



Client – und serverseitige Programmierung

Mirjam Pfättisch

26.11.2005



Was ist ein Server?*

- n Ein Programm, das auf die Kontaktaufnahme eines Client-Programmes wartet und nach Kontaktaufnahme mit diesem Nachrichten austauscht.





Was ist ein Client?*

Eine Anwendung, die in einem Netzwerk (hier: Internet) den Dienst eines Servers in Anspruch nimmt

à Client – Server - Prinzip



Statische und dynamische Webseiten

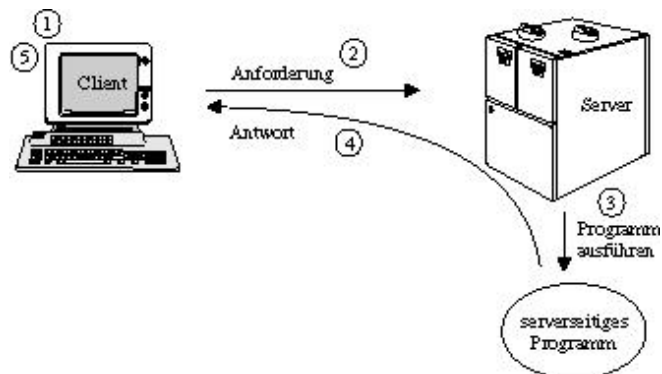
- | | |
|---|---|
| <ul style="list-style-type: none">n Statische Webseiteo Ursprüngliche Form der Webseiteno Technisch einfache Datei auf dem Webservero Benutzer kann lediglich Links aufrufen | <ul style="list-style-type: none">n Dynamische Webseiteo Häufigste Form von Webseiteno Werden im Moment der Anforderung erzeugt (z.B. aktuelle Nachrichten)o Benutzer kann interagiereno Server- und clientseitige Programmierung möglich |
|---|---|

Serverseitige Programmierung

Bei der Programmierung für Webseiten muss man zwischen serverseitiger und clientseitiger Programmierung unterscheiden:

- n serverseitige Programmierung (die Programme laufen auf dem Webserver)
- n clientseitige Programmierung (die Programme laufen auf dem PC des Benutzers)
- n Während clientseitige Programmausführungen von den individuellen Einstellungen jedes einzelnen Benutzers Ihrer Homepage abhängig sind, ist die serverseitige Ausführung nur vom Server-PC und den darauf installierten und aktivierten Softwarekomponenten abhängig.


Serverseitige Programmierung






Ausführung eines serverseitigen Programms (Beispiel)

1. Der Webbesucher füllt ein Formular aus und klickt auf den Abschicken-Schalter des Formulars. Der Autor der Webseite hat den Abschicken-Schalter des Formulars mit dem URL eines serverseitigen Programms verbunden.
2. Der Webbrowser schickt dem Webserver, der in dem URL angegeben ist, eine Aufforderung, das Programm aufzurufen und diesem die Formulardaten zu übergeben.
3. Der Webserver ruft das Programm auf. Das Programm wird auf dem Server-Rechner ausgeführt und verarbeitet die Formulardaten. Als Ergebnis erzeugt es beispielsweise den HTML-Code einer Antwortseite.
4. Der Webserver schickt die vom Programm generierte Webseite an den Browser zurück.
5. Der Webbesucher sieht in seinem Browser die Bestätigung, dass seine Formulardaten angekommen sind und verarbeitet wurden.



Beispiele der serverseitigen Programmierung

- n ASP (Active Server Pages): keine Programmiersprache, sondern eine von Microsoft entwickelte Technologie für die Verarbeitung interaktiver Internetseiten; verschiedene Skriptsprache können verwendet werden
- n PHP (Hypertext Preprocessor): Serverseitig interpretierte Skriptsprache zur Erstellung dynamischer Webseiten; Open-Source-Software
- n CFML (ColdFusion Markup Language): Programmiersprache (Tag-und Script-Syntax möglich)
- n Perl (Practical Extraction and Report Language): vielseitige Programmiersprache
- n JSP (Java Server Pages): eine Technologie, die zur einfachen dynamischen Erzeugung von HTML- und XML-Ausgaben eines Webservers dient



Vorteile der serverseitigen Programmierung

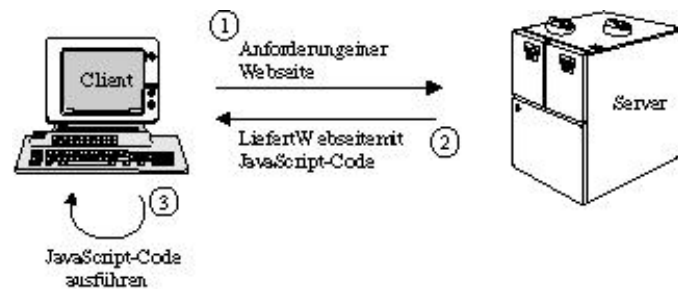
- n Keine speziellen Fähigkeiten beim Client (Browser) nötig und keine Inkompatibilitäten
- n Quelltext bleibt auf dem Server, nur der generierte Code ist für den Besucher einsehbar
- n Sicherer, da z.B. zwischen Datenbanken und Client keine direkte Verbindung aufgebaut wird
- n für anspruchsvollere Aufgaben oder das Zurückliefern von Daten an den Webserver unersetzlich



