

Crashkurs http - CGI/Servlets(JSF) - Viewer

Jan Krüger

`jkrueger(at)cebitec.uni-bielefeld.de`

- TCP Referenzmodell : ApplicationLayer
- zustandloses Protokoll
- textbasiert
- für Hypertext entwickelt ist es nicht darauf beschränkt
- Nachrichten :
 - Request : Client \Rightarrow Server
 - Response : Server \Rightarrow Client
- Aufbau : Header + Body, getrennt durch eine Leerzeile

http-request

```
GET / HTTP/1.1
HOST bibiserv.techfak.uni-bielefeld.de
User-Agent: Mozilla/5.0 (X11;SunOS i86x; rv:16.0) ...
Accept: text/html,application/xhtml+xml,application/xml ...
Accept-Language: en,de;q=0.7;en-us;q=0.3
Accept-Encoding: gzip,deflate
Connection: keep-alive
```

- GET** schickt eine Ressourcenanforderung (URL) an den Server, (kleine) Daten koennen als Argumente an die URL codiert werden
- POST** schickt (in der Größe unbegrenzte) Daten an den Server, Daten sind im Body als Key/Value abgelegt.
- HEAD** liefert den nur den Header (ohne Body)

...

http-response

```
200 OK
Connection: close
Date: Tue, 13 Nov 2012 08:42:19 GMT
Accept-Ranges: bytes
ETag: "aecf9-3e70-507420d3"
Server: Apache/1.3.29 (Unix) mod_ssl/2.8.16 OpenSSL/0.9.6g
Content-Length: 15984
Content-Type: text/html
Last-Modified: Tue, 09 Oct 2012 13:04:19 GMT
Client-Date: Tue, 13 Nov 2012 08:42:19 GMT
Client-Peer: 129.70.161.13:80
Client-Response-Num: 1
```

```
<?xml version="1.0"?>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
    "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html>
```

http - Fehlercodes

1xx information

2xx success (200 - ok)

3xx redirect (302 - temporarily moved)

4xx client error (403 - access denied, 404 - not found)

5xx server error (500 - internal server error)

Hilfsmittel

Firefox + Firebug

Chrome (mit integrierten Developer Tools)

MiniHTTP Server

- Binaries für Solaris, OSX, Linux, Windows(+CygWin)
- http:
`//www.techfak.uni-bielefeld.de/~jkrueger/mini_httpd`
- Defaults :
SERVERDIR \$HOME/WWW
data dir \$SERVERDIR/data
cgi suffix *.cgi
logfile \$SERVERDIR/log.txt
- CGI's müssen ausführbar sein !

Hello World!

Shell

```
#!/bin/sh
#Header
echo -e "Content-Type: text/plain\n"
#Body
echo "Hello World!"
```

Perl

```
#!/usr/bin/env perl
#Header
print "Content-Type :: text/plain\n\n";
#Body
print "Hello World!\n";
```

- informeller Standard
- Unterschied GET (Env) vs. POST (STDIN)
- händische Auswertung aufwändig (aber möglich)
- Perl Modul :: CGI;
- mehr unter
`http://www.techfak.uni-bielefeld.de/ags/pi/lehre/Perl12/CGI-screen.pdf`
- heute eher als Modul realisiert. z.B. modperl, modphp, modpython ...

CGI - in use

form.html

```
...  
<form action="fasta.cgi" method="GET">  
  <textarea rows="5" cols="60" name="fasta"></textarea>  
  <input type="submit" value="submit"/>  
</form>  
...
```

form.cgi

```
...  
my $cgi = new CGI;  
my $fasta = $cgi->param("fasta");  
...  
print $cgi->header("text/plain");  
print $result;
```

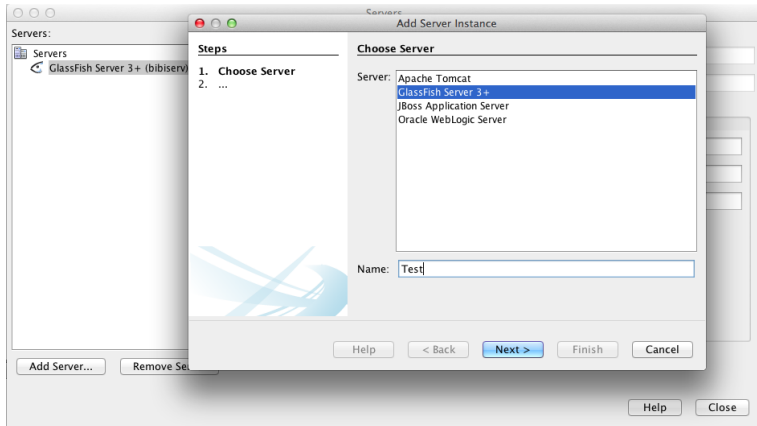
the Java way ...

