

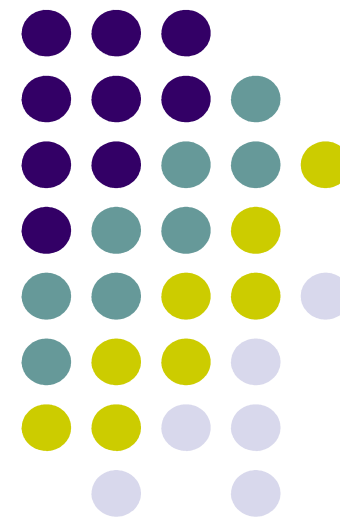
3.- WEB ORRIALDE DINAMIKOAK



Universidad
del País Vasco

Euskal Herriko
Unibertsitatea

Maidier Huarte Arrayago, Gorka Prieto Agujeta, Jasone Astorga Burgo, Nerea Toledo Gandarias
Komunikazioen Ingeniaritza Saila
Bilboko Ingeniaritza Goi Eskola Teknikoa
UPV/EHU



TELEK:O
UPV/EHU Bilbao

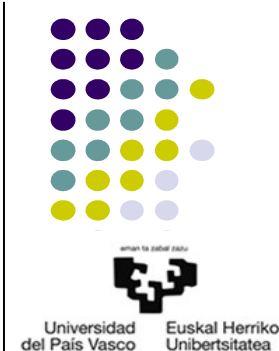
ZERBITZU TELEMATIKO AURRERATUAK: 3.- WEB ORRIALDE DINAMIKOAK



Copyright © 2015 Mainer Huarte Arrayago, Gorka Prieto Agujeta, Jasone Astorga Burgo, Nerea Toledo Gandarias

This is a derivative work from:

Mainer Huarte Arrayago, Gorka Prieto Agujeta
“Servicios Telemáticos Avanzados: 3.- Páginas Web Dinámicas”
OCW UPV/EHU 2014 (ISSN 2255-2316), 2014



ZERBITZU TELEMATIKO AURRERATUAK: 3.- WEB ORRIALDE DINAMIKOAK lana, Mainer Huartek, Gorka Prietok, Jasone Astorga Burgok eta Nerea Toledo Gandariasek egina, Creative Commons-en Attribution-Share Alike 3.0 Unported License baimenaren menpe dago. Baimen horren kopia bat ikusteko, <http://creativecommons.org/licenses/by-sa/3.0/> webgunea bisitatu edo gutun bat bidali ondoko helbidera: Creative Commons, 171 2nd Street, Suite 300, San Francisco, California, 94105, USA. Lan hau beste honen eratorria da: Mainer Huarte Arrayago, Gorka Prieto Agujeta, “Servicios Telemáticos Avanzados: 3.- Páginas Web Dinámicas”, OCW UPV/EHU 2014 (ISSN 2255-2316), 2014

ZERBITZU TELEMATIKO AURRERATUAK: 3.- WEB ORRIALDE DINAMIKOAK by Mainer Huarte, Gorka Prieto, Jasone Astorga Burgo and Nerea Toledo Gandarias is licensed under a Creative Commons Attribution-Share Alike 3.0 Unported License. To view a copy of this license, visit <http://creativecommons.org/licenses/by-sa/3.0/> or, send a letter to Creative Commons, 171 2nd Street, Suite 300, San Francisco, California, 94105, USA.

This is a derivative work from: Mainer Huarte Arrayago, Gorka Prieto Agujeta, “Servicios Telemáticos Avanzados: 3.- Páginas Web Dinámicas” OCW UPV/EHU 2014 (ISSN 2255-2316), 2014

WEB ORRIALDE DINAMIKOAK

INDIZEA

- 1.- Sarrera
- 2.- Client-side scripting
- 3.- Server-side scripting
- 4.- Diseinurako patroiak
- 5.- ERREFERENTZIAK

1.- SARRERA

- Web orrialdeak
 - Definizio sinplea
 - Estatikoak vs dinamikoak
 - Estatikoak
 - Dinamikoak
 - *Client-side*
 - *Server-side*

2.- Client-side scripting

2.1.- DHTML

- *Client-side scripting* moduko web orrialde dinamikoak
- Definizioa
- Osagaiak
 - HTML
 - CSS
 - Javascript
 - DOM

2.- Client-side scripting

2.1.- DHTML

- HTML: bezeroarekin dinamismoa eragiteko elementuak
 - CSS eta script kodeak: **<head></head>**
 - **<style></style>**
 - type="text/css"
 - **<link>**
 - **<script></script>**
 - **<noscript></noscript>**
 - HTML gertaerak
 - Zertarako?
 - Zenbait HTML elementuetarako atributuak:
gertaerarako_atributua="script_kodera_deia"
 - Leihoko gertaerak (<body>): **onload, onunload,...**
 - Formulategiko (eta barneko elementuetako) gertaerak : **onblur/onfocus, onchange, onselect, onsubmit,...**
 - Teklatuko gertaerak: **onkeydown/onkeyup, onkeypress**
 - Saguko gertaerak: **onclick,...**

2.- Client-side scripting

2.1.- DHTML

- CSS
 - DHTML estandarraren osagarrietako bat da
 - Ez du dinamismorik sartzen orrialdeetan
 - Orrialdeen itxura errazago kudeatzeko

2.- Client-side scripting

2.1.- DHTML

- JavaScript
 - Definizioa
 - JS kodeen kokapena
 - 1. aukera
 - 2. aukera
 - Exekuzioa
 - JS sententziak
 - Funtzioak

2.- Client-side scripting

2.1.- DHTML

- JavaScript

- Sintaxia:

