Interaktives HTML

Gemeint ist hier, daß der Webserver nicht nur wie bei simplen Webseiten einfach durch Schicken von HTML-Code als Response antwortet, sondern daß am Client erhobene Daten zum Server geschickt und verarbeitet werden. Das ist die technische Grundlage von e-commerce. Hier sind prinzipiell 2 Varianten möglich :

a) Client-sided

Im Clientsystem werden die (meist vom Benutzer eingegeben) Daten direkt lokal verarbeitet. Der Programmcode steckt in der HTML-Seite, die vorher vom Server geholt wurde. Sprache : Java oder Varianten.

b) Server-sided

Am Client werden Daten zwar erfasst (meist im Browser durch Eingabefelder o.Ä),dann aber nicht lokal verarbeitet, sondern an den Server geschickt, und dort verarbeitet. Sprache : früher PERL, heute meist PHP.

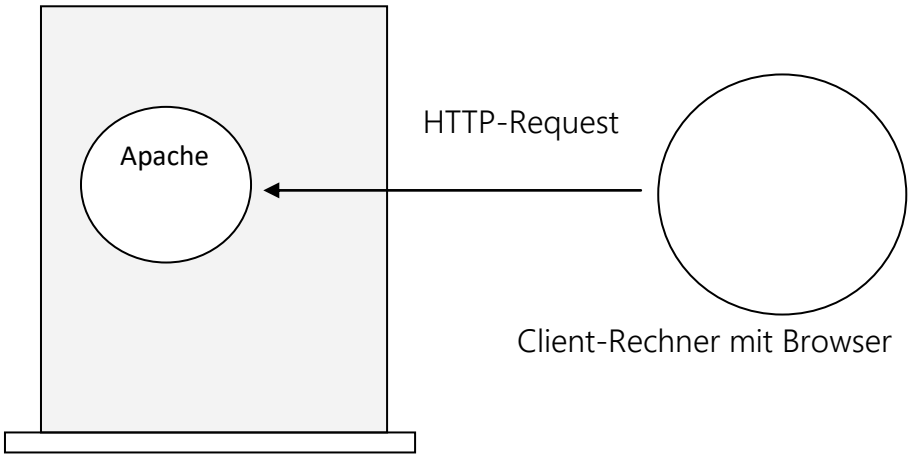
Variante a) ist schneller, hat aber den Nachteil, daß eine ungünstige Clientkonfiguration die Verarbeitung behindern oder unmöglich machen kann (und damit den kommerziellen Effekt verhindert).

Variante b) ist wegen der nötigen Übertragung langsamer, aber der Programmierer weiß, wie seine Maschine konfiguriert ist, und es läuft sicher …

Bei e-commerce wird man alle Programmfunktionen, die dem Komfort und dem Design dienen, client-sided ausführen, alle Funktionen die zum Gelingen des Geschäftsprozesses unabdingbar sind, server-sided.

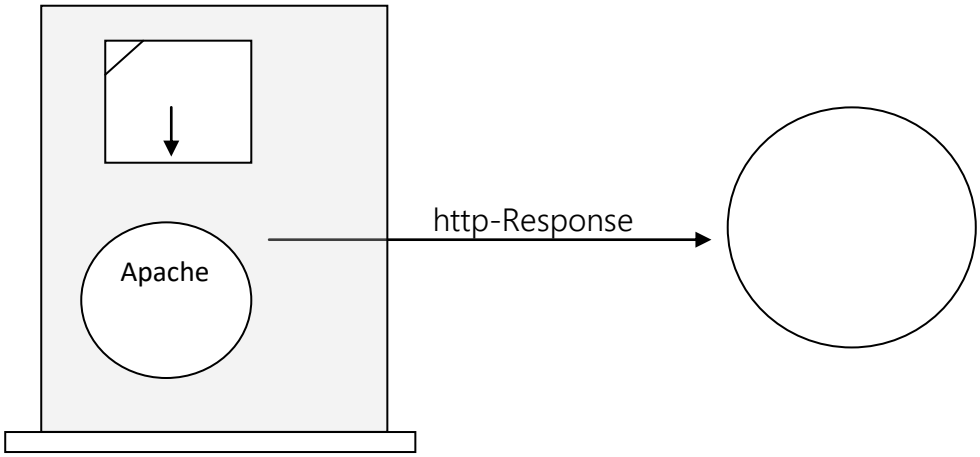
Wir gehen nun im Folgenden davon aus, daß ein Client eine Webseite anfordert, die dann einen wie oben beschriebenen Vorgang server-sided ermöglichen soll.

