Julius-Maximilians-
UNIVERSITÄT
WÜRZBURG

DATA DRIVEN
DECISIONS

Web Engineering
ASP.NET und MVC

Prof. Dr. Frédéric Thiesse
Sommersemester 2026

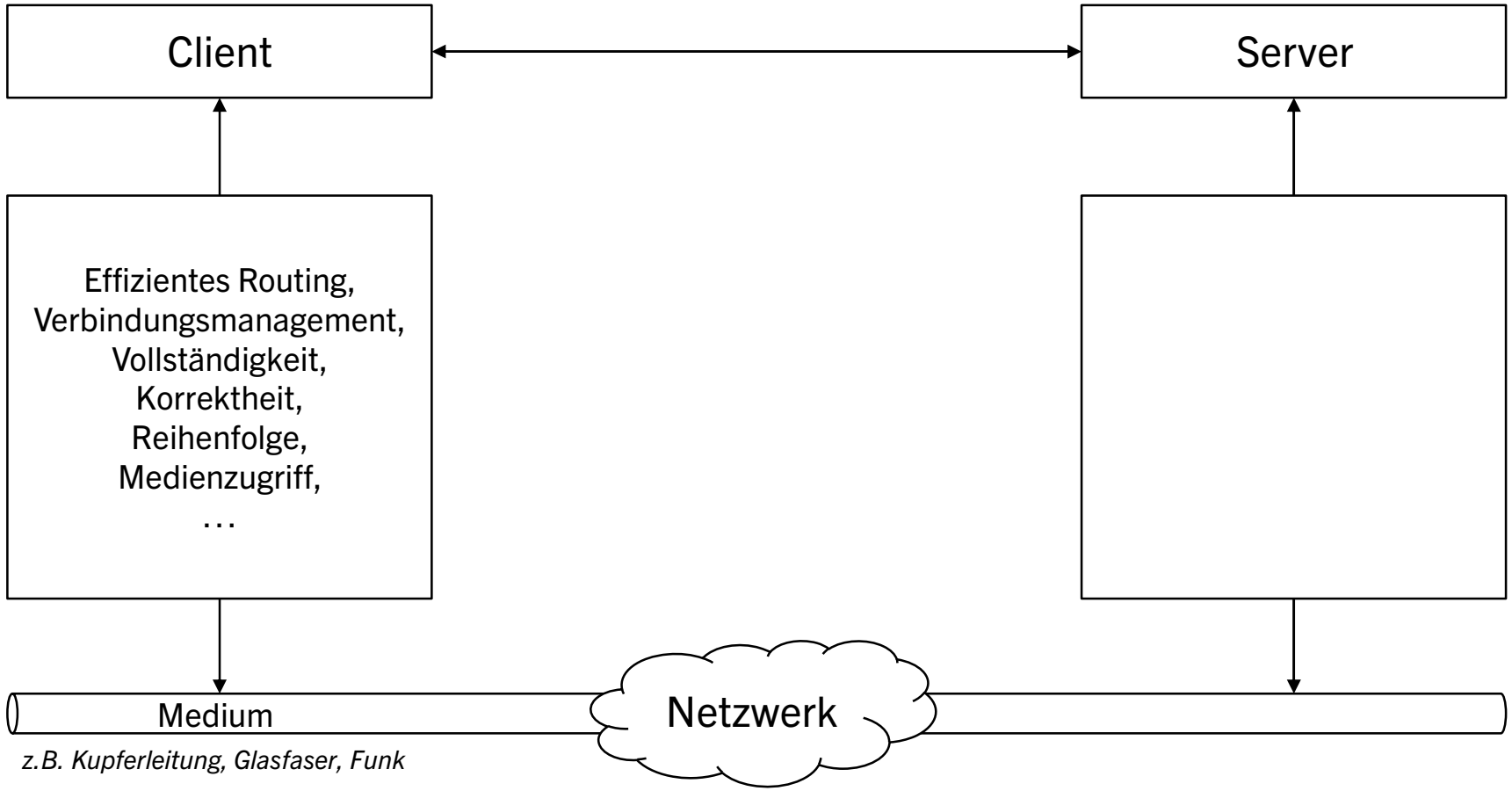
Julius-Maximilians-
UNIVERSITÄT
WÜRZBURG