- J2EE
- Servlets
- JavaServerFaces 2.x

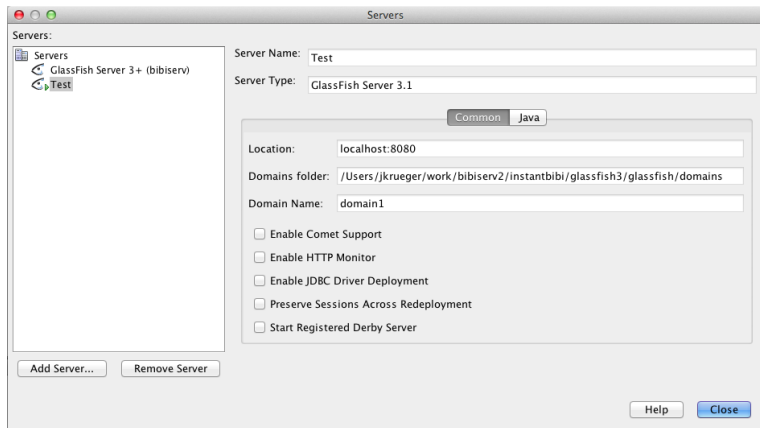
J2EE Applikation Server - NetBeans

- Glassfish über NetBeans IDE erstellen
- CeBiTec : rcinfo Paket "netbeans-7.2"

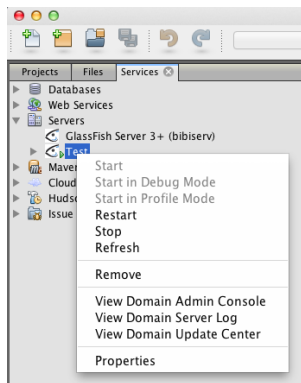
J2EE Applikation Server - NetBeans



J2EE Applikation Server - NetBeans



J2EE Applikation Server - NetBeans



WebProjekt - NetBeans

- WebProjekt mittels NetBeans IDE erstellen
- unterstützt alle gebräuchlichen J2EE Frameworks :
 - Servlets
 - JSP
 - Spring Web
 - JSF 2.x (+ PrimeFaces, IceFaces, ...)
 - ...

WebProjekt - NetBeans (Demo)

Steps

1. Choose Project
2. ...

Choose Project

Categories:

- Java
- JavaFX
- Java Web
- Java EE
- Maven
- NetBeans Modules
- Samples

Projects:

- Web Application
- Web Application with Existing Sources
- Web Free-Form Application

Description:

Creates an empty Web application in a standard IDE project. A standard project uses an IDE-generated build script to build, run, and debug your project.

Help < Back Next > Finish Cancel

WebProjekt - NetBeans (Demo)

New Web Application

Steps

1. Choose Project
2. **Name and Location**
3. Server and Settings
4. Frameworks

Name and Location

Project Name:

Project Location:

Project Folder:

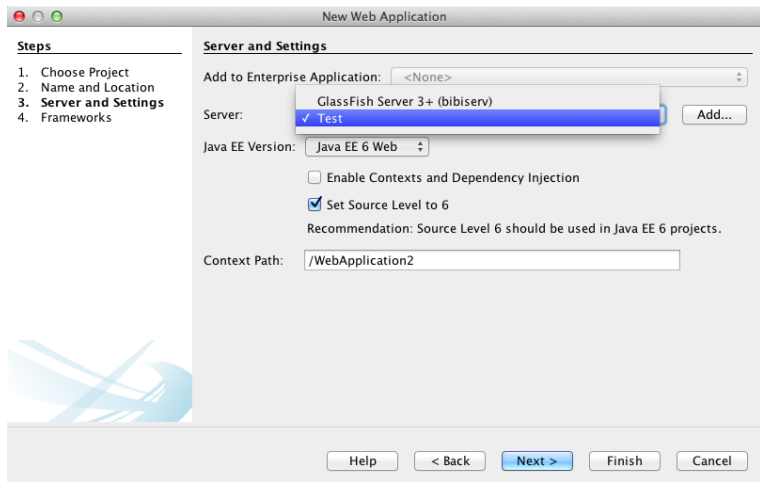
Use Dedicated Folder for Storing Libraries

Libraries Folder:

Different users and projects can share the same compilation libraries (see Help for details).



