache Tomcat sebagai salah satu web server yang mendukungnya.

Agar bisa menjalankan file-file JSP yang berbasis Java, diperlukan web server yang mampu memproses Java, atau minimal JSP engine yang dapat terintegrasi dengan web server.

### 1.3 Web Container

Menurut spesifikasi J2EE, dikenal EJB Container, Web Container dan Application Server. Web Container adalah services yang dijalankan oleh suatu Java Application Server hususnya untuk services yang compliance/kompatibel dengan Servlet dan JSP. Selain menjadi services oleh Java Application Server, Web Container dapat berdiri sendiri. Contoh Web Container adalah Tomcat, ServletExec, Resin, Jrun, Blazix. Web Container juga dapat bekerja sama dengan web server, misalnya Tomcat dengan Apache, Jrun dengan IIS.

Web Server adalah software untyk server yang menangani request melalui protokol HTTP yang digunakan oleh situs-situs web saat ini dalam menangani request file statik HTML, sepeti Apache dan Microsoft IIS. Web server sekarang sering “dibungkus” oleh Java Application Server sebagai HTTP Server.

Java Application Server adalah Server yang terdiri atas HTTP Server (Web Server), EJB Container maupun Web Container. Contoh Java Application Server: Sun J2EE RI 1.2/1.3, Borland AppServer 4.5/Enterprise Server 5.0, Oracle9i Application Server dan lainnya.

## 1.4 Jakarta Tomcat

Jakarta Tomcat adalah web application server, yang mempunyai kemampuan sebagai Servlet container dan JSP container di mana Anda bisa mendeploy Servlet dan JSP. Di atas Jakarta Tomcat, Servlet dan JSP akan bekerja melayani request dari client, yang lumrahnya adalah berupa browser.

Untuk bisa menjalankan Jakarta Tomcat, Anda membutuhkan Java Development Kit (JDK). Untuk instalasi Jakarta Tomcat, Anda bisa mendownload binary dari <http://jakarta.apache.org> , dalam format .zip, .tar.gz. Yang Anda perlu lakukan hanyalah mendecompress file tersebut.

Dalam bekerja dengan Jakarta Tomcat, Anda mempunyai sebuah directory yang dikenal sebagai TOMCAT\_HOME. TOMCAT\_HOME adalah directory di mana Jakarta Tomcat diinstall. Selanjutnya di bawah TOMCAT\_HOME Anda akan menemukan beberapa subdirectory, diantaranya bin/, conf/, logs/ dan webapp/. Di dalam subdirectory bin/ terdapat file-file executable terutama untuk menjalankan dan menghentikan Jakarta Tomcat. Di dalam subdirectory conf/ terdapat file-file untuk configuration. Di dalam subdirectory logs/ terdapat file-file log. Dan subdirectory webapp/ adalah di mana Anda bisa meletakkan aplikasi Web yang Anda bangun dengan Servlet dan JSP. Di bawah subdirectory webapp/ Anda bisa mengcreate subdirectory. Sub directory ini akan dijadikan sebagai Context oleh Jakarta Tomcat.

Anda menjalankan Jakarta Tomcat dengan mengexecute **startup.sh** di subdirectory bin/. Sedangkan untuk menghentikan Tomcat Anda mengexecute **shutdown.sh** di sub directory bin/ juga. Secara default Jakarta Tomcat siap melayani request dari client melalui port 8080. Melalui Web browser, Anda bisa menghubungi <http://localhost:8080>

## 1.5 Context

Sebuah **Context** adalah sebuah aplikasi Web yang terpisah, berdiri sendiri, independen. Sebuah Context mempunyai configuration masing-masing. Library dari sebuah Context juga tidak bisa dibaca oleh Context lain. Obyek di sebuah Context tidak bisa mengakses obyek di Context lain.

Di atas sebuah web application server seperti Jakarta Tomcat bisa dideploy lebih dari satu Context. Anda bisa membuat sebuah Context dengan mengcreate sebuah subdirectory di bawah **TOMCAT\_HOME/webapps/**. Dalam folder **webapps/** inilah file JSP ditaruh.