Internet

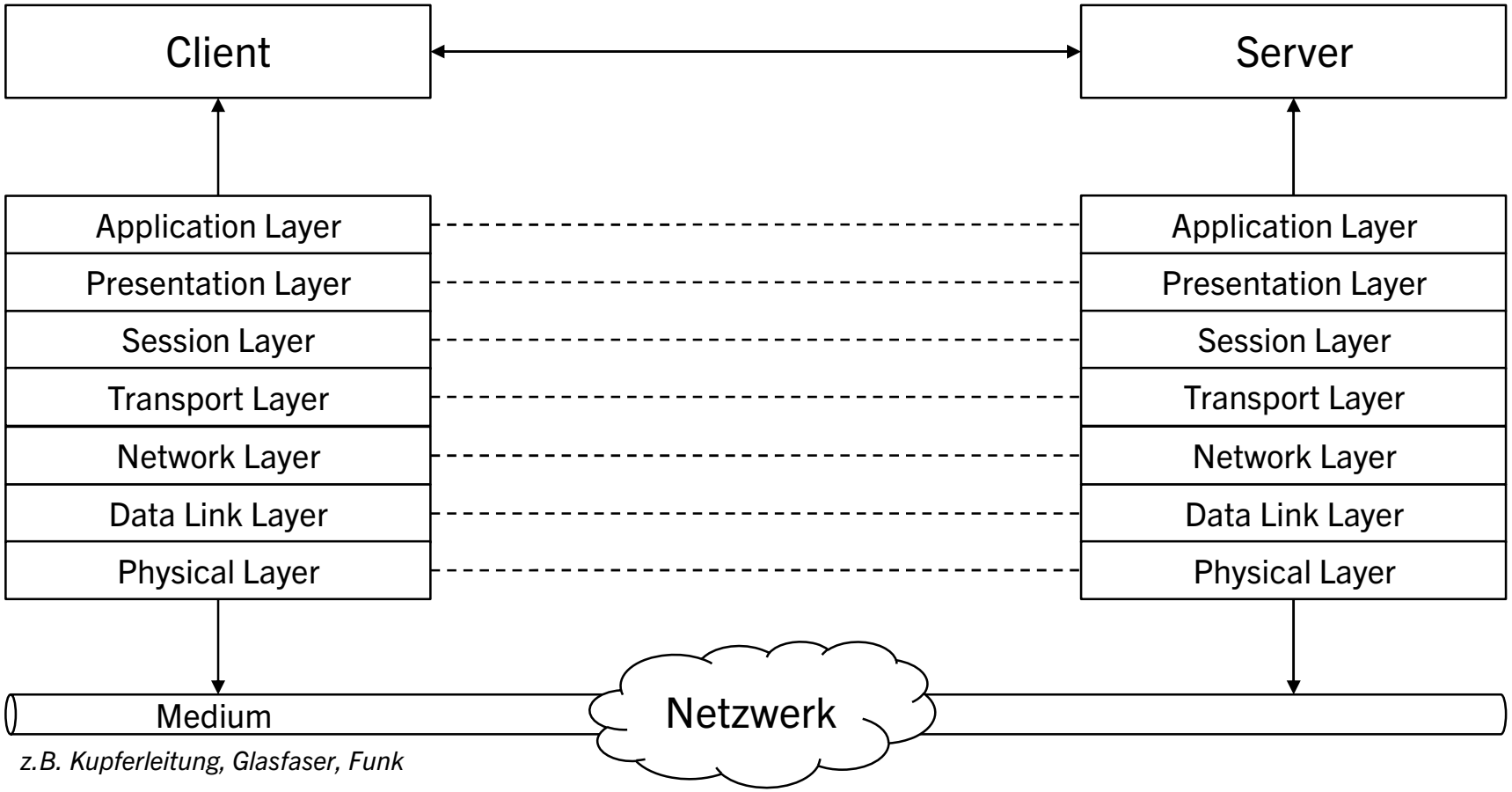
World Wide Web

ASP.NET MVC

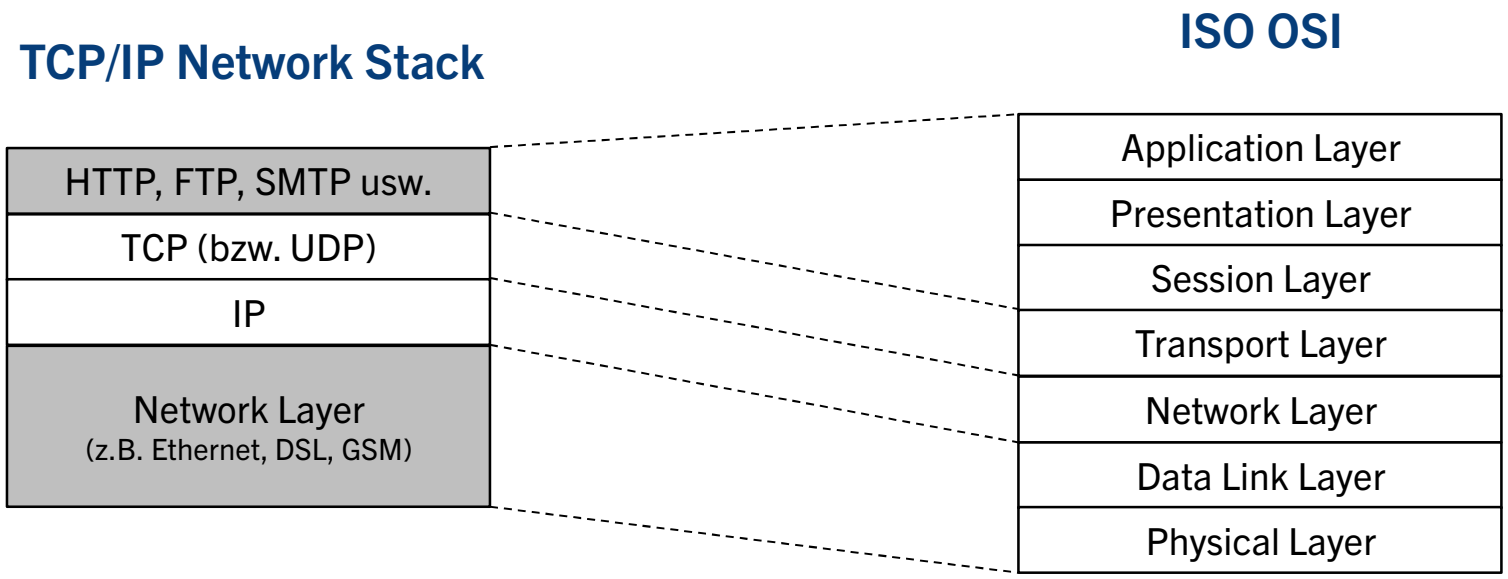
Kommunikation zwischen Client und Server im Netzwerk



