F10
Webbteknologier

EDA095 Nätverksprogrammering

Roger Henriksson
Datavetenskap
Lunds universitet

Dynamiska webbsidor

- HTML är statisk. En sida får sitt utseende bestämt när en webbdesigner skapar den.
- Ofta vill man ha mera dynamiska webbsidor:
  - Svar på en databasförfrågan.
  - Konstant uppdaterade webbsidor.
  - Dialog med användaren.
  - Animeringar.
  - Kontroll av inmatad information i ett formulär.

Serversidan eller klientsidan?

Klientsidan

- JavaScript
- Flash

Serversidan

- CGI – CommonGateway Interface
- JSP (Java Server Pages) och Servlets
- ASP – Active Server Pages
- PHP – "PHP:Hypertext Preprocessor"

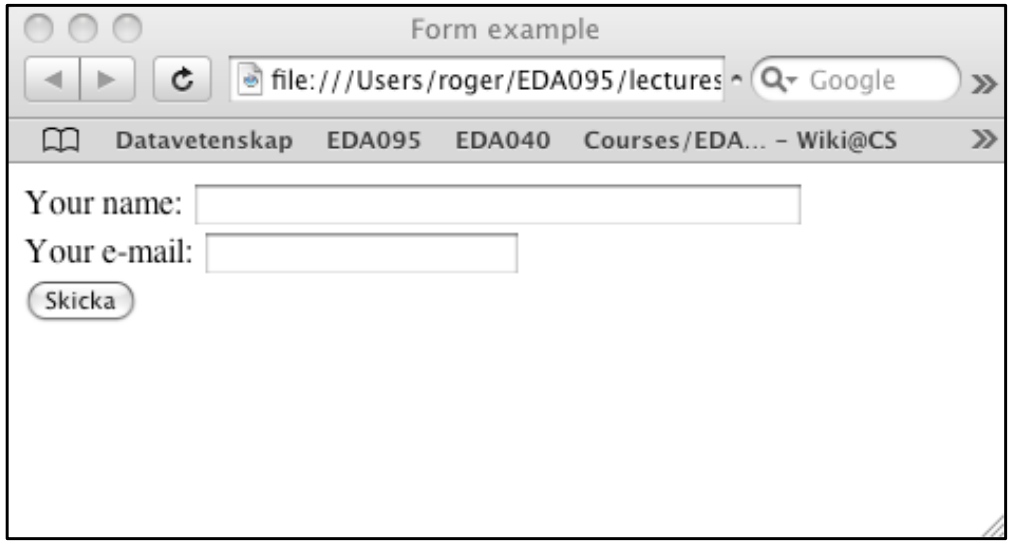
Varför inte en helt specialskriven webbserver?

Common Gateway Interface

1. När webbservern får en begäran om en webbsida med en särskild URL startar servern ett externt program – ett "CGI-skript".
2. Det externa programmet läser in eventuella parametrar i form av en "query string" antingen via standard input eller s.k. "environment-variabler".
3. Programmet genererar en HTML-sida baserat på parametrarna och skriver HTML-koden till standard output.
4. Programmet avslutas.

HTML och formulär

```
<html>
<head><title>Form example</title></head>
<body>
<form method="get" action="/cgi-bin/storeaddress.pl">
Your name: <input name="name" type="text" size=40>
<br>
Your e-mail: <input name="email" type="text" size=20>
<br>
<input type="submit">
</form>
</body>
</html>
```



HTTP-förfrågan

Föregående exempel genererar en TCP-uppkoppling till servern och en HTTP-förfrågan sänds:

```
GET /cgi-bin/storeaddress.pl?name=Roger+Henriks
son&email=roger%40cs.lth.se HTTP 1.0
```

1. Servern startar skriptet “storeaddress.pl”.
2. Frågesträngen (“query string”) överförs via en environment-variabel.

POST istället för GET

```
POST /cgi-bin/storeaddress.pl HTTP 1.0
Content-type: application/x-www-form-urlencoded
Content-length: 49

name=Roger+Henriksson&email=roger%40cs.lth.se
```

Begäran består av ett huvud (avslutat med dubbla radslut (CR+LF+CR+LF) och en frågesträng.

