"Hallo Internetz"

Franz Fiala

Es gibt zwar auch in Computersprachen zahlreiche Möglichkeiten, einen ersten Text auf den Bildschirm zu bringen, doch durch das Nebeneinander verschiedener Verfahren im Internet gibt es hier eine noch größere Vielfalt. Alle Methoden wurden an einem Windows-NT-Server getestet. Die Beispiele sind im Internet abrufbar.

Wie funktioniert ein Browser?

Jede Referenz des Users zu einem HTML-Dokument (z.B. durch Eingabe in der Adresszeile oder Anklicken eines Hyperlink) veranlasst den Server, dieses Dokument zum Client zu senden. Die Kommunikation wird durch das HTTP-Protokoll gesteuert. Die Elemente des HTTP-Protokolls sind für den Benutzer nicht sichtbar. Lediglich in der Schreibweise des URL kann man einige Regeln des HTTP-Protokolls erkennen.

Der Normalfall ist, daß der Server die Dokumente inklusive aller referenzierten Objekte unverändert zum Client schickt. Diese Möglichkeiten werden unter "Clientseiteige Methoden" gezeigt. Der Server hat hier nur eine Transportaufgabe zu erfüllen. Alle Dokumente mit der Endung `htm` oder `html` werden so behandelt.

Es gibt aber Situationen, in denen der Server anders reagiert. Im allgemeinen ist es die Dateiendung, die den Unterschied macht. Diese Möglichkeiten werden unter "serverseitige Methoden" besprochen.

Clientseitige Methoden

1 Text in HTML-Datei

2 Text in HTML-Datei indirekt formatiert mit Style-Sheets

3 Text durch Skript generiert

4 Text in Statuszeile

5 Text in eigenem Fenster

6 Text in neuem Browserfenster

7 Textvariation durch DHTML

8 Text aus Cookie

9 Text im Java Applet

Serverseitige Methoden

10 Text aus EXE-Datei

11 Text aus Perl-Skript

12 ASP aus ASP-Skript

13 Text aus Datenbank

Clientseitige Methoden

In den folgenden Beispielen wird der Text "Hallo Internetz" am Bildschim gezeigt.

1. HTML

HTML dient zur Strukturierung eines kontinuierlichen Fließtextes durch Tags und enthält nur wenige Elemente, die die Darstellung betreffen. Wie der Text aussieht, bestimmt die Einstellung am Browser.

1.htm
HTML-Datei

```
<BODY>
  <P>Hallo Internet<I>z</I></P>
</BODY>
```

Will man das Aussehen verändern, beginnen sich im Kode Strukturinformationen mit Darstellungsinformationen zu vermengen:

1a.htm
HTML-Datei mit Format-Anweisungen

```
<BODY BGCOLOR="#DDDDDD">
  <P>
    <FONT COLOR=green SIZE=4>
      Hallo Internet<I>z</I>
    </FONT>
  </P>
</BODY>
```

Vergleichbar ist dieser Stil mit einer Textverarbeitung, bei der alle Textauszeichnungen mit direkter Formatierung vorgenommen werden.

2. CSS

StyleSheets ermöglichen eine präzise Definition des Aussehens eines Dokuments in einer eigenen Datei. Im HTML-Dokument erfolgt eine Referenz über das `LINK`-Tag.

2.htm
HTML-Datei referenziert Style-Sheet und enthält nur Strukturanweisungen

```
<LINK REL="StyleSheet"
      HREF="2.css"
      TYPE="text/css">
<BODY>
  <P>Hallo Internet<I>z</I></P>
</BODY>
```

Im CSS-Dokument kann jeder Tag mit einer großen Zahl möglicher Attribute exakt beschrieben werden.

Der Server schickt zwei Dateien. Zuerst die HTML-Datei, die im Head-Teil einen `LINK`-Tag enthält, der die Style-Sheet-Datei benennt. Die Style-Sheet-Datei beschreibt, wie die vorhandenen Tags `BODY`, `I` und `P` darzustellen sind.

2.css
Style-Sheet enthält Formatanweisungen:

```
<STYLE>
{ }
BODY {
  background-color : #CCCCCC;
}
P  {
  font-family : Verdana;
  font-size : 10.00000 pt;
  color : #444444;
  background-color: #DDFFDD;
}
I {
  font-zize : 14 pt;
  font-weight : bold;
  font-style : italic;
}
</STYLE>
```

Diese Trennung in Strukturinformation hat eine ähnliche Wirkung wie die Druckformatvorlagen in der Textverarbeitung.

Anmerkung: das Klammernpaar `{}` am Beginn der Style-Sheet-Datei ist wichtig, sonst wird der erste Stil nicht interpretiert.