- Iruzkinak: `//`, `/* */`
- Sententziak: `;`
- Funtzioak:

```
function izena(argumentuak)
{
    Sententziak
}
```
- Aldagaiak eta datu-motak
 - Deklarazioa: **var** *izena*;
 - Deklarazioa+hasieratzea: **var** *izena=balioa*;
- Eragileak:
- Baldintzazko sententziak:
- Sententzia errepikakorrak:

2.- Client-side scripting

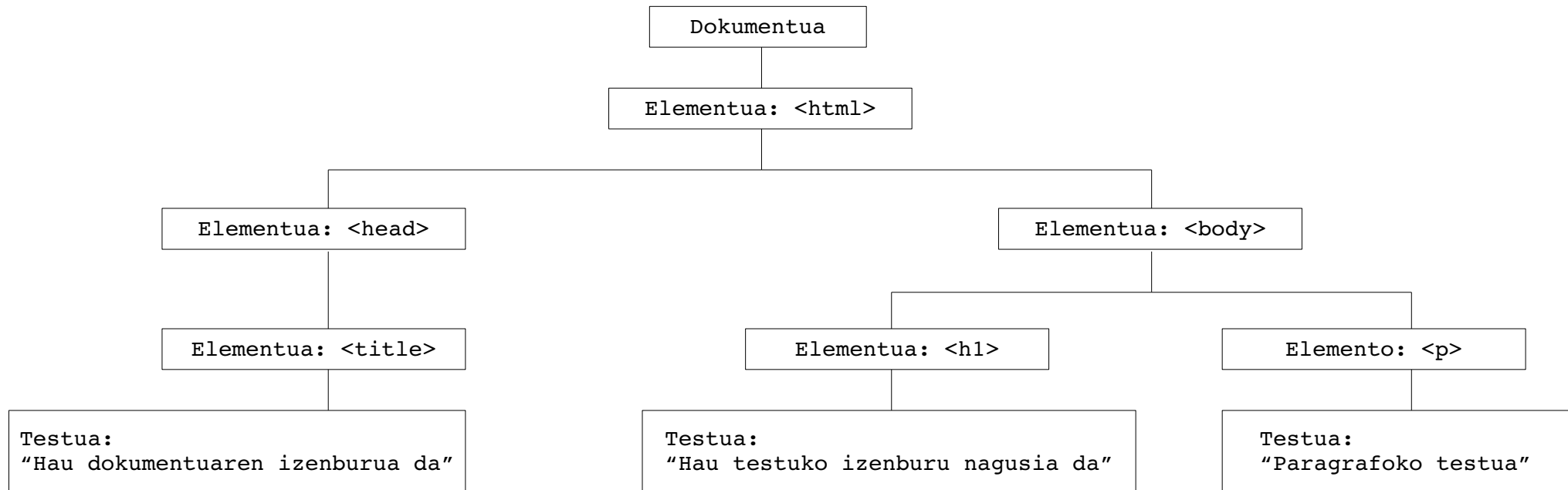
2.1.- DHTML

- JavaScript
 - Sintaxia:
 - Objektuak
 - Elementuak:
 - Propietateak: balioak
 - Metodoak: funtzioak
 - Jadanik definitutako objektuak
 - **String, Number, Array, Math**
 - **Window: alert(), confirm(), prompt(), open(), close()**
 - Eremuak erabiltzeko: `.` eragilea
 - Objektuak sortzea: **new** eragilea
 - Batzuk izan ezik:
 - Erroreak

2.- Client-side scripting

2.1.- DHTML

- DOM
 - Definizioa
 - Elementu eta edukien egitura hierarkikoa: nodoen zuhaitza



- Client-side scripting moduko lengoaiei, nodoak erabiltzea ahalbidetzeko

2.- Client-side scripting

2.1.- DHTML

- DOM
 - Jadanik definitutako DOM objektuak
 - **node**: edozein nodorentzat
 - Propietateak: **attributes, nodeName, childNodes**
 - Metodoak: **appendChild(), replaceChild(), removeChild()**
 - **document**: erro-nodoa, ez dauka amarik. **node** objektuko guztia +
 - Propietateak: **body, forms, images, title, ...**
 - Metodoak: **getElementById(), getElementsByTagName(), createElement(), renameNode(), open(), write(), close(),...**
 - **element**: edozein HTML elementurako. **node** objektuko guztia +
 - Propietateak: **id, innerHTML, tagName, ...**
 - Metodoak: **getAttribute(), setAttribute(),...**

2.- Client-side scripting

2.1.- DHTML

- DOM
 - Jadanik definitutako DOM objektuak
 - **body: node** eta **element** objektuetako guztia +
 - Propietateak: **background, bgColor, onload...**
 - Metodoak: -
 - **form: node** eta **element** objektuetako guztia +
 - Propietateak: **name, action, method, target, length...**
 - Metodoak: **reset(), submit()**
 - **checkbox: node** eta **element** objektuetako guztia +
 - Propietateak: **checked, disabled, name, value...**
 - Metodoak: -
 - **image, link, table: node** eta **element** objektuetako guztia +
 - **style:** CSS estiloekin lan egiteko