Nachteile der serverseitigen Programmierung

- n Jede Aktion des Benutzers erst bei erneutem Aufruf der Seite erfassbar
- n Die Ausführung ist aufgrund der Datenübertragung über das Netz relativ langsam
- n Jede Seite wird vom Server interpretiert → hohe Server-Belastung; Greifen z.B. Hunderte oder gar Tausende von Websurfern gleichzeitig auf das Programm auf dem Server zu, wird es hundert- oder tausendmal parallel auf dem Server-Rechner ausgeführt und kann diesen unter Umständen lahm legen

Clientseitige Programmierung



Ausführung eines clientseitigen Programms

- n Bei der clientseitigen Programmierung z.B. mit JavaScript schreibt der Webdesigner den Programmcode direkt in den HTML-Code der Webseite
- n Der **JavaScript-Code** wird **mit der Webseite** auf die Rechner der Webbesucher (Client-Rechner) heruntergeladen und dort mit Hilfe eines **Interpreters** ausgeführt



Beispiele der clientseitigen Programmierung

- n JavaScript: Objektorientierte Skriptsprache, die von der Firma Netscape entwickelt wurde, um statische HTML-Seiten dynamisch zu gestalten
- n Java Applet: kleines Computerprogramm zum Schreiben von Programmen für Webseiten, die im Webbrowser arbeiten; interagieren direkt mit dem Benutzer ohne Umweg über den Server
- n Flash (Macromedia Flash): Zur Erzeugung von Flash-Filmen im SWF-Format (unveränderbar); Grafik- und Animationsformat basierend auf Vektorgrafiken; Einsatz auf Webservern; proprietär
- n SVG (Scalable Vector Graphics): freies Konkurrenzformat zu Flash; veränderbar, da XML-Anwendung → kann als Dateiformat geöffnet werden (.svg); je nach Browser unterschiedlich gut unterstützt



Vorteile der clientseitigen Programmierung

- n Da der Programmcode direkt auf den Client-Rechnern ausgeführt wird, ist die Ausführung viel schneller, der Webserver wird nicht weiter belastet
- n Weil die Programme auch kein Sicherheitsrisiko mehr für den Webserver darstellen, entfällt der ganze Verwaltungsaufwand, der mit der serverseitigen Programmierung verbunden ist
- n Nützlich für dynamische Effekte und schnelle Interaktion mit dem Webbesucher



Nachteile der clientseitigen Programmierung

- n Spezielle Fähigkeiten beim Client (Browser) nötig
- n Quelltext einsehbar, d.h. veränderbar (kann je nach Anwendung auch ein Vorteil sein)
- n Nur für kleinere Aufgaben geeignet



Kompilation*

Hierzu wird ein spezielles Programm (der Compiler) aufgerufen, das den Quelltext des Programms in Maschinencode übersetzt. Das Ergebnis ist eine ausführbare Datei (EXE-Datei). Diese kann als eigenständiges Programm direkt aufgerufen und ausgeführt werden.



Kompilation Vor- & Nachteile*

- n Kompilierte Programme sind meist schneller und effizienter als vergleichbare interpretierte Programme.
- n Kompilierte Programme sind nicht portabel, da der vom Compiler erzeugte Maschinencode immer auf einen bestimmten Prozessortyp abgestimmt ist und von anderen Prozessoren nicht verarbeitet werden kann.
- n Beispiele für kompilierte Programmiersprachen sind [Pascal](#), [C/C++](#) und [Java](#).



Interpretation*

Hier fallen Übersetzung und Ausführung des Programms zusammen. Dies leistet der Interpreter. Er liest das Programm Zeile für Zeile ein, erzeugt entsprechenden Maschinencode und lässt diesen direkt ausführen.



Interpretation Vor-&Nachteile*

- n Interpretierte Programme sind in der Ausführung langsamer als vergleichbare kompilierte Programme, da im Hintergrund ja immer der Interpreter läuft und die Zeit für die Umsetzung in Maschinencode hinzugerechnet werden muss.
- n Interpretierte Programme können problemlos portiert werden, da sie nur als Quelltext vorliegen. Voraussetzung ist allerdings, dass auf den Rechnern, auf denen die Programme interpretiert und ausgeführt werden sollen, ein passender Interpreter installiert ist.
- n Beispiele für interpretierte Programmiersprachen sind [Basic](#), [Perl](#), [Java](#) und [JavaScript](#).