Crashkurs http - CGI/Servlets(JSF) - Viewer

Jan Krüger
jkrueger(at)cebitec.uni-bielefeld.de

http

- 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 Resourcenanforderung (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";
```

CGI

- 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;
```

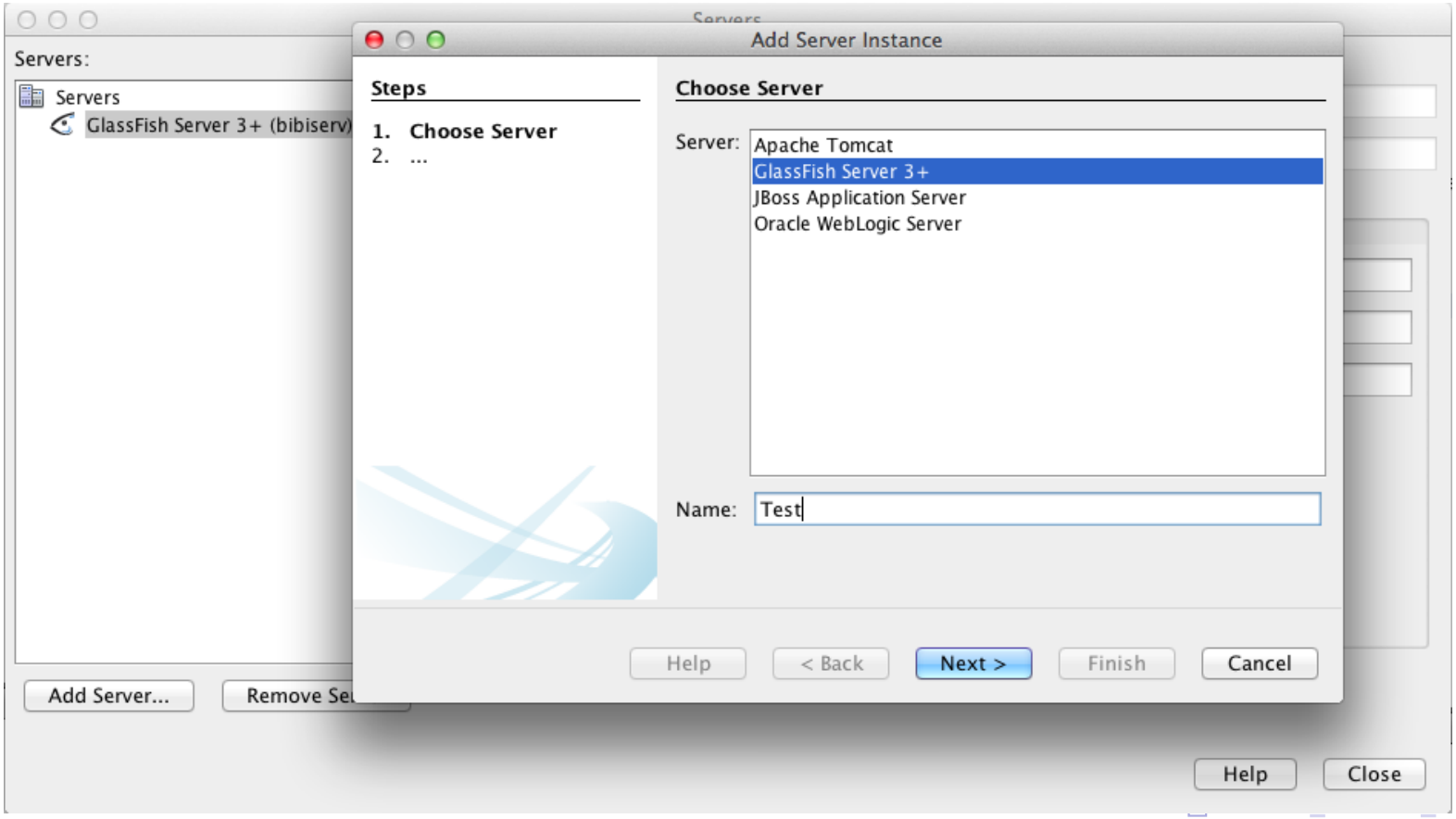
the Java way . . .

