Web-Baukasten

Jobst-Hartmut Lüddecke

23. März 2010

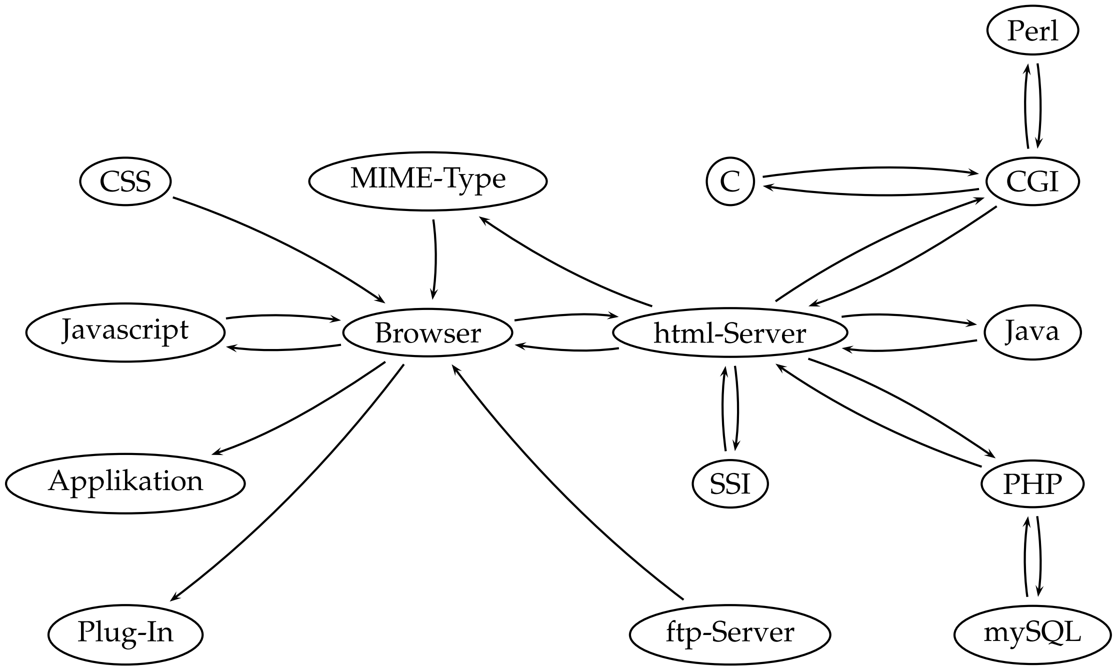
Zusammenfassung

In dieser ersten Lektion des 2. Semesters soll ein gewisser Zusammenhang verschiedener Schnittstellen von HTML zu anderen Produkten aufgezeigt werden, um die Behandlung von *Javascript* zu positionieren und weitere Möglichkeiten zu verdeutlichen. Dies ist ein **Ausschnitt** und erhebt **keinen Anspruch auf Vollständigkeit**.

Inhaltsverzeichnis

1 Überblick

Dass das *World Wide Web* nicht nur mit in der *Hypertext Markup Language* geschriebenen Seiten funktioniert, haben wir bereits mindestens mit den *Cascading Style Sheets* gesehen. Hier nun ein Überblick einiger Bausteine, ohne Anspruch auf Vollständigkeit.



2 Server-Seitig

Sofern der Web-Server entsprechend konfiguriert ist, die entsprechenden Module im Serverprogramm eingebunden sind und für die Anwendung, bzw. den Benutzer freigegeben sind, gibt es folgende Erweiterungen. Die Aktion wird dabei auf dem **Server** ausgeführt.