CGI-skriptet läser frågesträngen via standard input.

Lämpligt för stora datamängder.

Frågesträngen syns ej i URL:en.

Svar

CGI-skriptet skriver till standard output:

1. MIME-typ, typiskt "Content-type: text/html".
2. Blankrad
3. HTML-kod för den genererade sidan.

Exempel

```
Content-type: text/html

<html>
<head><title>Registration completed</title></head>
<body>
<h1>Registration completed</h1>
Roger Henriksson (roger@cs.lth.se) has been added to the
database.
</body>
</html>
```

CGI – Exempel

Implementera en CGI-baserad webbtjänst för temperaturmätning.

DEMO

CGI: fördelar och nackdelar

Fördelar

- Möjlighet att välja mellan många olika implementationsspråk.
- Väl beprövat och allmänt tillgängligt.

Nackdelar

- Ineffektivt: startar en ny operativsystemprocess för varje HTTP-begäran.
- Måste avkoda frågesträngen själv.
- Besvärligt att bevara tillstånd – måste spara på disk.

Servlets

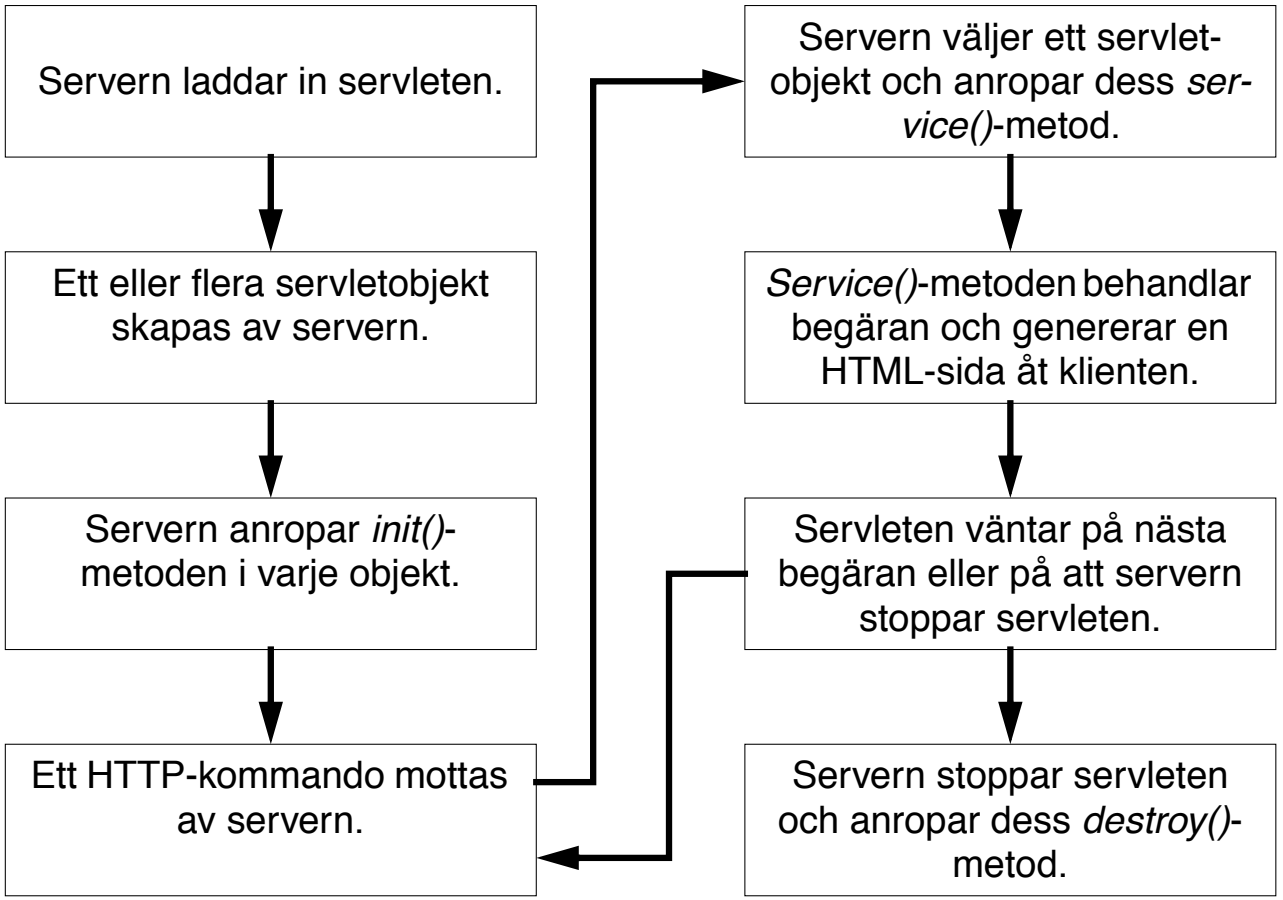
Servlet?

