Adressierung, Ankerpunkte und einfache Navigation

Jobst-Hartmut Lüddecke

24. Oktober 2011

Zusammenfassung

In dieser Lektion geht es um die *Adressierung*. Diese ist *relativ* und *absolut* möglich und sieht unter *Windows* etwas anders aus, als unter *Unix*. Dann kommen wir zur *Adressierung* in *html*. Diese orientiert sie an der *Unix-Syntax* und hat noch einige Erweiterungen. Am Ende soll in einer *html-Seite* sicher ein *relativer*, als auch ein *absoluter Ankerpunkt* (*link*) gesetzt werden können.

Inhaltsverzeichnis

Abbildungsverzeichnis

1 Dateiname

1.1 Name

Der Dateiname, also die Bezeichnungen von Dateien (files) bestehen aus zwei Teilen. Einmal dem **Namen**, der nur *ASCII-Zeichen* und davon nur **Buchstaben, Ziffern, Bindestriche und Unterstriche** enthalten darf. Dabei ist für *Unix* ein großer Buchstabe selbstverständlich ein **anderes Zeichen**, als ein kleiner Buchstabe. *DOS* kennt nur große Buchstaben und kleine Buchstaben werden als große Buchstaben interpretiert. *Windows* folgt dieser Sache aus *DOS* im Wesentlichen. Probleme kann es in *heterogenen Netzwerken* machen. In diesen Netzwerken arbeiten z.B. *Windows-Clients* mit *Unix-Servern* z.B. über *Samba* zusammen. Dies ist auch bei uns der Fall und so kann die strengere *Unix*-Auslegung wirken.

1.2 Extension

Der zweite Teil des Dateinamens ist die *Extension*. Sie ist durch einen Punkt vom eigentlichen Namen getrennt. Diese *Extension* ist **international genormt** und kennzeichnet jeweils ein bestimmtes *Datei-Format*. Die *Extensions html* und *css* haben Sie schon kennen gelernt. Andere wären z.B. *pdf* für eine Datei im *portable document format*, dass Sie mit den *Acrobat Reader* lesen könnten. Andere Beispiele sind *ps* für *Postscript*. Die Bezeichnung *doc* ist das alte *Microsoft Word Format*. Das neue von *Word 2007* ist dann *docx*. Dies lässt sich beliebig so weiterführen und man könnte eine eigene Vorlesung daraus machen. Wichtig ist: **Es muss auch drin sein, was drauf steht, sonst funktionieren viele Dinge nicht!**

Dann gibt es wieder Unterschiede in *DOS*, *Windows* und *Unix*. Bei *DOS* können die Namen nur 8 Zeichen und die Extension nur 3 Zeichen lang sein. Daher ist **je nach Serverkonfiguration** auch die Extension *htm* statt *html* für Web-Seiten zulässig. Unsere Server laufen mit *Unix* (hauptsächlich Sun *Solaris 10*, aber auch vereinzelt mit *Linux*) und dort haben Sie nur mit der Extension *html* den vollen Funktionsumfang.

Windows kann wesentlich längere Dateinamen, aber auch nur **einen Punkt**. Leider kann *Windows auch Lücken in Dateinamen*. **Dies sollten Sie aber auf alle Fälle unterlassen!** Die Datensicherung erfolgt unter *Unix* und dort ist eine Lücke ein **Trennsymbol**! Damit wird eine Datei mit einem *Space* im Dateinamen **nicht gefunden**, folglich auch **nicht gesichert** und lässt sich unter *Unix* auch **nicht öffnet, löschen oder anfassen**!

Unix kann mehrere Punkte in einem Dateinamen. Einige Archivierungsprogramme unter *Windows* – bei denen es sich um eine *Portierung* aus *Unix* handelt – können dies auch, aber generell gibt es unter *Windows* Probleme mit Dateinamen die mehrere Punkte enthalten, da Windows nach dem **ersten Punkt** eine **gültige Extension** erwartet.