2.1 Server Side Includes *(SSI)*

Nützliche Anweisungen für den *http*-Server (*Hypertext Transfer Protocol*), die innerhalb einer *html*-Seite stehen können. Damit kann man z.B. die *Größe der Datei* und den *Zeitstempel der letzten Modifikation* von Dateien im Web-Angebot ermitteln und ausgeben. Das ist sehr benutzerfreundlich bei Dateien mit großer Datenmenge, da man dann als Benutzer entscheiden kann, worauf man sich einlässt und z.B. wirklich die nächsten 2 Stunden mit seinem Telefonanschluss *Daten saugen* betreibt, oder die Datei doch nicht so dringend braucht. Auch mit einem DSL-Anschuss können Dateien mit so um die 60 Megabyte einen Anschluss einige Zeit blockieren. Weiterhin kann man mit *SSI* Datums- und Zeitangaben formatieren und ausgeben.

Allen **Server-Side-Includes** gemeinsam ist: Sie fangen mit **<!--#** an und enden mit **-->**.

2.1.1 Textmacro (Textbaustein)

Die allgemeine Form dieses Befehls lautet:

<!- -#include virtual="/pfad/textmacro.html"- ->

Dieses *Textmacro* ist nun ein *Textbaustein* (inklusive Pfad), der bei Abruf der HTML-Seite automatisch an der betreffenden Stelle eingefügt wird. Dies kann z.B. sehr nützlich bei gleich bleibenden Seiten-Füßen mit redaktionellen Angaben des Anbieters der Seite sein.

2.1.2 Definition des Zeitformates

Mit dem Server-Side-Include *config timefmt* kann man das Zeitformat für Zeitabfragen wie z.B. nach dem *Modifikationsdatum* konfigurieren. Gültige Parameter sind:

%a Wochentag in Englisch (Kurzform),

%A Wochentag in Englisch (Langform)

%d Monatstag (immer 2-stellig),

%e Monatstag (1- oder 2-stellig),

%b Monatsname (Kurzform),

%B Monatsname (Langform),

%m Monat als Zahl

%y Jahreszahl (2-stellig),

%Y Jahreszahl (4-stellig),

%H Stunde (24-stündig),

%I Stunde (12-stündig),

%p Tageszeit: AM/PM bei Verwendung der 12-stündigen Uhrzeit

%M Minuten (2-stellig),

%S Sekunden (2-stellig),

%j julianisches Datum,

%U Kalenderwoche,

%w Tag der Woche als Zahl: 1-7,

%Z Zeitzone.

Hier nun die Definitionen für eine Ausgabe des Datums, wie sie Deutschland üblich ist:

<!- -#config timefmt"%d. %b. %Y"- ->

oder

<!- -#config timefmt"%e. %m. %y"- ->

2.1.3 Abfrage des Modifikationsdatums der aktuellen html-Seite

Das folgende Server-Side-Include bietet sich an, um den geforderten Stand der Seite automatisch zu aktualisieren. Baut man den Befehl in einer html-Seite ein, wird immer das Datum der letzten Bearbeitung der Seite in der Form der vorher definierten Datumsausgabe ausgegeben.

<!- -#echo var="LAST_MODIFIED"- ->

Kleine Falle! Diesen Befehl sollte man nicht in einem Textmacro verwenden, denn dann wird das Modifikationsdatum des Textmacros und nicht das Modifikationsdatum der Seite, in die das Macro eingesetzt wird, ausgegeben.

2.1.4 Abfrage des Modifikationsdatums einer andern Datei

Man kann nicht nur das Modifikationsdatum der aktuellen Seite, sondern das Modifikationsdatum jeder Datei im Zugriff des WWW-Servers abfragen und ausgeben. Dies ist sehr nützlich um die Aktualität einer mit einem Link versehenen Datei zu kennzeichnen.

<!- -#flastmod virtual"/pfad/datei.extension"- ->

2.1.5 Abfrage der Größe einer andern Datei

Noch mehr als das Modifikationsdatum einer mit einem Link versehenen Datei, ist die Größe dieser Datei von Interesse. Mit dieser Angabe kann man sofort abschätzen ob man diese Datei (in der Regel ein Programm-Paket, ein Bild, ein Video- oder Ton-Dokument) wirklich jetzt und überhaupt herunterladen möchte. Damit kann man z.B. bei schlechten Leitungsverbindungen (z.B. Stau-Zeiten um die Mittagszeit) Abbrüche durch Benutzer (oder Time-outs) vermeiden und Übertragungen von großen Dateien auf freiere Zeiten verlegen (freie Bahn hat man morgens zwischen 5 und 7 Uhr).