Sebuah Context yang lengkap mempunyai subdirectory **WEB-INF/** di mana terdapat **web.xml** yang merupakan configuration file dari Context ini. Di dalam **WEB-INF/** bisa terdapat subdirectory **classes/** dan **lib/**. Subdirectory **classes/** adalah di mana file-file **.class** diletakkan, sedangkan **lib/** adalah di mana file-file **.jar**, yang merupakan kumpulan file-file **.class**, diletakkan.

## 1.6 Java Virtual Machine

Sebelum menginstal Web Container sebagai prasyarat untuk menjalanka JSP, maka terlebih dulu harus menginstal Java Virtual Machine. Java Virtual Machine adalah software yang berfungsi untuk menerjemahkan program Java supaya dapat dimengerti oleh komputer. Untuk memiliki Java Virtual Machine di komputer, maka perlu mendownload JDK (Java Development Kit) yang tersedia di <http://java.sun.com> karena untuk development diperlukan class-class API. Apabila tidak melakukan proses development dan hanya perlu menjalankan program, maka yang diperlukan hanya JRE (Java Runtime Environment).

## 1.7 Percobaan

### 1. Instalasi Java Virtual Machine

- a. Lakukan Java Virtual Machine sampai selesai.
- b. Lakukan setting path dan classpath dengan cara sebagai berikut:
  - Buka Control Panel - System
  - Pilih tab : Advanced
  - Pilih button: Environment Variables
  - Di bagian system variables lakukan setting PATH dan CLASSPATH sebagai berikut:

Pada variabel PATH tambahkan :

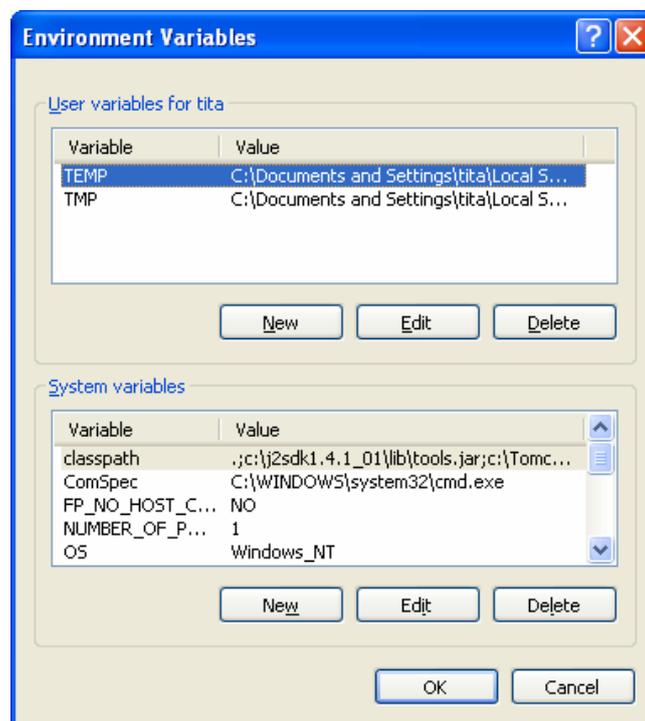
```
c:\nama_folder_tempat_instal\bin
```

Pada variabel CLASSPATH tambahkan:

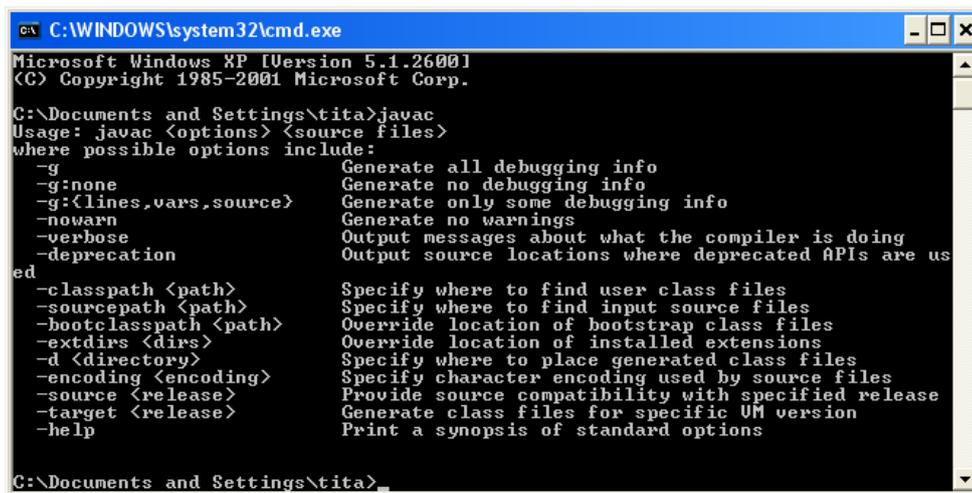
```
.;c:\nama_folder_tempat_instal\lib\tools.jar
```

Bila variabel CLASSPATH belum ada maka buat variabel baru dengan menekan tombol new.

- c. Lakukan pengecekan hasil instalasi dengan cara buka command prompt, Ketikkan perintah **javac**, bila keluar instruksi cara penggunaan maka instalasi telah berhasil.



Gambar 1.2 Contoh setting Environment Variables



```
C:\WINDOWS\system32\cmd.exe
Microsoft Windows XP [Version 5.1.2600]
(C) Copyright 1985-2001 Microsoft Corp.

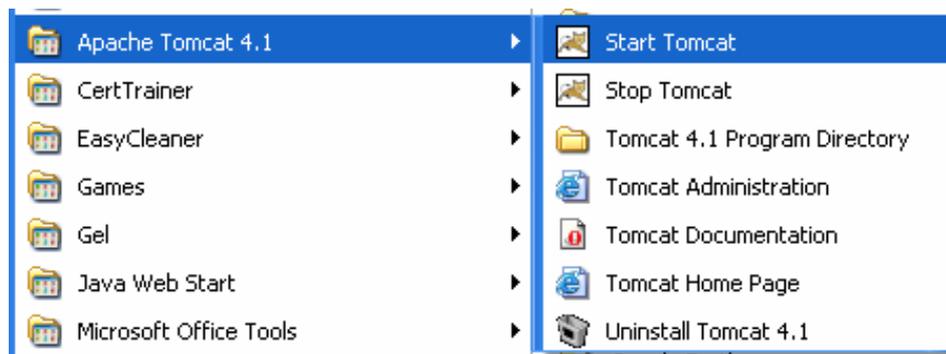
C:\Documents and Settings\tita>javac
Usage: javac <options> <source files>
where possible options include:
  -g                Generate all debugging info
  -g:none           Generate no debugging info
  -g:<lines,vars,source> Generate only some debugging info
  -nowarn           Generate no warnings
  -verbose          Output messages about what the compiler is doing
  -deprecation      Output source locations where deprecated APIs are used
  -classpath <path> Specify where to find user class files
  -sourcepath <path> Specify where to find input source files
  -bootclasspath <path> Override location of bootstrap class files
  -extdirs <dirs>   Override location of installed extensions
  -d <directory>   Specify where to place generated class files
  -encoding <encoding> Specify character encoding used by source files
  -source <release> Provide source compatibility with specified release
  -target <release> Generate class files for specific VM version
  -help            Print a synopsis of standard options

C:\Documents and Settings\tita>
```