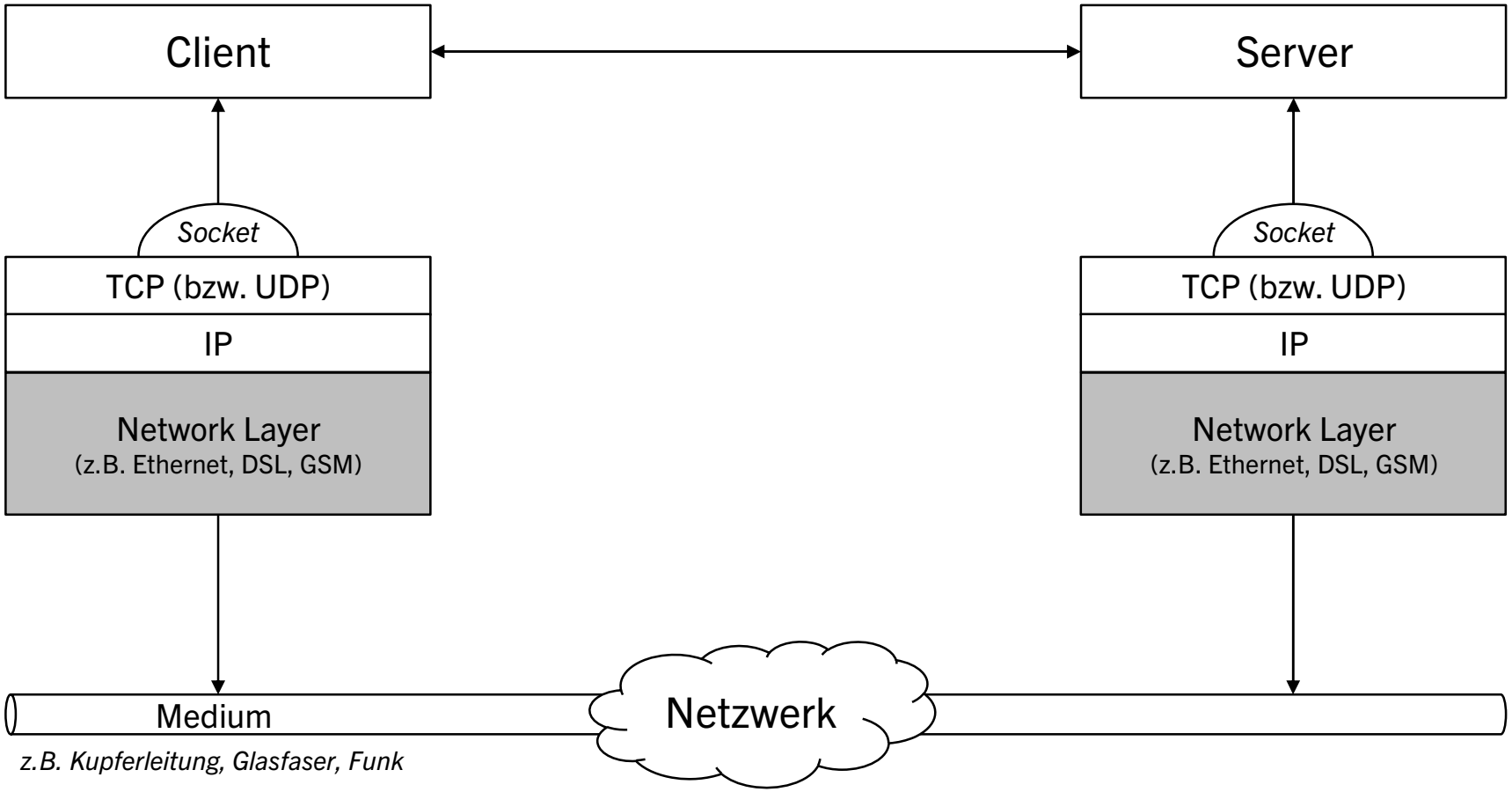
Das ISO OSI-Referenzmodell (Open Systems Interconnection)