2.- Client-side scripting

2.1.- DHTML

- JAVASCRIPT+DOM ADIBIDEAK

```
<!DOCTYPE html>
<html>
<head>
<script>
function agurraBistaratu()
{
  var e1=prompt("Nor zara?");
  document.getElementById("demo").innerHTML="Kaixo "+e1+"! hau da une honetako data eta ordua: "+Date();
}
</script>
</head>
<body>
<h1>Nire lehenengo JavaScript kodea duen orrialdea</h1>
<p id="demo">Erantzun galdera eta hemen agurtuko zaitut...</p>
<button type="button" onclick="agurraBistaratu()">Ikusi agurra</button>
</body>
</html>
```

```
<!DOCTYPE html>
<html>
<head>
<script>
function kontuzWrite()
{
  document.write("SUNTSIKETA!!!!");
}
</script>
</head>
<body>
<h1>Nire bigarren JavaScript kodea duen orrialdea</h1>
<p id="demo">Hau ere testu paragrafo bat da.</p>
<button type="button" onclick="kontuzWrite()">Ikusi zelan dabilen document.write</button>
</body>
</html>
```

2.- Client-side scripting

2.1.- DHTML

- JAVASCRIPT+DOM ADIBIDEAK

```
<!DOCTYPE html>
<html>
<head>
<script>
function aldatuNodoak()
{
  var ps=document.getElementsByTagName("p");
  ps[0].innerHTML+=" ";
  var subp0=document.createElement("strong");
  subp0.innerHTML="Hau paragrafoaren amaierako testu garrantzitsua da";
  ps[0].appendChild(subp0);

  var ns=document.body.childNodes;
  document.body.removeChild(ns[5]);
  document.body.removeChild(ns[4]);
}

function aldatuEstiloa()
{
  var p1=document.getElementById("p1"); // Elementua atzitu
  p1.style.backgroundColor="red"; // Estiloa aldatu
}
</script>

</head>
<body>
<h1>HAU DOKUMENTUKO LEHENENGO IZENBURUA DA</h1>
<p id="p1">Hau dokumentuko lehenengo paragrafoa da</p>
<h2>HAU DOKUMENTUKO LEHENENGO IZENBURUKO LEHENENGO AZPI-IZENBURUA DA</h2>
<p>Hau dokumentuko bigarren paragrafoa da</p>
<p>Hau dokumentuko hirugarren paragrafoa da</p>

<button type="button" onclick="aldatuNodoak()">Aldatu paragrafoak</button>
<button type="button" onclick="aldatuEstiloa()">Aldatu atzealdea</button>

</body>
</html>
```

2.- Client-side scripting

2.1.- DHTML

- JAVASCRIPT+DOM ADIBIDEAK

```
<!DOCTYPE html>
<html>
<head>
<script>
function egiaztatuFormularioa()
{
  var e=true;
  var o=document.getElementById("i1").value;
  var m=document.getElementById("i2").value;

  if(o==null||o=="||o<"0"||o>"23"||m==null||m=="||m<"0"||m>"59")
  {
    alert("Ordua ez da baliozkoa");
    document.getElementById("i1").value="";
    document.getElementById("i2").value="";
    e=false;
  }
  else
    alert("Ordua egokia da");
  return e;
}
</script>
</head>
<body>
<h1>Nire laugarren JavaScript: formularioak egiaztatzeko</h1>
<form action="" onsubmit="return egiaztatuFormularioa()" method="post">
Sar ezazu ordua <br>
<input id="i1" type="text" name="ordua" maxlength="2">:<input id="i2" type="text" name="min" maxlength="2"><br>
<input type="submit" value="Submit">
</form>
</body>
</html>
```


3.- Server-side scripting

- Exekuzio ingurunea
 - Web zerbitzaria+CGI: Apache
 - Web zerbitzaria+Aplikazio Zerbitzarirako *conector*
 - Apache zerbitzaria Tomcat-ekin
 - Aplikazioen Zerbitzaria bakarrik
 - Tomcat
 - Glassfish
 - JBoss
 - Jetty
- Bezeroak: nabigatzaileak

3.- Server-side scripting

3.1.- CGI

- Definizioa
- Ingurune-aldagaiak eta S/I estandarra
 - Edozein lengoaia
 - Ohikoena, PERL
- HTTP-request
 - GET
 - Datuak URL eremuan
 - QUERY_STRING
 - POST
 - Datuak gorputzean
 - Nola lortu?

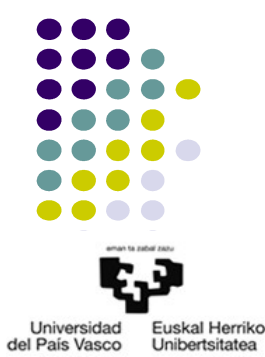
3.- Server-side scripting

3.1.- CGI

- HTTP-response
 - Nola idatzi?
 - Edukiera
- Beste ingurune-aldagaiak
 - REQUEST_METHOD, HTTP_USER_AGENT, REMOTE_ADDR, REMOTE_USER, HTTPS,...

3.- Server-side scripting

3.2.- Servletak



- Aurretik berdinerako bazegoen beste CGI teknologiaren ordezkoa
- Servleten Java hierarkia

```
java.lang.Object
```

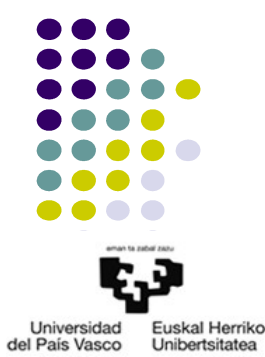
```
↳ javax.servlet.GenericServlet (abstract): javax.servlet.Servlet, ...
```

```
↳ javax.servlet.http.HttpServlet (abstract)
```