WebProjekt - NetBeans (Demo)



WebProjekt - NetBeans (Demo)

New Web Application

Steps

1. Choose Project
2. Name and Location
3. Server and Settings
4. **Frameworks**

Frameworks

Select the frameworks you want to use in your web application.

- Spring Web MVC
- JavaServer Faces
- Struts 1.3.10
- Hibernate 3.2.5

JavaServer Faces Configuration

Libraries Configuration Components

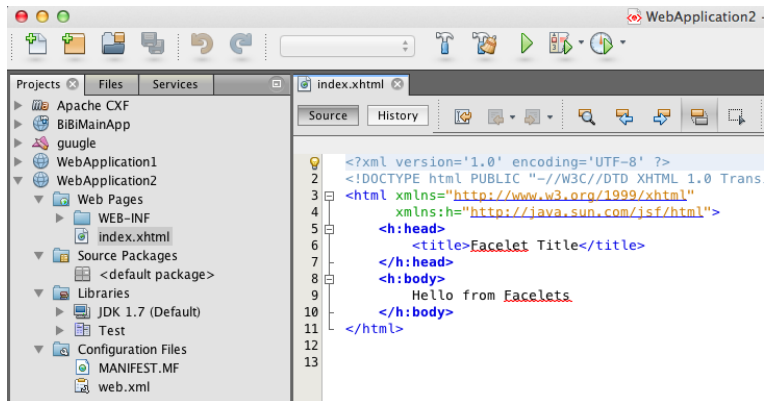
Server Library: JSF 2.1

Registered Libraries: JSF 2.1

Create New Library

JSF Folder or JAR:

WebProjekt - NetBeans (Demo)



Servlets - NetBeans

- vergleichbar mit CGI-Skripten
- direkten Zugriff auf die http request/response Nachrichten
- ...

Servlets - NetBeans (Demo)

New Servlet

Steps

1. Choose File Type
2. **Name and Location**
3. Configure Servlet Deployment

Name and Location

Class Name:

Project:

Location:

Package:

Created File:

Servlets - NetBeans (Demo)

Steps

1. Choose File Type
2. Name and Location
3. **Configure Servlet Deployment**

Configure Servlet Deployment

Register the Servlet with the application by giving the Servlet an internal name (Servlet Name). Then specify patterns that identify the URLs that invoke the Servlet. Separate multiple patterns with commas.

Add information to deployment descriptor (web.xml)

Class Name:

Servlet Name:

URL Pattern(s):

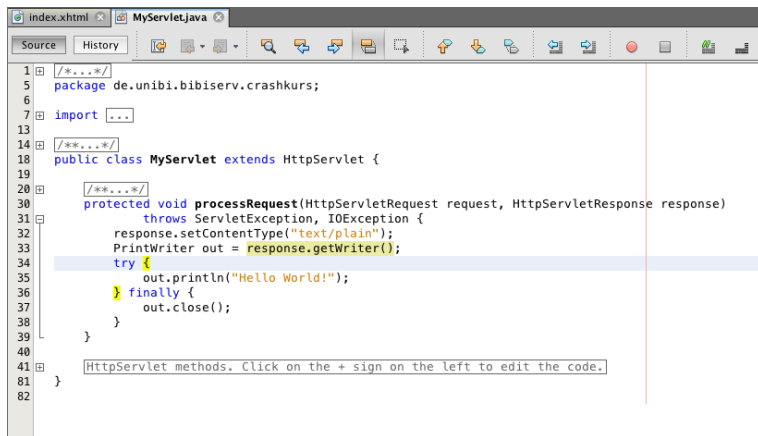
Initialization Parameters:

Name	Value
------	-------

New
Edit...
Delete

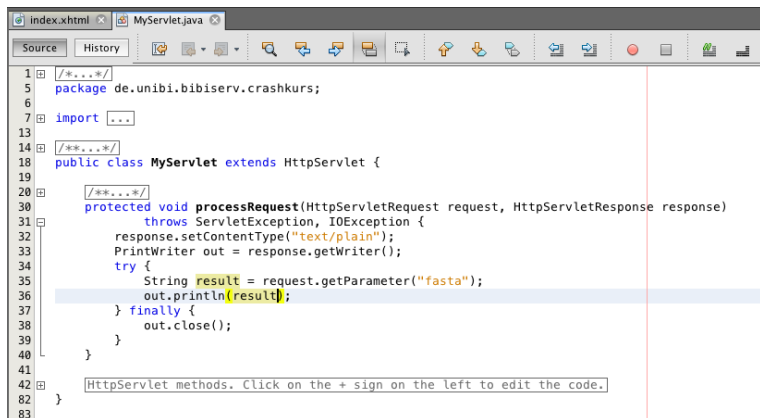
Help < Back Next > **Finish** Cancel

Servlets - NetBeans (Demo)



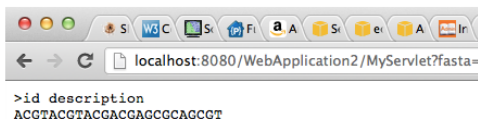
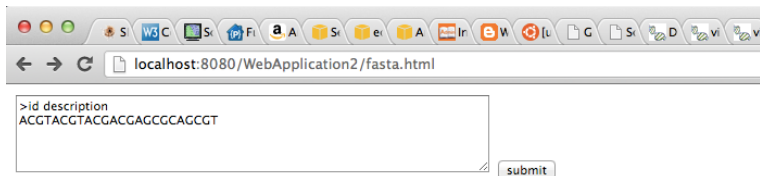
```
1  [ ] /****/  
5  package de.unibi.bibiserv.crashkurs;  
6  
7  [ ] import ...  
13  
14 [ ] /****/  
18 public class MyServlet extends HttpServlet {  
19  
20 [ ] /****/  
30 protected void processRequest(HttpServletRequest request, HttpServletResponse response)  
31     throws ServletException, IOException {  
32     response.setContentType("text/plain");  
33     PrintWriter out = response.getWriter();  
34     try {  
35         out.println("Hello World!");  
36     } finally {  
37         out.close();  
38     }  
39 }  
40  
41 [ ] HttpServlet methods. Click on the + sign on the left to edit the code.  
81 }  
82
```


Servlets - NetBeans (Demo)



```
1  [ ] /*...*/
5  package de.unibi.bibiserv.crashkurs;
6
7  import [ ] ...
13
14 [ ] /*...*/
18 public class MyServlet extends HttpServlet {
19
20 [ ] /*...*/
21     protected void processRequest(HttpServletRequest request, HttpServletResponse response)
22         throws ServletException, IOException {
23         response.setContentType("text/plain");
24         PrintWriter out = response.getWriter();
25         try {
26             String result = request.getParameter("fasta");
27             out.println(result);
28         } finally {
29             out.close();
30         }
31     }
32 }
33
34 [ ] HttpServlet methods. Click on the + sign on the left to edit the code.
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
```

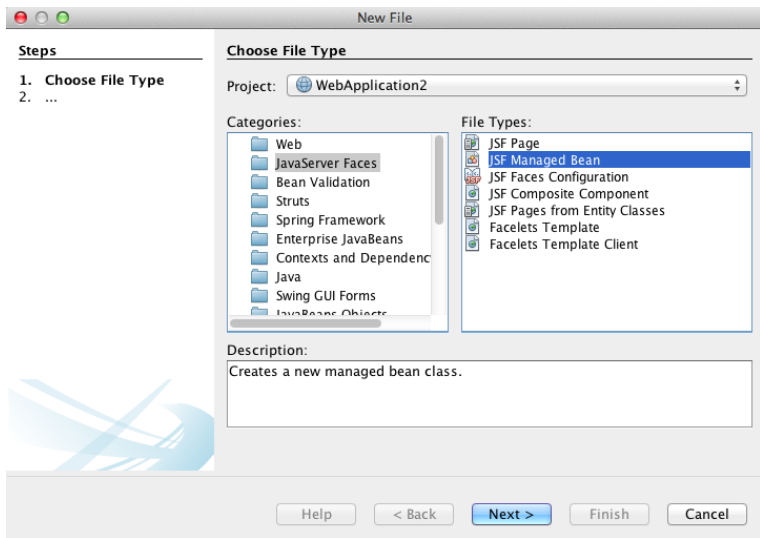
Servlets - NetBeans (Demo)



JSF - NetBeans

- vollständiges WebFramework auf der Basis von Servlets
- Parameterzugriff sehr einfach (via JavaBeans)
- viele (OS) Erweiterungen
- Aber, weniger Kontrolle !