<!- -#fsize virtual"/pfad/datei.extension"- ->

2.2 Common Gateway Interface *(CGI)*

Ein *Interface* zwischen einer *html*-Seite und einem Programm. Es ist also ein Programm mit einem Übergabemechanismus, mit dem man Daten aus einem *html*-Formular strukturiert an ein anderes Programm zur Weiterverarbeitung durchreichen und die Ergebnisse dynamisch wieder an die *html*-Seite zurück geben kann. Dabei ist es unerheblich in welcher Programmiersprache das Programm zur Weiterverarbeitung geschrieben ist. Weit verbreitet sind Programme in *Perl* und *C*.

Da über das *Common Gateway Interface* ein recht ungehinderter Zugriff unbekannter, externer Personen auf Programme des Web-Servers möglich sind, wird aus Gründen der Sicherheit dieser Mechanismus ungern verwendet und verliert dadurch an Bedeutung.

2.3 Java

Ein von der Hardware unabhängiger objektorientierter Programm-Interpreter der Firma *Sun Microsystems*, der auf der *Java Virtual Machine* ausgeführt wird. Ähnlich der Programmiersprache *C* braucht es damit nur einen kleinen, prozessorabhängigen und betriebssystemabhängigen Teil um diese Programme ausführen zu können.[1] Damit ist das in Java geschriebene Programm völlig unabhängig von der Plattform auf der die *Java-VM* läuft. Man erreicht damit also ein Maximum an Kompatibilität.

Java wurde im Auftrag von *Sun Microsystems* im Zeitraum von 18 Monaten entwickelt und im Sommer 1992 fertig gestellt. Die maßgeblichen Entwickler sind: *James Gosling, Patrick Naughton, Mike Sheridan und Bill Joy.*

Der Name geht auf die von den Entwicklern bevorzugte Verköstigung mit *Java-Kaffee* zurück. Dies ist kein neuer Einfall, denn *Apple-Computer* ist ja auch aus *Steven Jobbs* Apfel-Diät hervorgegangen.

Der Durchbruch für *Java* kam 1995 mit der Kooperation von *Sun Microsystems* und *Netscape Communications* und der daraus folgenden Integration der *Java-VM* in den *Netscape Navigator* und der Ankopplung von *Javascript*.

Kleine Anekdote: Der Kooperationsvertrag wurde morgens um 4 Uhr in einem Hotelzimmer per Handschlag zwischen *Eric Schmidt* und *George Poalini* von *Sun Microsystems* und *Mark Andreesen* von *Netscape Communications* abgeschlossen.[2]

Nach diesem Durchbruch im *Web* war *Java* der *Hype* in der Software-Technologie und in der Fachpresse. Dies hat sich aber wieder beruhigt, nachdem klar wurde, dass der *Kaffee* auch nur mit *Wasser gekocht* wird und einiges an Ressourcen kostet um auf verschiedenen Plattformen gleich zu funktionieren. Die Stärke liegt in der Verbreitung und Kompatibilität[3], nicht in der Geschwindigkeit.

2.4 *PHP*

Für die Abkürzung *PHP* gibt es keine eindeutige Auflösung. Es geht von *Personal Home Page* bis zu *Perl Hypertext Preprocessor*. Auf jeden Fall ist es ein *Interface* zwischen einer *html*-Seite und einer Datenbank, wie z.B. *mySQL*. Über *PHP* ist es somit möglich eine definierte Datenbankabfrage mit in einem Formular empfangenen Parametern (Suchbegriffen) zu starten und das Ergebnis der Recherche dynamisch an die *html*-Seite zur Ausgabe zurückzugeben.

PHP wurde ursprünglich von *Rasmus Lerdorf* als ein *Werkzeugkasten* – in *Perl* geschrieben – 1995 entwickelt. Dann noch einmal von ihm komplett in *C* neu geschrieben.