- J2EE
- Servlets
- JavaServerFaces 2.x

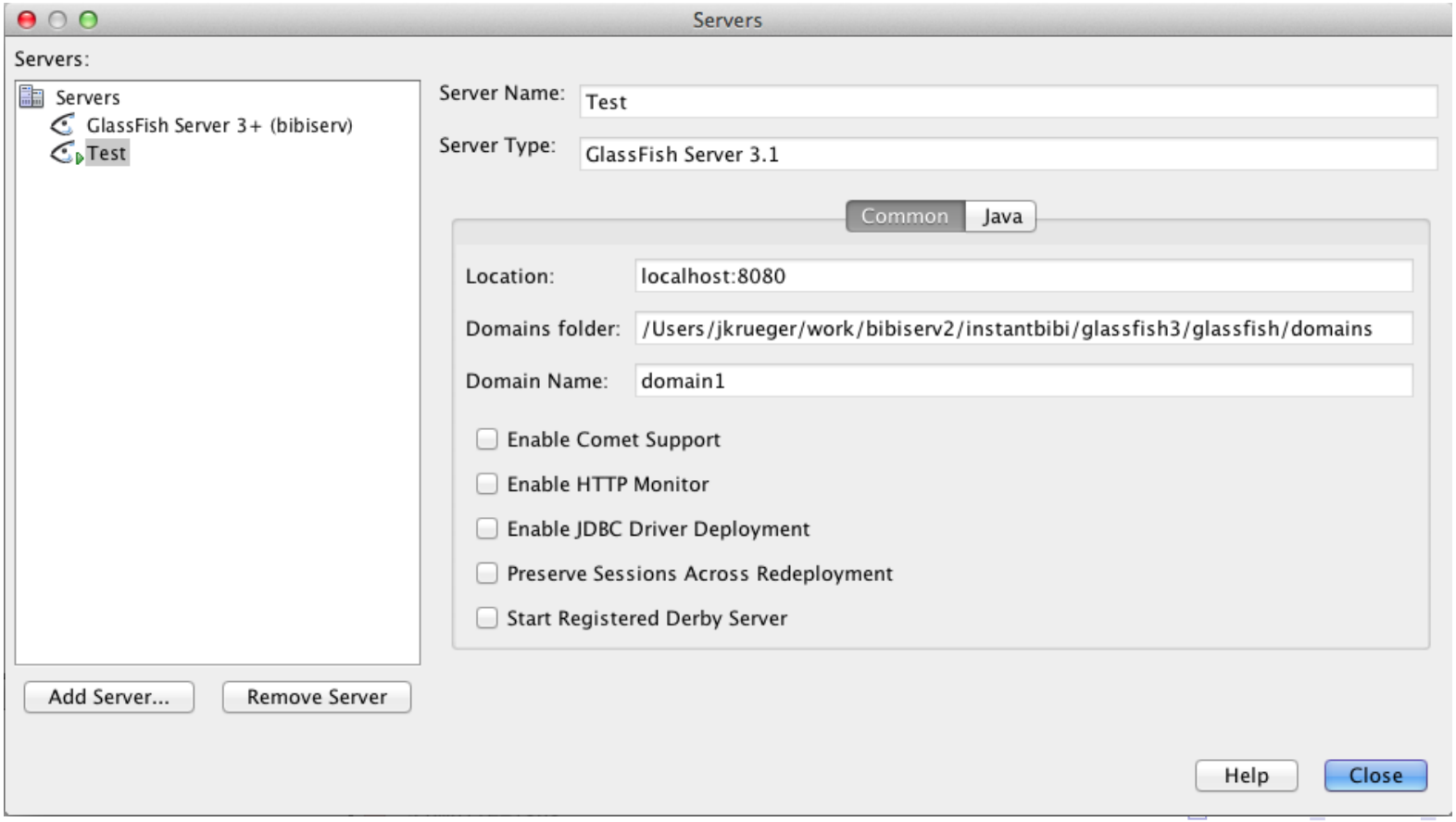
J2EE Applikation Server - NetBeans

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

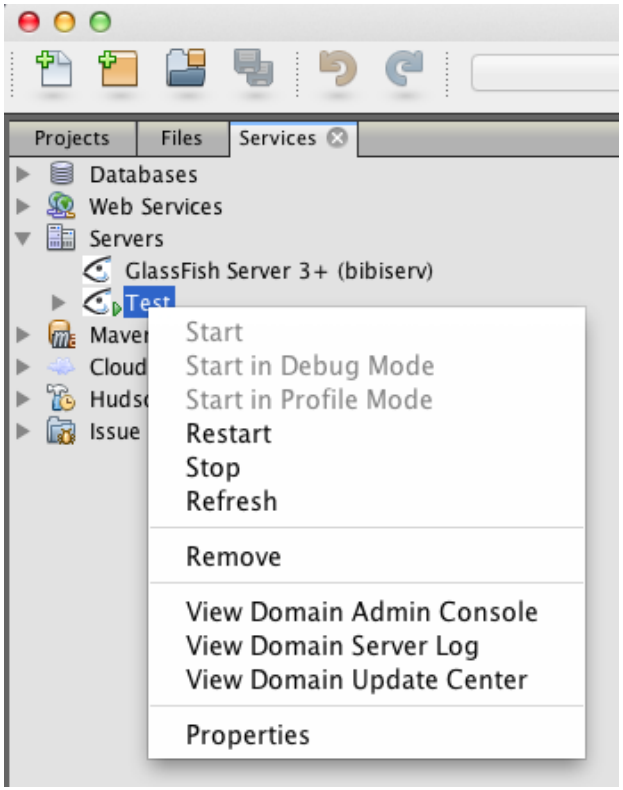
J2EE Applikation Server - NetBeans



J2EE Applikation Server - NetBeans



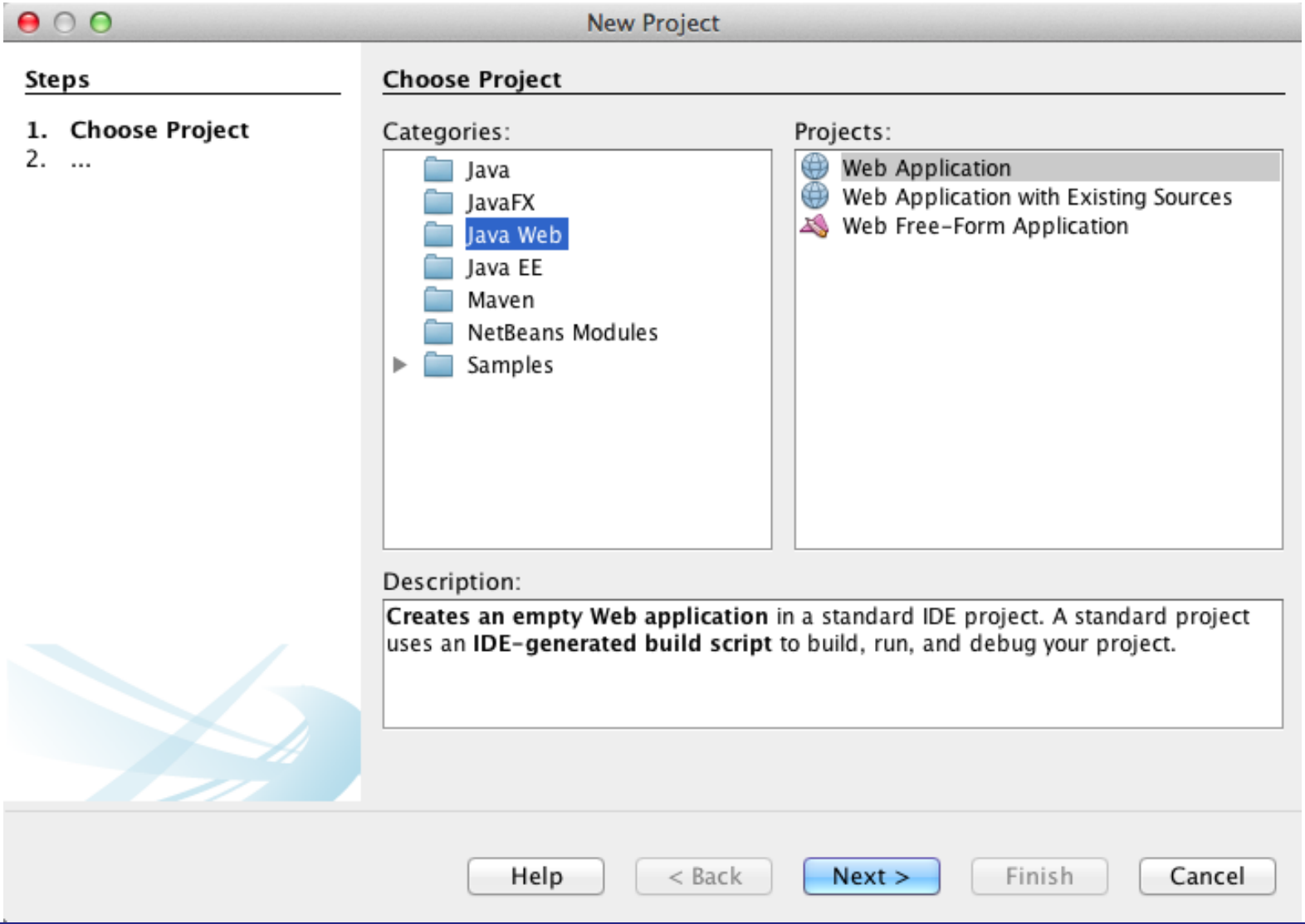