Applet - "liten applikation", Servlet - "liten server"

Som CGI, fast:

- Skrivna i Java.
- Systemoberoende.
- Skapar inte ny operativsystemsprocess varje gång. Effektivare!
- Startar inte om för varje HTTP-begäran. Kan komma ihåg information från gång till gång.

En servlets liv



Klassöversikt

Paket

```
import javax.servlet.*;      // Standard i J2EE
import javax.servlet.http.*; // Standard i J2EE
import java.io.*;
```

Klasser/interface

- HttpServlet – superklass för webbservlets.
- HttpServletRequest – klientens HTTP-begäran.
- HttpServletResponse – servletens svar till klienten.
- ServletConfig – information om servern.

HttpServlet

Ett servletobjekt skapas av webbservern.

Initialisering

Implementera en av nedanstående:

```
public void init(ServletConfig config)
                                    throws ServletException;
public void init();
```

Terminering

```
public void destroy();
```

Implementera denna för att till exempel stänga databasuppkopplingar/stänga öppna filer när servleten avslutas av servern.

service()

När servern tar emot ett HTTP-kommando anropas:

```
protected void service(HttpServletRequest request,
                       HttpServletResponse response)
                throws ServletException, IOException;
```

Denna kan implementeras för att behandla en begäran från klienten.

- request – information om klientens begäran
- response – används för att skicka svar till klienten.

Ofta vill vi göra olika saker beroende på typ av begäran:

```
if (request.getMethod().equals("GET")) {
  ...
} else {
   if (request.getMethod().equals("POST")) {
     ...
```

Alternativ till service()

Standardimplementationen av service() undersöker vilken typ av kommando klienten skickade (GET/POST/HEAD etc) och anropar en av:

```
protected void doGet(HttpServlerRequest request,
                     HttpServletResponse response)
                         throws ServletException,IOException;
protected void doPost(HttpServlerRequest request,
                      HttpServletResponse response)
                         throws ServletException,IOException;
```

Likadant för:

doHead
doPut
doDelete

HttpServletRequest

Information om klientens begäran.

Metoder

`public String getParameter(String name);`
Returnerar värdet för angiven parameter, t.ex. innehållet i ett textfält i ett HTML-formulär.

`public String getRemoteAddr();`

`public String getRemoteHost();`
IP-nummer / namn på klientdatorn.

`public String getMethod();`
Typ av begäran (GET, POST, etc.).

Med flera...

HttpServletResponse

Används för att skicka svar till klienten.

1. Ange MIME-typen för svaret:
   ```
   response.setContentType("text/html");
   ```
2. HTML-koden skickas genom en ström:
   ```
   PrintWriter output = response.getWriter();
   ```
3. Skriv HTML-koden till strömmen.
4. Stäng strömmen:
   ```
   output.close();
   ```

Tillståndsinformation

Kommunikationen mellan klient och webbserver är ofta en dialog - jmf en webbshop.