3. Skriptsprachen

Browser haben Skriptsprachen integriert. Es sind zwar keine vollständigen Computersprachen, erlauben aber viele Gestaltungsmöglichkeiten. Als Sprachen können JavaScript [NetScape] (JScript [MS]) oder VBScript [MS] verwendet werden. Die Skriptsprache behandelt alle Teile eines HTML-Dokuments, die in `<SCRIPT>..</SCRIPT>` eingebettet sind. Wie das folgende Beispiel zeigt, kann das gesamte Dokument als eine Folge von Skriptanweisungen dargestellt werden.

3a.htm
Skriptsprache generiert Dokument

```
<SCRIPT LANGUAGE="JavaScript">
  var dtmVar = new Date();
  document.write ("<BODY BGCOLOR=#A0FFA0>");
  document.write ("<P>Hallo Internet");
  document.write
    ("am ",dtmVar.getDate(),".",
     dtmVar.getMonth()+1,".",
     dtmVar.getYear(),"</P>");
```

```
  document.write ("</BODY>");
</SCRIPT>
```

Für eingefleischte Visual Basic Programmierer stehen durch VBScript auch alle gewohnten Visual Basic Steueranweisungen zur Verfügung. Das Programm unterscheidet sich nur unwesentlich von der JavaScript-Version:

3b.htm
Skriptsprache generiert Dokument

```
<SCRIPT LANGUAGE="VBScript">
  Document.Write "<BODY BGCOLOR=#A0A0FF>"
  Document.Write _
    "<P>Hallo Internet<I>z</I></P>"
  Document.Write "</BODY>"
</SCRIPT>
```

Während JavaScript unempfindlich gegenüber einem Zeilenumbruch an Trennstellen ist, muss man bei VBScript die Zeile mit "_" verlängern. Außerdem entfallen bei VBScript die Klammern für die Übergabeparameter und die Strichpunkte am Zeilenende.

Dieses Beispiel zeigt nicht gerade die Vorteile der Skriptsprache, doch hätte man beispielsweise eine komplizierte Tabelle zu editieren, kann auch beim Bildaufbau ein JavaScript-Programm sehr hilfreich sein. (Beispiel siehe PCNEWS-53, Seite 76).

4 Text in Statuszeile

Skriptsprachen können natürlich noch mehr. Beispielsweise kann man auch das Drumherum eines Fensters steuern, etwa den Inhalt der Statusleiste:

4.htm
Text in Status-Zeile:

```
<INPUT
  TYPE="button"
  VALUE="hier klicken"
  onClick=
    "window.status='Hallo Internetz'">
</INPUT>
```

5 Text in eigenem Fenster

Skriptsprachen können Botschaften auch in einem eigenen Fenster generieren. Wie auch im vorigen Beispiel hat man jetzt aber keinen Einfluß mehr auf das Aussehen des Textes, zumindest nicht mit Mitteln von HTML.

5a.htm
Nachricht in Fenster (JavaScript):

```
<SCRIPT LANGUAGE="JavaScript">
  alert ("Hallo Internetz");
</SCRIPT>
```

VBScript wartet hier mit einer Besonderheit auf: einerseits würde in VBScript das obige Programm auch ablaufen, wenn man nur die Klammern beim Text und den Strichpunkt entfernt, anderseits kann man auch die von Visual-Basic gewohnte Message-Box verwenden:

5b.htm
Nachricht in Fenster (VBScript):

```
<SCRIPT LANGUAGE="VBScript">
  MsgBox "Hallo Internetz"
</SCRIPT>
```

6 Text in neuem Browserfenster

Skriptsprachen können den Text auch in einem neuen Browser-Fenster erscheinen lassen, das vielfältig parametrisiert werden kann. Dazu muß keine weitere Serververbindung bestehen, denn alle Daten sind mit dem ersten HTML-Dokument geliefert worden:

6.htm
Neues Browser-Fenster

```
<SCRIPT Language="JavaScript">
window2=open
  ("","", "width=150, height=400");
window2.document.write
  ("<P>Hallo Internet<I>z</I></P>");
window2.document.bgColor="#FFCCCC";
</SCRIPT>
```

7. DHTML

Dynamisches HTML ist die logische Fortsetzung der Benutzung von Skriptsprachen in einem HTML-Dokument. Während JavaScript und VBScript zunächst nur mir den Formular-Elementen zusammenarbeiten, ist DHTML die Möglichkeit, Skripts auf Bereiche des HTML-Dokuments anwenden zu können.