J2EE Applikation Server - NetBeans



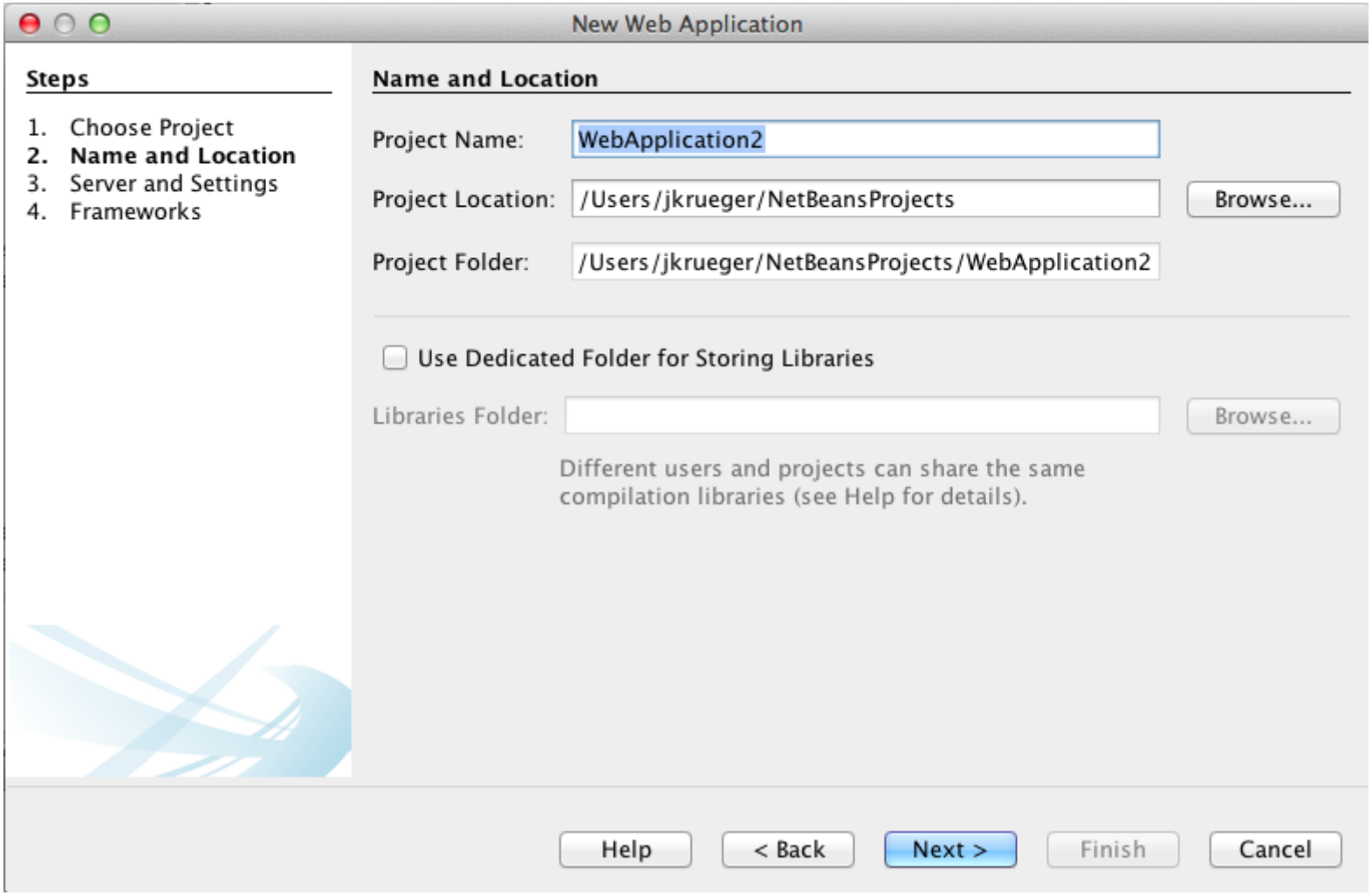
WebProjekt - NetBeans

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

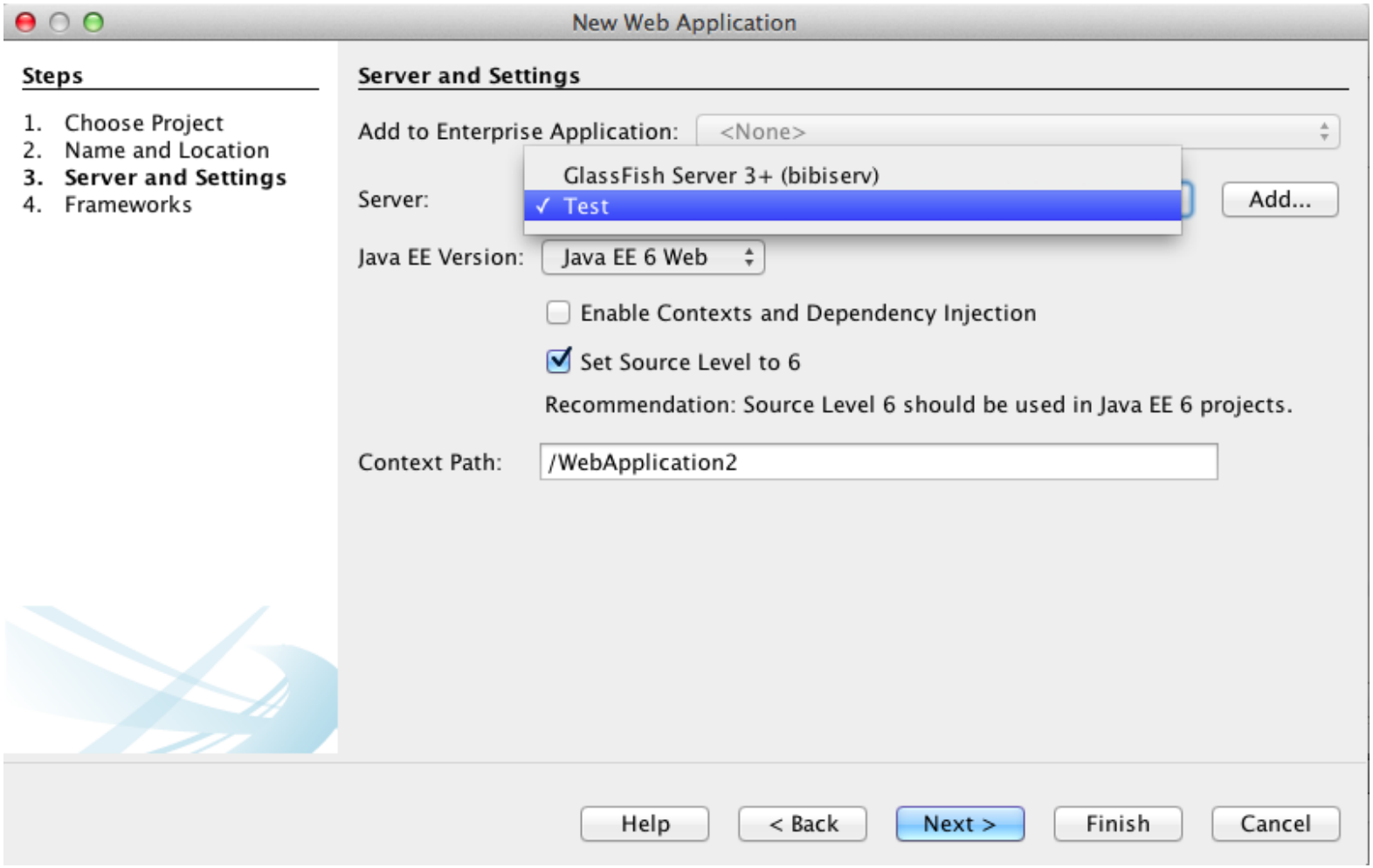
WebProjekt - NetBeans (Demo)



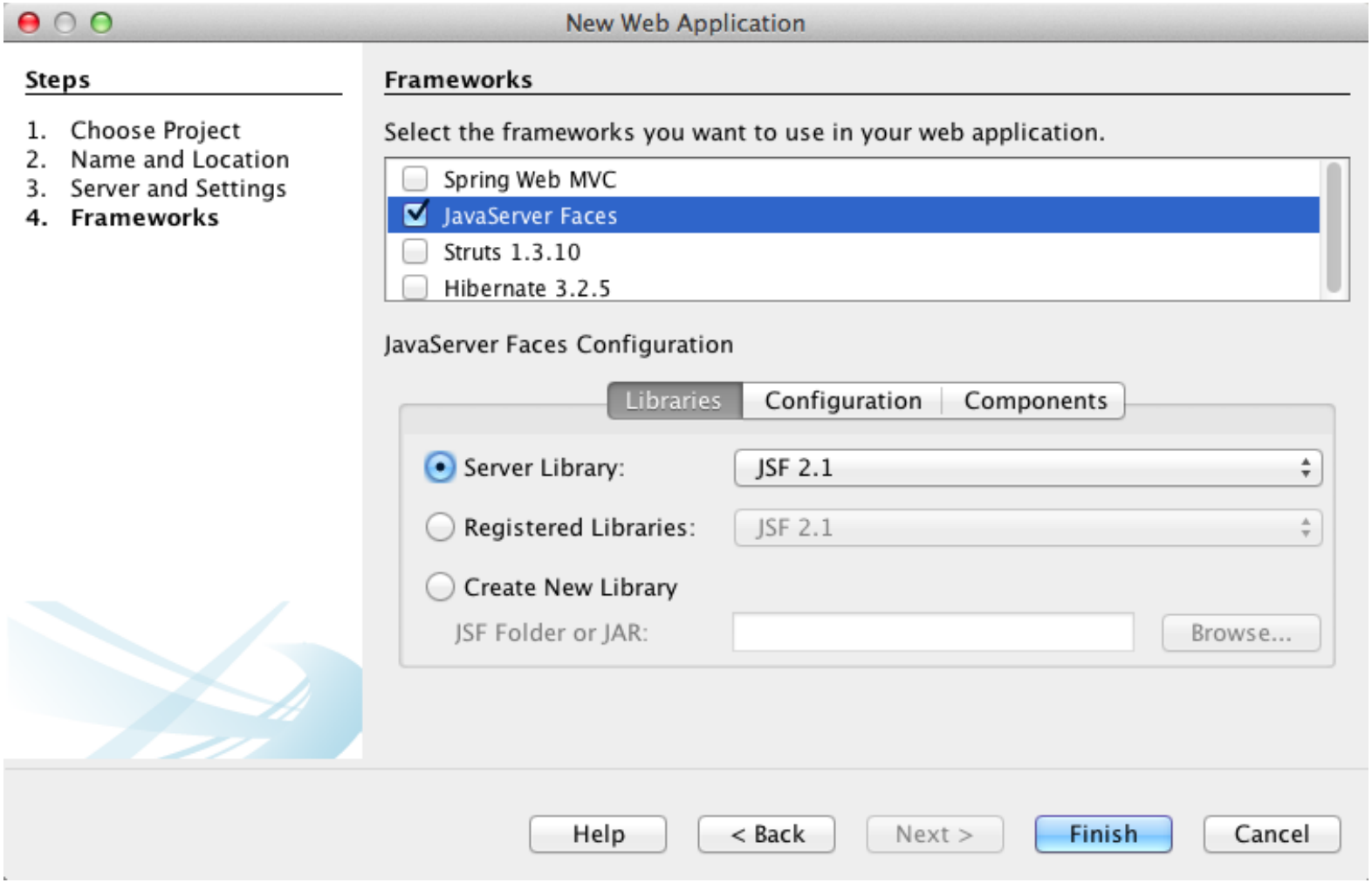
WebProjekt - NetBeans (Demo)