Die Internetprotokollfamilie TCP/IP



Zugriff auf den Network Stack in der Praxis über Sockets



Julius-Maximilians-
UNIVERSITÄT
WÜRZBURG

Internet

World Wide Web

ASP.NET MVC

Julius-Maximilians-
UNIVERSITÄT
WÜRZBURG

HTTP-Protokoll

- **HTTP = HyperText Transfer Protocol**
  - **Grundlage für Bereitstellung bzw. Abruf von Webseiten im World Wide Web**
- **Anwendungsprotokoll auf der Basis von TCP**
  - **Zustandslos, d.h., jede Anfrage wird (prinzipiell) unabhängig von allen vorherigen Anfragen bearbeitet**
- **Geschichte**
  - **1996: Version 1.0**
  - **1999: Version 1.1**
    - Seither verschiedene Verbesserungen / Verfeinerungen
  - **2015: Version 2.0**
  - **2022: Version 3.0**

Julius-Maximilians-
UNIVERSITÄT
WÜRZBURG

HTTP-Anweisungen an den Server

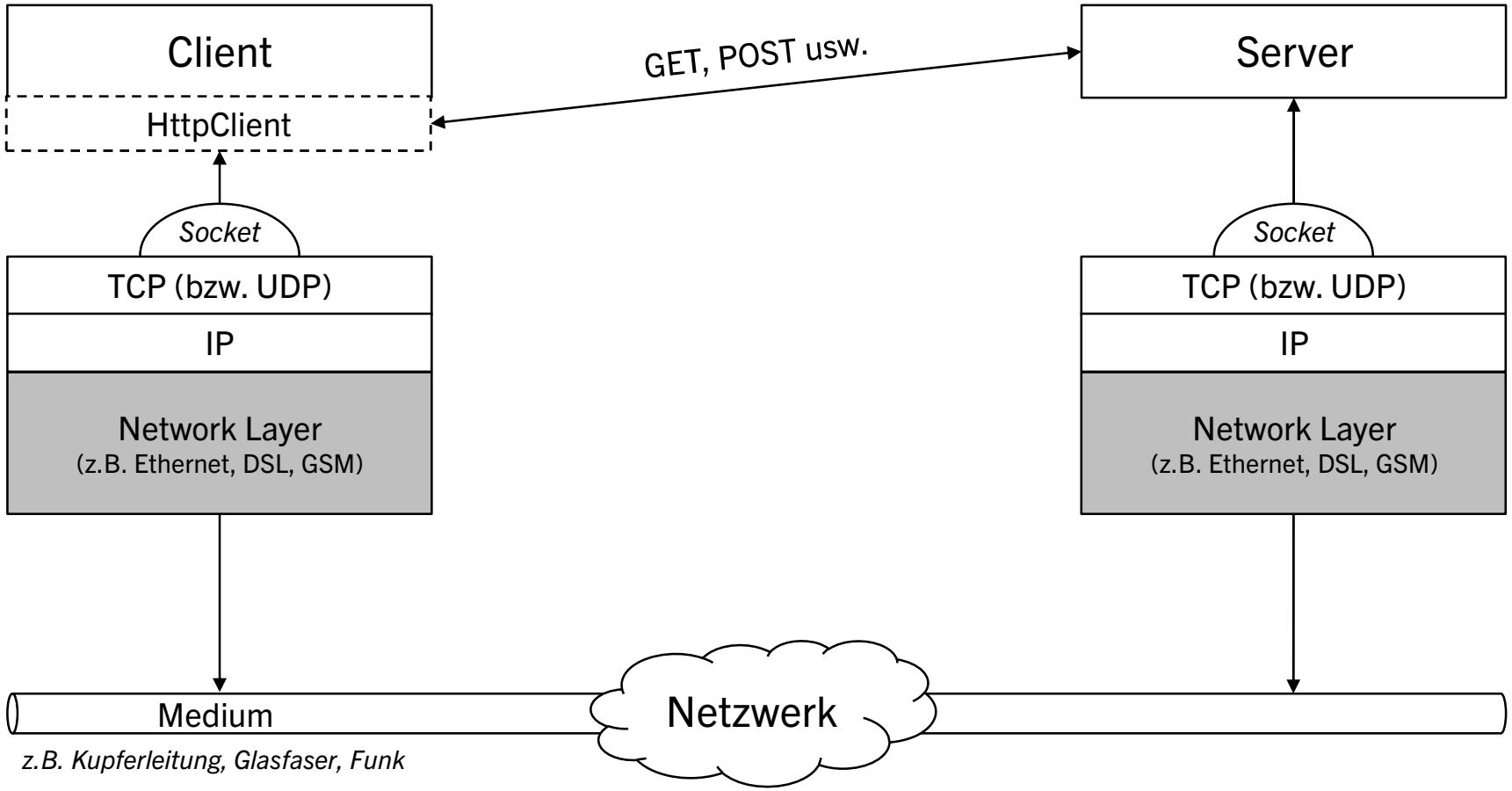
- **GET**
  - Abruf einer Ressource (optional mit Parametern in der URL)
- **POST**
  - Übertragung von (Formular-)Daten
- **HEAD**
  - Wie GET, aber nur Abruf des Dokumentenkopfs
- **PUT**
  - Upload einer Ressource (z.B. HTML-Dokument)
- **DELETE**
  - Löschen einer Ressource
- **TRACE**
- **OPTIONS**
- **CONNECT**

