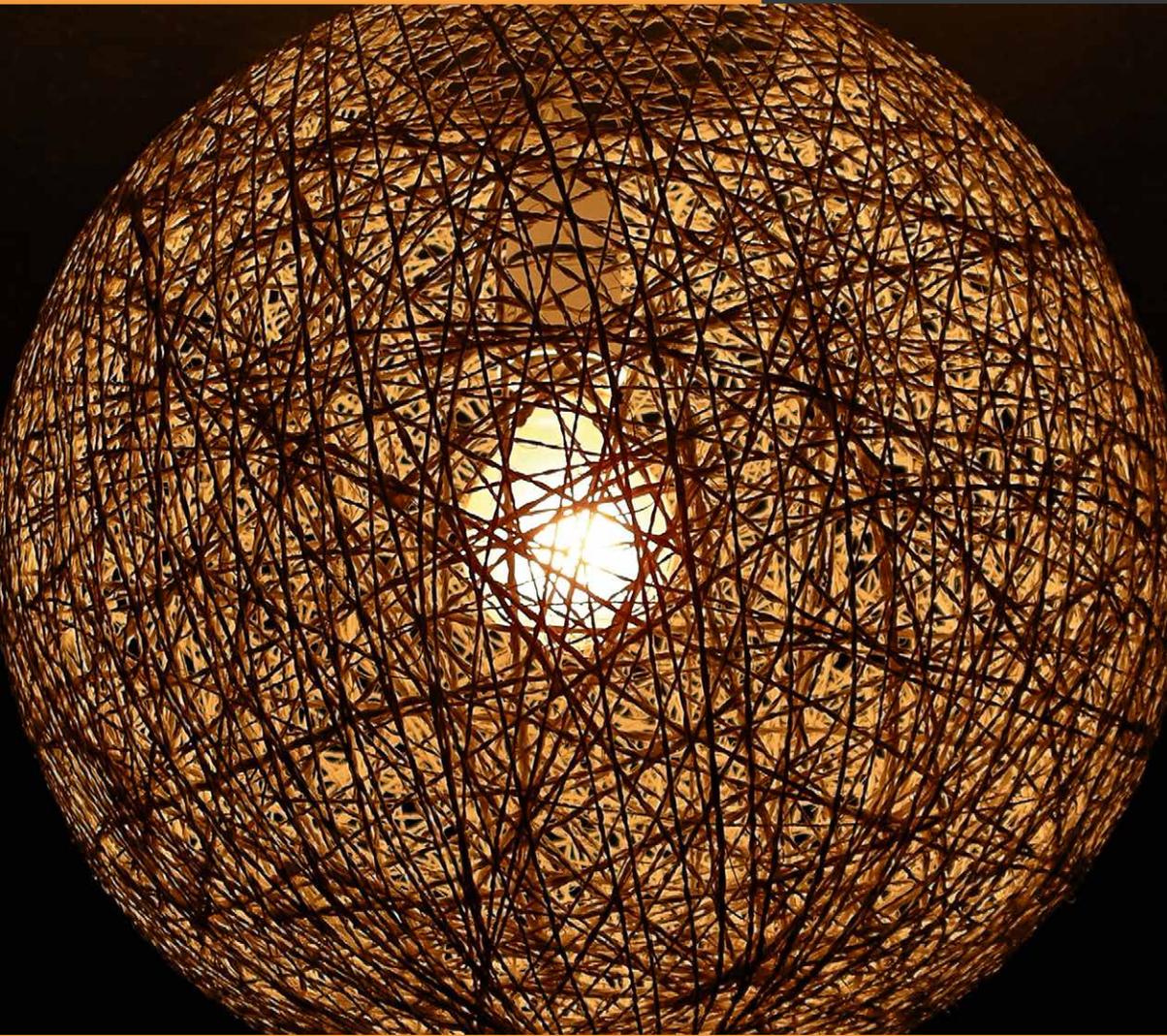




Steinbeis-Transferzentrum  
Verteilte und mobile  
Anwendungen



Sandro Leuchter

# Internet-Kommunikation – Eine praktische Einführung mit Java



*Sandro Leuchter*

Internet-Kommunikation – Eine praktische Einführung mit Java



**Steinbeis-Edition**

## Reihe „Verteilte Architekturen“ | Band 1



**Sandro Leuchter** hat seit 2016 die Professur für verteilte und mobile Anwendungen der Fakultät für Informatik der Hochschule Mannheim inne, ist Gründungsmitglied des Kompetenzzentrums Lehre & Lernen der Hochschule Mannheim und leitet das Steinbeis-Transferzentrum Verteilte und mobile Anwendungen. Vorher war er u. a. „Head of Software Engineering and Infrastructure Software“ bei Atlas Elektronik, einem gemeinsamen Unternehmen von ThyssenKrupp und EADS. 2009 promovierte er an der Fakultät für Verkehrs- und Maschinensysteme der Technischen Universität Berlin zum Dr.-Ing. Das Diplom in Informatik hat er 1999 ebenfalls an der Technischen Universität Berlin bekommen.



Steinbeis-Transferzentrum  
Verteilte und mobile  
Anwendungen

Sandro Leuchter

# **Internet-Kommunikation – Eine praktische Einführung mit Java**

## Impressum

2019 Steinbeis-Edition

Dieses Werk ist unter einer Creative Commons Lizenz vom Typ *Namensnennung – Nicht-kommerziell – Weitergabe unter gleichen Bedingungen 4.0 International* zugänglich. Um eine Kopie dieser Lizenz einzusehen, konsultieren Sie <http://creativecommons.org/licenses/by-nc-sa/4.0/> oder wenden Sie sich brieflich an Creative Commons, Postfach 1866, Mountain View, California, 94042, USA.



Sandro Leuchter: Internet-Kommunikation – Eine praktische Einführung mit Java  
Reihe „Verteilte Architekturen“ | Band 1

1. Auflage, 2019 | Steinbeis-Edition, Stuttgart  
ISBN 978-3-95663-189-4

Diese Publikation ist auch abrufbar unter: <https://doi.org/10.5281/zenodo.1042168>