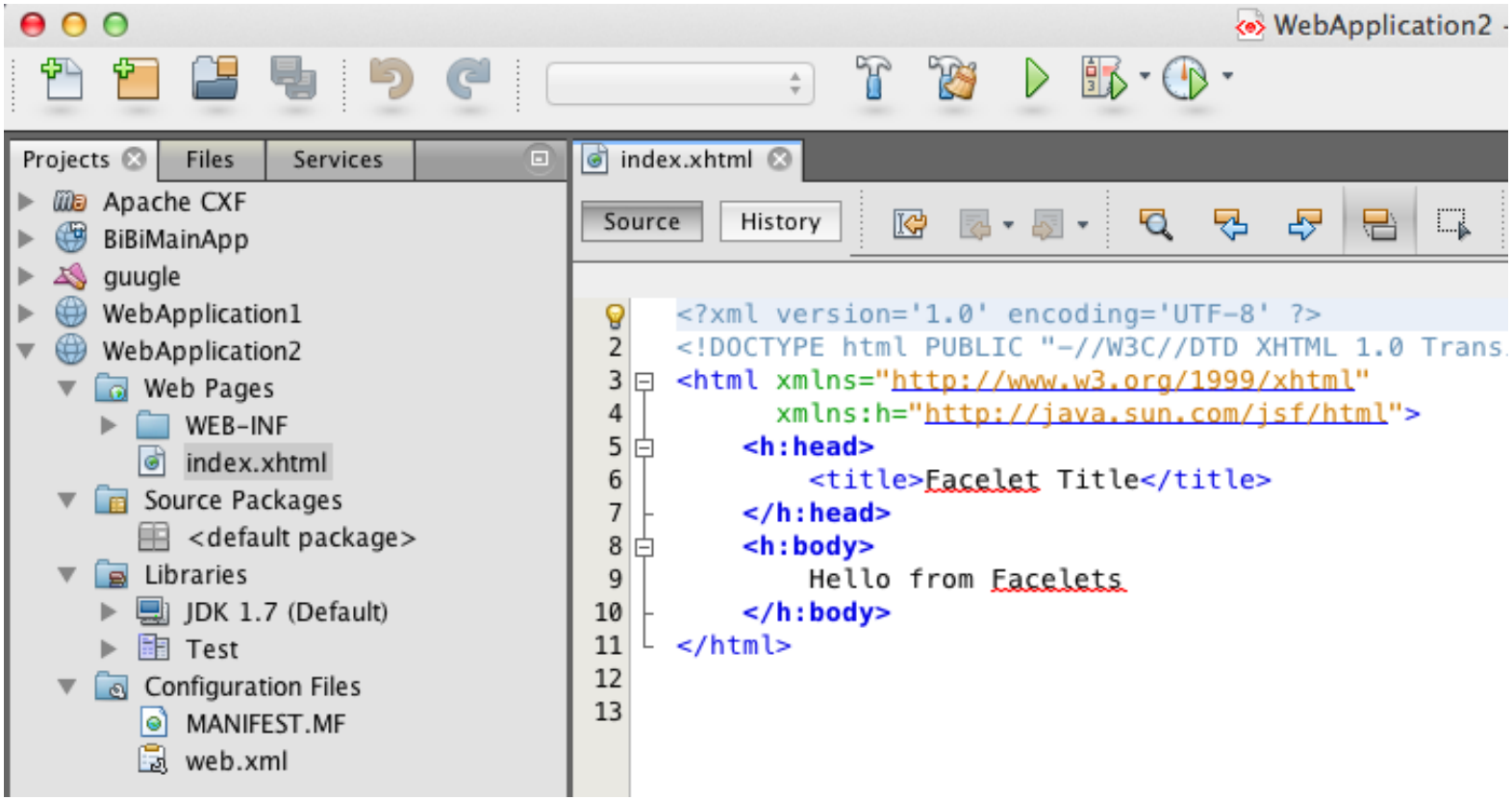
WebProjekt - NetBeans (Demo)



WebProjekt - NetBeans (Demo)



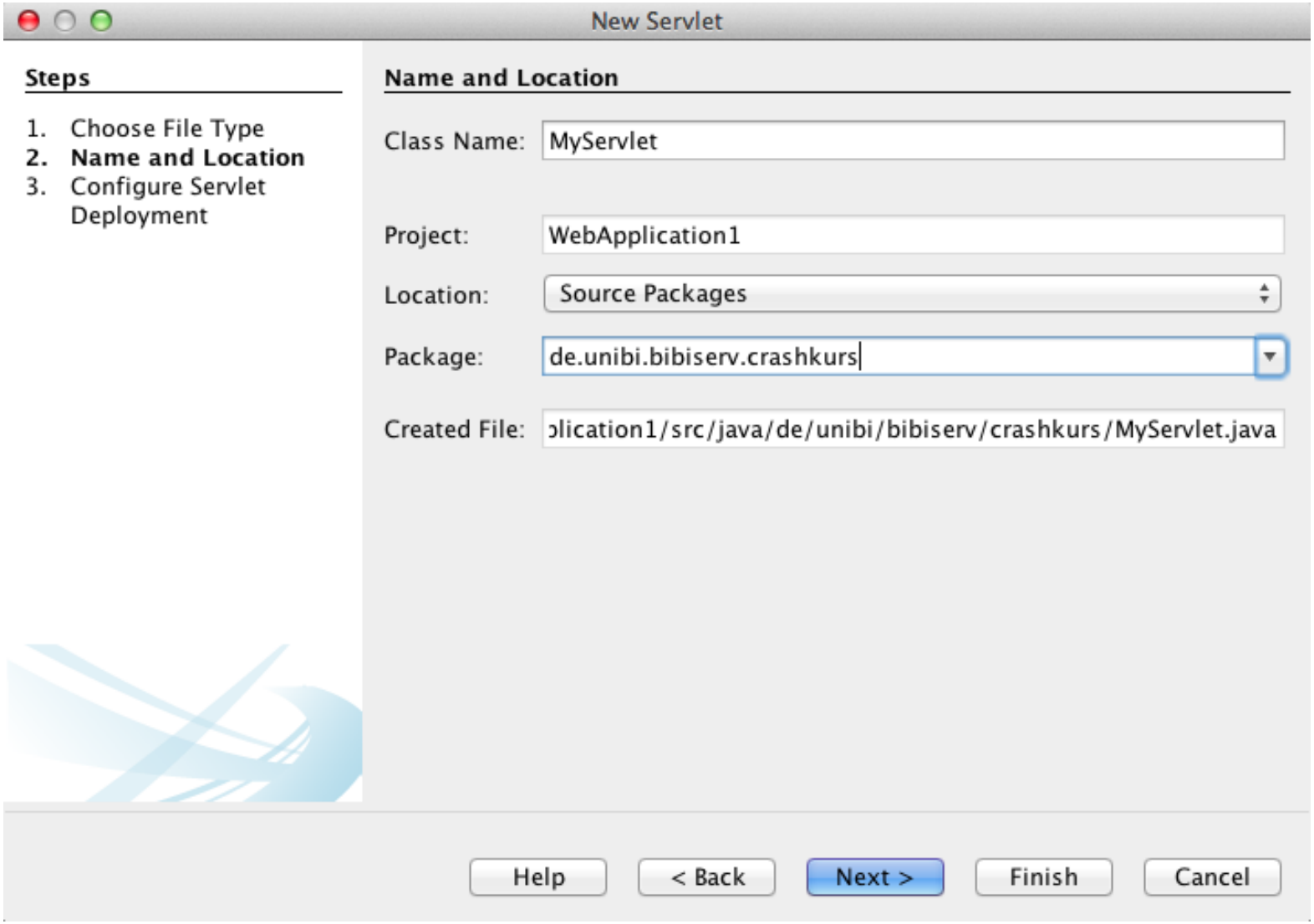
WebProjekt - NetBeans (Demo)