Servern måste hålla reda på vad som hänt under dialogen, dvs hålla reda på ett tillstånd.

Tillstånd kan lagras i servleten, men:

Vi måste kunna skilja på olika klienter!

- Gömda fält i formulär

```
<input type="hidden" name="number" value="42">
```

- Cookies
- HttpSession

Klassen Cookie

Namn/värdepar som lagras på klienten.

Paket

```
import javax.servlet.http.*;
```

Konstruktor

```
public Cookie(String name, String value);
```

Metoder

```
public String getName();
public String getValue();
```

Med flera...

Skriva/läsa cookies

Metoder i klasserna HttpServletRequest/HttpServletResponse.

HttpServletRequest

```
public Cookie[] getCookies();
```

Returnerar en vektor med samtliga cookies från denna webbplats.

HttpServletResponse

```
public void addCookie(Cookie cookie);
```

Skriver ner en ny cookie eller ny version av en gammal cookie till klienten.

DEMO – CookieGuess

JSP – Java Server Pages

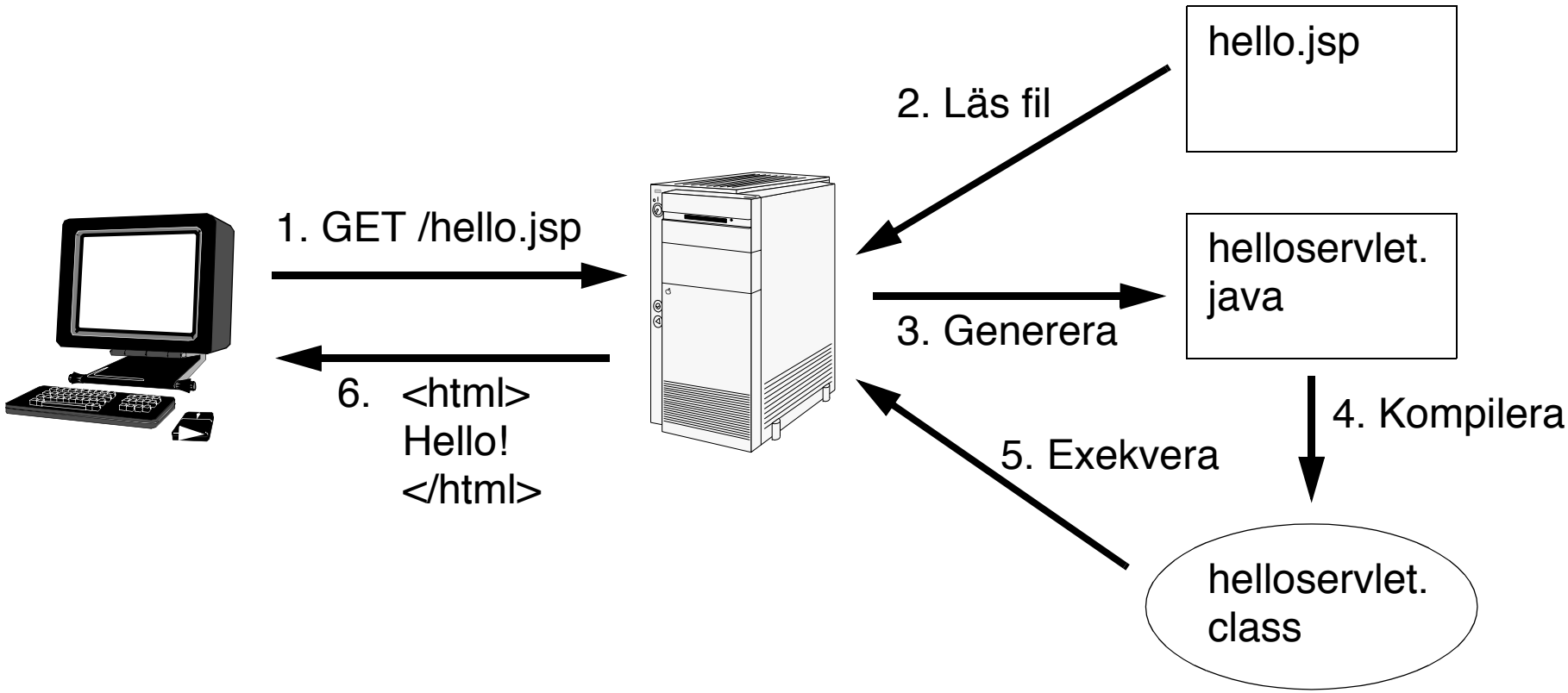
Idé

"Programkoden i HTML-koden" istället för "HTML-koden i programkoden".

Implementation

- En JSP-fil (.jsp) är en HTML-fil med några extra element i.
- JSP-elementen ("tags") anger var dynamisk HTML-kod ska infogas i dokumentet och hur den ska genereras.
- När en klient begär JSP-filen tolkas innehållet av servern och görs automatiskt om till en servlet första gången.

Översättning av JSP-fil



JSP – Tags

Direktiv
Anvisningar för översättningen till servlet. "<%@ ... %>"

Deklarationer
Deklarationer av attribut motsvarande servletattribut. "<%! ... %>"

Uttryck
Anger javauttryck vars värde stoppas in på sidan. "<%= ... %>"

Scriptlets
Block av javakod som exekveras när JSP-sidan anropas. "<% ... %>"

Kommentarer
"<%-- ... -- %>"

JSP – Direktiv

Styr översättningen till en servlet.

"page"

Styr servletens struktur: Importerar externa klasser, ändrar "content type", ändrar servletens superklass.

Exempel:

```
<%@ page import="java.util.*" %>
<%@ page contentType="text/plain" %>
```

"include"