– javax.servlet.Servlet

3.- Server-side scripting

3.2.- Servletak



- HTTPrako servlet baten oinarrizko egitura

```
import java.io.*;
import javax.servlet.ServletException;
import javax.servlet.annotation.WebServlet; //JavaEE motako proiektuetan
import javax.servlet.http.*;

@WebServlet("/LehenengoServlet") //JavaEE motako proiektuetan
public class LehenengoServlet extends HttpServlet
{
    //Atributuak

    private static void doGet(HttpServletRequest request, HttpServletResponse response)
        throws ServletException, IOException
    {
        //GET moduko HTTP-request eskaerari erantzuteko exekutatu beharreko kodea
    }

    private static void doPost(HttpServletRequest request, HttpServletResponse response)
        throws ServletException, IOException
    {
        //POST moduko HTTP-request eskaerari erantzuteko exekutatu beharreko kodea
    }

    //Beste metodoak, berriak edo oinordetuak (doHead,...)
}
```

HTTPrako servlet baten oinarrizko egitura

- Gaur egun, edozein programaziorako IDE berrik txantiloiak eskaintzen ditu

3.- Server-side scripting

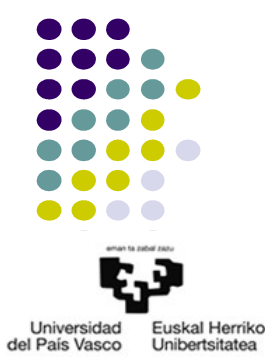
3.2.- Servletak

- HTTPrako servlet baten oinarritzko egitura
 - Zerbitzu-metodoen parametroak
 - **request**: zein informazio?
 - **.getMethod()**
 - **.getHeader("HTTP_goiburua")**
 - Erabiltzailearen datuak (adibidez, formularioan sartuak)
 - **.getParameter("izena")**
 - **.getParameterValues("izena")**
 - ...

```
Metodoa path_edo_URL HTTP_bertsioa
Goiburu1:balio1
Goiburu2:balio2
...
Lerro_hutsa
Mezuaren_gorputza
```

3.- Server-side scripting

3.2.- Servletak



- HTTPrako servlet baten oinarritzko egitura

- Zerbitzu-metodoen parametroak

- **Response**: zein informazio?

- **DERRIGORREZKOA?**

- Html kodea

- **PrintWriter** objektuan idatzi

- **response.getWriter()**

- Fitxategi bitarra

- **ServletOutputStream** objektuan idatzi

- **response.getOutputStream()**

- **AUKERAZKOA**

- HTTP-response mezuaren 1. lerroko eremuak

- HTTP-response mezuaren goiburuak: **.setContentType("mime")**,
.addCookie(Cookie),...

```
HTTP_bertsioa Kodea Azalpena
Goiburu1:balio1
Goiburu2:balio2

...
lerro_hutsa
Mezuaren_gorputza:_adib_HTML_kodea
```

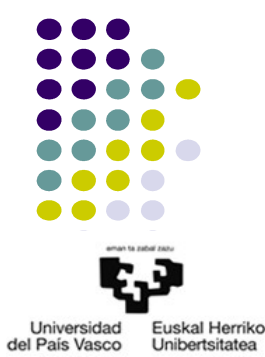
3.- Server-side scripting

3.2.- Servletak

- HTTPrako servlet baten oinarritzko egitura
 - Zerbitzu-metodo baten eskema orokorra
 - HTTP-request mezutik informazioa atera:zein?
 - Kanpoko baliabideak atzitu
 - Datuekin lan egin
 - HTTP-response mezurako idatzi: nola?
 - HTTP-response mezuaren goiburuak
 - HTTP-response mezuaren gorputza
 - HTML kodea edo bitarra
 - Erabilitako *stream* objektuak itxi

3.- Server-side scripting

3.2.- Servletak



- SERVLET ADIBIDEAK: HTTP-request mezuan jasotako informazioaz arduratu gabe, HTML kodea sortzen duen Servlet sinplea

```
//import lerroak

@WebServlet("/KaixoWWW")
public class KaixoWWW extends HttpServlet //HTTP-request mezuko informaziorik erabiltzen ez duen Servleta
{
    protected void doGet(HttpServletRequest request, HttpServletResponse response)
        throws ServletException, IOException
    {
        PrintWriter out = response.getWriter(); //HTTP-response mezuan HTML kodea idazteko
        out.println("<!DOCTYPE html>");
        out.println("<html>");
        out.println("<head>");
        out.println("<title>Kaixo WWW</title>");
        out.println("</head>");
        out.println("<body>");
        out.println("<h1>KAIIXO MUNDUA!</h1>");
        out.println("</body>");
        out.println("</html>");
        out.close();

        return;
    }
}

//content-type eta content-length goiburuak web zerbitzariak sartuko ditu

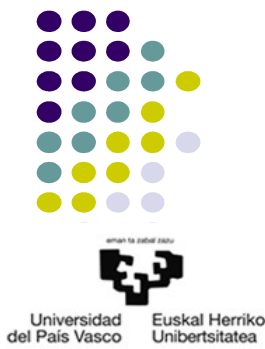
protected void doPost(HttpServletRequest request, HttpServletResponse response)
    throws ServletException, IOException
{
    doGet(request,response);

    return;
}
}
```

Servlet sinplea

3.- Server-side scripting

3.2.- Servletak



- SERVLET ADIBIDEAK: Servlet sinplea, klase laguntzaile (POJO) batekin hobetua

```
//import lerroak

@WebServlet("/KaixoWWW_MOD")
public class KaixoWWW_MOD extends HttpServlet //HTTP-request mezuko informaziorik erabiltzen ez duen Servleta
{
    protected void doGet(HttpServletRequest request, HttpServletResponse response)
        throws ServletException, IOException
    {
        response.setContentType("text/html");//html koderako ez da behar, egia esan

        PrintWriter out = response.getWriter();
        out.println(ServletErab.DOCTYPE);
        out.println(ServletErab.headTitle("KaixoWWW_MOD"));
        out.println("<body>");
        out.println("<h1>KAIXO MUNDU MODULARRA!!!!</h1>");
        out.println(ServletErab.AMAIERA);
        out.close();

        return;
    }

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

        return;
    }
}
```