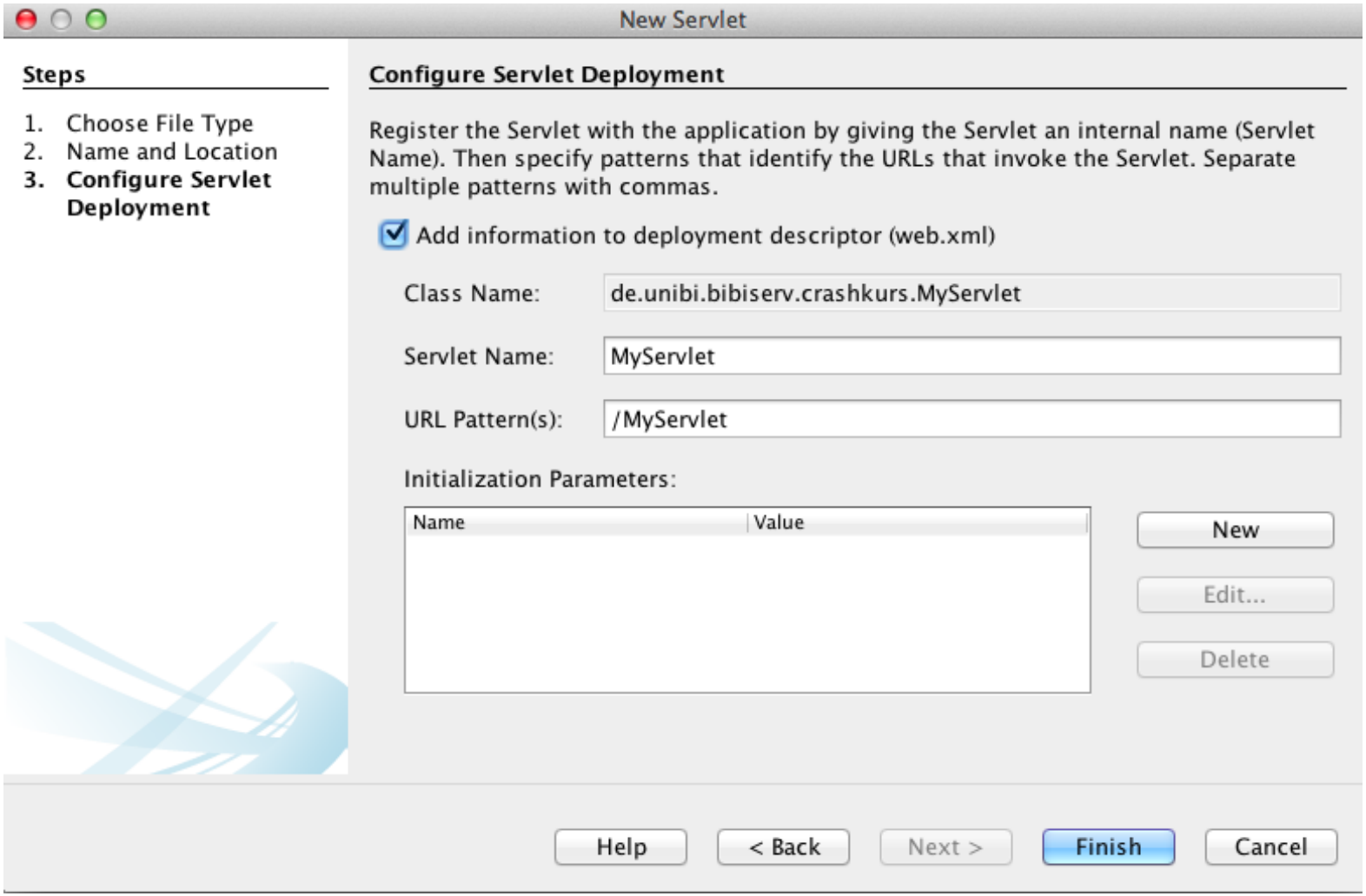
Servlets - NetBeans

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

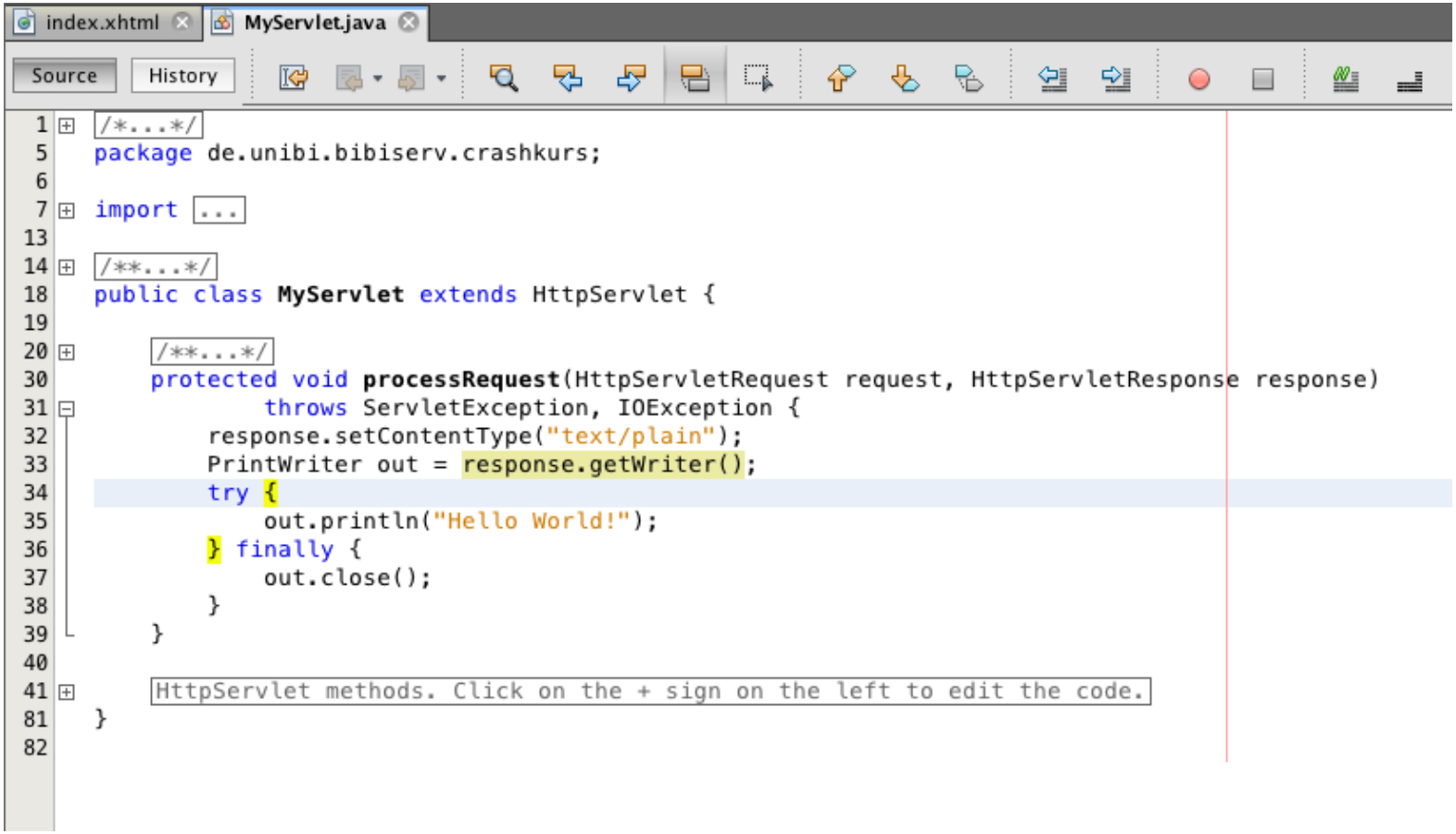
Servlets - NetBeans (Demo)