All dies müssen Sie sich unbedingt vor Augen halten und in *heterogenen Netzwerken* (wie hier) den **kleinsten gemeinsamen Nenner** wählen um die Funktionsfähigkeit sicher zu stellen!!!

2 Adressierung

Jede Struktur eines Verzeichnisses ist bei allen mir bekannten Betriebsystemen eine *Baumstruktur*[1]. Bei *DOS* und *Windows* beginnt die *Wurzel* am *Device*[2] und hat einen **Buchstaben** gefolgt von einem **Doppelpunkt**. Danach werden die einzelnen Ordner-Stufen durch *Backslash* (\) voneinander getrennt.[3]

Bei allen Variationen von *Unix* (*Aix, BSD, Linux, Solaris, System V* usw.) beginnt die Baumstruktur immer beim *Wurzelverzeichnis* (*root*) *Slash* (/). Die einzelnen Ordner-Stufen werden ebenfalls durch ein *Slash* getrennt. Dies ist möglich, weil unter *Unix* eine Platte direkt in die Verzeichnisstruktur eingehängt (*mount*) werden kann. Damit ist die eigentliche Struktur der Platten für den Benutzer nicht sichtbar und er braucht sich nicht darum zu kümmern.

Sowohl in *Windows*, als auch in *Unix* gibt es die speziellen Ordner **.** (Punkt), als auch **..** (Punkt Punkt). Das Verzeichnis **.** bezeichnet den aktuellen Ordner, also das Verzeichnis in dem Sie sich zur Zeit befinden. Der Ordner **..** bezeichnet das übergeordnete Verzeichnis, also eine Stufe dichter am *Wurzelverzeichnis*. Dabei kann man mit **..** nicht höher als bis zum *Wurzelverzeichnis* steigen.[4]

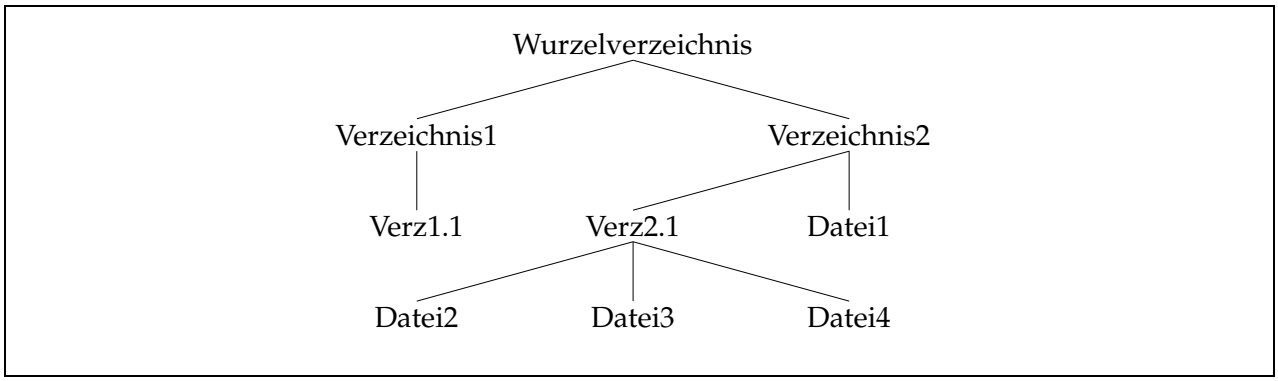

Abbildung 1: Einfacher Verzeichnisbaum

Bitte achten Sie unbedingt darauf, dass die **Namen von Ordnern** genauso, wie die **Namen von Dateien** nur *ASCII-Zeichen* enthalten dürfen und davon auch **keine** Sonderzeichen wie Stern[5], Fragezeichen[6], Anführungszeichen[7], senkrechte Striche[8] usw. Lücken (Spaces, Leertaste usw.)[9] oder Umlaute[10] bewirken auch die bereits oben erklärten Probleme und die folgenden Sachen funktionieren nicht!