PHP3 wurde dann erneut komplett von *Andi Gutmans* und *Zeev Suraski* in Kooperation mit *Rasmus Lerdorf* neu geschrieben um Bedürfnisse des *e-commerce* und der Sicherheit zu befriedigen.

PHP4 wird nun von der von *Andi Gutmans* und *Zeev Suraski* gegründeten Firma *Zend Technologies Ltd* gegründeten Firma weiterentwickelt und enthält als Kern die mehrfach geehrte *Zend Engine*.[4]

PHP5 bekam die neue *Zend Engine II*, Sicherheits- und Stabilitätsupdates und Interfaces zu *MySQLi, SQLite, SimpleXML*, eine *DOM*-Unterstützung und mit *JSON* ein Zusammenspiel mit *Ajax*. PHP5 enthält ab dieser Version auch *ZIP* um solche Archivdateien lesen und erzeugen zu können.

[1] Die *Java-VM* ist auch selbst in *C++* geschrieben worden.

[2] siehe auch Wikipedia: *Java Technologie*

[3] Die Programme laufen auf unterschiedlichen Rechnern mit unterschiedlichen Prozessoren und unterschiedlichen Betriebssystemen weitgehend gleich

[4] siehe auch Wikipedia: *PHP*

PHP6 (aktuelle Entwicklung) kann jetzt *Unicode* und hat eine bessere Unterstützung der *Mime-Types* und erfährt erneut eine *Durchforstung*.

Kritikpunkt an PHP : Teilweiser *Wildwuchs* der Funktionen mit mehreren Funktionen unterschiedlichen Namens und bedingt gleicher Funktion. Dies hatte dann mehrere *Durchforstungen* und damit eine nicht kompatible Änderung des Sprachumfangs zur Folge. Damit sind die einzelnen Hauptversionen von *PHP* nicht mehr kompatibel zueinander. Ein Update so einer PHP-Version hat dann zwangsläufig eine Überarbeitung der in *PHP* geschriebenen Programme zur Folge.

3 Client-Seitig

Sofern die Aktion auf dem Client freigegeben ist, wird die Aktion auf dem **Client**-Rechner ausgeführt.

3.1 Hypertext Markup Language *(html)*

Seitenbeschreibungssprache, die *interpretativ* von einem Browserprogramm auf dem Client zur Ausgabe umgesetzt wird. Dabei können unterschiedliche *Browser* durchaus abweichende Ergebnisse liefern. Der Sprachumfang der jeweiligen *html*-Version ist zwar eindeutig vom *W3-Consortium* festgelegt worden, aber einerseits nicht vollständig und eindeutig von den Browser-Herstellern umgesetzt, dafür auf der anderen Seite mit vom Hersteller abhängigen Erweiterungen ergänzt worden. Hier ist darauf zu achten, keinen *proprietären Code*, der nur mit einem bestimmten Browser funktioniert, zu erzeugen, wie es einige *html-Generatoren* der entsprechenden Hersteller mit klarer Absicht tun!

3.2 Cascading Style Sheets *(CSS)*

Wie schon gesehen, eine Möglichkeit das *Erscheinungsbild* der einzelnen *html-Tags* verbindlich festzulegen, sowie die Möglichkeit eigene *Klassen* und eigene *Tags* zu definieren.

3.3 Javascript

Vom Namen her ein *Trittbrettfahrer*. Diese Entwicklung der ehemaligen Firma *Netscape Communications Corporations*[5] sollte eigentlich *LiveScript* heißen und wurde 1995 von *Brendan Eich* entwickelt.

Im Dezember 1995 verkündeten *Netscape* und *Sun* eine Kooperation aus der dann *LiveConnect* und damit eine Schnittstelle zwischen *LiveScript*, *Java* und *HTML* und damit ein Zusammenwirken im *Netscape Navigator* entwickelt wurde.

Danach wurde *LiveScript* in *Javascript* umgetauft.

