Prendo l'es. 6 e aggiungo ajax e json.
Tasto destro in chrome → ispeziona elemento: compare un albero della pagina.

Esempio di codice .html con json:

```
// Pagina html 5 chiamata  "prove.html"
// Nello scritp di javascript scrivo le funzioni:
// Appena carico una pagina voglio eseguire una pagina e lo faccio con onLoad nel body il quale chiama una funzione javascript
function paginaCaricata(tipo){
    alert("la pagina è stata caricata");
    eval(alert("la pagina è stata caricata"));  // eval passa come stringa un pezzo di sorgente javascript da interpretare e viene eseguito (a compile time è una stringa e basta)

    eval(tipo + "('la pagina è stata caricata')");  // "tipo" è una variabile passata alla funzione che non serve per le altre.
    // Creo i due href nel body con alert e confire legati a questo
    // eval vuole uno statement

    var risultato = eval("1 + x");  // x è una variabile dichiarata prima
    // In genere nell'eval noi useremo solo costanti

    // Mappe su java
    x = {"chiave1" : "valore1", "chiave2" : "valore2");
    var risultato = x['chiave1'];  // Mostrerà valore1

    var json = "{ 'studente1' : {'nome' = 'Gian' , 'cognome' = 'Rossi'} ,
    'studente2' : {'nome' = 'Mario' , 'cognome' = 'Bianchi'} }";  // Mappa di mappe; json (javascript object notation): modo per scrivere stringhe che se mandate a javascript sono facili da leggere.
    // In altri linguaggi per usare json servono librerie in più, qui basta eval.
    eval("risultato =" + json);
    var s = risultato["studente2"];  // Un bravo compilatore mi direbbe che questo è un errore perché "risultato" non è stato mai dichiarato.
    alert(s["cognome"]);

// FINE ESEMPIO
```

ECLIPSE

Esercizio7

src

- it.unipr.informatica.esercizio7
  - Sessione.java
- it.unipr.informatica.esercizio7.database
  - DatabaseManager.java
- it.unipr.informatica.esercizio7.modello
  - Modello.java
  - ModelloException.java
  - Studente.java
- it.unipr.informatica.esercizio7.web
  - ListaStudentiServlet.java
  - **ListaStudentiJSONServlet.java**
  - AggiornaStudenteServlet.java
  - DettagliStudenteServlet.java
  - InserisciStudenteServlet.java
  - ModificaStudenteServlet.java
  - RimuoviStudenteServlet.java
- configurazione.properties

WebContent

- WEB-INF
  - lib
    - derbyclient.jar
  - web.xml
- index.html
- errore.html
- lista_studenti.jsp
- **lista_studenti_json.jsp**
- aggiungi_studente.jsp
- modifica_studente.jsp
- dettagli_studente.jsp
- **stile.css**

