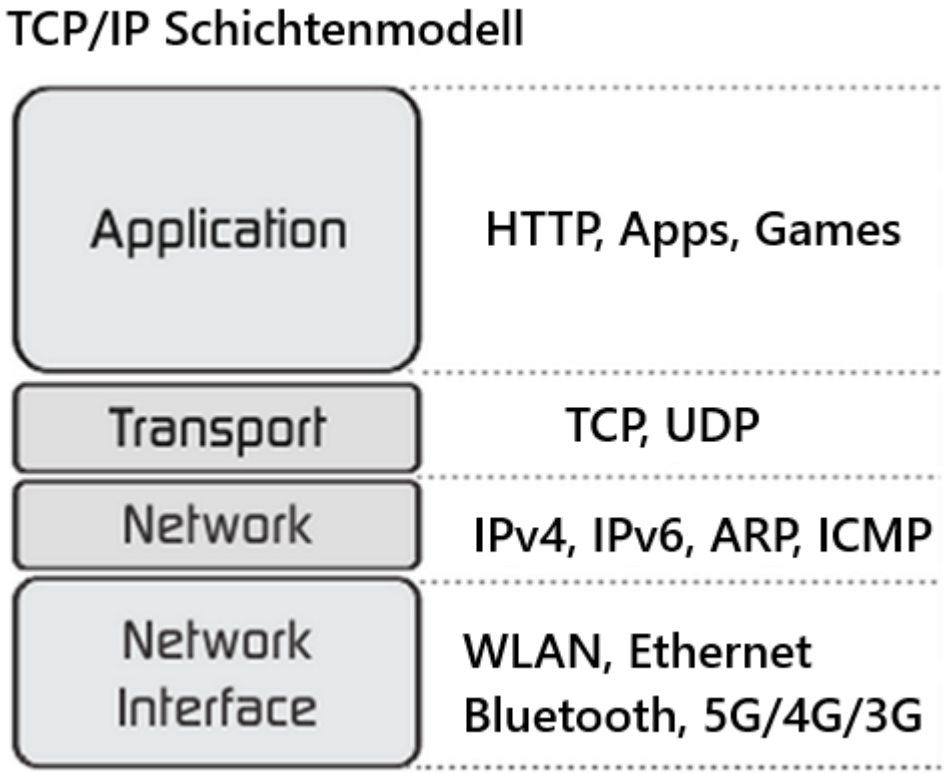
Netzwerke und Internet II – Teil A

Dr. Philipp Hurni, Kantonsschule Sursee

Netzwerke und Internet II

- Repetition Konzepte Netzwerke
  - Schichtenmodell
  - Verbindungsschicht
  - Netzwerkschicht
  - Transportschicht
  - Anwendungsschicht (WWW/Web)
- Experimentieren mit
  - Transportschicht
  - HTTP/Web
  - Meine erste Webseite!

Das Netzwerk-Schichtenmodell nach TCP/IP



Anwendung des Users (z.B. Web-Browser, Spotify-App, Game, etc.)

Ermöglicht (zuverlässige) **Punkt-zu-Punkt Kommunikation** zwischen zwei Computern, gegeben dass der Pfad gefunden ist.

Findet den **Pfad zum Ziel** im Netz

Steuert das **Sende- und Empfangs-Gerät**

Sorgt für die Kommunikation über das Medium. Konvertiert 1en und 0en zu Signalen.

Schichtenmodell "Pizzeria"

"obere Schicht befiehlt"

"untere Schicht handelt"