Julius-Maximilians-
UNIVERSITÄT
WÜRZBURG

HTTP-Statuscodes des Servers

- **1xx – Informationen**
  - z.B. 102 = Processing
- **2xx – Erfolgreiche Operationen**
  - z.B. 200 = OK
- **3xx – Umleitung**
  - z.B. 308 = Permanent Redirect
- **4xx – Client-Fehler**
  - Z.B. 404 = Not Found
- **5xx – Server-Fehler**
  - z.B. 503 = Service Unavailable
- **9xx – Proprietäre Statuscodes**

HTTP-Kommunikation mit Hilfe der .NET-Klasse „HttpClient“



Julius-Maximilians-
UNIVERSITÄT
WÜRZBURG

Hypertext im WWW

- **Inhalte im WWW basieren auf der „Hypertext Markup Language“ (HTML)**
  - HTML erlaubt die geräteunabhängige Beschreibung von Text- und Mediendokumenten, die durch ein einheitliches Verweissystem verbunden sind

- **HTML-Befehle haben (meist) folgende Syntax:**
  - Sie bestehen aus einem Starttag und einem Endtag
  - Die Schlüsselwörter werden in spitzen Klammern angegeben
    - Endtags wird ein „/“ vorangestellt
  - Bei HTML-Tags wird nicht zwischen Groß- und Kleinschreibung unterschieden

<tagname> … TEXT … </tagname>

Julius-Maximilians-
UNIVERSITÄT
WÜRZBURG

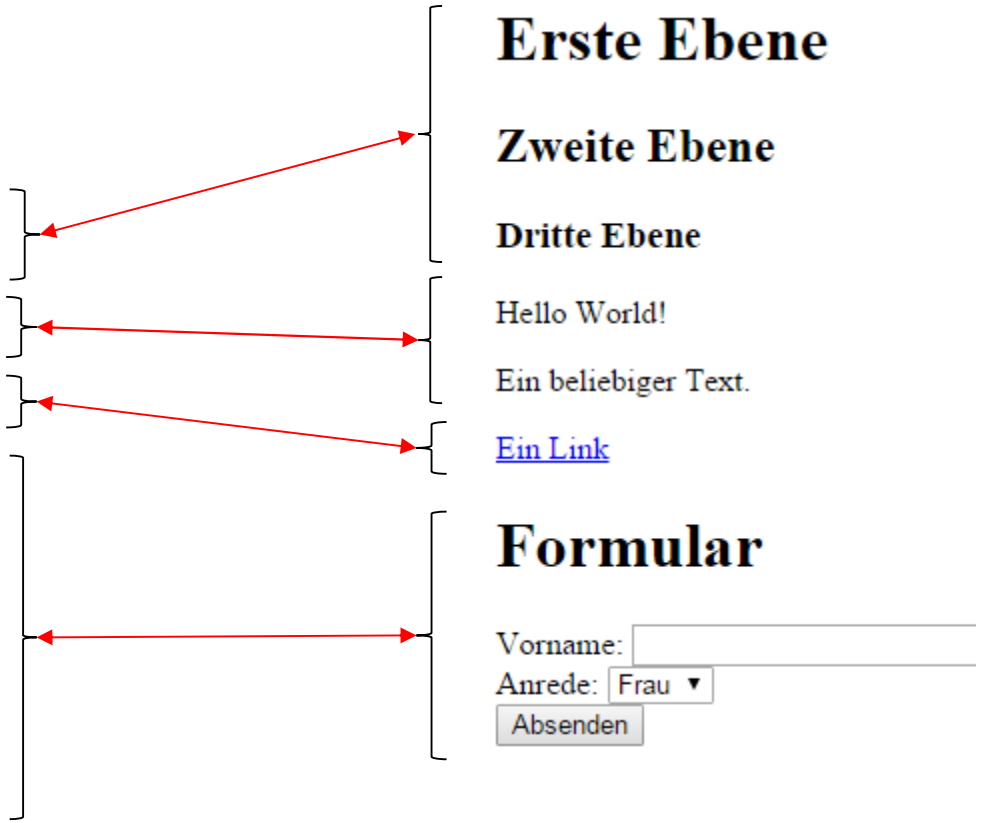
Aufbau von HTML-Dokumenten

- **Einige Tags sind in jedem HTML-Dokument notwendig:**
  - **Das Tag <html>…</html> umrahmt das gesamte Dokument**
  - **Das Tag <head>…</head> markiert den Dokumentkopf**
    - Im Kopf werden Metainformationen zum Dokument angegeben, die nicht vom Browser angezeigt werden, z.B. Autor, Datum
  - **Das Tag <body>…</body> umrahmt den eigentlichen Inhalt**

```
<html>
    <head>
        <title>Dies ist mein erstes HTML-Dokument</title>
    </head>
    <body>
        Hello World!
    </body>
</html>
```