Servlets - NetBeans (Demo)



Servlets - NetBeans (Demo)



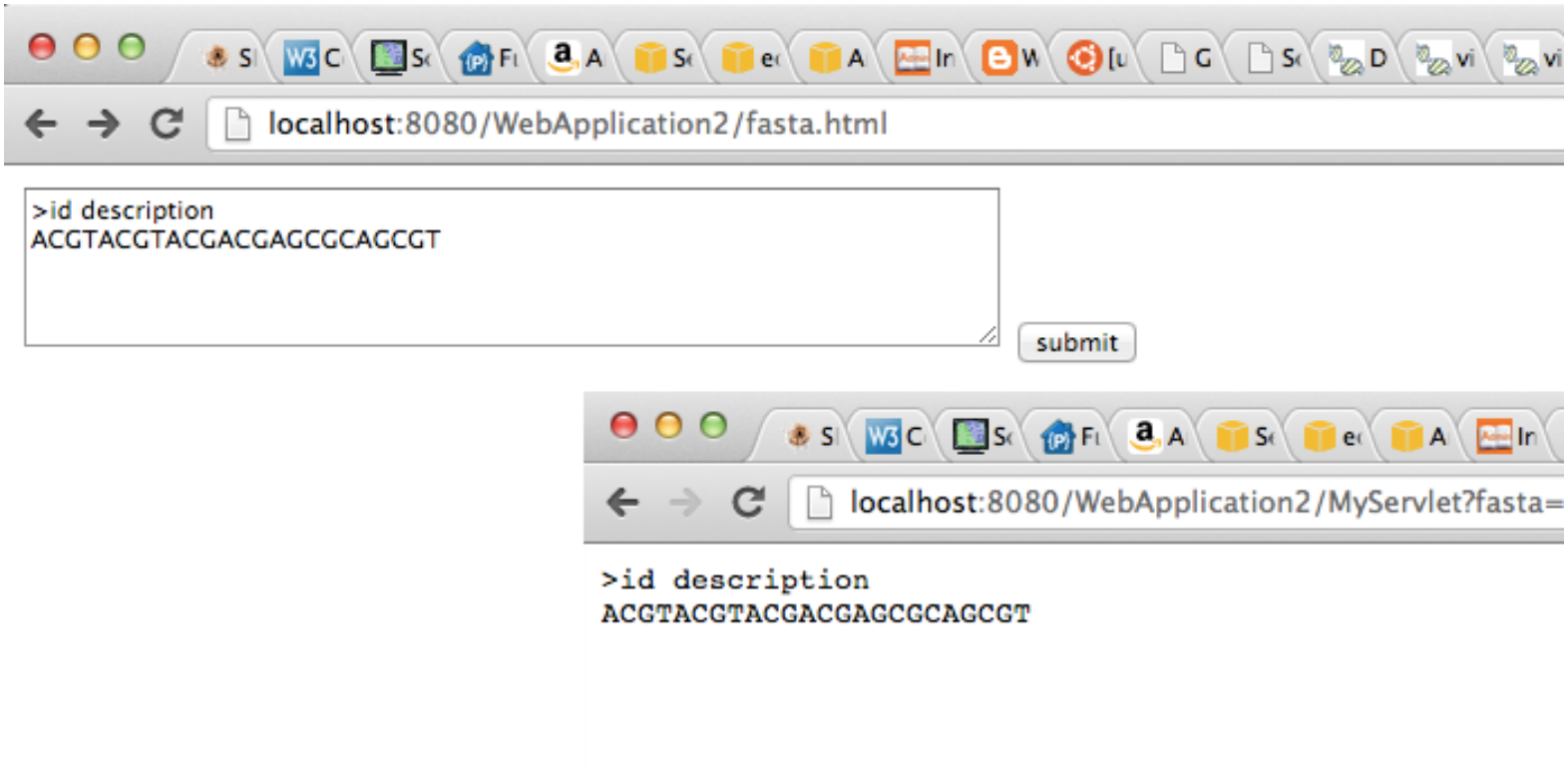
Servlets - NetBeans (Demo)

index.xhtml | MyServlet.java

Source | History

```
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             String result = request.getParameter("fasta");
36             out.println(result);
37         } finally {
38             out.close();
39         }
40     }
41
42     HttpServlet methods. Click on the + sign on the left to edit the code.
82 }
83
```

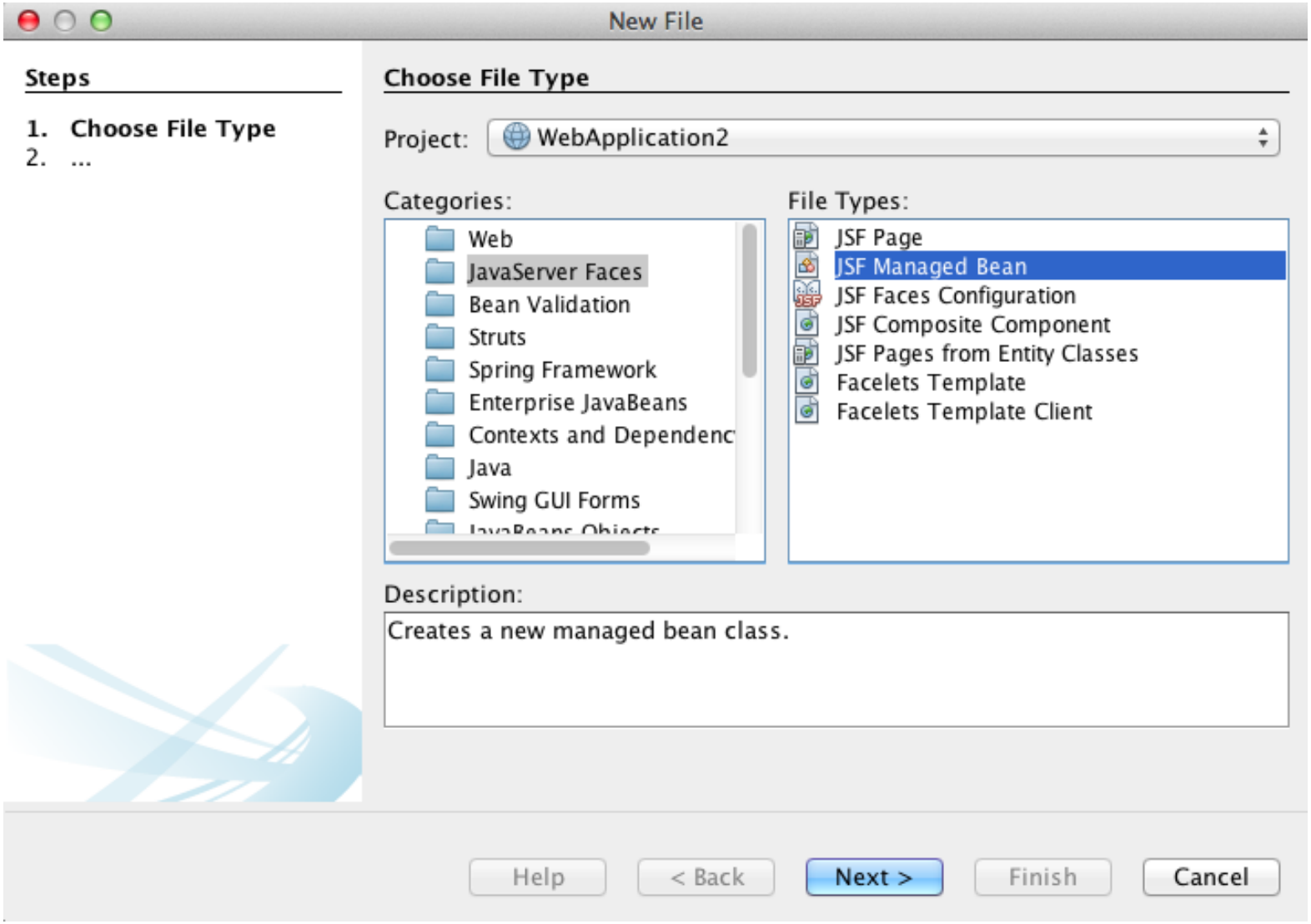
Servlets - NetBeans (Demo)