Modularitatearekin hobetutako Servlet sinplea

3.- Server-side scripting

3.2.- Servletak

- SERVLET ADIBIDEAK: Servlet sinplea, klase laguntzaile (POJO) batekin hobetua

```
public class ServletErab
{
    public static final String DOCTYPE = "<!DOCTYPE html>";

    public static String headTitle(String tit)
    {
        String s="<html>";
        s+="\n<head>";
        s+="\n<title>" + tit + "</title>";
        s+="\n</head>";

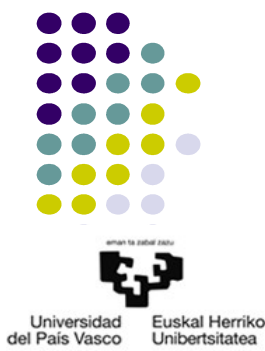
        return s;
    }

    public static final String AMAIERA = "</body>\n</html>";
}
}
```

Ataza arruntetarako POJO moduko klasea

3.- Server-side scripting

3.2.- Servletak



- SERVLET ADIBIDEAK: Web Aplikazio sinplea, Servlet eta klase laguntzaileekin (POJO)

```
<!DOCTYPE html>
<html>
<head>
<title>Aplikazio aritmetikoaren hasierako orrialdea</title>
<link rel="stylesheet" type="text/css" href="aplikAritEstiloak.css">
</head>
<body>
<h1>Sar itzazu beharrezko datuak</h1>
<form method="post" action="FormularioaJasoEtaEmitzaSortzekoServlet">
  <p>
    <input type="number" name="eragigai1" maxlength="2" size="12" value="1. eragigaia">
    <select name="eragiketa">
      <option value="+">+</option>
      <option value="-">-</option>
      <option value="*">*</option>
      <option value="/">/</option>
    </select>
    <input type="number" name="eragigai2" maxlength="2" size="11" value="2. eragigaia">
  </p>
  <p>
    <input type="submit" value="Bidali">
  </p>
</form>
</body>
</html>
```

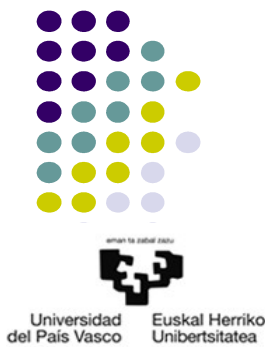
Erabiltzailearen datuak jasotzeko html orrialde estatikoa

```
h1{text-align:center;}
p{text-align:center;}
```

aplikAritEstiloak.css

3.- Server-side scripting

3.2.- Servletak



- SERVLET ADIBIDEAK: Web Aplikazio sinplea, Servlet eta klase laguntzaileekin (POJO)

```
//import arruntak
import logika.AplikaziokoLogika;

@WebServlet("/FormularioaJasoEtaEmaitzaSortzekoServlet")
public class FormularioaJasoEtaEmaitzaSortzekoServlet extends HttpServlet
{
    protected void doGet(HttpServletRequest request, HttpServletResponse response) throws ...
    {
        PrintWriter out = response.getWriter();
        out.println(ServletErab.DOCTYPE);
        out.println(ServletErab.headTitleStyle("Atzipen errorea","aplikAritEstiloak.css"));
        out.println("<body>");
        out.println("<h1>ERROREA: servlet hau POST metodoarekin bakarrik atzitu daiteke</h1>");
        out.println(ServletErab.AMAIERA);
        out.close();
    }

    protected void doPost(HttpServletRequest request, HttpServletResponse response) throws ...
    {
        String sEr1=request.getParameter("eragigai1");
        String sEr=request.getParameter("eragiketa");
        String sEr2=request.getParameter("eragigai2");

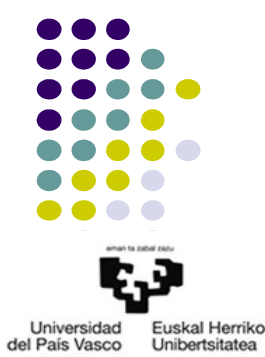
        AplikaziokoLogika aL=new AplikaziokoLogika(sEr1,sEr,sEr2);
        aL.kalkulatu();

        PrintWriter out = response.getWriter();
        out.println(ServletErab.DOCTYPE);
        out.println(ServletErab.headTitleStyle("Emaitza","aplikAritEstiloak.css"));
        out.println("<body>");
        out.println("<h1>ESKATUTAKO ERAGIKETAREN EMAITZA</h1>");
        out.println("<p>"+sEr1+sEr+sEr2+"="+aL.getEm()+"</p>");
        out.println(ServletErab.AMAIERA);
        out.close();
    }
}
```

Formularioan sartutako datuak jaso, prozesatzera bidali eta bistartzeko HTML kodea sortzen duen Servleta

3.- Server-side scripting

3.2.- Servletak



- SERVLET ADIBIDEAK: Web Aplikazio sinplea, Servlet eta klase laguntzaileekin (POJO)

```
public class ServletErab
{
    public static final String DOCTYPE = "<!DOCTYPE html>";

    public static String headTitleStyle(String tit,String styleSheetFileName)
    {
        String s="<html>";
        s+="\n<head>";
        s+="\n<title>"+tit+"</title>";
        s+="\n<link rel=\"stylesheet\" type=\"text/css\" href="+styleSheetFileName+">";
        s+="\n</head>";

        return s;
    }

    public static final String AMAIERA = "</body>\n</html>";
}
}
```