lista_studenti_json.jsp

```
<%@ page import="it.unipr.informatica.esercizio7.Sessione"%>
<%@ page import="it.unipr.informatica.esercizio7.modello.Studente"%>
<%@ page import="java.util.List"%>
<%@ page import="it.unipr.informatica.esercizio7.database.DatabaseManager"%>
<%@ page language="java" contentType="text/html; charset=ISO-8859-1"
    pageEncoding="ISO-8859-1"%>
<!DOCTYPE html>
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-1">
<title>Esercizio 7</title>
<script type="text/javascript">
function rimuovi(matricola) {
      var sicuro = confirm("Sei sicuro di voler cancellare lo studente con matricola: " + matricola);
```

```
    if(sicuro == true)
        location.href = "rimuovi_studente?matricola="+matricola;
}

function nuovo_stato_connessione() {
    if(richiesta.readyState != 4)  // Solo se la connessione è avvenuta con successo (4), proseguo.
        return;
    if(richiesta.status != 200)  // Solo se ho dei dati (200), proseguo.
        return;

    var json = richiesta.responseText;
    eval("risultato = " + json);
-   alert(json);  // Per controllare
    // Al posto della alert posso fare un for che aggiunga testo html per formattare il testo originale.
/* ajax è un asincrono di javascrip e xml.
Non si applica quando cambio una pagina ma quando arriva un evento (come ad es. il caricamento di una
pagina).
Usa javascript; parla con il server per farsi dare dell'xml quindi, al posto di farci dare dell'xml, uso eval che
interpreta codice java molto facile quindi più veloce (per interpretare xml serve una libreria).
Questo tipo di approccio (anche senza xml) si chiama "richieste rest".
Molti servizi vengono offerti tramite questa interfaccia rest.
L'unica difficoltà è che si usano tante cose insieme ma tecnicamente è una cosa molto semplice.
È meta programmazione che significa che un programma scrive parte di se stesso.*/

    var indice = 0;  // Usa lo 0 come numero e non come stinga.
    while(risultato[indice] != null) {  // java ha null e basta; javascript ha anche undefined diverso da null;
    // Cercare qualcosa che non c'è dovrebbe ritornare undefine
    // Variabili:
        var studente = risultato[indice];
        var matricola = studente["matricola"];
        var cognome = studente["cognome"];
        var nome = studente["nome"];
        var riga = document.createElement("tr");  // Creo la riga che andrò ad inserire.
        var stringa = "<td><a href='dettagli_studente?matricola="+ matricola +"'>Dettagli</a></td>";
        stringa += "<td><a href='modifica_studente?matricola="+ matricola +"'>Modifica</a></td>";
        stringa += "<td><a href='javascript:rimuovi(" + matricola + ")'>Rimuovi</a></td>";
        stringa += "<td><a href='dettagli_studente?matricola=" + matricola + "'>" + matricola +
                                                                    "</a></td>";
        stringa += "<td><a href='dettagli_studente?matricola=" + matricola + "'>" + cognome +
                                                                    "</a></td>";
        stringa += "<td><a href='dettagli_studente?matricola=" + matricola + "'>" + nome +
                                                                    "</a></td>";
        riga.innerHTML = stringa;  // Scrivo all'interno della riga "tr", la "stringa" creata
        tabella.appendChild(riga);  // Aggiungo la riga (come figlia) alla tabella
        indice++;
    }
}

function caricaTabella() {
    tabella = document.getElementById("tabella");  // Mi faccio dare l'elemento tramite id "tabella"
    tabella.innerHTML = "";  // Va a scrivere "" tra inizio e fine tabella (la svuota)
```

```
        // Esiste un metodo javascript che viene chiamato dalla inner e che permette di modificare l'html
        var riga = document.createElement("tr");
        // Creo intestazione:
        riga.innerHTML = "<td></td><td></td><td></td><td class='intestazione'>Matricola</td>
                        <td class='intestazione'>Cognome</td><td class='intestazione'>Nome</td>";
        tabella.appendChild(riga);
        richiesta = null;

        if(window.XMLHttpRequest)
            richiesta = new XMLHttpRequest();  // In internet explorer (eclipse) non funziona perché esso usa un oggetto diverso da XMLHttpRequest
        else
            richiesta = new ActiveXObject("Microsoft.XMLHTTP");

        richiesta.onreadystatechange = nuovo_stato_connessione;  // Indica quale funzione chiamare quando lo stato della connessione cambia (richiesta fallita, risultato non disponibile, se faccio open, send etc)
// Creo una nuova funzione “nuovo_stato_connessione” che indica lo stato della connessione, in cui ho copiato molto codice che prima era contenuto in questa funzione “caricaTabella”.
        richiesta.open("GET", "lista_studenti_json", true);  // “false” vuol dire che viene fatta la richiesta, e solo quando ottengo un risultato, la chiamata va avanti; con “true” va avanti subito.
        // "lista_studenti_json" url della servlet
        richiesta.send(null);  // E' necessario.
}
</script>
<link href="stile.css" rel="stylesheet" />  // Collega la pagina css.
        // La specifica di stile più vicina, vince.
        // Lo stile non posso applicarlo al testo libero ma solo ad un tag (assegno un tag span al testo).
        // P.S.: ci sono normative che obbligano a inserire la partita IVA in un certo posto della pagina.
</head>
<body>
<h1>Sistema di Gestione Studenti</h1>
<h2>Attività disponibili</h2>
<body onLoad="caricaTabella()">  // “onLoad” al caricamento della pagina esegue la funzione indicata.
                // In questo caso, carico la tabella quindi nell'html di partenza la tabella è vuota.
<%
        String messaggio = (String)session.getAttribute("messaggio");

        if(messaggio != null) {
            session.removeAttribute("messaggio");
%>
<p class="testo_rosso"><%=messaggio %></p>
<%
        }
%>
<p><a href="aggiungi_studente.jsp">Aggiungi studente</a></p>
<p><a href="javascript:ricaricaTabella()">Ricarica tabella</a></p>
<table class="tabella" id="tabella">
</table>
</body>
</html>
```