Inkluderar andra JSP-filer vid översättningen.

Exempel:

```
<%@ include file="filetoinclude.jsp" %>
```

JSP – Deklarationer

Används för att deklarera variabler som sedan kan användas i uttryck och i "scriptlets".

Motsvarar attributen i en servletklass.

Exempel:

```
<%! int counter = 0; %>
<%! Date today = new Date(); %>
```

Flera deklarationer kan samlas:

```
<%!
   int counter = 0;
   Date today = new Date();
%>
```

JSP – Uttryck

Används för att stoppa in resultatet av en beräkning eller annat uttryck i HTML-koden.

Exempel:

```
<%= counter %>
<%= today.toString() %>

Pris (inkl. moms): <%= pris*1.25 %> kronor.
```

Kan innehålla godtyckliga javauttryck.

JSP – Scriptlets

Anger Javakod som ska exekveras när sidan hämtas.

Exempel:

```
<%
    total = 0.0;
    for(int i=0;i<myArray.length;i++) {
       total = total+myArray[i];
    }
%>

Average: <%= total/myArray.length %>
```

JSP – Implicita objekt

Ett antal standardobjekt finns alltid tillgängliga utan explicit deklaration.

`request` – HTTP-begäran från klienten.

`response` – HTTP-svaret till klienten.

`session` – HTTPSession-objekt associerat till den aktuella användaren/sessionen.

`application` – Refererar till "globala" objekt som ska delas mellan alla sessioner, t.ex. databasanslutning.

`out` – Objekt som används för att skriva till den utgående svarsströmmen (till klienten).

Med flera...

JSP – Exempel

DEMO – guessinggame.jsp

PHP – PHP: Hypertext Preprocessor

- Skriptspråk särskilt lämpligt för webbapplikationer.
- Öppen, gratis, programvara.
- HTML-kod med PHP-taggar inlagda där dynamiskt beteende behövs (jämför JSP).
- Vanlig och mycket spridd teknik:
  - CMS – Content Management Systems (ex LTH).
  - Wiki – Många wikiimplementationer i PHP.
  - Webbkopplingar till databassystem, t.ex. webbshopgränssnitt.
- URL: www.php.net

Struktur/taggar

HTML-kod med PHP-taggar insprängda.

```
<?php
    ...
?>
```

All text utanför PHP-taggarna skickas oförändrad till klienten.