Leider gehen hier Netscape und Microsoft getrennte Wege. Auf der Strecke bleiben die User - weil nur ein geringer Teil der Möglichkeiten aus Kompatiblitätsgründen genutzt werden kann - und die Programmierer - weil die Implementierung browserunabhängigen Kodes sehr aufwendig wird.

Netscape führt einen neuen Tag, den `LAYER`-Tag ein, der allein durch Skripts beeinflussbar ist. Microsoft beschränkt sich auf die bereits vorhandenen Tags `DIV` und `SPAN`, mehr noch, Skriptelemente sind bei der MS-Lösung in allen Tags möglich.

Hier das kurze Beispiel mit Elementen von DHTML angereichert. Eine Mausbewegung über den Text verändert dessen Farbe:

7.htm
DHTML (MS-Version)

```
<div id="Abschnitt"
  onMouseover=
    "Abschnitt.style.color='red'"
  onMouseout=
    "Abschnitt.style.color='green'">
  <P>Hallo Internet<I>z</I></P>
</div>
```

8. Cookie

Ein Cookie ist ein Stück Text, das ein Server am Computer des Users speichern kann. Der Server kann das aber nur, wenn der User diese Erlaubnis in den Browser-Einstellungen gegeben hat.

Für den Server sind Cookies praktisch, wenn er gleichzeitig viele User verwalten soll. Die für die einzelnen Sessions benötigten Variablen, z.B. die Daten eines Einkaufs, können am User-Rechner gespeichert werden.

Die `cookie`-Eigenschaft von `document` wird in einer ersten HTML-Datei dazu benutzt, das Cookie zu setzen:

```
<SCRIPT>
  document.cookie = "Hallo Internet";
</SCRIPT>
```

Ein anderes Dokument kann das Cookie lesen und wie im folgenden Beispiel ausgeben:

```
<SCRIPT>
```