Beispiel

```
<!DOCTYPE html>

<html lang="de">

<head>
<title>HTML-Test</title>
<meta charset="utf-8" />
</head>

<body>

<h1>Erste Ebene</h1>
<h2>Zweite Ebene</h2>
<h3>Dritte Ebene</h3>

<p>Hello World!</p>
<p>Ein beliebiger Text.</p>

<a href="http://www.google.com">Ein Link</a>

<form>
<h1>Formular</h1>
Vorname:
<input type="text" name="Vorname">
<br>
Anrede:
<select name="Anrede">
<option value="Frau">Frau</option>
<option value="Herr">Herr</option>
</select>
<br>
<input type="submit" value="Absenden" action="abc">
</form>

</body>

</html>
```



Julius-Maximilians-
UNIVERSITÄT
WÜRZBURG

Cascading Style Sheets

- **Der Fokus von HTML lag in den Anfangszeiten vor allem auf der „logischen“ Auszeichnung**
  - Logische Auszeichnung = Deklaration der Struktur von Seiten und der Rolle einzelner Elemente (z.B. Überschriften)
  - Die Formatierung (d.h. „physische“ Auszeichnung) wurde hingegen dem Browser überlassen

- **Je mehr HTML in den Alltag einzog, umso mehr wollten Webdesigner auch die Darstellung der Webseiten unter Kontrolle haben**

- **Um die zunehmend komplexen Formatierungsangaben handhaben zu können, wurde das Konzept der „Cascading Style Sheets“ für HTML umgesetzt**

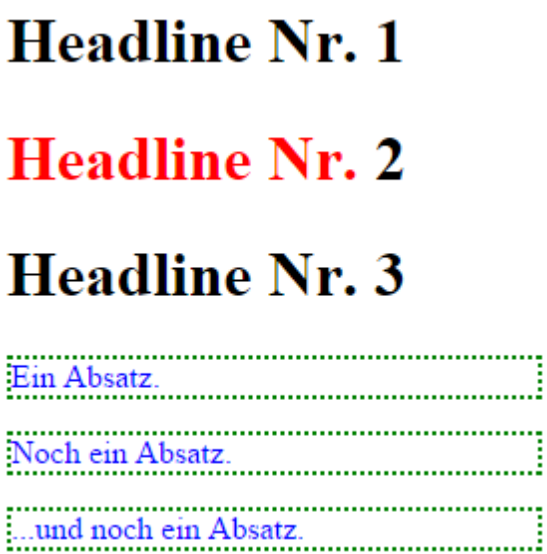
Beispiel

```
<!DOCTYPE html>
<html lang='de'>

<head>
<meta charset="UTF-8" />
<title>CSS-Beispiel</title>

<style>
p
{
color: blue;
border: 2px dotted green;
}

.RoteSchrift
{
color: red;
}

h1
{
font-size: 80px;
}
</style>

</head>

<body>

<h1>Headline Nr. 1</h1>
<h1><span class="RoteSchrift">Headline Nr.</span> 2</h1>
<h1>Headline Nr. 3</h1>
<p>Ein Absatz.</p>
<p>Noch ein Absatz.</p>
<p>...und noch ein Absatz.</p>

</body>
</html>
```

Style für normale Absätze („Paragraphs“)

Style für Elemente mit Klasse „RoteSchrift“

Style für h1-Elemente



Julius-Maximilians-
UNIVERSITÄT
WÜRZBURG

Internet

World Wide Web

ASP.NET MVC

Dynamische Webseiten

- **Webseiten zu Beginn als rein statische Inhalte**

- **Common Gateway Interface (CGI)**
  - **Erster Ansatz zur dynamischen Generierung von Webinhalten**
  - **Nachteil: Ressourcenverbrauch bzw. Performance**

- **Heute zwei Ansätze (auch in Kombination) üblich**
  - **Serverseitig**
    - Code wird innerhalb des Webservers ausgeführt
    - Implementierung mit PHP, Ruby, Python, C#, Java usw.
  - **Clientseitig**
    - Code wird mit HTML ausgeliefert und im Browser ausgeführt
    - Implementierung mit Javascript und JS-Bibliotheken (z.B. JQuery)
    - Extremfall „Single Page Applications“ (SPA)

Julius-Maximilians-
UNIVERSITÄT
WÜRZBURG

Active Server Pages (ASP)