Filändelse: .php

Datatyper

Dynamiska typer, ingen variabeldeklaration behövs.

boolean	Sant/falskt.
integer	Heltal.
float	Flyttal.
string	Strängar.
array	Vektorer.
object	Objekt.
resource	Referens till externa resurser, t.ex. filer.
NULL	Tomt värde.

Variabler inleds med $, exempel: `$value`.

Språkkonstruktioner

Några språkkonstruktioner:

```
$a = $b*3;  // Tilldelning, uttryck

echo “Svar: “.$a.”\n”; // Utskrift,
                       strängkonkatenering

if ($a==0) { ... } else { ... } // If-sats

for($i=0;$i<10;$i++) { ... } // For-sats
```

Vektorer

Vektorer är associativa! (Egentligen en "ordered map".)
Index kan vara av godtycklig typ.

```
$arr = array();    // Tom array
$arr["roger"] = "duktig";
$arr[2] = 45;
$arr[] = "NP"; // Index: högsta numeriska
               // index + 1
```

Iterator över vektorn:

```
foreach($arr as $key => $value) {
   echo $key.":".$value."\n";
}
```

Funktioner

```
<?php
function sum($arg_1, $arg_2, $arg_3)
{
    echo “Example function\n”.
    $retval = $arg_1+$arg_2+$arg_3;
    return $retval;
}
?>

Summan är: <?php echo sum(2,5,8); ?>
```

Biblioteksfunktioner

Stor mängd inbyggda biblioteksfunktioner.
Betoning på webbrelaterade uppgifter.

Audio	*Autentisering*	*Datum/tid*
Komprimering	*Kreditkorts-betalningar*	*Filhantering*
Teckenkodning	*Stränghantering*	*Bildmanipulering*
Mail	*Matematik*	*Nätverk*
Kryptering	*Databaser (SQL)*	*Processhantering*

Med mera...

Koppling till webbservern

Request

Parametrar accessas genom "superglobala" variabler, t.ex:

`$_GET`	Parametrar vid GET-anrop. Vektor.
`$_PUT`	Parametrar vid PUT-anrop. Vektor.
`$_COOKIE`	Cookies. Vektor.

Response

HTML-kod skrivs till standard output.

Cookies: `setcookie(...);`

En elektronisk shoppinglista

Fallstudie:

Implementera en webbaserad tjänst för att hantera en familjs gemensamma shoppinglista.

- Lista över dagligvaror som saknas i hemmet.
- Enkel inläggning av varor från Internetanslutna enheter.
- Stöd både för att skapa listan och vid shoppingtillfället.

DEMO – shoppinglist.php

Design

Lagring

- Textfil med en rad per artikel.

Webbsida

Formulär med:

- Checkbox för varje artikel.
- Textinmatningsfält för ny artikel

Implementation

PHP-skript:

1. Läs in artiklar från fil.
2. Om checkbox ikryssad: tag bort motsvarande artikel.
3. Om textinmatningsfält ifyllt: lägg till artikel.
4. Om ändringar gjorts: spara artiklar på fil.
5. Skapa nytt formulär med aktuella artiklar.

JavaScript

Skriptspråk för webbläsare

Syntax som påminner lite om Java. I övrigt har JavaScript och java inget med varandra att göra.

Historia

Skapat av netscape. Ursprungligen kallat LiveScript. Fick namnet JavaScript i Netscape 2.0.

Microsofts variant, JScript, nästan kompatibelt.

Vad kan JavaScript göra?

JavaScript kan t.ex:

- Göra beräkningar på tal och strängar.
- Modifiera utseende och innehåll i ett webbläsarfönster.
- Skapa nya fönster och ladda in nya sidor.
- Manipulera HTML-element på en webbsida, t.ex. innehåll i ett formulär.