Satz: Sandro Leuchter, Steinbeis-Transferzentrum Verteilte und mobile Anwendungen, Karlsruhe  
Bildquellen: siehe S. 341  
Druck: e.kurz+co druck und medientechnik gmbh, Stuttgart

Steinbeis ist mit seiner Plattform ein verlässlicher Partner für Unternehmensgründungen und Projekte. Wir unterstützen Menschen und Organisationen aus dem akademischen und wirtschaftlichen Umfeld, die ihr Know-how durch konkrete Projekte in Forschung, Entwicklung, Beratung und Qualifizierung unternehmerisch und praxisnah zur Anwendung bringen wollen. Über unsere Plattform wurden bereits über 2.000 Unternehmen gegründet. Entstanden ist ein Verbund aus mehr als 6.000 Experten in rund 1.100 Unternehmen, die jährlich mit mehr als 10.000 Kunden Projekte durchführen. So werden Unternehmen und Mitarbeiter professionell in der Kompetenzbildung und damit für den Erfolg im Wettbewerb unterstützt. Die Steinbeis-Edition verlegt ausgewählte Themen aus dem Steinbeis-Verbund.

203683-2019-07 | [www.steinbeis-edition.de](http://www.steinbeis-edition.de)

---

## Vorwort

Die Mehrzahl der IT-Systeme basiert auf einer verteilten Architektur: Sei es, dass Menschen in sozialen Netzwerken miteinander interagieren wollen, Ressourcen in der Unternehmens-IT zwischen mehreren Nutzern geteilt oder redundant vorgehalten werden um Lastspitzen bzw. Ausfälle abzufedern oder dass Sensoren in weltumspannende Messnetze eingebunden werden. Für IT-Entwicklerinnen und Entwickler aus nahezu allen Anwendungsbereichen ist das Aufteilen der Funktionalität auf voneinander getrennte Einheiten, die durch Netzwerkkommunikation zusammenwirken, heute allgegenwärtig. Immer mehr wird bei der Umsetzung auf Internet-Technologien gesetzt. Das ermöglicht die Implementierung durchgehender Funktionsketten ohne Protokollbrüche und erleichtert die Integration von bisher isolierten Systemen im Zuge der Digitalisierung aller Lebensbereiche.