Ressourcen Für DHTML

W3C: DOM Document Object Model

http://www.w3.org/TR/WD-DOM/

Microsoft

http://www.microsoft.com/workshop/author/dhtml/

Netscape

http://developer.netscape.com/docs/manuals/communicator/dynhtml/

Beginners Guide to DHTML

http://members.tripod.com/~toolmandavid/

Dynamic Drive DHTML Code Library

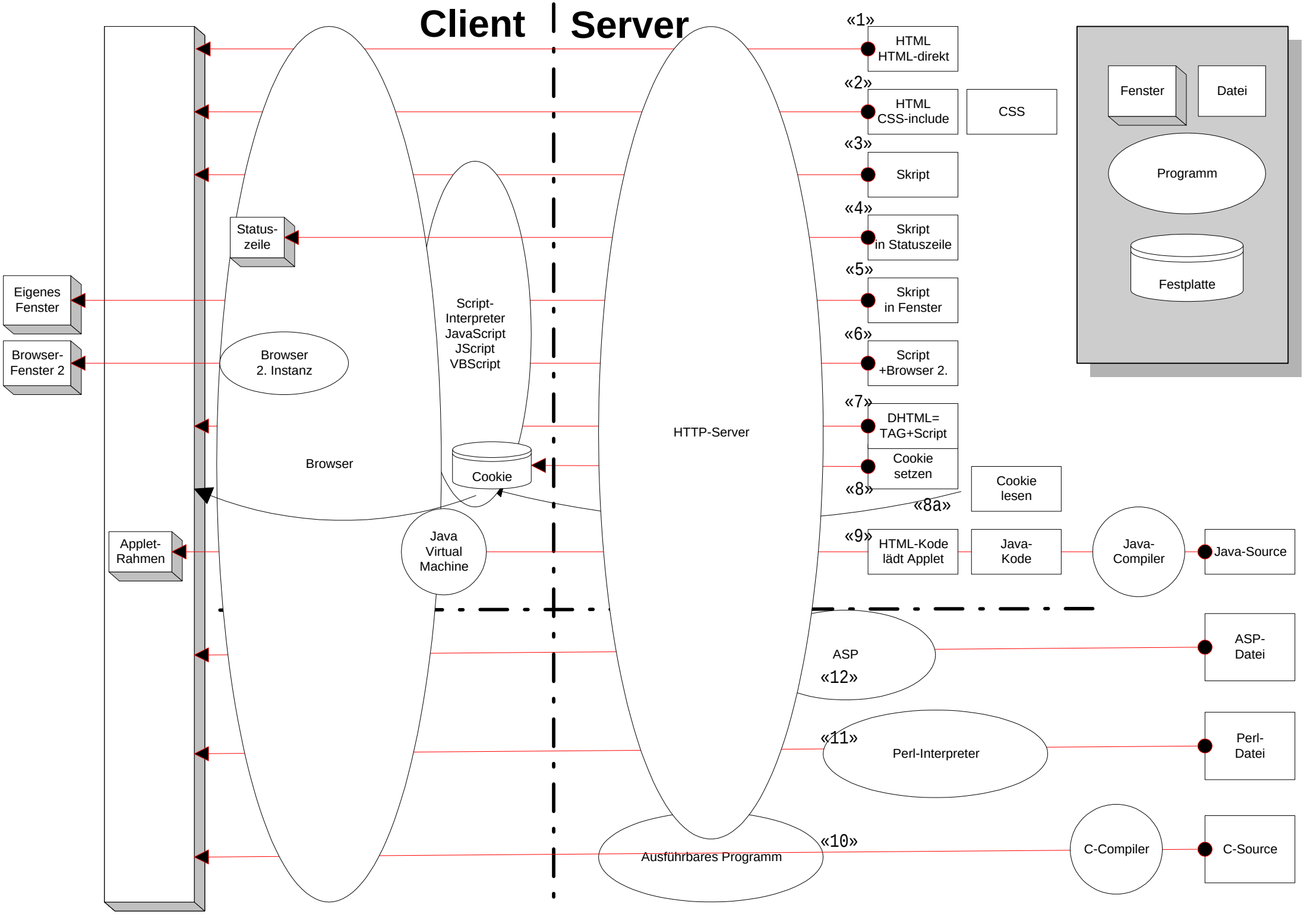
http://dynamicdrive.com/

WebCoder.com - Your home for JavaScript and Dynamic HTML on the Web.

http://webcoder.com/

Stefan Münz: Selfhtml

http://www.netzwelt.com/selfhtml/tf.htm


Client
Server
«1»
HTML
HTML-direkt
«2»
HTML
CSS-include
CSS
«3»
Skript
«4»
Skript
in Statuszeile
«5»
Skript
in Fenster
«6»
Script
+Browser 2.
«7»
DHTML=
TAG+Script
«8»
Cookie
setzen
«8a»
Cookie
lesen
«9»
HTML-Kode
lädt Applet
Java-
Kode
Java-
Compiler
Java-Source
«10»
Ausführbares Programm
C-Compiler
C-Source
«11»
Perl-Interpreter
Perl-
Datei
«12»
ASP
ASP-
Datei
HTTP-Server
Fenster
Datei
Programm
Festplatte
Status-
zeile
Eigenes
Fenster
Browser-
Fenster 2
Browser
2. Instanz
Browser
Script-
Interpreter
JavaScript
JScript
VBScript
Cookie
Java
Virtual
Machine
Applet-
Rahmen

```
  document.write (document.cookie);
</SCRIPT>
```

Das Cookie ist lediglich ein Textstück. Weitergehende Informationen über den Server, das Datum, die Uhrzeit... muss man durch einen vereinbarten Aufbau dieses Textes festlegen. Man findet Beispiele dazu in der Dokumentation von JavaScript 1.1. (http://pcnews.at/edu/tk/javascr/12ref/doc1.htm#1010655)

9. Java-Applet

HTML beschränkt sich auf die Darstellung von Texten. Komplexere Bildausgaben, ein windows-ähnliches Interface erfordert den Ablauf eines clientseitigen Programms. Dieses Programm darf kein ausführbares Programm wie ein EXE-Programm sein, denn sonst würde es ja nur auf einem Windows-Rechner ablaufen können. Sogenannte Java-Applets bestehen daher aus einem Zwischenkode, der durch die *Java Virtual Machine* im Browser ausgeführt wird. Ein Applet verhält sich wie ein Programm, allerdings hat es Beschränkungen hinsichtlich des Zugriffs auf lokale Ressourcen. Das folgende Applet schreibt den gewünschten Text in einen Fensterbereich.

drawtest.java
Kode für JavApplet

```
import java.awt.*;
import java.applet.*;
public class drawtest extends Applet {
  public void paint (Graphics g) {
    g.drawString
      ("Hallo Internetz!", 50, 25);
  }
}
```

Kompiliert man mit dem JDK, Sun oder Visual J++, MS erhält man die Datei `drawtest.class`, die in eine HTML-Datei eingebunden wird:

8.htm
HTML-Kode, der Applet einbindet

```
<applet code=drawtest.class
        width=400 height=400>
</applet>
```

Serverseitige Methoden

Der Normalfall der Kommunikation mit dem Client ist, dass der Server eine HTML-Datei oder ein Bild zum Client schickt. In manchen Fällen ist es aber wünschenswert, dass am Server Daten gespeichert werden oder die HTML-Seiten dynamisch generiert werden. Jedenfalls findet eine Erweiterung der Servereigenschaften durch externe Programme statt.