Ataza arruntetarako POJO moduko klasea

3.- Server-side scripting

3.2.- Servletak

- SERVLET ADIBIDEAK: Web Aplikazio sinplea, Servlet eta klase laguntzaileekin (POJO)

```
package logika;

public class AplikaziokoLogika
{
    private float er1;
    private char er;
    private float er2;
    private float em;

    public AplikaziokoLogika(String sEr1,String sEr,String sEr2)
    {
        er1=Float.valueOf(sEr1);
        er=sEr.charAt(0);
        er2=Float.valueOf(sEr2);
    }

    public void kalkulatu()
    {
        switch(er)
        {
            case '+':
                em=er1+er2;
                break;

//Beste kasu posible guztiak
        }

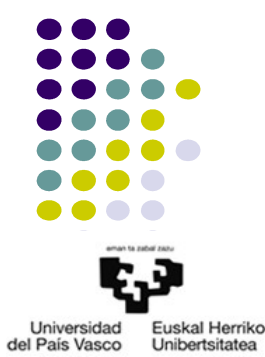
        return;
    }

    public float getEm()
    {
        return em;
    }
}
```

Aplikazioko logikarako POJO moduko klasea

3.- Server-side scripting

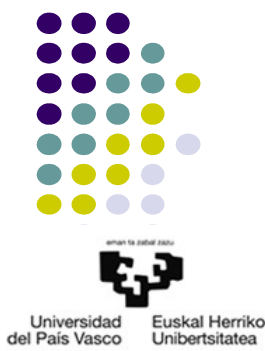
3.2.- Servletak



- Servletekin sesioak kudeatzen
 - HTTPk ez du egoera mantentzen
 - Zerbitzariak memorian mantendu dezake egoeraren informazioa eta dagokion HTTP-request mezuekin erlazionatu, ondokoen bitartez
 - Cookiak
 - URL
 - Egoeraren informazioa servletetatik atzitzeko
 - **HttpSession** objektua: **request.getSession()**
 - Egoeraren identifikatzailea: **.getId()**
 - Informazioa sartu/atera: **.setAttribute("izena",objektua), .getAttribute("izena")**
 - Sesioa amaitzea: **.invalidate()**
 - Nabigatzea: servletetan prestatutako loturen bitartez
 - **response.encodeURL("loturaren_url")**

3.- Server-side scripting

3.2.- Servletak



- SERVLET ADIBIDEAK: Sesioen kudeaketa servletekin

```
//import arruntak
```

Servlet1 klasea: sesioa sortzen du

```
@WebServlet("/Servlet1")
public class Servlet1 extends HttpServlet {
    protected void doGet(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {
        HttpSession s=request.getSession();

        if(s!=null)
        {
            s.setAttribute("kontagailua", "0");
            response.setContentType("text/html");
            PrintWriter out = response.getWriter();

            out.println(ServletErab.DOCTYPE);
            out.println(ServletErab.headTitleStyle("Servlet1","aplikAritEstiloak.css"));
            out.println("<body>");
            out.println("<h1>SESIO BERRI BAT SORTU DA: "+s.getId()+"</h1>");

            String urlNoCookie=response.encodeURL("Servlet2");
            //HTTP-request mezua cookie-rik gabe iritsi bada, cookie berria bidaltzeaz gain (set-cookie), Id gehitzen du badaezpada

            out.println("<form action=\""+urlNoCookie+"\" method=\"get\">");
            out.println("<p><input type=\"submit\" value=\"Hasi exekuzioa\"></p>");
            out.println("</form>");
            out.println("<a href=\""+urlNoCookie+"\" target=\"_blank\">Hurrengoa</a>");

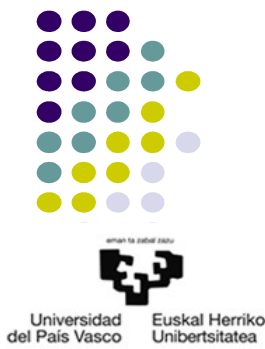
            out.println(ServletErab.AMAIERA);
            out.close();

        }
        else
            System.out.println("EZIN IZAN DA SESIORIK HASI");
    }

    protected void doPost(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {
        // TODO Auto-generated method stub
    }
}
```

3.- Server-side scripting

3.2.- Servletak



- SERVLET ADIBIDEAK: Sesioen kudeaketa servletekin

```
//import arruntak
```

Servlet2 klasea: sesioa erabiltzen du

```
@WebServlet("/Servlet2")
public class Servlet2 extends HttpServlet {
    protected void doGet(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {
        HttpSession s=request.getSession();

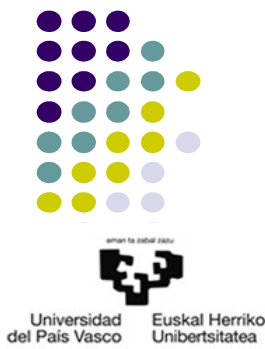
        if(s!=null&&!s.isNew()) {
            try {
                int c=Integer.parseInt((String)s.getAttribute("kontagailua"));
                System.out.println("S2 kontagailua: "+c);

                String[] inc=request.getParameterValues("gehikuntzak");
                for(int i=0;i<inc.length;i++)
                    c+=Integer.parseInt(inc[i]);
                s.setAttribute("kontagailua", ""+c);
            }
            catch(Exception ex) {
                System.out.println("S2: "+ex.getStackTrace());
            }
        }

        //HEMENGO KODE GUZTIA HURRENGO ORRIALDEAN DAGO
    }
}
```