JSF - NetBeans (Demo)



The screenshot shows the 'New File' dialog in NetBeans. The 'Project' is 'WebApplication2'. Under 'Categories', 'JavaServer Faces' is selected. Under 'File Types', 'JSF Managed Bean' is selected. The description reads: 'Creates a new managed bean class.' The 'Next >' button is highlighted.

Steps

1. Choose File Type
2. ...

Choose File Type

Project:

Categories:

- Web
- JavaServer Faces**
- Bean Validation
- Struts
- Spring Framework
- Enterprise JavaBeans
- Contexts and Dependenc
- Java
- Swing GUI Forms
- JavaBeans Objects

File Types:

- JSF Page
- JSF Managed Bean**
- JSF Faces Configuration
- JSF Composite Component
- JSF Pages from Entity Classes
- Facelets Template
- Facelets Template Client

Description:

Creates a new managed bean class.

Buttons: Help, < Back, **Next >**, Finish, Cancel

JSF - NetBeans (Demo)

Steps

1. Choose File Type
2. **Name and Location**

Name and Location

Class Name:

Project:

Location:

Package:

Created File:

Add data to configuration file

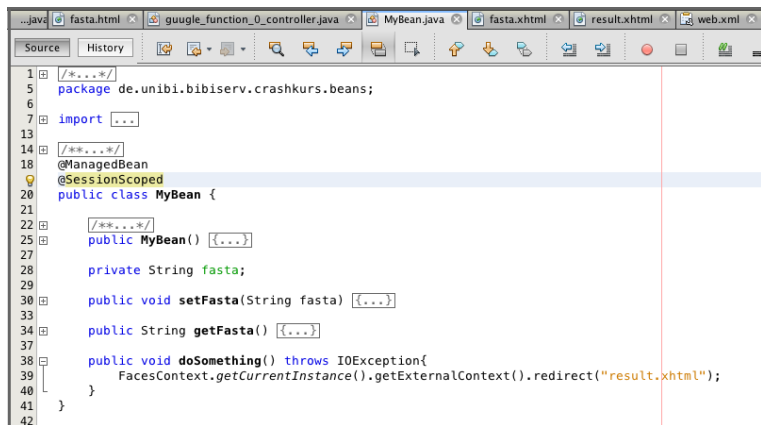
Configuration File:

Name:

Scope:

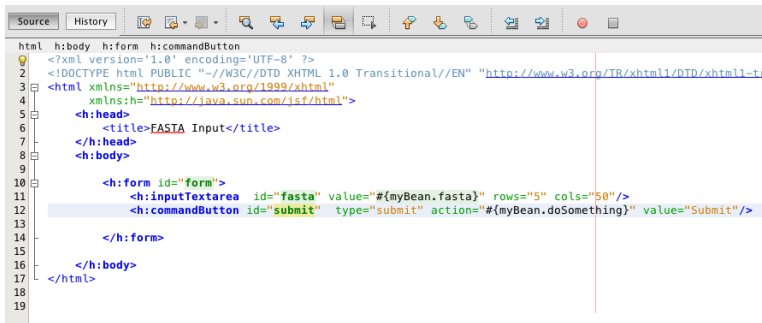
Bean Description:

JSF - NetBeans (Demo)



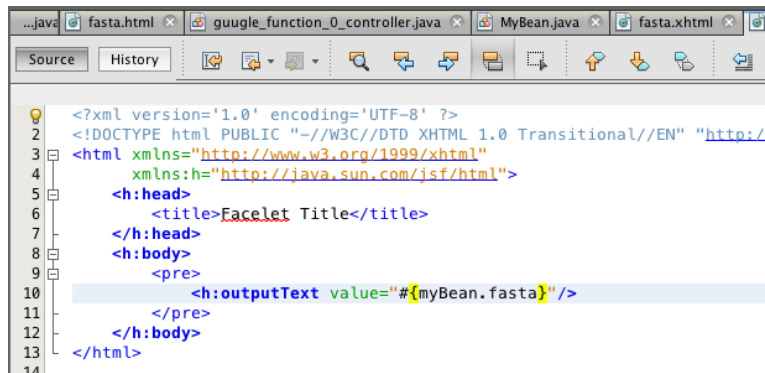
```
1  /**...*/
5  package de.unibi.bibiserv.crashkurs.beans;
6
7  import ...
13
14  /**...*/
18  @ManagedBean
19  @SessionScoped
20  public class MyBean {
21
22  /**...*/
25  public MyBean() {...}
27
28  private String fasta;
29
30  public void setFasta(String fasta) {...}
33
34  public String getFasta() {...}
37
38  public void doSomething() throws IOException{
39      FacesContext.getCurrentInstance().getExternalContext().redirect("result.xhtml");
40  }
41  }
42
```