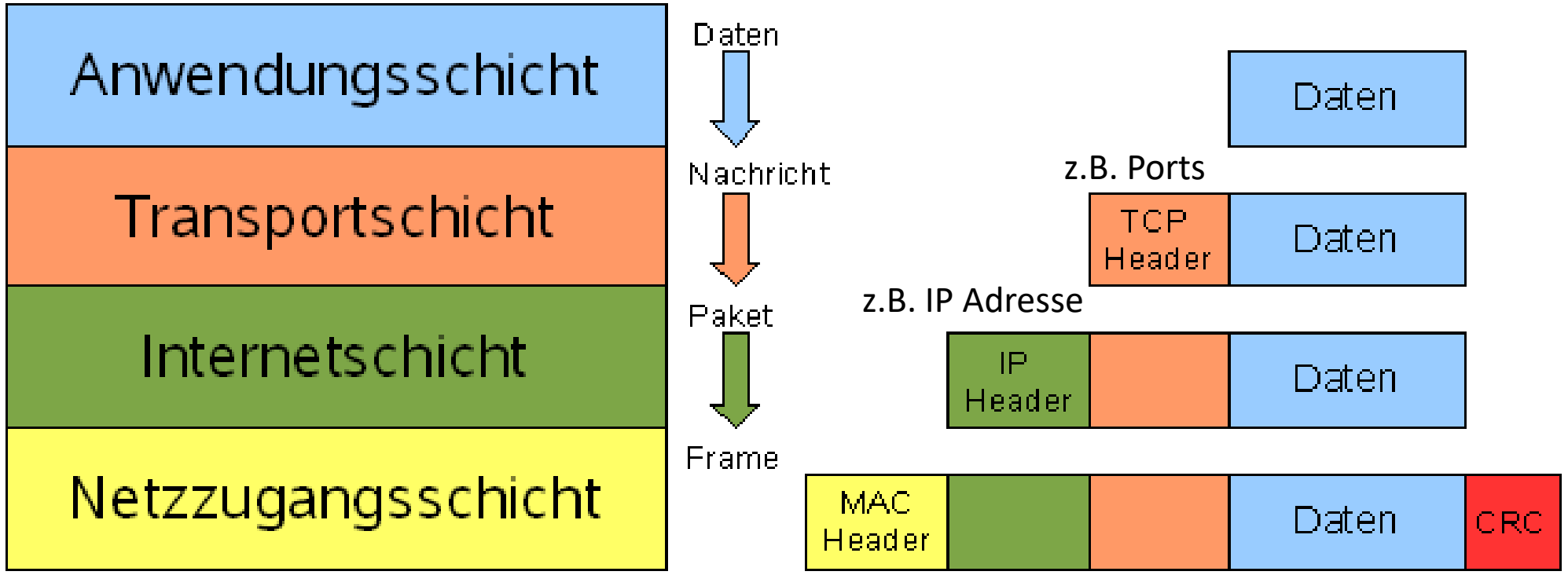
Pizzeria

Service-Schicht: der Kellner

Küchen-Schicht: der Pizzaiolo

Zulieferer-Schicht: der Grosshändler

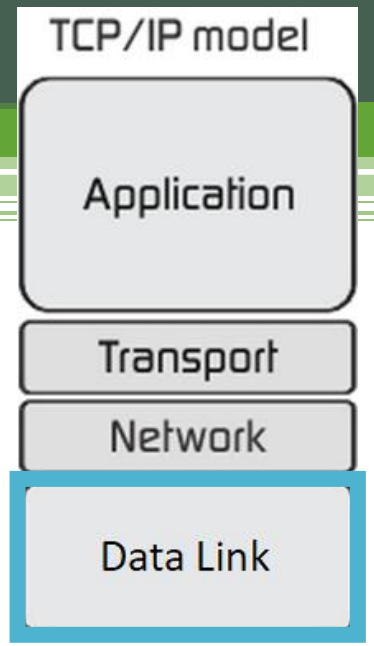
Das Netzwerk-Schichtenmodell nach TCP/IP



Schichten

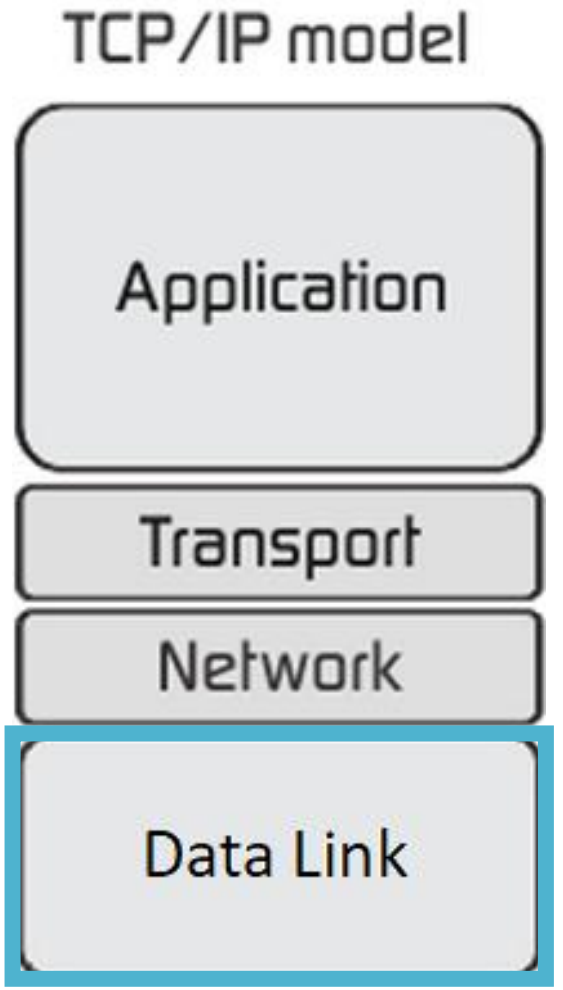
Entstehende Pakete / Frames

Verbindungsschicht / Netzzugangsschicht / Data Link Layer

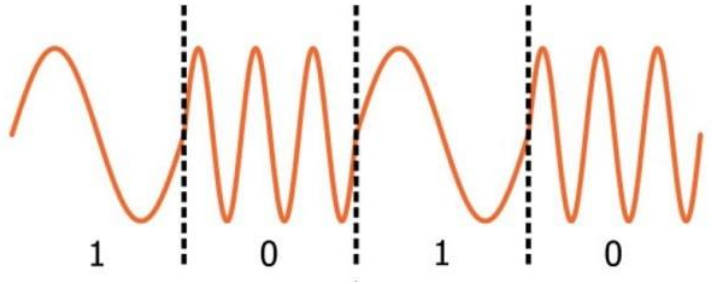


Verbindungsschicht (Data Link Layer)