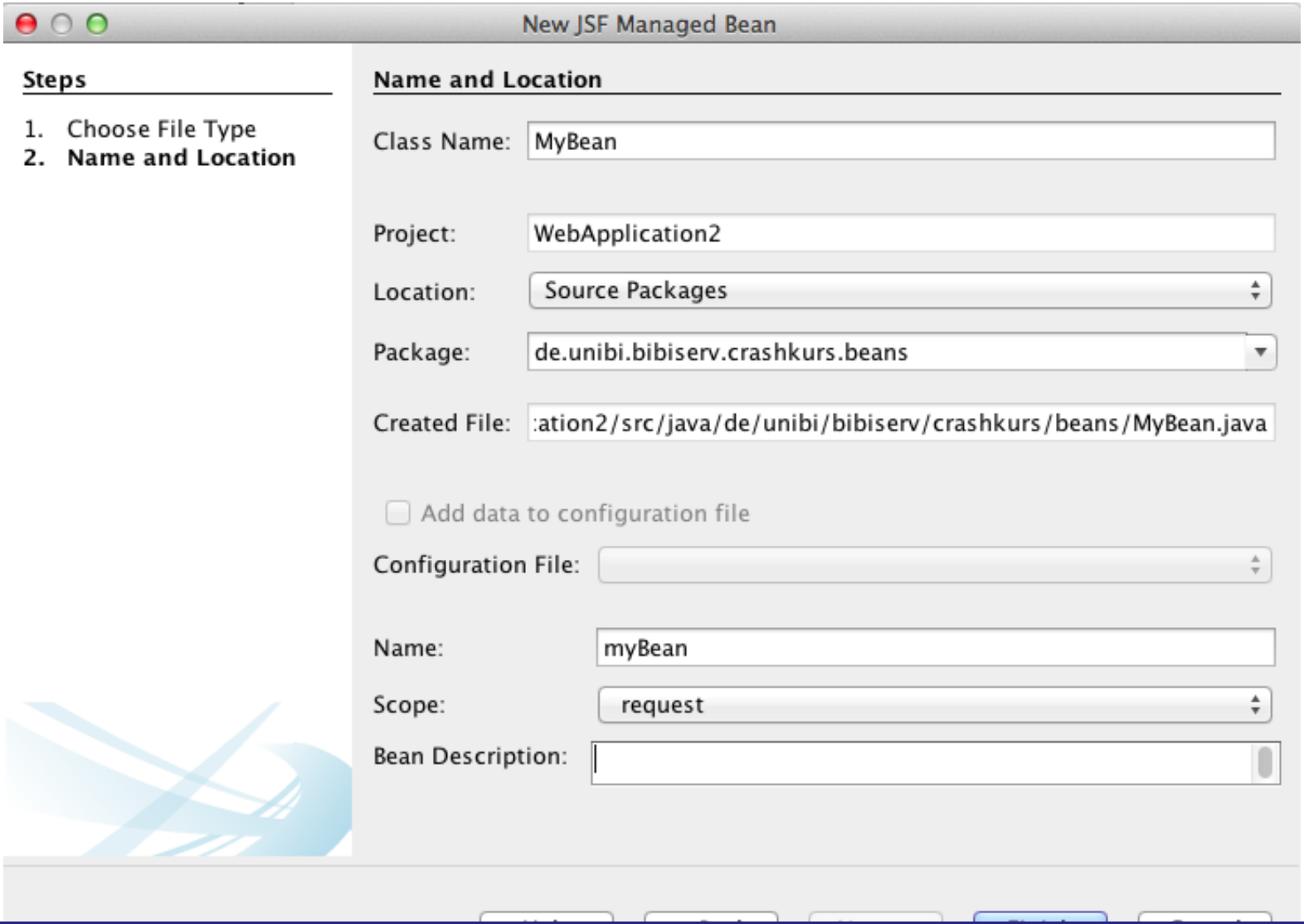
JSF - NetBeans

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

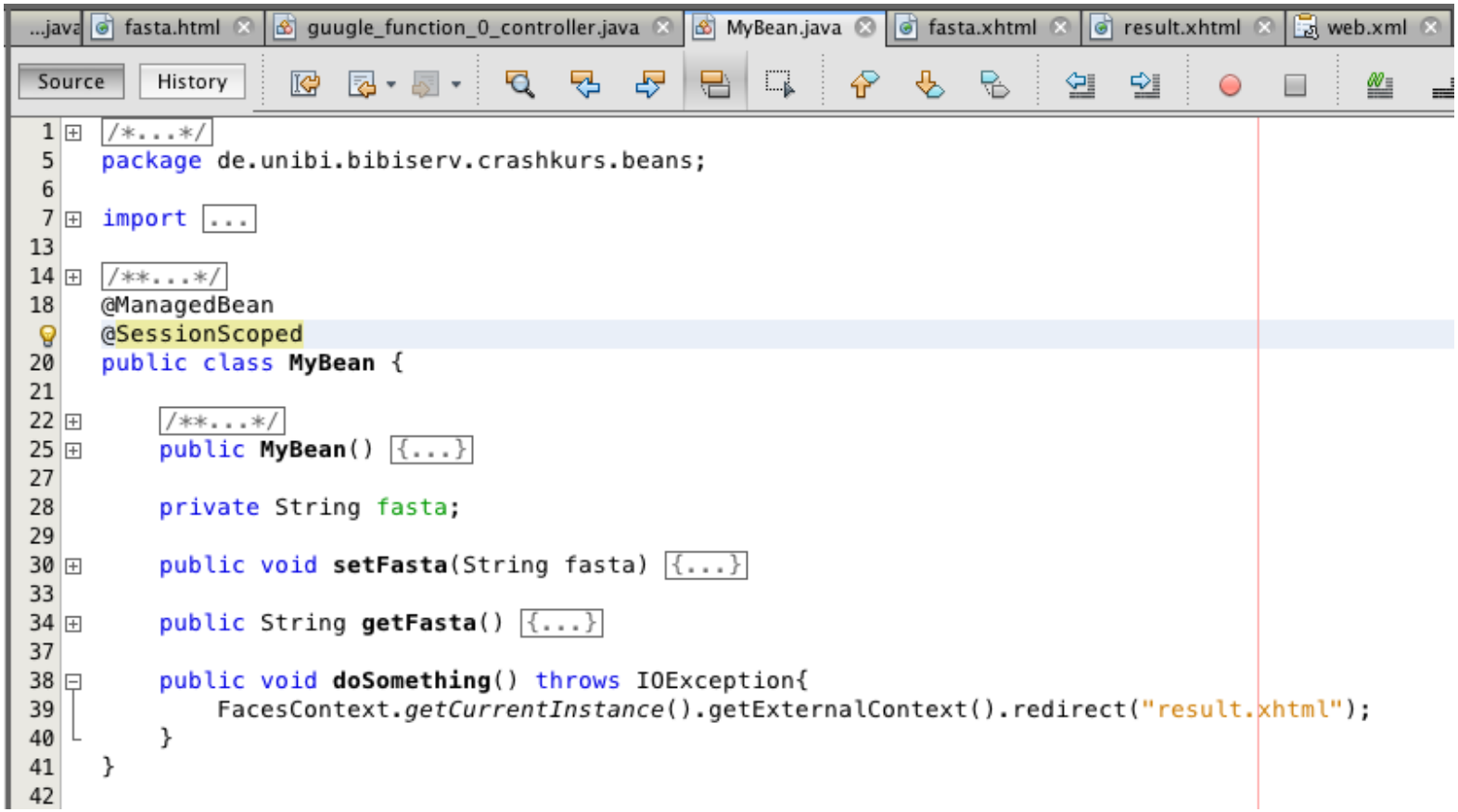
JSF - NetBeans (Demo)