Die Entwurfsparadigmen verteilter Architekturen haben dabei vielfältige Auswirkungen auf die nichtfunktionalen Anforderungen an IT-Systeme. Sie ermöglichen z. B. die Berücksichtigung von Ausfallsicherheit und Skalierbarkeit, erleichtern aber auch gleichzeitig das unerlaubte Ausspähen oder Stören von Komponenten verteilter Anwendungen. Deshalb ist es für alle Rollen, die zum Entwicklungsprozess moderner IT-Systeme beitragen, immens wichtig, die Funktionsweise der Internet-Kommunikation zu kennen und ein breites Repertoire für Gestaltungsalternativen für verteilte Architekturen zur Hand zu haben.

Diese Buchreihe ist an alle gerichtet, die aufbauend auf grundlegenden Programmierkenntnissen die Kompetenz erlangen wollen, verteilte Architekturen zu planen, umzusetzen und zu betreiben. Die Reihe ist als Lehrbuch für das Selbststudium konzipiert. Am Ende sind die Leserinnen und Leser in der Lage selbstständig alle architekturellen Anforderungen an verteilte Anwendungen zu analysieren, geeignete Architekturen für verteilte Anwendungen auszuwählen und zu bewerten, eine geeignete *Middleware* und Ablaufumgebung auszuwählen sowie verteilte Anwendungen kompetent zu entwerfen, zu implementieren und in Betrieb zu nehmen.

Die Algorithmik dieses Lehrprogramms ist zunächst leicht. Die Hürden für den Einsatz der vorgestellten Technologiekonzepte für die Umsetzung verteilter Architekturen liegen nicht in der Komplexität der Programme, sondern in der Ablaufumgebung: Aufgrund der Abhängigkeiten zwischen Elementen der Ablaufumgebung kann es ziemlich aufwändig und fehleranfällig sein eine Entwicklungsumgebung für Programmierung und Test verteilter Architekturen aufzubauen.

---

Deshalb ist diese Lehrbuchreihe als Praktikum aufgebaut: Zu jedem Technologiebereich gibt es eine sehr kurze Einführung in die Konzepte. Darauf folgt immer ein umfangreiches Praktikum, in dem sich Leserinnen und Leser die jeweilige Technologie mit einer Schritt-für-Schritt-Anleitung selbstständig erarbeiten.

### **Aktivitätsschritt 0.1:**

#### **Ziel: Aufbau des Buches verstehen**

Jeder Aktivitätsschritt beschreibt eine abgrenzbare Tätigkeit, die jeweils ein konkretes Ziel hat. Zu Beginn jedes Praktikums sind die Aktivitätsschritte noch ziemlich ausführlich erläutert. In späteren Schritten werden die Funktionalität und Leistungsfähigkeit der erarbeiteten Technologie durch praktische Experimente genauer untersucht und die Anleitungen dazu werden weniger detailliert, sind aber immer durch Transfer aus den früheren Aktivitätsschritten zu meistern.

Die Aktivitätsschritte bauen immer aufeinander auf und sollten in der vorgegebenen Reihenfolge bearbeitet werden. Am Ende jedes Praktikums sollten die Leserinnen und Leser sich ein gutes überblicksartiges Verständnis der architekturellen Möglichkeiten der jeweiligen Technologie erarbeitet haben und in der Lage sein sie selbstständig einzusetzen. Zu allen Programmierpraktika gibt es Musterlösungen.

Die Auswahl der Themen geht über das Minimum hinaus, das für eine pragmatische Beherrschung aktueller Technologien für die Umsetzung verteilter Architekturen erforderlich ist. Der thematischen Erweiterung in dieser Buchreihe liegt die Überzeugung zugrunde, dass für eine kompetente Anwendung der benutzten Technologien das Verständnis für mindestens eine darunterliegende Ebene vorhanden sein sollte.



