

Esempio di pagina JSP

```
<html>
  <head>
    <title>Hello World</head>
  </head>
  <body>
    <h1>Hello, World!</h1>
    Current Date and Time is:
    <%= new java.util.Date() %>
  </body>
</html>
```

Identificazione del DB

- Le informazioni di identificazione e modalità di accesso (driver, eventuali parametri, ecc.) vengono specificate in una stringa con formato stile URL:

`jdbc: subprotocol : nome_e_parametri_DB`

Specifica il driver da usare

Formato dipendente dal subprotocol

- Esempi:

`jdbc:odbc://www.polito.it:5000/myDB`

`jdbc:oracle:thin:@ORCL:1025`

Caricamento dei Driver

- Perché il Driver Manager possa attivare un driver jdbc, la sua classe deve essere stata caricata.
- Esistono 2 modalità di caricamento:
 - Chiamando esplicitamente il metodo `Class.forName()`:

```
Class.forName("sun.jdbc.odbc.JdbcOdbcDriver");
```
 - Impostando la proprietà di sistema `jdbc.drivers` (stringa contenente l'elenco delle classi da caricare, separate da ':')
L'impostazione può essere fatta
 - » all'avvio dell'interprete

```
java -Djdbc.drivers=...
```
 - » importando un file di proprietà

```
Properties props = new Properties();  
props.load(new FileInputStream(filename));
```

Apertura di una Connessione

- Una connessione è rappresentata da un oggetto della classe Connection
- Per ottenere una connessione si usa il metodo statico getConnection della classe DriverManager:

```
url = "jdbc:odbc://www.polito.it:5000/myDB";  
user = "sisto";  
pass = "@%%@"  
Connection con =  
DriverManager.getConnection(url, user, pass);
```

Esecuzione di una Query

- Qualsiasi istruzione SQL sul DB viene eseguita tramite un oggetto della classe **Statement**
- I risultati di una Query sono rappresentati da un oggetto di tipo **ResultSet**
- Esempio di esecuzione di una query:

```
Statement stmt = con.createStatement();  
ResultSet rs =  
    stmt.executeQuery("SELECT * FROM BOOKS");
```
- Lo stesso oggetto Statement può servire per più query

Ispezione di un ResultSet

- Il ResultSet corrisponde al record set di ODBC: esso contiene le righe del risultato
- Il record set può essere ispezionato sequenzialmente:

```
while (rs.next()) {  
    String s = rs.getString(1); // accesso alla riga  
}
```

- Il record set contiene un cursore che scorre le righe
- il cursore inizialmente punta prima della prima riga
- il trasferimento dei dati della riga dal server al client avviene solo alla chiamata di un metodo getXXXXX

I Metodi di Accesso

- Vi è un metodo get per ciascun tipo di dato SQL
- Il parametro può specificare
 - il numero d'ordine della colonna da leggere
 - il nome della colonna da leggere
- Esempi:

```
String s = rs.getString(1); // prima colonna
```

```
double d = getDouble("Price"); // colonna Price
```

Corrispondenza tra tipi SQL e Java

INTEGER o INT	int
SMALLINT	short
NUMERIC(n,m), DECIMAL(m,n) o DEC(m,n)	java.sql.Numeric
FLOAT(n)	double
REAL	float
DOUBLE	double
CHARACTER(n) o CHAR(n) o VARCHAR(n)	String
BOOLEAN	boolean
DATE	java.sql.Date
TIME	java.sql.Time
TIMESTAMP	java.sql.TimeStamp
BLOB	java.sql.Blob
CLOB	java.sql.Clob
ARRAY	java.sql.Array

Aggiornamento dei Dati

- L'aggiornamento dei dati (comandi SQL UPDATE, INSERT, DELETE, e tutti i comandi DDL) avviene usando il metodo `executeUpdate`
- Il valore di ritorno è il numero di righe aggiornate
- Esempio:

```
String command = "UPDATE BOOKS "+  
    "SET PRICE=PRICE-5.00 "+"WHERE TITLE NOT LIKE "%HTML%";  
int n = stmt.executeUpdate(command);
```