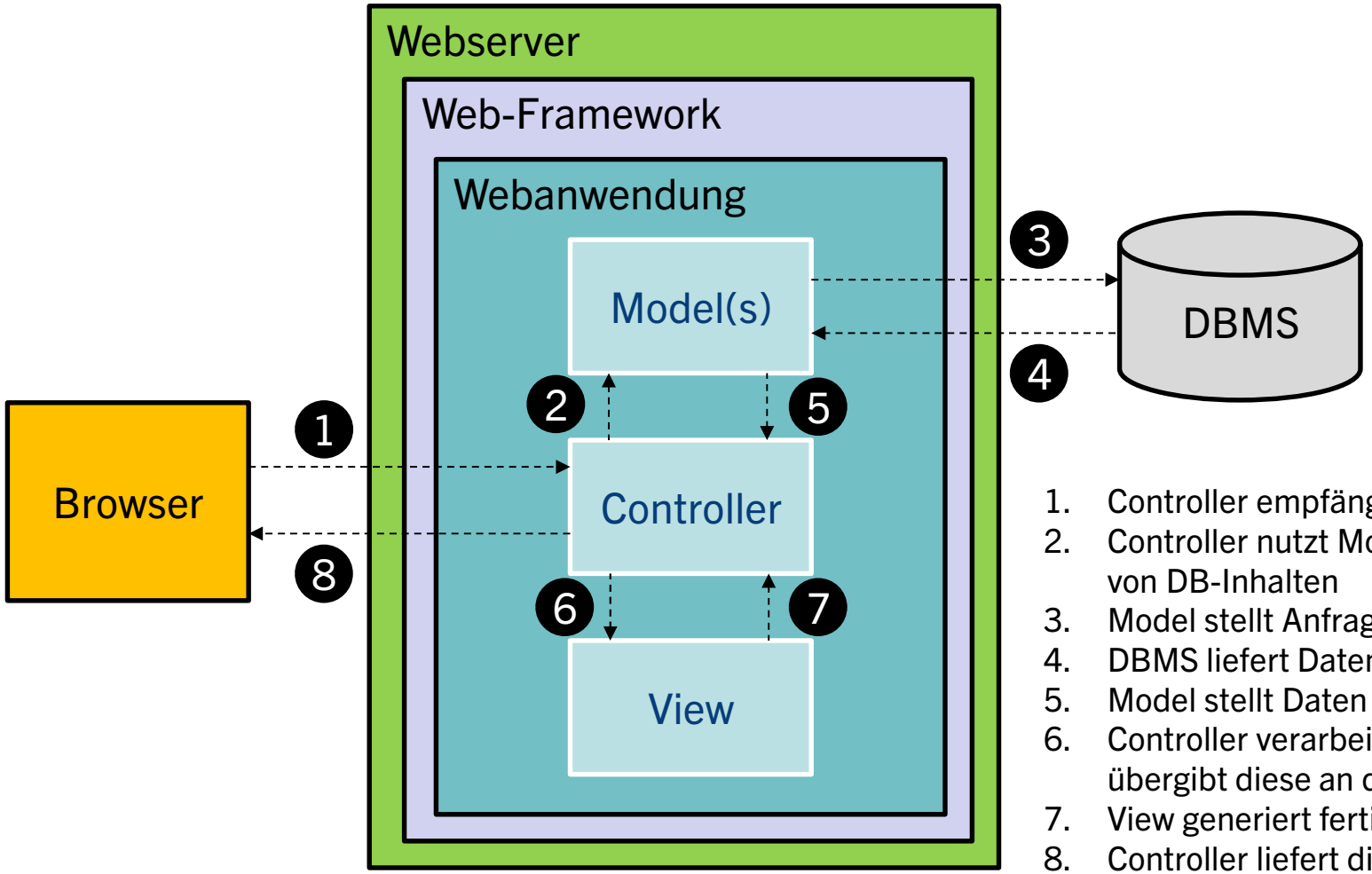
- **ASP = Oberbegriff für Webtechnologien von Microsoft**
  - Seit 2002 auf Grundlage von .NET
- **Ansatz „ASP.NET Websites“**
  - Konkurrenztechnologie insb. zu PHP
  - HTML-Seiten mit eingebettetem Code
- **Ansatz „ASP.NET Web Forms“**
  - Entwicklung analog zur Desktop-Programmierung
  - u.a. Performancenachteil durch Kommunikationsoverhead
- **ab 2007: Ansatz „ASP.NET MVC“**
  - 2016: Komplette Neukonzeption als „ASP.NET Core“