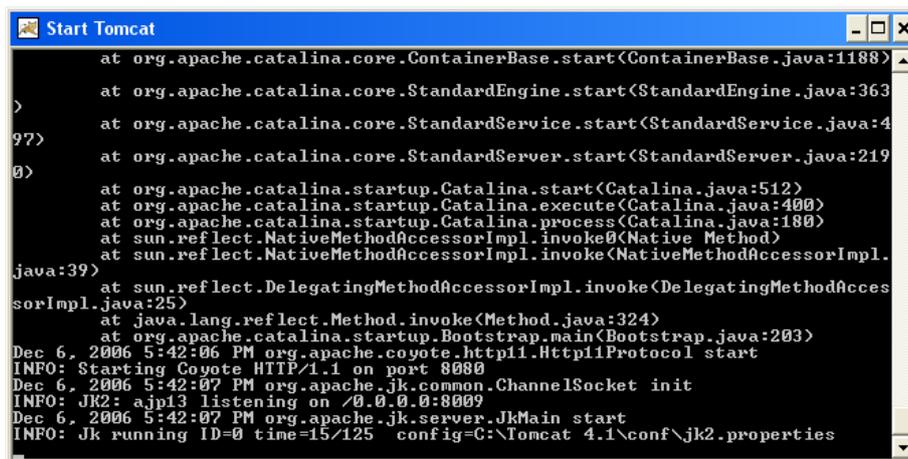
Gambar 1.3 Tampilan bila instalasi j2sdk berhasil

## 2. Instalasi Web Server Tomcat

- a. Instal Web Server Tomcat sampai selesai
- b. Jalankan Web Server Tomcat dengan cara memilih menu Start Tomcat dari pop up menu seperti pada gambar dan akan keluar tampilan seperti pada Gambar 1.3 dan selanjutnya akan keluar tampilan seperti pada Gambar 1.5. Untuk menghentikan Web Server maka pilih menu Stop Tomcat.



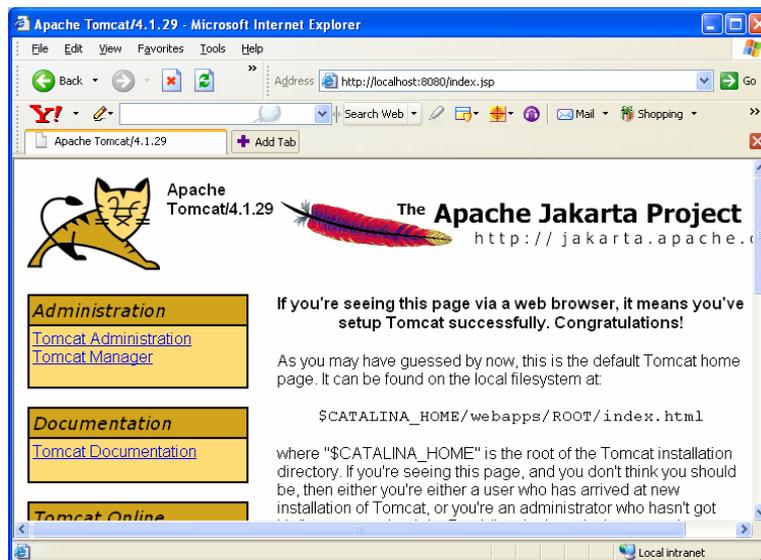
Gambar 1.4 Menu untuk menjalankan Web Server Tomcat



```
at org.apache.catalina.core.ContainerBase.start(ContainerBase.java:1188)
>
at org.apache.catalina.core.StandardEngine.start(StandardEngine.java:363)
>
at org.apache.catalina.core.StandardService.start(StandardService.java:4
97)
at org.apache.catalina.core.StandardServer.start(StandardServer.java:219
0)
at org.apache.catalina.startup.Catalina.start(Catalina.java:512)
at org.apache.catalina.startup.Catalina.execute(Catalina.java:400)
at org.apache.catalina.startup.Catalina.process(Catalina.java:180)
at sun.reflect.NativeMethodAccessorImpl.invoke0(Native Method)
at sun.reflect.NativeMethodAccessorImpl.invoke(NativeMethodAccessorImpl.
java:39)
at sun.reflect.DelegatingMethodAccessorImpl.invoke(DelegatingMethodAcces
sorImpl.java:25)
at java.lang.reflect.Method.invoke(Method.java:324)
at org.apache.catalina.startup.Bootstrap.main(Bootstrap.java:203)
Dec 6, 2006 5:42:06 PM org.apache.coyote.http11.Http11Protocol start
INFO: Starting Coyote HTTP/1.1 on port 8080
Dec 6, 2006 5:42:07 PM org.apache.jk.common.ChannelSocket init
INFO: JK2: ajp13 listening on /0.0.0.0:8080
Dec 6, 2006 5:42:07 PM org.apache.jk.server.JkMain start
INFO: Jk running ID=0 time=15/125 config=C:\Tomcat 4.1\conf\jk2.properties
```