Die allerwichtigsten Konzepte und Zusammenhänge sind mit einem Ausrufezeichen am Rand markiert.

Jeder Band stellt den Stoff eines Semesters (von vier bis fünf ECTS *Credit Points*, also ca. 120-150 Stunden Aufwand) dar. Aufgrund der Vielzahl der architekturellen Ansätze ist das Konzept dieser Lehrbuchreihe: «Breite vor Tiefe». An vielen Stellen gibt es interessante weiterführende Informationen, die nicht zu den Basisinformationen des jeweiligen Themas gehören und letztlich auch nicht zu den beschriebenen Lernzielen beitragen. Leser, die einem straffen Programm folgen wollen, können diese als «Exkurs» gekennzeichneten Abschnitte überspringen.

---

### Exkurs: Abschweifungen für mehr Kontext

Trotzdem sind diese jeweils kurz dargestellten Themen in die Lehrbuchreihe aufgenommen worden, um Leserinnen und Lesern mehr Kontext und Anknüpfungspunkte zu bieten und zu eigener Recherche zu motivieren.

Letztlich ist die Stoffauswahl und der Umfang der Praktika ziemlich knapp. Es hat sich gezeigt, dass einige Leserinnen und Leser den Stoff gerne über den Standardumfang hinaus vertiefen wollen – zum einen um die neu angeeigneten Fertigkeiten zu festigen, zum anderen um die Technologien stärker in der Tiefe zu erforschen und weitergehende Konzepte zu entdecken. Deshalb gibt es am Ende jedes Praktikums Anregungen für eigene Projekte.

#### **Aktivitätsschritt 0.2 (fakultativ):**

**Ziel:** Fakultative Aktivitätsschritte als Anregung begreifen

Bei über den Standardumfang hinausgehenden Aufgaben sind die Aktivitätsschritte als «fakultativ» gekennzeichnet. Das bedeutet, dass es sich um Vertiefungen handelt. Für die meisten dieser Vertiefungsaufgaben gibt es Musterlösungen wie für die regulären.

Auf der *Companion Website* <http://verteiltearchitekturen.de/> stehen die Bände der Buchreihe als PDF zum Download bereit. Außerdem gibt es dort die Eclipse-Projekte für Praktika sowie Musterlösungen als Eclipse-Projekte. Bücher und Quelltexte werden unter einer «Creative Commons BY-NC-SA»-Lizenz weitergegeben. Jeder hat also das Recht die Inhalte und Dateien der Buchreihe für nichtkommerzielle Zwecke zu verwenden und unter Autorennennung und unter den gleichen Bedingungen weiterzugeben. Dozierende bekommen auf Anfrage sehr gerne Folien sowie Aufgaben und Musterlösungen für Testate zur Verfügung gestellt.

Karlsruhe, Oktober 2018

*Sandro Leuchter*



# Inhaltsverzeichnis

<b>I</b>	<b>Internet-Standards</b>	<b>1</b>
1	Internet-Adressierung	3
1.1	IP-Adressen . . . . .	4
1.2	Subnetze und ihre Adressierung . . . . .	6
1.3	Forwarding und Routing . . . . .	10
1.4	Dynamic Host Configuration Protocol (DHCP) . . . . .	12
1.5	Domain Name System (DNS) . . . . .	14
1.5.1	DNS-Architektur . . . . .	17
1.5.2	Daten im DNS . . . . .	21
1.5.3	Auflösung eines textuellen <i>Host</i> -Namens zu einer numerischen IP-Adresse . . . . .	21
1.5.4	Abläufe . . . . .	22
1.6	Interaktion mit dem Domain Name System (DNS) . . . . .	28
1.7	Alternative DNS Server . . . . .	32
1.8	Domaininhaber feststellen . . . . .	36
1.9	Paketvermittlung im Internet: Routing . . . . .	37
1.10	Netzwerkconfiguration ermitteln und ändern . . . . .	41
1.10.1	Windows: <code>ipconfig</code> . . . . .	41
1.10.2	Unix (BSD, Linux, macOS): <code>ifconfig</code> . . . . .	43
<b>2</b>	<b>Socket-Kommunikation mit dem User Datagram Protocol (UDP)</b>	<b>47</b>
2.1	Lebenszyklus von <code>java.net.DatagramSocket</code> -Objekten . . . . .	48
2.1.1	UDP-Socket-Konstruktoren . . . . .	48
2.1.2	Vor der Verwendung: <i>Port binding</i> . . . . .	49
2.1.3	Am Ende: Socket schließen . . . . .	49
2.2	Lesen und Schreiben über einen UDP Socket . . . . .	50
2.2.1	Timeout beim Lesen von einem UDP Socket . . . . .	51
2.3	Aufgabe: Echo Service . . . . .	55
2.3.1	Server für den Echo Service . . . . .	57
2.3.2	Client für den Echo Service . . . . .	57
2.4	Aufgabe: Time Service . . . . .	62