- Häufig eingesetzte Technologien:
  - Ethernet 
  - WLAN 
  - Fiber 
  - 3G/4G/5G 



Historischer Exkurs: Audiosignale in der Telefonleitung



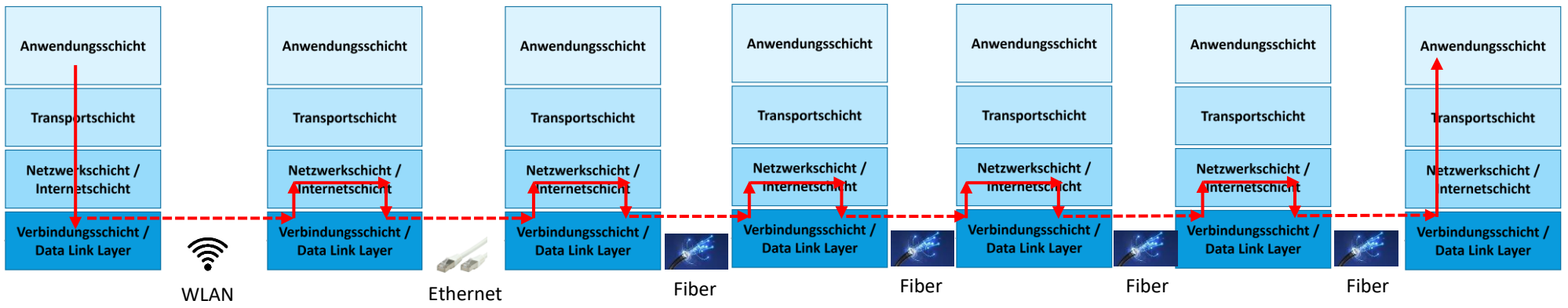
Netzwerkschicht

Netzwerkschicht (Network Layer)