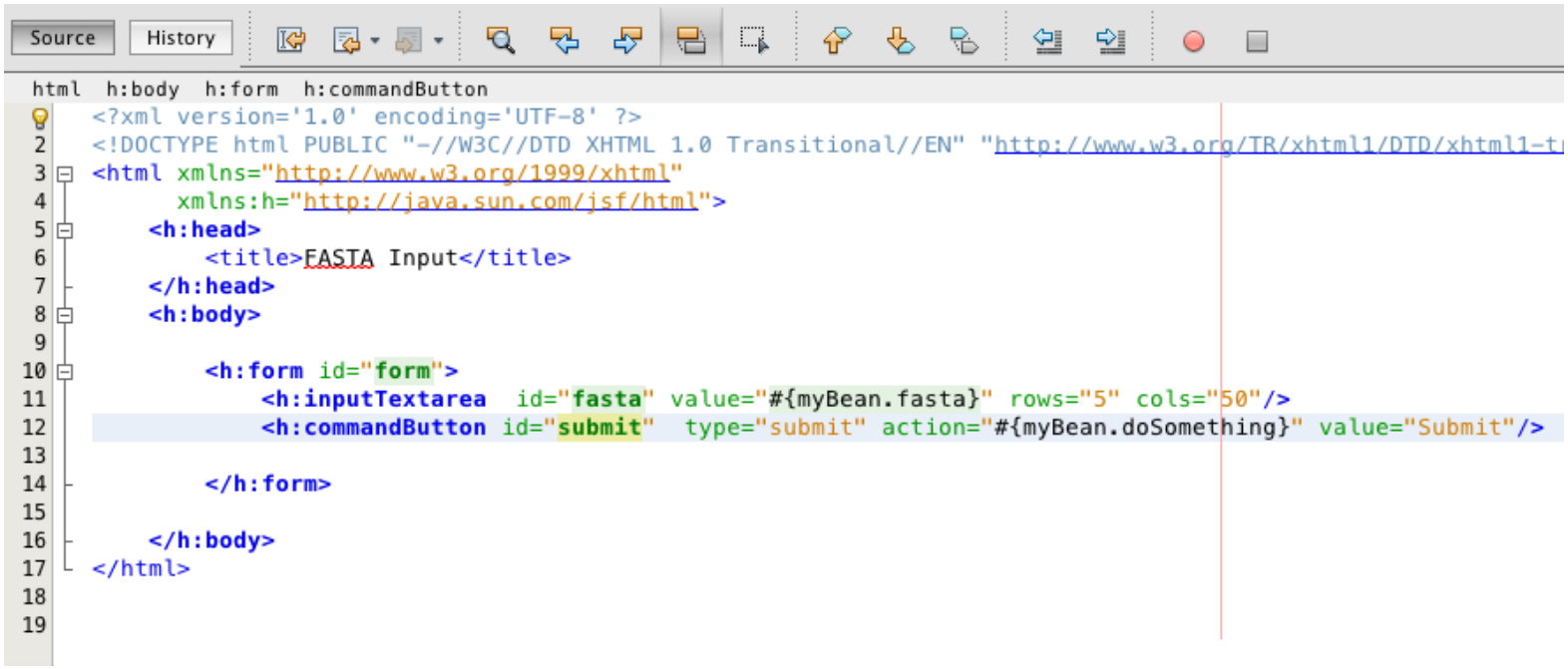
JSF - NetBeans (Demo)



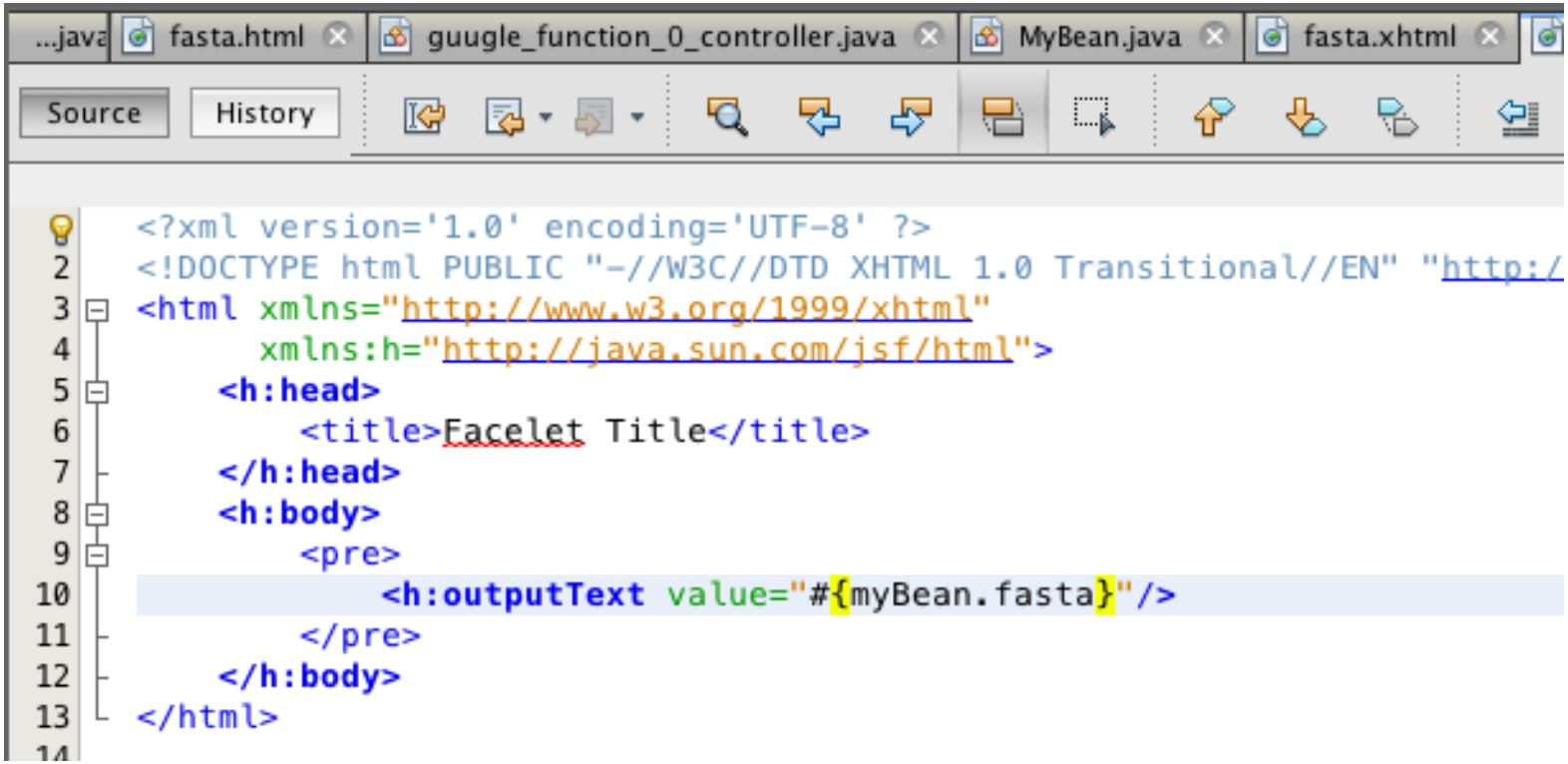
JSF - NetBeans (Demo)



JSF - NetBeans (Demo)



JSF - NetBeans (Demo)



Viewer

- 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 läd 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 = handleresponse;
xmlhttp.send(data);
...
// show progressbar or something like that
...

function handleresponse () {
    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 ? $\rightarrow$ /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