2.4.1	Server für den Time Service . . . . .	65
2.4.2	Client für den Time Service . . . . .	66
2.5	Aufgabe: Messwert Service . . . . .	66
2.5.1	Server für den Messwert Service . . . . .	68
2.5.2	Client für den Messwert Service . . . . .	70
2.6	Anregung ReactionGame: Ein einfaches Reaktionsspiel mit Timeouts	71
<b>3</b>	<b>Socket-Kommunikation mit dem Transmission Control Protocol (TCP)</b>	<b>75</b>
3.1	Das Transportprotokoll TCP . . . . .	76
3.1.1	Dienstgüteparameter von TCP . . . . .	76
3.1.2	TCP Sockets . . . . .	77
3.2	Aufgabe: Echo Service . . . . .	83
3.2.1	Client für den Echo Service . . . . .	83
3.2.2	Iterativer Server für den Echo Service . . . . .	87
3.2.3	<i>Threaded</i> Server für den Echo Service . . . . .	89
3.3	Aufgabe: File Service . . . . .	95
3.3.1	Client für den File Service . . . . .	95
3.3.2	Iterativer Server für den File Service . . . . .	97
3.3.3	<i>Threaded</i> Server für den File Service . . . . .	100
3.4	Aufgabe: Time Service . . . . .	103
3.4.1	Time Service: iterativer Server für Zeitstempel in ms . . . . .	104
3.4.2	Time Service: iterativer Server für Zeitstempel in lokalisier- tem Textformat . . . . .	106
3.4.3	Client für den Time Service . . . . .	108
3.5	Ausblick und Anregungen für eigene Projekte . . . . .	109
3.5.1	<i>Thread Pool</i> : Skalierung der Anzahl der Clients . . . . .	109
3.5.2	Funktionale Erweiterung von Kommunikationsprotokollen . .	112
<b>II</b>	<b>Indirekte, gepufferte Kommunikation</b>	<b>115</b>
<b>4</b>	<b>Indirekte, gepufferte Kommunikation</b>	<b>117</b>
4.1	Enterprise Application Integration . . . . .	118
4.2	Message Oriented Middleware . . . . .	119
<b>5</b>	<b>Java Message Service (JMS)</b>	<b>123</b>