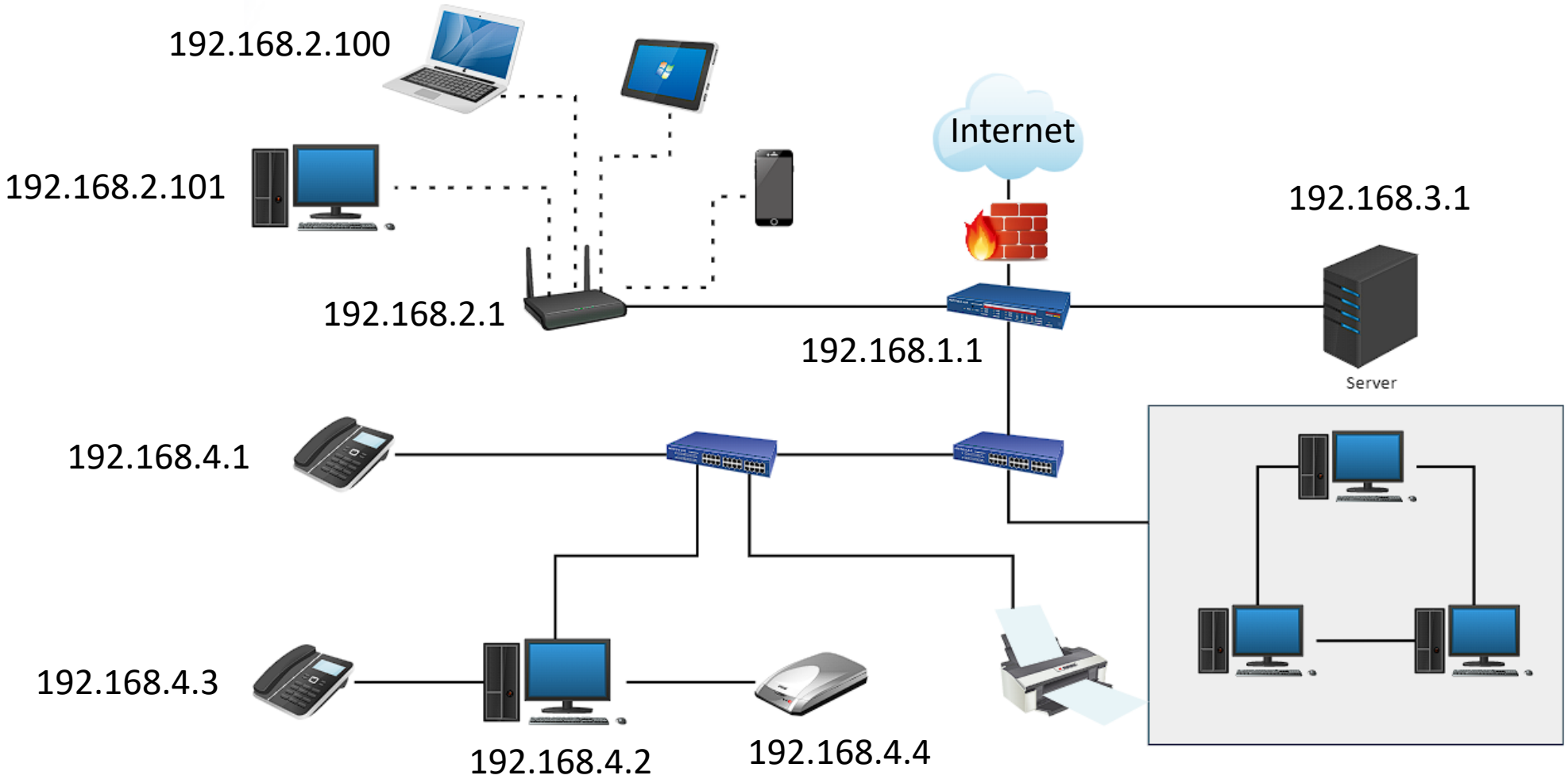
Kantonsschule Sursee
kssursee.lu.ch





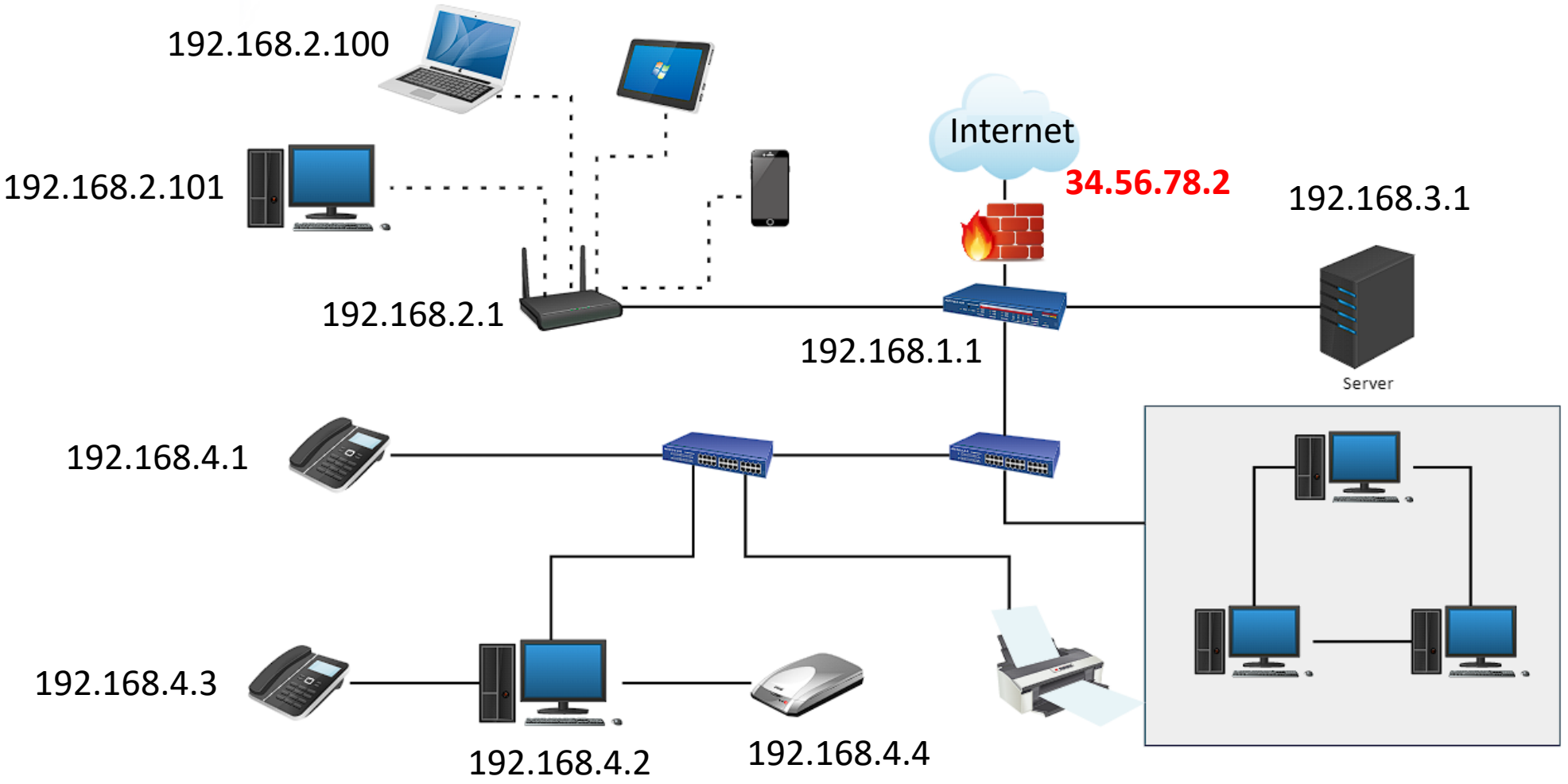



Lokale Netzwerke

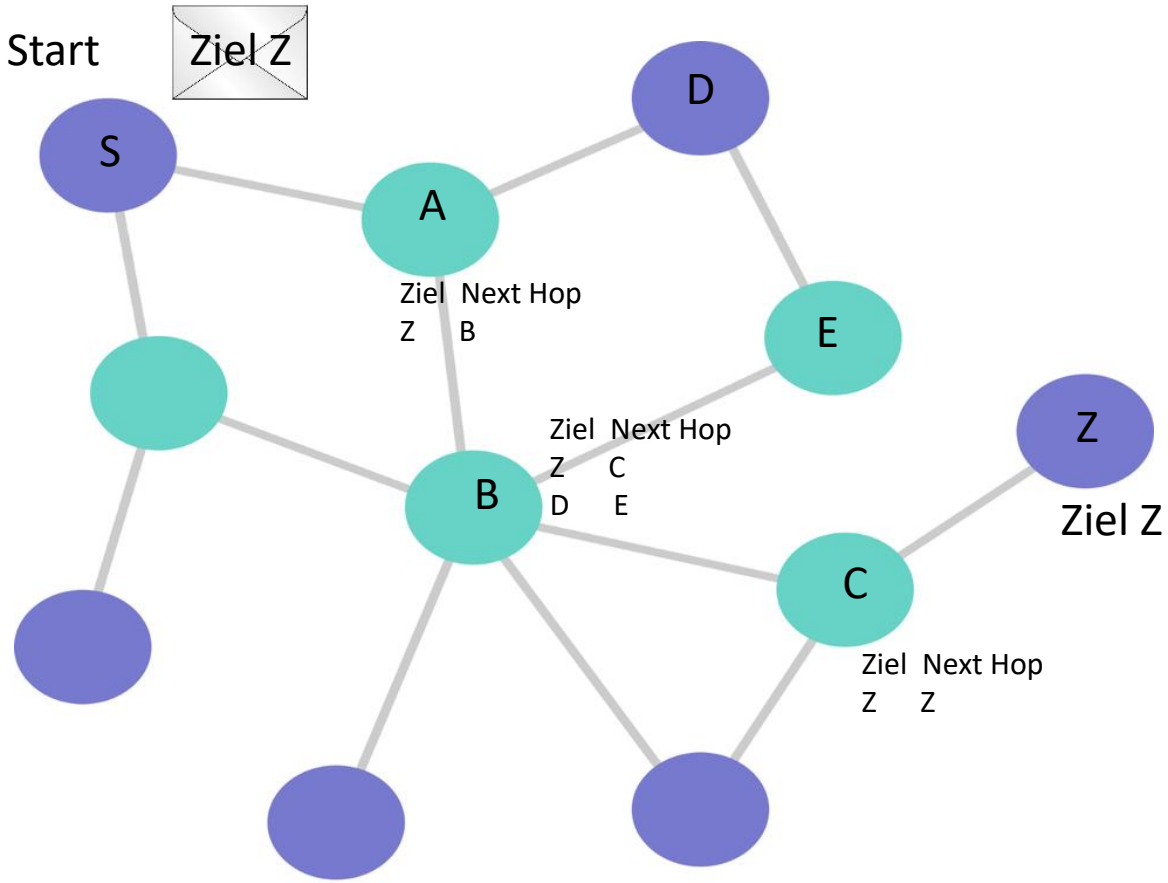


Globale IP Adressen

https://www.whatsmyip.org/

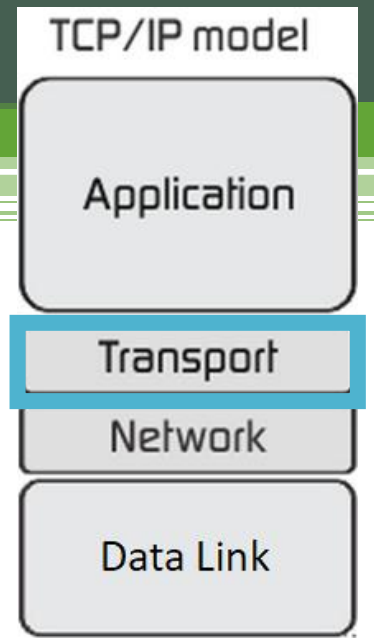


Tabellenbasiertes Routing



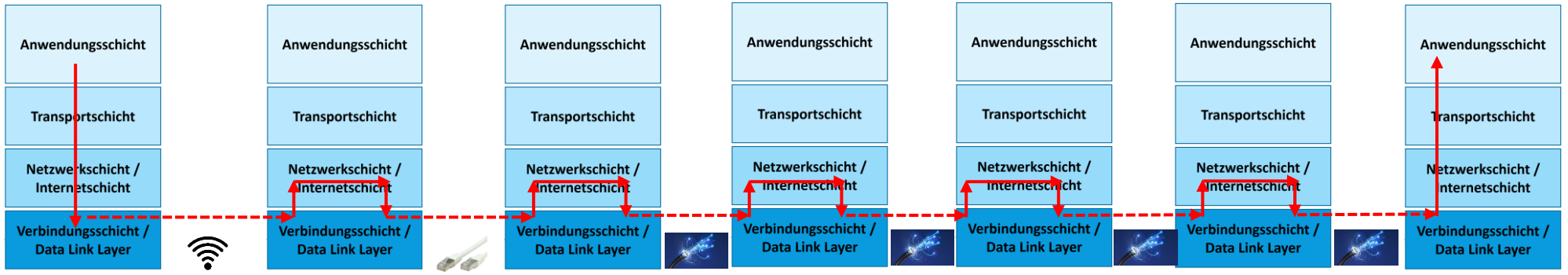
- **Intelligenz:** Die Router sind **intelligent**. Das Paket ist nicht intelligent.