3.- Server-side scripting

3.2.- Servletak



- SERVLET ADIBIDEAK: Sesioen kudeaketa servletekin

```
//HEMENGO KODE GUZTIA AURREKO ORRIALDEAN DAGO
```

Servlet2 klasea: sesioa erabiltzen du

```
response.setContentType("text/html");
PrintWriter out = response.getWriter();

out.println(ServletErab.DOCTYPE);
out.println(ServletErab.headTitleStyle("Servlet2", "aplikAritEstiloak.css"));
out.println("<body>");
out.println("<h1>Sesio berdinean gaude: "+s.getId()+"</h1>");
out.println("<h1>Kontagailuaren balioa orain: "+s.getAttribute("kontagailua")+"</h1>");

String urlNoCookie=response.encodeURL("Servlet2");
out.println("<form action='"+urlNoCookie+"' method='get'>");
out.println("<p>Aukeratu zenbat gehitu nahi duzun kontagailua:</p>");

out.println("<p>+1<input type='checkbox' name='gehikuntzak' value='1'></p>");
out.println("<p>+2<input type='checkbox' name='gehikuntzak' value='2'></p>");
out.println("<p>+3<input type='checkbox' name='gehikuntzak' value='3'></p>");

out.println("<p><input type='submit' value='Gehitu kontagailuan'></p>");
out.println("</form>");
urlNoCookie=response.encodeURL("Servlet3");
out.println("<a href='"+urlNoCookie+"' target='_blank'>Hurrengoa</a>");

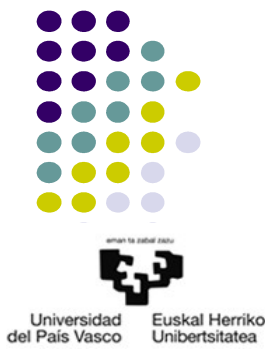
out.println(ServletErab.AMAIERA);
out.close();
```

```
}
else
    System.out.println("EZ DAGO SESIORIK EDO SESIOA EZ DA GORDE");
}
```

```
protected void doPost(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {
    // TODO Auto-generated method stub
}
}
```

3.- Server-side scripting

3.2.- Servletak



- SERVLET ADIBIDEAK: Sesioen kudeaketa servletekin

```
//import arruntak
```

Servlet3 klasea: sesioa amaitzen du

```
@WebServlet("/Servlet3")
public class Servlet3 extends HttpServlet {

    protected void doGet(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException
    {
        HttpSession s=request.getSession();

        if(s!=null&&!s.isNew())
        {
            response.setContentType("text/html");
            PrintWriter out = response.getWriter();

            out.println(ServletErab.DOCTYPE);
            out.println(ServletErab.headTitleStyle("Servlet3","aplikAritEstiloak.css"));
            out.println("<body>");
            out.println("<h1>Sesioaren amaiera: "+s.getId()+"</h1>");
            out.println("<h1>Kontagailuaren azken balioa: "+s.getAttribute("kontagailua")+ "</h1>");

            s.invalidate();

            out.println(ServletErab.AMAIERA);
            out.close();
        }
        else
            System.out.println("EZ DAGO SESIORIK EDO SESIOA EZ DA GORDE");
    }

    protected void doPost(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {
        // TODO Auto-generated method stub
    }
}
```

```
public class ServletErab {
//AURREKO ADIBIDEKO KODE BERDINA
}
```

Ataza arruntetarako POJO moduko klasea

4.- Diseinurako patroiak

4.1.- Sarrera

- Definizioa
- Garrantzia
- Patroi motak
 - Sorkuntza: factory, builder, singleton,...
 - Egitura: composite, adapter,...
 - Portaera: iterator, observer,...
 - Concurrence: lock, monitor, scheduler,...
 - Arkitektura: module, layer, tier, MVC,...
- Kontzeptu orokorrak
 - Lengoaia bakoitzak bere implementazio zehatza

4.- Diseinurako patroiak

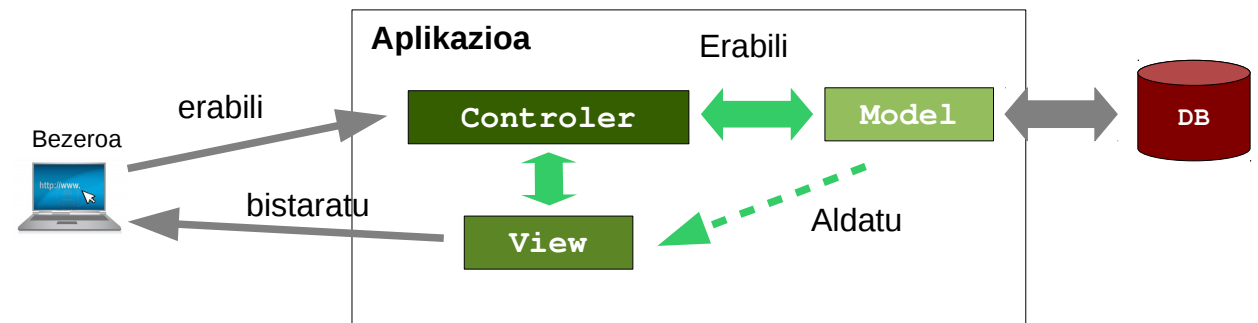
4.2.- POJO eta JavaBeans

- POJO (Plain Old Java Object)
 - Zer dira?
 - Baldintzarik?
- JavaBean
 - Zer dira?
 - Derrigorrezko baldintzak?
 - 1
 - 2
 - Beste baldintzak?
 - Atributuen atzipenerako getter eta setter publikoak
 - getter: **getAtributulzena()**
 - setter: **setAtributulzena(atributurakoBalioa)**
 - **Serializable** interfazea inplementatzea