Transazioni

- Possono essere eseguite solo se il DB le supporta (altrimenti viene generata un'eccezione)
- Per default, la connessione è in modalità autocommit (commit automatico dopo ogni statement SQL). Per disabilitare il commit automatico si usa il comando:

```
con.setAutoCommit(false);
```

- Per eseguire manualmente il commit (o il rollback) si usano i comandi:

```
con.commit()
```

```
con.rollback()
```

Esempio

```
con.setAutoCommit(false);
Statement stmt = con.createStatement();
stmt.executeUpdate(command1);
stmt.executeUpdate(command2);
stmt.executeUpdate(command3);
...
if (allOK)
    con.commit();
else
    con.rollback();
```

Metadati

- Forniscono informazioni sulla struttura di un DB (nomi delle tabelle, nomi e tipi delle colonne, ecc.)
 - sono utili per scrivere applicazioni indipendenti dalla struttura del DB o in grado di funzionare con diverse strutture
- In JDBC vi si accede tramite un oggetto della classe `DatabaseMetaData`:

```
DatabaseMetaData md = con.getMetaData();  
ResultSet rs = md.getTables(null, null, null,  
    new String[] {"TABLE"});  
...
```

Metadati

- E' anche possibile ottenere (meta) informazioni su un record set, con la classe ResultSetMetaData:

```
ResultSetMetaData rsmd = rs.getMetaData();  
int nc = rsmd.getColumnCount();  
String name1 = rsmd洗columnName(1);  
...
```

Elementi di una pagina JSP

- Template text
 - testo HTML standard, copiato tale e quale
- Commenti
 - `<%-- questo è un commento --%>`
 - Eliminati nella compilazione jsp
- Direttive
 - `<%@ direttive %>`
 - Comandi per il compilatore jsp (include, ecc.)

Elementi di una pagina JSP in Linguaggio di Scripting

- Dichiarazioni

<%! dichiarazioni %>

- Dichiarazioni di variabili e metodi che verranno inserite nel corpo del servlet

- Scriptlet

<% scriptlet %>

- Codice Java che controlla la generazione di codice HTML

- Espressioni

<%= espressione %>

- Espressione che verrà valutata dal servlet e scritta in posizione corrispondente

Il Linguaggio di Scripting

- Può essere Java o Javascript
- Ha accesso ad oggetti del servlet:
 - Request
 - Response
 - Out
 - PageContext
 - » dà accesso agli oggetti del contesto del servlet:
 - Session
 - Application (è il ServletContext)
 - Config

Esempio

```
<html>
<head>
  <title>datetime</title>
</head>
<%@ page import="java.util.*, java.text.*" %>
<%!
  DateFormat dateFormat = new SimpleDateFormat("EEEE d MMMM yyyy");
  DateFormat timeFormat = new SimpleDateFormat("H:mm");
%>
<% Date dateTime = new Date(); %>
<body style="font-size: 16pt;">
Oggi &egrave; <%= dateFormat.format(dateTime) %><br>
e sono le ore <%= timeFormat.format(dateTime) %>
</body>
</html>
<html>
```

Esempio di Pagina legata ad un Form

```
<html>
<head>
  <title>Hello</title>
</head>
<body style="font-size: 20pt;">
<% String name = request.getParameter("name"); %>
<% if (name == null || name.length() == 0) { %>
Ciao, chiunque tu sia!
<% } else { %>
Ciao <%= name %>!
<% } %>
</body>
</html>
```

Esempio di Form

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0  
  Transitional//EN">  
<html>  
<head>  
  <title>Hello</title>  
</head>  
<body style="font-size: 16pt;">  
<form action="Hello.jsp" method="post">  
Scrivi il tuo nome  
<input type="text" name="name">  
<input type="submit" value="Clicca qui">  
</form>  
</body>  
</html>
```