- **Wegfindung:** Jeder Router sieht sich die Zieladresse des eingehenden Datenpakets an.
- Er schlägt diese Adresse in seiner **Routing-Tabelle** nach. Die Tabelle teilt dem Router mit, zum welchem nächsten Sprung **(Next Hop)** das Paket gesendet werden muss, um dem Ziel näherzukommen.
- Die Routeninformationen (für den nächsten Schritt) sind **dezentral** in den Routing-Tabellen jedes Routers gespeichert.

Transport



Transportschicht (nur Endpunkte)

Kantonsschule Sursee
kssursee.lu.ch



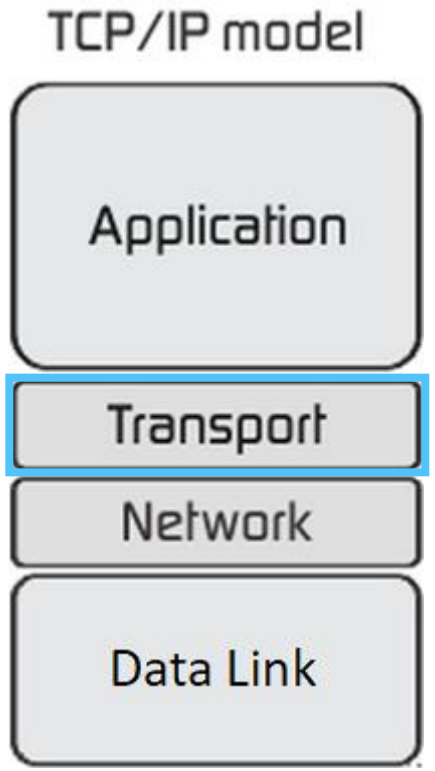
TCP - Protokoll

TCP ist ein **Verbindungsorientiertes Protokoll** – d.h. Server und Client etablieren zuerst eine Verbindung bevor Daten gesendet werden.

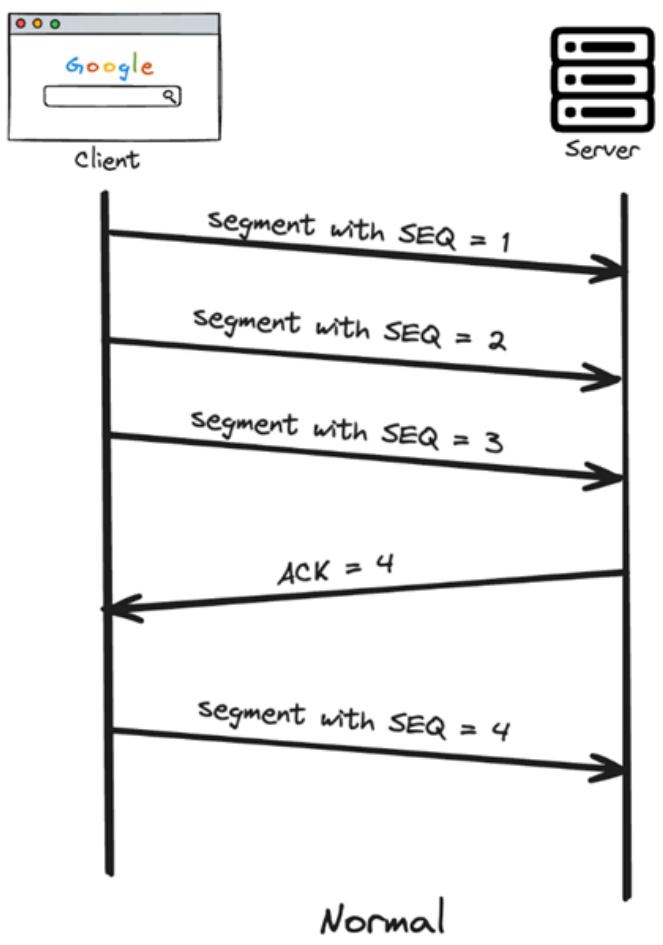
Es ist das Fundament für praktisch alle **Web-Technologien** im Internet.