1) Client schickt (vom Benutzer ausgelöst) einen http-Request :



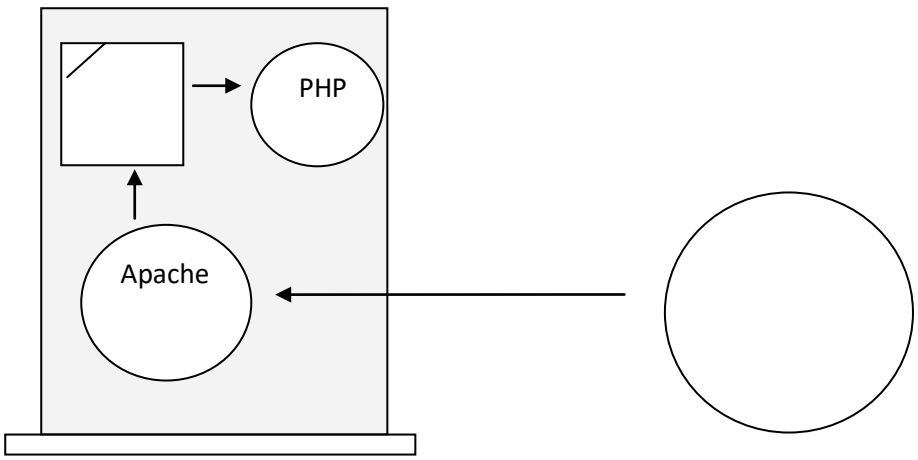
Server-Rechner mit Apache-Installation

2) Der Server sucht das Dokument im angegebenen Pfad (unter Document-Root), und schickt es an den Client. In diesem HTML-File sind graphische Elemente enthalten, die dem Benutzer die Eingabe von Daten ermöglichen (Textfelder usw..) : html-forms



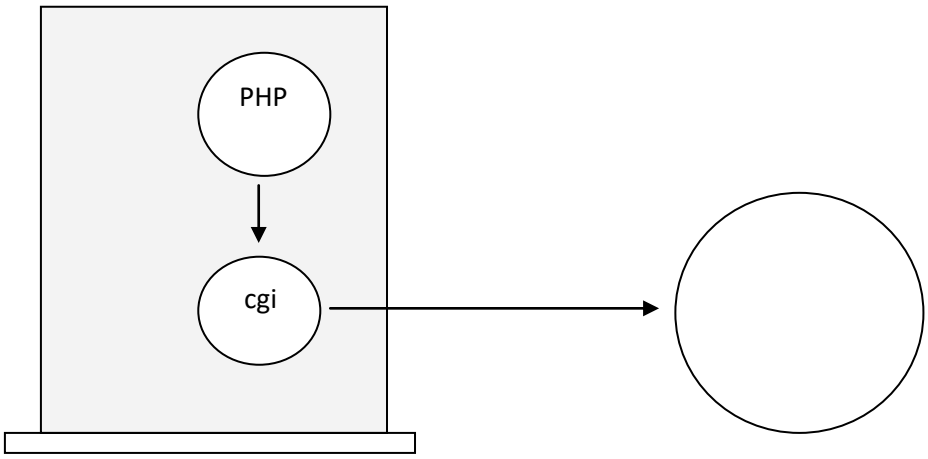
3) Der Benutzer schreibt seine Daten rein, und schickt sie mit einem Knopfdruck im Browser wieder an den Server.