Julius-Maximilians-
UNIVERSITÄT
WÜRZBURG

MVC-Pattern

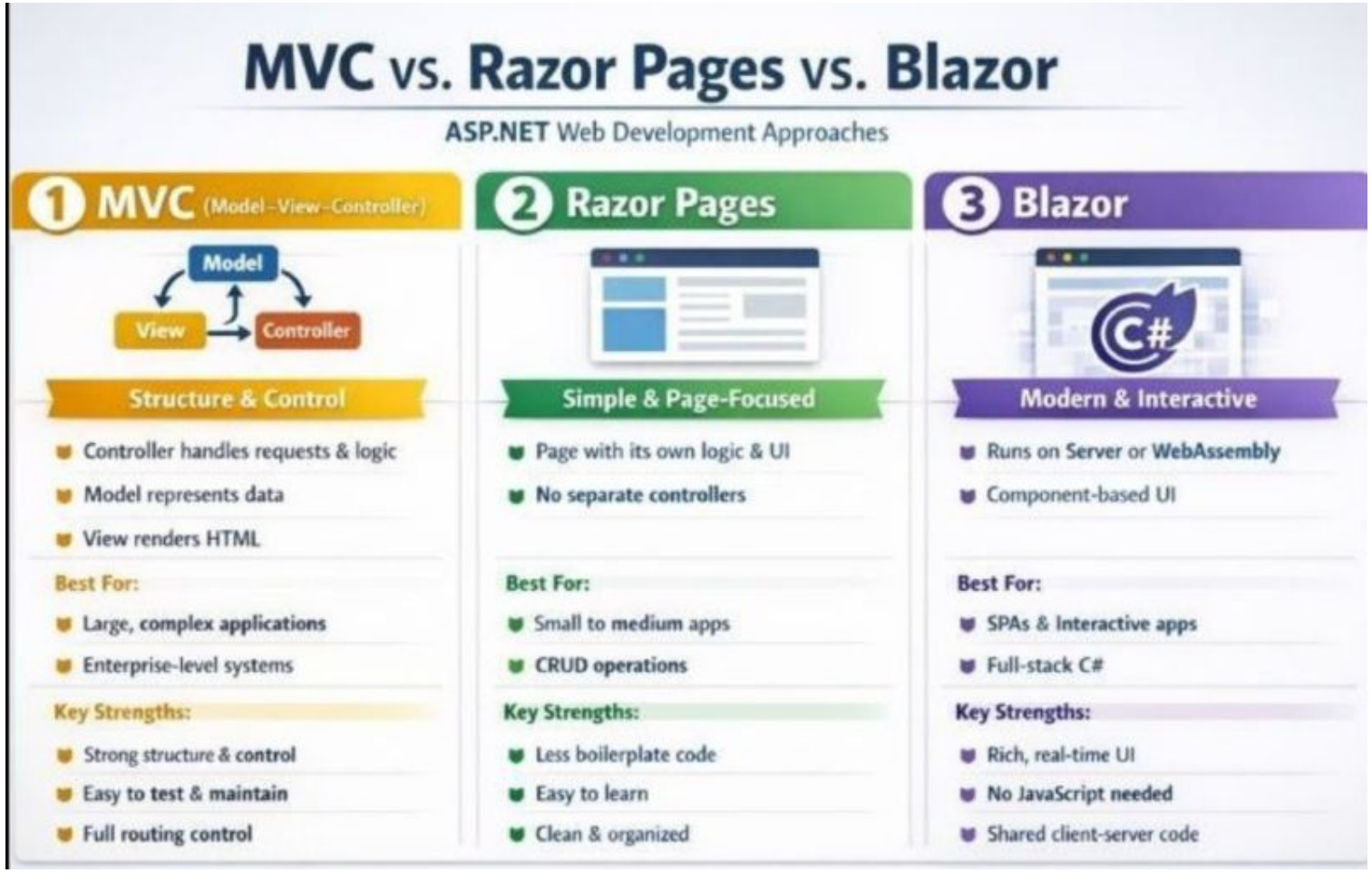
- **Ursprünge in der Entstehung der ersten objekt-orientierten Programmiersprachen (insb. Smalltalk)**
- **Entkopplung von Datenspeicherung, Verarbeitung und Darstellung**
- **Drei Arten von Komponenten**
  - Models = Komponenten zur Kommunikation mit der Datenbank
  - Views = Templates für die HTML-Seiten, welche dynamisch mit Daten gefüllt werden
  - Controllers = Komponenten zur Entgegennahme von Anfragen des Browsers und Generierung der Ergebnisse

Ablauf einer Anfrage an den Server



1. Controller empfängt Anfrage
2. Controller nutzt Model zur Abfrage von DB-Inhalten
3. Model stellt Anfrage an DBMS
4. DBMS liefert Daten zurück
5. Model stellt Daten bereit
6. Controller verarbeitet Daten und übergibt diese an die View
7. View generiert fertige Webseite
8. Controller liefert diese dem Client

MVC and beyond



Quelle: https://www.linkedin.com/posts/bandaravikas_csharp-sqlserver-aspdotnetcore-activity-7410144844662304768-YmZr

Vielen Dank!

Prof. Dr. Frédéric Thiesse
Lehrstuhl für Wirtschaftsinformatik und Systementwicklung
Julius-Maximilians-Universität Würzburg
Sanderring 2, 97070 Würzburg

+49 (0)931 31-80242
frederic.thiesse@uni-wuerzburg.de
www.wiwi.uni-wuerzburg.de/wise