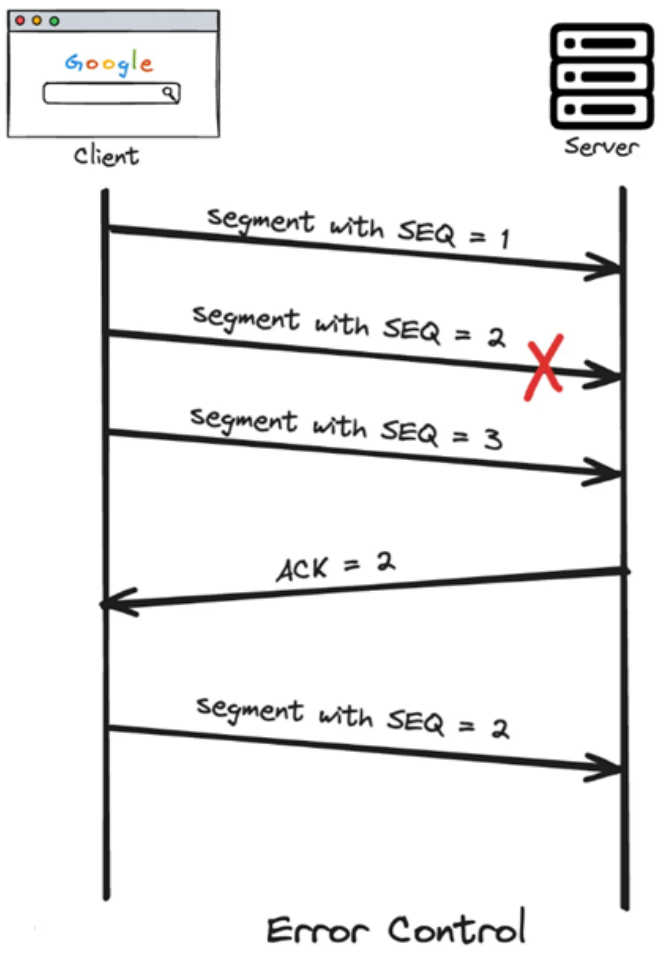
Eine Anwendung auf deinem Computer möchte nun eine Verbindung herstellen mit einer anderen Anwendung auf einem anderen Computer – beispielsweise einem Web Server.

Dein Computer formuliert nun eine Anfrage, und muss aber dabei angeben, mit welchem **Port** er kommunizieren will.