Gambar 1.5 Tampilan Start Tomcat

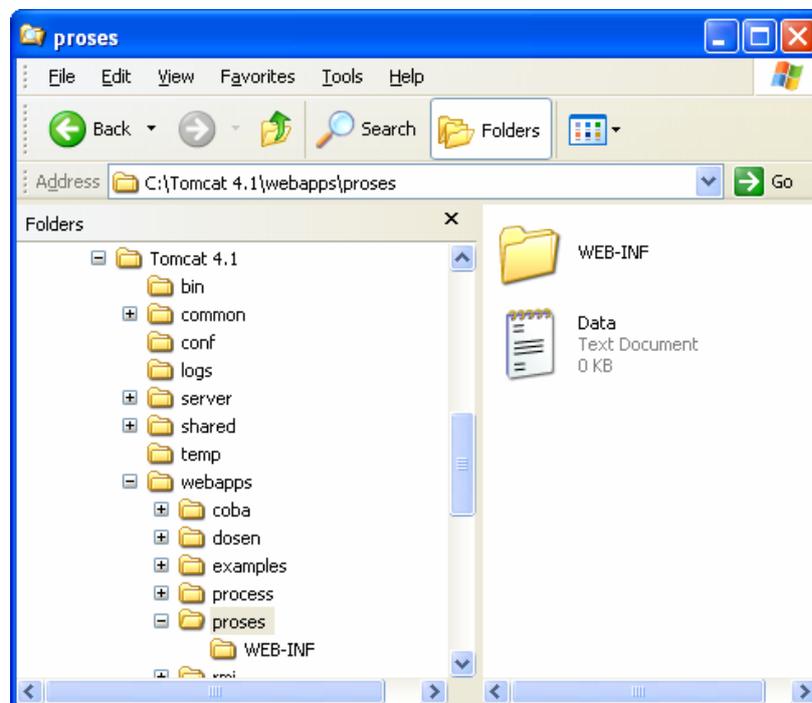
- c. Jalankan browser dan akses alamat <http://localhost:8080>. Bila instalasi web server Tomcat berhasil maka akan keluar tampilan seperti pada Gambar 1.6.

Gambar 1.6 Tampilan [http://localhost :8080](http://localhost:8080)

### 3. Membuat context.

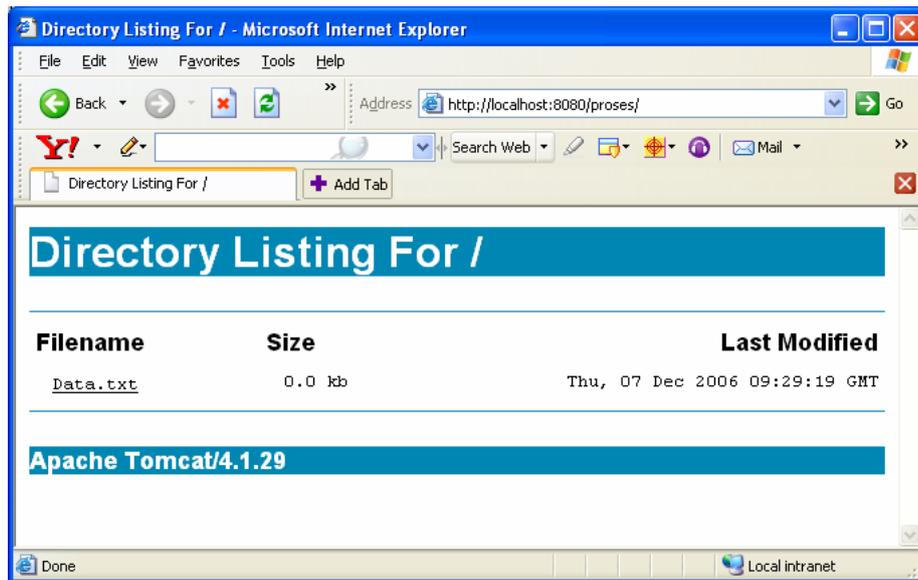
- a. Buatlah sebuah folder (misal dengan nama 'proses') didalam TomcatHome/webapp/. Maka akan didapat hirarki sebagai berikut TomcatHome/webapp/proses.