Im Juni 1997 wurde für Javascript ein ECMA-Standard (ECMA-262) formuliert und dieser im April 1998 zur ISO-Norm (ISO-IEC 16262:1998). Seit Juni 2002 wird Javascript im Rahmen des Mozilla-Projektes weiterentwickelt und liegt in der aktuellen Version von *Mozilla Firefox 3.6 - Februar 2010* in der Version 1.9.2 mit Teilen von *Gecko 20100115* vor.

[5] Wurde als *spin-off* des *NCSA-Mozilla*-Projektes am 4. April 1994 von *Marc Andreesen* gegründet. Der *Netscape Navigator* wurde dann sehr schnell zum weltweit führenden Web-Browser und konnte diese Position bis ca. zum Jahr 2000 mit über 50% Marktanteil halten. Die Firma wurde 1998 von *AOL* gekauft, *AOL* wiederum von *Warner Bros.* und im Browser-Krieg mit *Microsoft* ging es stetig bergab. Ende 2007 ist dann *AOL* aus der Browser-Entwicklung ausgestiegen. Teile wurden von *HP* gekauft.

Seit Oktober 1997 wird *Javascript* in der Version 1.1 vom *Internet Explorer 4* der Firma *Microsoft* verstanden und verarbeitet. Welche Version von *Javascript* vom aktuellen *Internet Explorer 8* unterstützt wird, konnte ich noch nicht mit Sicherheit herausfinden.

Ansonsten hat *Javascript* mit *Java* so viel zu tun, dass sich die Beiden sehr gut ergänzen und über *LiveConnect* zusammen arbeiten können. *JavaScript* kann auf dem Client das Verhalten des Browsers steuern und den Inhalt, bzw. Eingaben kontrollieren. *JavaScript* kann keine Grafiken zeichnen, hat keine Netzwerk-Funktionen und keine Kontrolle über den Server. *Java* hingegen hat keinen Einfluss auf den Browser, beherrscht hingegen Grafik, Netzwerk, Multithreading und kann Einfluss auf den Server über *Java Applets* und *NetBeans* nehmen.

Wie wir noch sehen werden kann man mit *Javascript* wunderbar **Eingabeformulare** erstellen, deren Eingaben *syntaktisch*[6] geprüft werden, bevor sie an den Server zwecks inhaltlicher Weiterverarbeitung übergeben werden. Damit kann man schon einmal die gröbsten Eingabefehler am Client abfangen.

4 MIME-Types

Multipurpose Internet Mail Extensions sind international genormte Datentypen mit in der Norm festgelegten Extensionen. Eine Übersicht der Normen findet man unter **http://www.iana.org**.

Es macht hier **überhaupt keinen Sinn** sich selbst irgendwelche Extensionen auszudenken und dann zu hoffen, dass es mit den *Applikationen* und dem *MIME-Type Mechanismus* klappt.[7]

Dieser Mechanismus ist notwendig um andere Datentypen als HTML-Dateien transportieren zu können. Transportieren bedeutet in diesem Fall im World Wide Web lesbar machen, bzw. als Anlage einer e-mail verschicken zu können.

Dieser Mechanismus funktioniert nur, wenn diese Mime-Types sowohl beim **Sender**[8], als auch beim **Empfänger**[9] bekannt sind[10] und weiter verarbeitet werden können[11].

Der Umkehrschluss: Um die mögliche Verbreitung der eigenen Seiten soweit wie möglich sicherzustellen sollte man nur stark verbreitete Datentypen verwenden. Gleiches gilt für das Versenden von Anlagen von e-mails. Gegebenenfalls muss man sich vorher beim Empfänger versichern, ob er diesen Datentyp als *Mime-Type* eingetragen hat und weiterverarbeiten kann.

Beispiel anhand einer Postscript Datei. Der Server überprüft die Extension der Datei und identifiziert **ps**, also eine Postscript Datei[12]. Dann wird in die Datei nach bestimmten Merkmalen geschaut.[13] Ist der Fund erfolgreich, teilt der Server nun dem Client mit: *Es kommt eine Datei des Types* **application/postscript** und wählt eine entsprechende Übertragungsform, wie z.B. **7-Bit ASCII**, oder **8-Bit binär** aus.