5.1	Java Naming and Directory Interface (JNDI) . . . . .	125
5.2	Programmierung von JMS Clients . . . . .	127
5.2.1	Behandlung von Ausnahmen . . . . .	127
5.2.2	Verbindungsaufbau zum JMS Provider . . . . .	127
5.2.3	Ressourcen im JMS Client freigeben . . . . .	129
5.2.4	Nachrichten mit JMS synchron empfangen . . . . .	129
5.2.5	Nachrichten mit JMS asynchron empfangen . . . . .	132
5.2.6	Nachrichtentypen in JMS . . . . .	132
5.2.7	Nachrichten mit JMS versenden . . . . .	134
5.2.8	Message Properties . . . . .	134
5.2.9	Filter beim Nachrichtenaustausch über JMS . . . . .	135
5.3	Kommunikationsmuster bei JMS . . . . .	138
5.3.1	Point To Point . . . . .	138
5.3.2	Request/Reply . . . . .	140
5.3.3	<i>Publish/Subscribe</i> . . . . .	142
5.4	Aufgabe: Installation von Apache ActiveMQ als JMS Provider . . . . .	150
5.5	Aufgabe: Logger Service . . . . .	155
5.6	Aufgabe: Textumdreher-Service . . . . .	156
5.7	Aufgabe: Chat Service ( <i>Publish/Subscribe</i> ) . . . . .	160
5.8	Ausblick und Anregungen für eigene Projekte . . . . .	162
5.8.1	Skalierbare und robuste verteilte Architekturen mit JMS . . . . .	162
5.8.2	Ein Framework für JMS-basierte Anwendungen . . . . .	165
<b>6</b>	<b>Message Queue Telemetry Transport (MQTT)</b> . . . . .	<b>167</b>
6.1	Topics . . . . .	169
6.2	Das MQTT-Protokoll . . . . .	171
6.2.1	Quality of Service . . . . .	171
6.2.2	Retained Messages . . . . .	172
6.2.3	Last Will and Testament (LWT) . . . . .	172
6.2.4	Persistent Session . . . . .	172
6.2.5	Nachrichten . . . . .	173
6.3	Broker . . . . .	176
6.4	MQTT Clients in Java mit der Paho Library . . . . .	176
6.4.1	Publisher mit Paho . . . . .	177
6.4.2	Subscriber mit Paho . . . . .	177

- 6.5 Aufgabe: unterschiedliche Broker für eine prototypische MQTT-Anwendung . . . . . 181
  - 6.5.1 Eclipse-Projekt und Client Library . . . . . 181
  - 6.5.2 Broker . . . . . 183
  - 6.5.3 ActiveMQ lokal installiert . . . . . 183
  - 6.5.4 HiveMQ öffentlich gehostet . . . . . 188
  - 6.5.5 MQTT Clients . . . . . 189
- 6.6 Aufgabe: Smart-Home-Anwendung . . . . . 190
  - 6.6.1 Szenario und Beispielanwendung . . . . . 190
  - 6.6.2 Datenmodell für die Beispielanwendung . . . . . 190
  - 6.6.3 Publisher: Sensorsimulator . . . . . 191
  - 6.6.4 Subscriber: Alarm . . . . . 193
  - 6.6.5 Subscriber: Logger mit Wildcard-Verwendung . . . . . 194
  - 6.6.6 Selbstbeschreibender Sensor mit *Retained Message* und *Last Will and Testament* . . . . . 195
- 6.7 Ausblick und Anregungen für eigene Projekte . . . . . 198
  - 6.7.1 Subscriber für Lagebilddarstellungen . . . . . 198

**III Webbasierte Kommunikation 201**

- 7 Hypertext Transport Protocol (HTTP) 203**
  - 7.1 Identifikation von Ressourcen im World Wide Web . . . . . 205
    - 7.1.1 MIME Type . . . . . 205
    - 7.1.2 Uniform Ressource Identifier/Uniform Ressource Locator . . 206
  - 7.2 Hypertext Transfer Protocol . . . . . 210
    - 7.2.1 HTTP-Protokollablauf («non persistent» und «persistent») . . 211
    - 7.2.2 HTTP Request . . . . . 211
    - 7.2.3 Header-Zeilen . . . . . 215
    - 7.2.4 HTTP Response . . . . . 215
    - 7.2.5 Sessions . . . . . 217
    - 7.2.6 Cookies . . . . . 218
    - 7.2.7 Conditional GET . . . . . 221
  - 7.3 Webserven . . . . . 223
    - 7.3.1 Statische und dynamische Inhalte . . . . . 223