JavaScript kan inte:

- Kommunicera direkt via nätverket (via UDP/TCP).
- Använda filsystemet.

JavaScript och webbsidor

JavaScript kan läggas in som en del av HTML-koden för en webbsida.

JavaScript kan antingen fås att exekvera när webbsidan laddas, eller när någon speciell händelse inträffar.

JavaScript vid laddning av en webbsida

Skriptet körs i samband med att webbsidan ritas upp. Utdata från skriptet kompletterar den övriga HTML-koden före uppritning.

Händelsestyrda JavaScript

Skriptet körs när en viss händelse inträffar, t.ex. att användaren klickar i webbläsarfönstret. Kan påverka webbläsarfönster eller någon HTML-komponent.

<script>-tagen

HTML-kod för att definiera ett JavaScript

Syntax:

```
<script>
...
</script>
```

eller

```
<script language="JavaScript">
...
</script>
```

Generell HTML-tag för skript oavsett skriptspråk.
JavaScript är oftast standardskriptspråk.
Andra möjligheter: VBScript.

Hello, world!

```
<html>
<head><title>HelloWorldScript</title></head>
<body>
<h1>A greeting from our JavaScript:</h1>
<script language="JavaScript">
document.write("Hello, World!");
</script>
</body>
</html>
```

Datatyper

Variabeldeklaration – dynamiska typer!

```
var name;
```

Typer

tal	Numeriska tal, både heltal och flyttal.
sträng	Texter ungefär som i Java.
boolean	Sant eller falskt.
undefined	Värde oinitialiserade variabler har.
null	Tomt värde.

Dessutom finns vektorer och objekt.

Några satskonstruktioner

Tilldelning

```
a = a + 3;
```

If-sats

```
if (a>b) {
   document.writeln("Tal A är störst!");
} else {
   document.writeln("Tal A är INTE störst!");
}
```

For-sats

```
for(i=1;i<=10;i++) {
   document.writeln(i+" "+i*i);
}
```

Väldigt likt Java!

Funktioner

Precis som man kan definiera metoder i Java kan man definiera funktioner i JavaScript.

```
<script language="JavaScript">
function printSquare(x) {
   document.writeln(x+" "+x*x+"<br>");
}

var i;
for(i=1;i<=30;i++) {
   printSquare(i);
}
</script>
```

Kan till exempel användas för att beskriva vad som ska hända när någon viss händelse inträffar.

Standardobjekt

För att påverka omvärlden, t.ex. webbläsarfönstret, används olika s.k. standardobjekt.

Exempel:

navigator	Tillhandahåller information om webbläsaren.
window	Funktionalitet för att öppna/påverka fönster.
document	Motsvarar HTML-dokumentet i webbläsaren och ger tillgång till alla HTML-komponenter i detta, t.ex. formulär.

Standardklasser

Date, Math, Array.

Formulär och JavaScript

JavaScript är bland annat bra för att behandla inmatade data i ett HTML-formulär.

Låt användaren fylla i ett värde samt anropa sedan JavaScript-funktionen "compute()":

```
<form name="inputform">
<input name="field" type="text">
<input type="button" value="Enter!"onClick="compute()">
</form>
```

Ett JavaScript kan nå fälten i formuläret via "document"-objektet:

```
var userValue;
userValue = document.inputform.field.value;
```

DEMO - Gissa talet (game.html)

Ajax – Asynchronous JavaScript and XML

- Samling av relaterade tekniker för interaktiv webb.
- Förbättrar svarstider genom att HTML-sidor genereras lokalt mha JavaScript istället för på servern. Endast små datamängder överförs.
- Data/skript överförs asynkront i bakgrunden.
- Bygger på JavaScript och (ofta, men inte alltid) XML för överföring av data till/från servern.
- Sidans struktur kan manipuleras dynamiskt.

Exempel:

- Sidor med sökresultat. Endast själva sökresultatet behöver överföras/genereras och bytas ut på sidan.