Wenn jetzt auf dem Client für diesen Typ ein Programm eingetragen ist, welches diesen Typ weiterverarbeiten kann (z.B. Ghostview mit Ghostscript), dann wird diese Applikation oder das Plug-in gestartet und die empfangene Datei an dieses Programm zur Weiterverarbeitung übergeben.

Ist der *Mime-Type*, aber keine Applikation beim Client eingetragen, dann kann man noch versuchen die Datei abzuspeichern.[14]

[6] Ob sie den formalen Vorgaben entsprechen

[7] Alles schon erlebt! Auch Professoren technischer Fakultäten sind nicht davor gefeit!

[8] hier der Server

[9] hier der Browser

[10] In der Datei **mime.types** eingetragen sind

[11] Welche Aktion bei diesem Datentyp erfolgen soll

[12] Zulässig sind dafür nur **ai**, **eps** und **ps**

[13] Dies kann mit mehr oder weniger Aufwand, je nach Datentyp passieren und wird in der Norm geregelt. Bei Postscript bietet sich z.B. die *Header*-Zeile an, in der auch das *Postscript-Level* steht. Alleine darüber gibt es ganze Buchserien.

[14] Hoffnungsweise ist wenigstens eine brauchbare Übertragungsform verwendet worden.

Ist der Mime-Type nicht eingetragen, dann hat man Übertragungskapazität und ggf. Telefonkosten nutzlos verschwendet, da man nur nicht weiter nutzbaren *Daten-Schrott* erhalten hat.[15]

Aus leidvoller Erfahrung kann ich berichten, dass es z.B. reinen Windows-Anwendern aus der Hochschul-Verwaltung schwer klar zu machen ist, dass man mit Microsoft doc-Dateien als Mail-Anhängsel auf Unix-Maschinen wenig anfangen kann, weil der MIME-Typ und die Applikation *Microsoft Word* fehlt. Zumindest *frisst* die aktuelle und für unterschiedliche Unix-Systeme verfügbare Version von *OpenOffice* auch diese Datentypen.

Um sich ein Bild zu machen, welche *Mime-Types* beim Browser bekannt und damit gültig sind, kann man z.B. bei *Mozilla* aus das Pull-Down-Menü

Extras → **Einstellungen** → **Anwenungen**

gehen

4.1 Applikationen

Eine Applikation ist ein eigenständiges Programm zur Weiterverarbeitung bestimmter Datentypen. Bei einem funktionierenden MIME-Type Mechanismus wird diese Applikation gestartet und die empfangenen Daten an dieses Programm übergeben.

So kann z.B. bei einer **ps**-Datei *Ghostview mit Ghostscript*, bei einer **pdf**-Datei der *Acrobat-Reader* und bei einer **doc**-Datei *Microsoft Word* anspringen und die Datei lesbar machen.

4.2 Plug-Ins

Ein *Plug-In* ist eine Erweiterung der Funktionalität des Browsers durch ein *eingestöpseltes* Programm. Der Mechanismus der MIME-Types funktioniert genauso wie bei den Applikationen, es wird nur nicht ein eigenständiges Programm gestartet, sondern das *Plug-In* ermöglicht es dem Browser diesen Datentyp zusätzlich zu verarbeiten.

Zum Beispiel gibt es den *Acrobat Reader* als Applikation und als Plug-In. Je nachdem, welches Verhalten bei den *MIME-Types* eingetragen ist, wird beim Empfang einer **pdf**-Datei entweder der *Acrobat Reader* gestartet, oder das *Plug-In* ermöglicht es dem Browser diese **pdf**-Datei im Browserfenster anzuzeigen. Das Ergebnis diese **pdf**-Datei lesen zu können ist gleich.

[15]Im 7-Bit ASCII Mode übertragene Bild-Dateien und Ton-Dokumente machen sich besonders gut!