[1] Bei *Computerbäumen* sind die Wurzeln immer oben und die Äste verzeigen sich nach unten.

[2] In etwa *Gerät*, also z.B. eine Festplatte, Diskette, CD usw.

[3] Unter DOS werden die *Optionen* eine Befehls mit einem *Slash* (/) gekennzeichnet und *Windows* hat dies geerbt. Unter *UNIX* werden die Optionen mit *Minuszeichen* (–) eingeleitet. Es macht also auch wenig Sinn die Dateinamen mit so einem Strich beginnen zu lassen.

[4] Weitere **..** werden ignoriert.

[5] Der Stern ist ein *Joker* für mehrere, beliebige Zeichen. So kann man mit einem *del *.** alle Dateinamen mit allen Extensionen – also alles – in einem Verzeichnis löschen

[6] Das Fragezeichen aktiviert entweder die Hilfefunktion, oder kann in Dateinamen als Joker für **ein** beliebiges Zeichen verwendet werden.

[7] Kennzeichnet bestimmte Expansionsebenen von Variablen

[8] Senkrechte Striche sind unter *Unix* das *Pipe*-Symbol, mit dem man die Ausgaben eines Prozesses zu Eingaben eines anderen Prozesses machen kann.

[9] Noch einmal: Lücken sind **Trennsymbole**!

[10] Umlaute sind **keine** ASCII-Zeichen!!!

2.1 Relativ zum aktuellen Ordner

Dies ist die einfachste Art der Adressierung, denn man braucht nur den Dateinamen anzugeben. Wenn man ganz sicher gehen will, kann man zusätzlich mit der Verwendung des Verzeichnisses **.** (Punkt) auf genau dieses Verzeichnis verweisen. Dies kann wichtig werden, wenn der *Suchpfad* (*path*) für diese Anwendung unpraktisch eingestellt ist und dadurch zunächst in anderen Ordnern gesucht wird. Beim ersten Fund ist dann die Suche abgeschlossen. Dieser Fund kann dann aber eine andere Datei gleichen Namens in einem anderen Verzeichnis sein, welches durch den Suchpfad vor dem aktuellen Verzeichnis durchsucht worden ist.

Im *Suchpfad* wird die Reihenfolge bestimmt in welchen *Verzeichnissen* (*directories*) die *Datei* (*file*) gesucht werden soll. Dabei gibt es mindestens zwei *Fallgruben* (*traps*). Einerseits kann eine gleichnamige Datei in einem früher durchsuchten *Verzeichnis* gefunden werden, weil das *aktuelle Verzeichnis* (Punkt) erst später im *Suchpfad* aufgeführt ist, oder andererseits wird die Datei gar nicht gefunden, weil das *aktuelle Verzeichnis* (Punkt) im Suchpfad überhaupt nicht mit aufgeführt ist. Solche Dinge sollte man unbedingt beachten, wenn man den *Suchpfad* (*path*) verändert.