7.4	Aufgabe: Webserver programmieren . . . . .	226
7.4.1	Grundgerüst TCP-Server . . . . .	227
7.4.2	HTTP Request prüfen . . . . .	227
7.4.3	HTTP Response generieren . . . . .	231
7.4.4	Anfragen, die keine Datei betreffen, behandeln . . . . .	231
7.5	Ausblick und Anregungen für eigene Projekte . . . . .	234
7.5.1	Mediathek mit Jugendschutz . . . . .	235
<b>8</b>	<b>Servlet Container</b>	<b>237</b>
8.1	HttpServlet . . . . .	238
8.2	Deployment Descriptor . . . . .	241
8.3	HttpServlet-Zustand und Thread-Sicherheit . . . . .	241
8.4	Aufgabe: Tomcat als Servlet Container verwenden . . . . .	244
8.4.1	Deployment von Web-Anwendungen . . . . .	248
8.5	Aufgabe: «Dynamic Web Project» als Entwicklungsumgebung für Servlets in Eclipse verwenden . . . . .	250
8.5.1	Serverkonfigurationen in Eclipse . . . . .	251
8.5.2	Dynamic Web Project mit Server anlegen . . . . .	253
8.6	Aufgabe: Hello World in einem «Dynamic Web Project» . . . . .	257
8.6.1	Statische Inhalte in der Web-Anwendung: HTML File . . . . .	257
8.7	Aufgabe: Dynamische Inhalte in der Web-Anwendung: HttpServlet . . . . .	261
8.7.1	HttpRequest . . . . .	265
8.7.2	Aufruf Parameter für Servlets . . . . .	267
8.8	Ausblick und Anregungen für eigene Projekte . . . . .	269
8.8.1	Mediathek mit Jugendschutz . . . . .	269
<b>9</b>	<b>Deployment auf einer Cloud-basierten Plattform</b>	<b>273</b>
9.1	Cloud Computing . . . . .	273
9.2	Cloud-Service-Modelle . . . . .	274
9.3	Platform as a Service . . . . .	276
9.4	Skalierung . . . . .	278
9.5	Statische Inhalte in PaaS Servlet Containern . . . . .	278
9.6	Aufgabe: Web-Anwendung für Wahl in Google App Engine deployen . . . . .	279
9.6.1	Serverseitiges Projekt in der Google App Engine anlegen . . . . .	281
9.6.2	Lokales (Eclipse-seitiges) Projekt anlegen . . . . .	286

9.6.3	Deployment in lokaler Sandbox . . . . .	291
9.6.4	Projekt in Google App Engine deployen . . . . .	293
9.6.5	Erreichbarkeit unter eigenem Domain-Namen . . . . .	303
9.6.6	Ausblick: Zustandslosigkeit und Instanzen der Web-Anwendung	308
<b>IV</b>	<b>Anhang</b>	<b>317</b>
<b>10</b>	<b>Index</b>	<b>319</b>
<b>11</b>	<b>Abkürzungsverzeichnis</b>	<b>335</b>
<b>12</b>	<b>Bildnachweise</b>	<b>341</b>

**Teil I**

**Internet-Standards**





# Internet-Adressierung

---

Prinzipiell kann man für die Kommunikation in verteilten Architekturen ganz unterschiedliche Netzwerktechnologien verwenden. Im Anwendungsbereich der Prozesssteuerung chemischer Anlagen kommen dafür z. B. besondere Feldbustechnologien zum Einsatz, die spezialisierte echtzeitfähige Kommunikationsverfahren benutzen. In dieser Buchreihe wird hingegen grundsätzlich Internet-Technologie für die Netzwerkkommunikation benutzt.

In ersten Teil dieses Bandes von *Verteilte Architekturen* werden die Grundlagen für die Programmierung von Netzwerkkommunikation gelegt. Im weiteren Verlauf werden dann unterschiedliche Techniken behandelt um Nachrichten in verteilten Systemen auszutauschen. Zur Repräsentation von Absender und Empfänger ist aber immer eine Benennung und Adressierung von Rechnern im Internet erforderlich. In diesem Kapitel lernen Sie deshalb mehrere Arten der Benennung von vernetzten Rechnern kennen: zwei unterschiedliche Arten von *IP-Adressen* und textuelle Rechnernamen, die vom *Domain Name System* verwaltet werden. Außerdem geht es um die Grundlagen der Kommunikation im Internet und die dynamische Verteilung von IP-Adressen über das *Dynamic Host Configuration Protocol*.