ListaStidentiJSONSServlet.java

```
package it.unipr.informatica.esercizio7.web;
import it.unipr.informatica.esercizio7.Sessione;
...

@SuppressWarnings("serial")
public class ListaStudentiJSONServlet extends HttpServlet {
	protected void doGet(HttpServletRequest request, HttpServletResponse response) throws
	ServletException, IOException {
		try {
			HttpSession session = request.getSession();
			Sessione sessione = (Sessione)session.getAttribute("sessione");
			if(sessione == null) {
				sessione = new Sessione();
				session.setAttribute("sessione", sessione);
			}

			DatabaseManager databaseManager = sessione.getDatabaseManager();
			List<Studente> studenti = databaseManager.ricaricaStudenti();
			response.setContentType("text/plain");  // Risposta; di default lo trasforma in testo

			String risultato = "{ ";  // Testo json che si apre e chiude sempre con una graffa
			int indice = 0;  // Contatore per sapere di quale studente parlo
			for (Studente studente : studenti) {  // Fa scorrere gli studenti ( "studente" conterrà ad ogni passo
del ciclo, uno studente diverso della lista "studenti").
				risultato += "'" + indice + "' : {";
				risultato += "'matricola' : '" + studente.getMatricola() + "', ";
				risultato += "'cognome' : '" + studente.getCognome() + "', ";
				risultato += "'nome' : '" + studente.getNome() + "'";
				risultato += " }";
				indice++;

				if(indice < studenti.size())
					risultato += ", ";
			}

/* webservice: genera contenuto lato server; posso mandare parametri tramite get e post, e genera un html
strutturato, deve si ricevono dati tramite https o http.
Dopo webservice scrivo rest e so che riceverò i parametri in un certo formato (separati da $).
C'è anche un modo con il quale si ricevono i dati come unico parametro (che è un testo xml) dove metto
dentro i parametri (e come risultato ottengo un xml).
Manipolare xml è complicato quindi si preferisce sempre di più il metodo rest con risultato json.
Con webservice non posso mettermi in attesa di un evento, ma sono comunque i più usati per i servizi.
P.S. Si pronuncia "webservice" (senza s finale). */
			risultato += " }";

			System.out.println(risultato);  // Non necessario

			PrintWriter out = response.getWriter();  // Prendo ciò che ho stampato e lo metto nella variabile
"out" di classe "PrintWriter"
			out.println(risultato);  // Stampo il "risultato" come testo di risposta
			out.flush();
```

```
        } catch(Throwable throwable) {
            request.getRequestDispatcher("errore.html").forward(request, response);
        }
    }

    protected void doPost(HttpServletRequest request, HttpServletResponse response) throws
    ServletException, IOException {
    doGet(request, response);
    }
}
```

stile.css

```
@CHARSET "ISO-8859-1";

html {
    font-family: sans-serif;
}

body {  // Uso parentesi graffe; fa riferimento a tutti i body
    background-color: #e0e0e0;
}

h1 {
    text-align: center;
    margin-top: 1em;  //1em è uno spazio alto come la media del font usato
    margin-bottom: 1em;
}

.testo_rosso {  // Fa riferimento ad un id
    background-color: red;
}

.tabella {
    width : 80%;
    margin-left: 10px;
}

td.intestazione {  // Si applicano solo ai td che sono di class="intestazione"
    font-style: italic;
    margin-top: 0.5em;
    margin-bottom: 0.5em;
}
```