Allgemein: `datei.ext`

Windows: `.\datei.ext`

Unix: `./datei.ext`

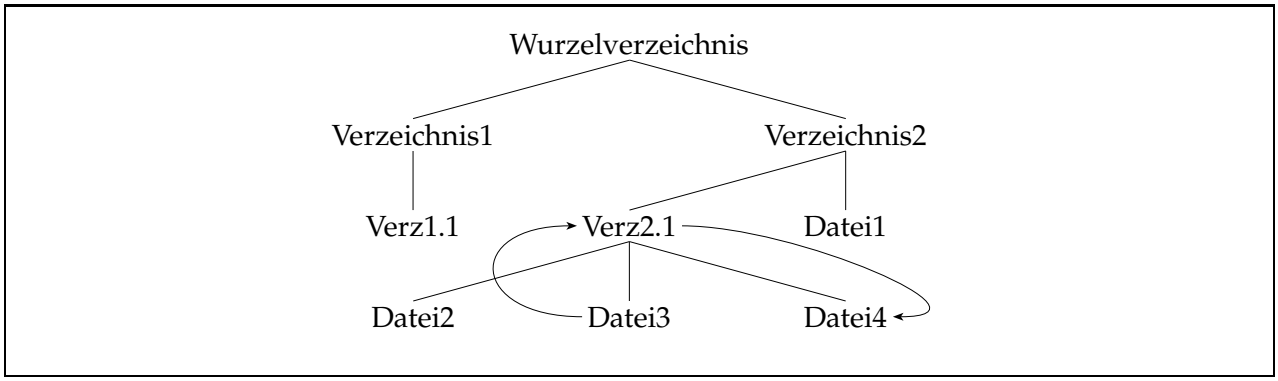

Abbildung 2: Verzeichnisbaum mit relativem Bezug im gleichen Verzeichnis

2.2 Relativ zum übergeordneten Ordner

Dies geht ganz einfach mit der Verwendung des Verzeichnisses **..** . Wie o.a. ist beim *Wurzelverzeichnis* das Ende erreicht, denn darüber gibt es kein *übergeordnetes Verzeichnis* mehr.

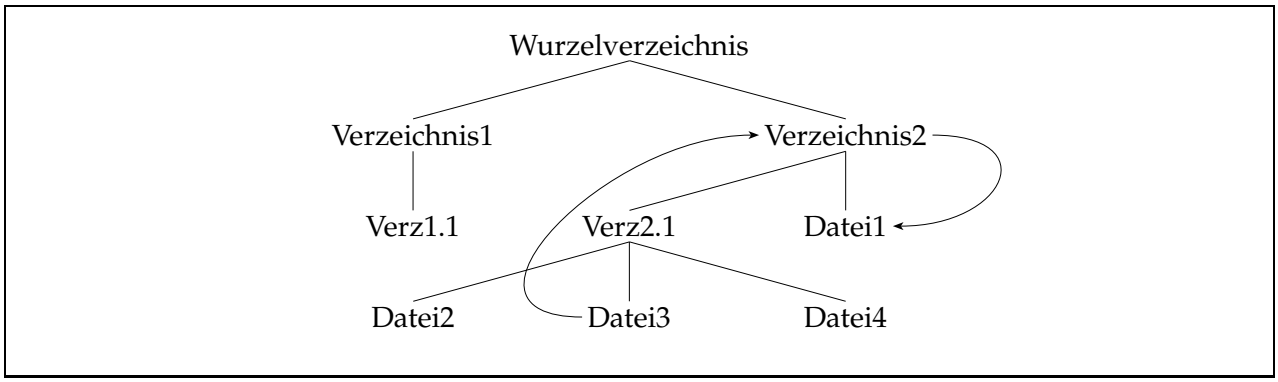

Abbildung 3: Verzeichnisbaum mit relativem Bezug zum übergeordneten Verzeichnis

Windows: `..\datei.ext`

Unix: `../datei.ext`

2.3 Relativ zu einem untergeordneten Ordner

Hier wird lediglich der Names des untergeordneten Verzeichnisses eingefügt.