4.- Diseinurako patroiak

4.3.- MVC patroia

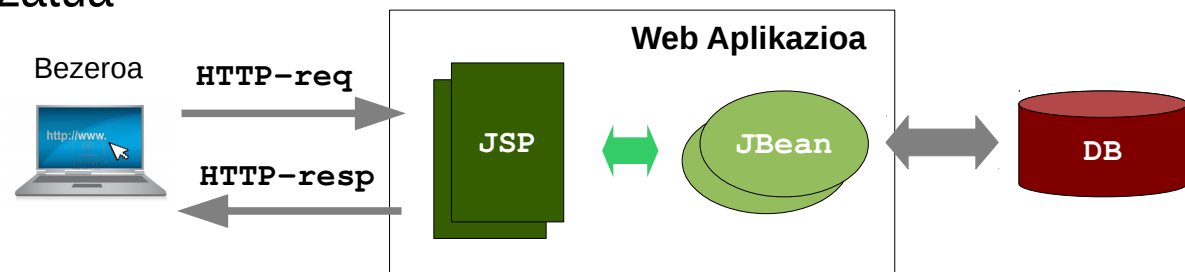
- Erabiltzaile interfazeak sortzeko oso erabilia
- Osagaiak
 - Model: M
 - Zer hartzen du?
 - Beste bi osagaiekin erlazionatzen da
 - View: V
 - Zer egiten du?
 - Erabiltzaileari datuak sartu eta C erabiltzeko gaitasuna ere ematen dio
 - Controller: C
 - Erabiltzailetik?
 - M?
 - V?



4.- Diseinurako patroiak

4.4.- JavaEEn erabilitako patroien adibideak

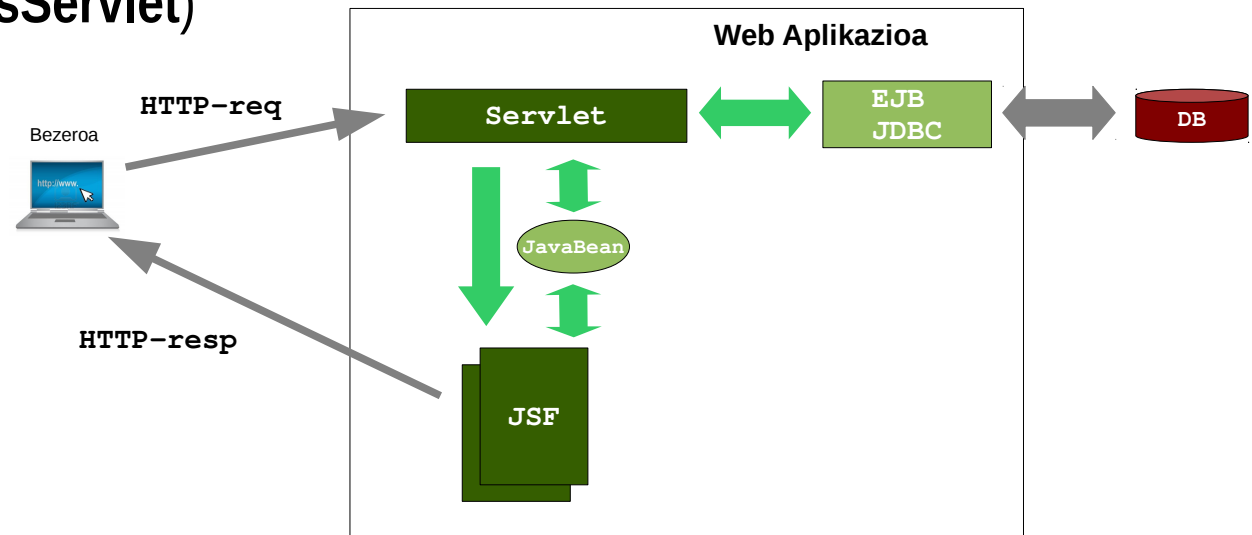
- Model 1
 - Zertarako?
 - JavaBeans
 - HTTP-response mezuaren gorputza sortzeko informazioa
 - Aplikazio konplexuetan arazoak
 - JSPak: HTML eta *scriptlet* kode zatien nahasketa
 - Nabigatze deszentralizatua



4.- Diseinurako patroiak

4.4.- JavaEEn erabilitako patroien adibideak

- Model 2
 - Web Aplikazio konplexuak: kasu orokorra
 - MVC patroia adaptazioa
 - Model: JavaBeans, EJBs, POJOs
 - View: JSFs (hasiera batean JSPs)
 - Controler: Servlet (**FacesServlet**)



5.- ERREFERENTZIAK

- W3CSCHOOLS: <http://www.w3schools.com/>
 - HTML: <http://www.w3schools.com/html/default.asp>
 - CSS: <http://www.w3schools.com/css/default.asp>
 - JavaScript: <http://www.w3schools.com/js/default.asp>
 - DOM: <http://www.w3schools.com/jsref/default.asp>
- Java Servlet Technology: <http://docs.oracle.com/javaee/7/tutorial/servlets.htm#BNAFD>
- Design Patterns. Elements of Reusable Object-Oriented Software - Erich Gamma, Richard Helm, Ralph Johnson, John Vlissides - Addison Wesley (GoF- Gang of Four)