- b. Selanjutnya kopi folder WEB-INF dari TomcatHome/webapp/ROOT ke TomcatHome/webapp/proses sehingga didapat hirarki TomcatHome/webapp/proses/WEB-INF.
- c. Dengan demikian maka didapat contex baru yang bernama proses. Dalam contex baru inilah file-file jsp ditaruh.
- d. Untuk mencoba web server yang dibuat jalan atau tidak maka taruhlah file dalam contex proses yang baru dibuat misal file Data.txt. Hasil seperti tampak pada Gambar 1.7.



Gambar 1.7 Hirarki folder pada webapps

- e. Buka browser dan ketikkan alamat url <http://localhost:8080/proses>. Bila context berhasil dibangun maka akan didapat tampilan seperti pada Gambar 1.8.



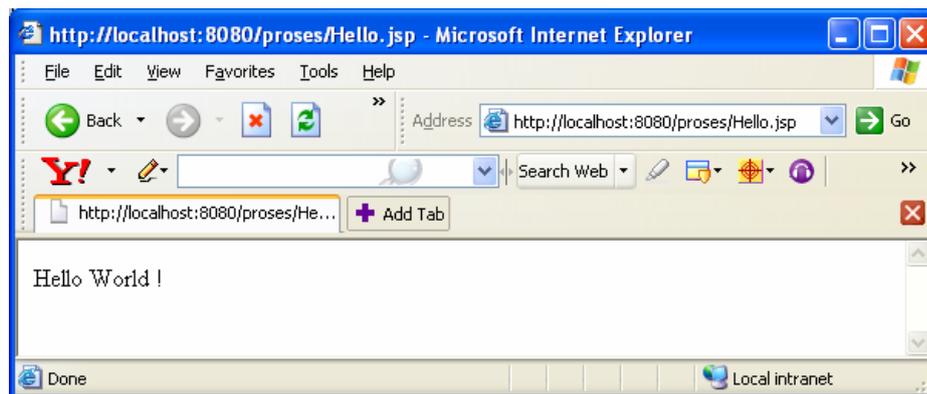
Gambar 1.8 Isi context proses

4. Membuat program sederhana untuk menampilkan halaman JSP pada browser.
  - a. Buatlah file seperti pada Listing 1.1, simpan dengan nama Hello.jsp. Taruh file ini dalam folder proses.

```
<html>
<body>
<% out.println("Hello World !"); %>
</body>
</html>
```

Listing 1.1 Hello.jsp

- b. Akses Hello.jsp dengan cara menuliskan alamat <http://localhost:8080/proses/Hello.jsp> dan akan didapatkan tampilan seperti pada Gambar 1.9.



Gambar 1.9 Tampilan Hello.jsp

## 1.8 Soal Latihan

1. Apa yang dimaksud dengan bahasa scripting?
2. Apa yang dimaksud dengan web server?
3. Apa yang dimaksud dengan web container?
4. Apa yang dimaksud dengan context?
5. Apa kegunaan web server?
6. Sebutkan macam tag HTML, jelaskan kegunaannya, dan berikan contoh masing-masing tag tersebut!
7. Buat context baru yang bernama coba!
8. Membuat program seperti pada Listing 1.2. Simpan dengan nama Pertama.jsp.

```
<HTML>
<HEAD>
  <TITLE>Contoh JSP</TITLE>
</HEAD>
<BODY>
  <H1>
    <%= "Program JSP Pertamaku!" %>
  </H1>
</BODY>
</HTML>
```

Listing 1.2 Pertama.jsp

9. Tampilkan Hai.jsp pada browser!