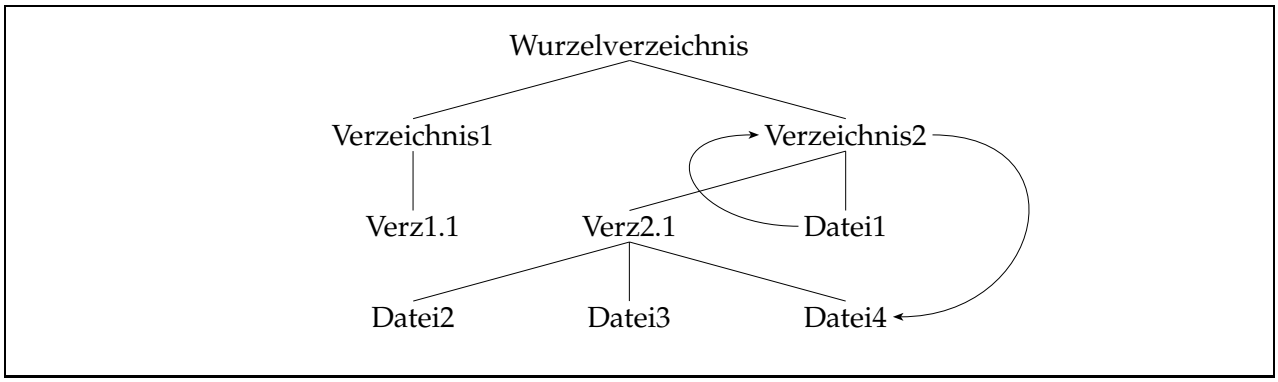

Abbildung 4: Verzeichnisbaum mit relativem Bezug zum untergeordneten Verzeichnis

Windows: `unterverzeichnis\datei.ext`

Unix: `unterverzeichnis/datei.ext`

2.4 Relativ zu einem parallelen Ordner

Dies ist eine Kombination aus den letzten beiden Adressierungsarten. Man begibt sich mit **..** einen Schritt höher und adressiert von dort aus das untergeordnete Verzeichnis.

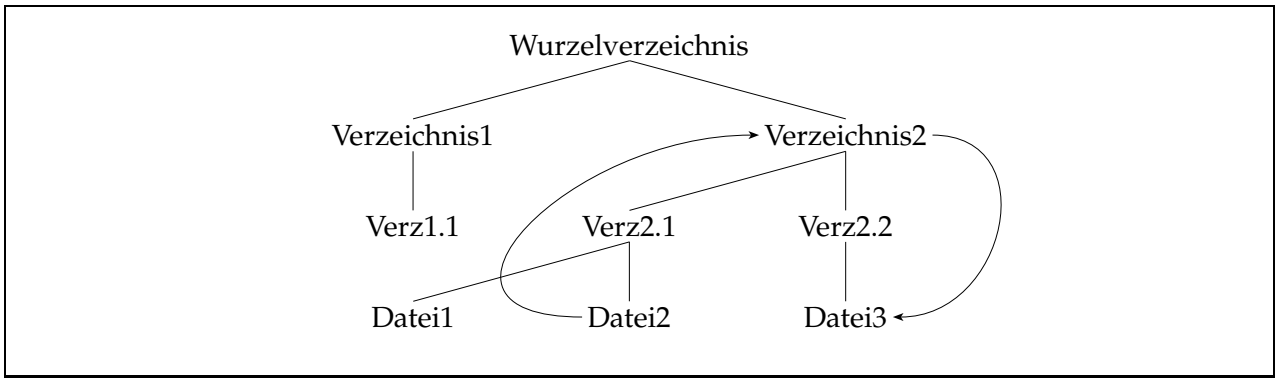

Abbildung 5: Verzeichnisbaum mit relativem Bezug zum parallelen Verzeichnis

Windows: `..\parallelverzeichnis\datei.ext`

Unix: `../parallelverzeichnis/datei.ext`

2.5 Absolut

Hier wird jeweils der vollständige Pfad ab dem Wurzelverzeichnis angegeben:

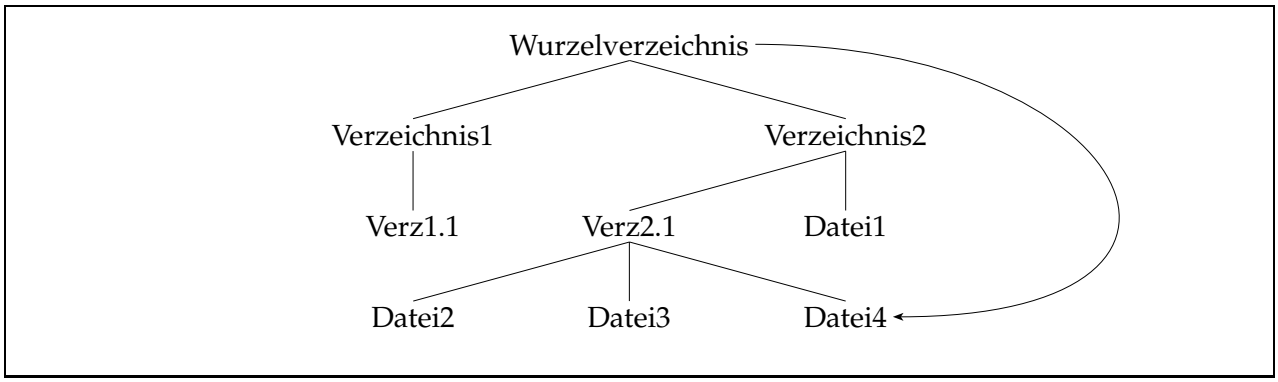

Abbildung 6: Verzeichnisbaum mit absolutem Bezug

Windows: `h:\verzeichnis\unterverzeichnis\datei.ext`

Unix: `/verzeichnis/unterverzeichnis/datei.ext`

3 Ankerpunkte

Die Ankerpunkte sind die Links innerhalb einer html-Seite auf andere Dateien (in der Regel andere html-Seiten). Sie bestehen aus dem **a-Tag** und einer **Ortsangabe**, sowie einem **Bezeichner**, der als *klickbar* gekennzeichnet wird.

Vergessen Sie auf keinen Fall den a-Tag wieder zu **schließen**, sonst ist der Rest der Seite der Bezeichner und damit *klickbar*. Weitere Ankerpunkte funktionieren dann nicht mehr.

```
<a href="Ortsangabe">Bezeichner</a>
```

3.1 Relativ

Dies ist wieder die einfache Version. *Ortsangabe* folgt der Unix-Syntax und bezieht sich auf das *aktuelle Zugriffsprotokoll*, dem *aktuellen Server* und dem *aktuellen Verzeichnis*. Dies entspricht also der *relativen Adressierung im aktuellen Ordner*. Die anderen Formen der *relativen Adressierung* lassen sich leicht herleiten und sind übertragbar.

3.2 Absolut

Hier muss für **Ortsangabe** alles genau spezifiziert werden um einen gültigen *Uniform Resource Locator* (*URL*) zu erhalten. Als erstes kommt das *Zugriffsprotokoll*. Handelt es sich um einen *lokalen Zugriff* auf die eigene Festplatte, dann ist das *Zugriffsprotokoll* ein *file*. Bei einer Webseite auf einem Web-Server wird das *Hypertext transport protocol* verwendet und somit kommt *http* zum Zuge. Gefolgt wird das *Zugriffsprotokoll* von einem *Doppelpunkt* und zwei *Slashes*.

Nach dem *Zugriffsprotokoll* mit *Doppelpunkt* und den *Slashes* kommt der *Servername* in der gängigen Syntax von *TCP/IP*, dem *Internet*. Beim *file*-Zugriff auf einem *Windows*-Rechner kommt statt des *Servernamens* ein dritter *Slash* und das *Device*.

Nach dem *Servernamen* kommt die *absolute Adresse* der Datei in der *Unix*-Syntax. Damit wird jetzt hoffentlich klar, warum Sie dies heute verstehen mussten ☺.

Beispiel:

Home

Dabei ist *http:* das Zugriffsprotokoll *hypertext transfer protocol*, der Server ist *//www.bui.haw-hamburg.de* und der vollständige Pfad ist */pers/jobst-hartmut.lueddecke/index.html*. Dann kommt *Home* als klickbarer *Ankerpunkt* und abgeschlossen wird dies mit dem **.