Im HTML-Form ist auch eine Information enthalten, welches Programm im Server nun diese Daten bekommen und verarbeiten soll : form-action. Dies ist wieder eine HTML- Seite mit der Besonderheit, daß ausführbarer Code (meist PHP) enthalten ist. Apache holt sich diese Seite wieder aus Document-Root, und prüft nun (Steuer-Tags auf der Seite) welche Teile davon er dem PHP-Interpreter zur Verarbeitung weitergeben muß



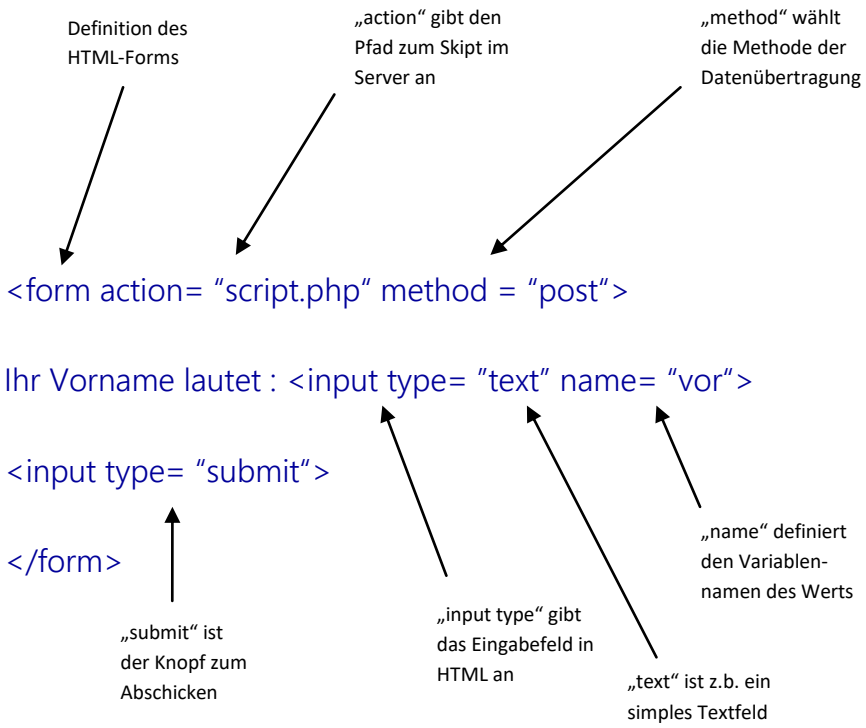
Der Programmteil auf der Seite (meist ein PHP-Skript) bekommt die Daten, verarbeitet sie, und schickt möglicherweise auch eine Antwort in Form einer Ausgabe. Diese Ausgabe wird nun aber natürlich nicht am Monitor des Servers angezeigt, sondern zurück zum Client geschickt, und dort an den Browser zur Anzeige übergeben.

Diese Funktion (Ausgabe des richtigen Skripts an den richtigen Client im Internet) leistet das common gateway interface (cgi).



Um das Ausprobieren zu können, benötigen wir nun zunächst die Eingabemöglichkeit am Browser, das HTML-Form.

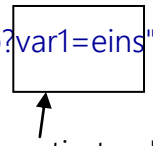
Hier der elementare Kern , in eine HTML-Seite reingeschrieben :



Als Datenübertragungsmethode stehen get und post zur Verfügung :
Get schickt die Daten einfach sichtbar an die Server-URL mit einem Fragezeichen angehängt
Post schickt die Daten unsichtbar und verschlüsselt.

Man kann bei der GET-Methode die zu übertragenden Daten auch selber anhängen, und dann mit einer beliebigen anderen Aufrufmethode (z.b. in einem Link) arbeiten :

```
<a href = "php1.php?var1=eins"> hier_aufrufen</a>
```



Das sind die transportierten Daten, Variable var1 ist „eins“

Klar ist das alles bisher ziemlich nutzlos, weil wir auf dem Server mit den Daten nichts anfangen können. In den folgenden Paketen beschäftigen wir uns deshalb mit „server-sided“ Programmierung