JSF - NetBeans (Demo)



```
html h:body h:form h:commandButton
1 <?xml version='1.0' encoding='UTF-8' ?>
2 <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-t
3 <html xmlns="http://www.w3.org/1999/xhtml"
4   xmlns:h="http://java.sun.com/jsf/html">
5   <h:head>
6     <title>FASTA Input</title>
7   </h:head>
8   <h:body>
9
10     <h:form id="form">
11       <h:inputTextarea id="fasta" value="#{myBean.fasta}" rows="5" cols="50"/>
12       <h:commandButton id="submit" type="submit" action="#{myBean.doSomething}" value="Submit"/>
13
14     </h:form>
15
16   </h:body>
17 </html>
18
19
```

JSF - NetBeans (Demo)



The screenshot shows the NetBeans IDE interface with several tabs open: `..java`, `fasta.html`, `google_function_0_controller.java`, `MyBean.java`, and `fasta.xhtml`. The `Source` tab is active, displaying the following XML code:

```
1 <?xml version='1.0' encoding='UTF-8' ?>
2 <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://
3 <html xmlns="http://www.w3.org/1999/xhtml"
4     xmlns:h="http://java.sun.com/jsf/html">
5     <h:head>
6         <title>Facelet Title</title>
7     </h:head>
8     <h:body>
9         <pre>
10            <h:outputText value="#{myBean.fasta}"/>
11        </pre>
12    </h:body>
13 </html>
14
```


- Allgemein: Komponente zur visuellen Darstellung von Daten
- zwei mögliche Herangehensweisen
 - 1 Ergebnisseite wird nach Anfrage vom Server dynamisch zusammengebaut (z.B. Template + Datenaus der Datenbank)
 - 2 Ergebnisseite lädt via JavaScript Daten dynamisch vom Server nach
- Vorteile, Nachteile des jeweiligen Verfahrens ?

XMLHttpRequest

XMLHttpRequest Objekt ermöglicht http Anfrage aus einer Seite auf den Server von dem die Seite geladen wurde!

- GET / POST möglich
- synchrones / asynchrones Handling möglich

typische Anwendung

```
if (window.XMLHttpRequest) {  
    xmlhttp=new XMLHttpRequest();  
} else {  
    xmlhttp=newActiveXObject("Microsoft.XMLHTTP");  
}
```

Beispiel synchron/GET

```
var url = ...
var data = ...

xmlhttp.open("GET", url, false);
xmlhttp.send(data);
if (xmlhttp.status == 200) {
    var response = xmlhttp.response;
    // manipulate page
    // ...
} else {
    // error handling
    // ...
}
```

Alternative zu response : responseText bzw. responseXML !

Beispiel asynchron/POST

```
var url = ...;
var data = ...;
xmlhttp.open("POST", url, true);
xmlhttp.setRequestHeader("Content-type","application/x-www-form-urlencoded");
xmlhttp.setRequestHeader("Content-length", data.length);
xmlhttp.setRequestHeader("Connection","close");
xmlhttp.onreadystatechange = handlerresponse;
xmlhttp.send(data);
...
// show progressbar or something like that
...

function handlerresponse () {
    if (xmlhttp.readyState == 4) {
        if (xmlhttp.status == 200) {
            var response = xmlhttp.response;
            // manipulate page
            ...
        } else {
            // error handling
            ...
        }
    }
}
```

Beispiel : BiBiServ Statistiken

- `http://bibiserv.cebitec.uni-bielefeld.de/statistics`
- „Viewer“ für aufbereitete WebServer Log Daten.
- kein HTML5, sondern `xhtml + svg`
- Daten vom Server via CGI (BiBiServ) bzw. Servlet (BiBiServ2)
- Sourcen ? → `/vol/bibidev/statistics` bzw. Browser

nützliche Referenzen

Specs

- W3C

Perl

- Skriptsprachen

- Quickreference

JSF

- TagLib Reference Documentation

- PrimeFaces

JavaScript

- W3Schools XMLHttpRequest

- Selfhtml (auch Perl/CGI & HTML)

- JQuery