Grundsätzlich sind dazu alle Programmiersprachen wie C, C++ geeignet.

Erfolgt aber die Referenz zu einem serverseitigen, ausführbaren Programm oder zu einem Skript, dann hängt das weitere Verhalten vom Verzeichnis ab, indem sich das Skript oder das Programm befindet. Handelt es sich um ein Nur-Lese-Verzeichnis, wird die Datei zum Download angeboten, weil das Dateiformat im allgemeinen durch das HTTP-Protokoll nicht exekutierbar ist. Handelt es sich aber um ein Verzeichnis mit Execute-Rechten, wird das Skript oder das Programm ausgeführt.

Der Server macht daher sein Verhalten von der Dateitype und vom Speicherort abhängig. Zum Beispiel kann man besondere Dateiendungen am Server definieren, die den Server veranlassen, die Datei von der Absendung durch den Server nach bestimmten Texten zu untersuchen (Server Side Includes). Der besondere Vorteil dieser Technik ist neben dem Einfügen von Variablen in den Text (Zugriffszähler) auch die Möglichkeit, ein HTML-Dokument in vielen verschiedenen Dokumenten einzubetten und dennoch nur einmal editieren zu müssen.

10Ausführbares Programm

Ein Beispiel, wie der Begrüßungstext auch durch ein C-Programm generiert werden kann, zeigt das folgende Beispiel

```
#include <stdio.h>
void main(void)
{
  printf("Content-Type: text/html\n\n");
  printf("<BODY>");
  printf ("<P>Hallo Internet</P>");
  printf ("</BODY>");
}
```

Das Programm muss mit einem 32-Bit-Compiler generiert werden.

Die aufrufende HTML-Datei enthält eine Referenz, die die EXE-Datei referenziert:

```
<A HREF="/scripts/hallo.exe">Hallo</A>
```

Alle ausführbaren Programme werden in eigenen Verzeichnissen abgelegt (hier `/scripts`), wo Rechte zur Ausführung bestehen (ist normalerweise im HTML-Verzeichnis nicht der Fall).

Der Nachteil dieser Methode ist, dass man in einem verhältnismäßig aufwendigen Compiliervorgang eine EXE-Datei generieren muss. Außerdem sind die Standard-Hochsprachen nicht auf die Bedürfnisse der Web-Programmierung zugeschnitten. Der Vorteil ist, dass diese Programme schnell ablaufen.

11 PERL-Skript

Speziell für die Bedürfnisse der Internet-Anwendungen optimiert ist die Skriptsprache PERL, die auf praktisch allen Plattformen existiert. Hier entfällt das Kompilieren, doch ist das Erlernen einer sehr mächtigen aber auch kryptischen Programmiersprache angesagt. Der Text wird wie folgt zurückgeschickt.

```
print STDOUT
  ('Content-Type: text/html'. "\n\n");
print STDOUT ("<BODY>");
print STDOUT ("<P>Hallo Internet</P>");
print STDOUT ("</BODY>");
```

Die Verwandschaft zu C ist groß, allerdings auch die Unterschiede.

Referenziert wird die Datei durch

```
<P><A HREF="/scripts/hallo.pl">Hallo</A></P>
```

In UNIX-Systemen muss in der Perl-Datei in der ersten Zeile ein Hinweis über den Pfad des Perl-Interpreters stehen. In Windows-NT wird Perl über eine Pfadangabe gefunden, die außerhalb der Datei in `CONFIG.SYS` spezifiziert wird.

12 ASP-Skript

Active Server Pages sind eine Microsoft-Technologie. Es ist eigentlich keine neue Skriptsprache, sondern die Möglichkeit, den HMTL-Kode durch beliebige Skriptsprachen erweitern zu können.

Während in Perl alle HTML-Zeilen in ähnlicher Form wie in einem C-Kode mit einem print-Befehl an den Client gesendet werden, bestehen ASP-Dateien grundsätzlich aus HTML-Kode. Lediglich Bereiche die mit `<%...%>` eingerahmt sind, werden durch die Skriptsprache interpretiert, gemäß den Anweisungen ergänzt und werden danach zu einer reinen HTML-Datei.

Im folgenden Beispiel wird serverseitig mit JavaScript ein Datum eingefügt.

```
<%@ LANGUAGE = JScript %>
<% var dtmVar = new Date(); %>
<P>Hallo Internetz am <%=dtmVar.getDate()%>.
<%=dtmVar.getMonth()+1%>.
<%=dtmVar.getYear()%></p>
```