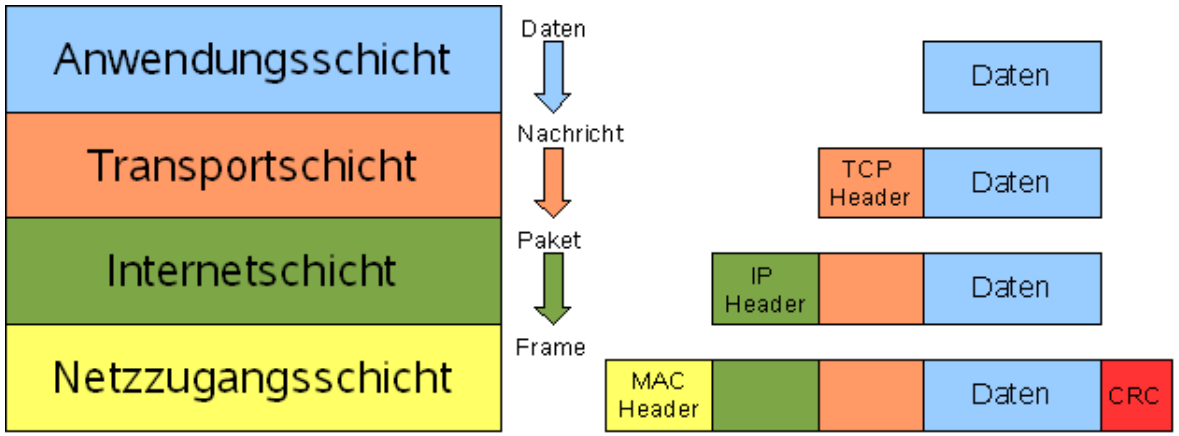


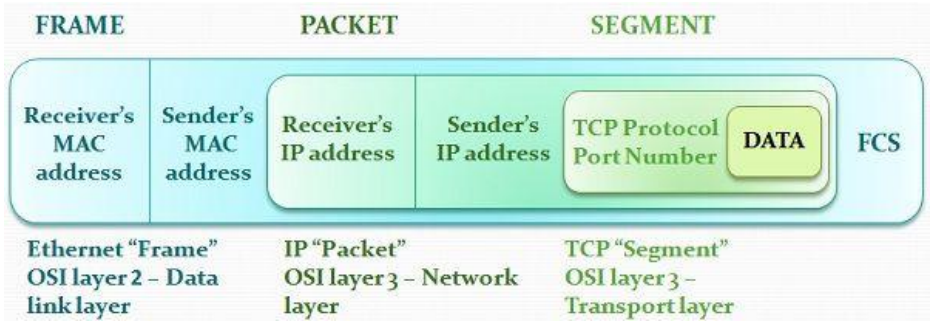
Fehlerbehebung mit TCP



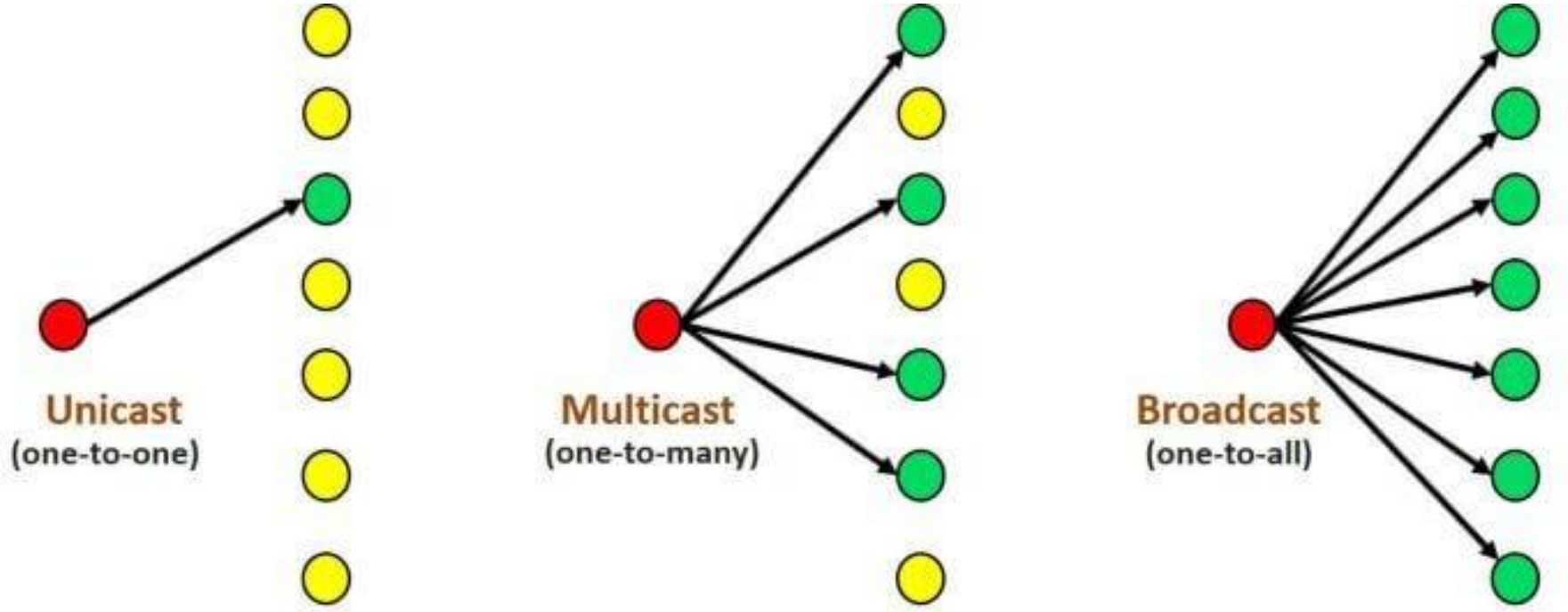


TCP Segment in IP Packet in Ethernet Frame



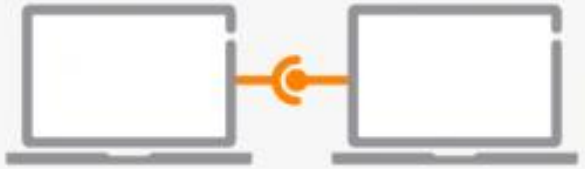


Übermittlungsmodi



TCP vs UDP

TCP



- Slower but more reliable transfers
- Typical Applications:
  - File Transfer Protocol (FTP)
  - Web Browsing
  - Email

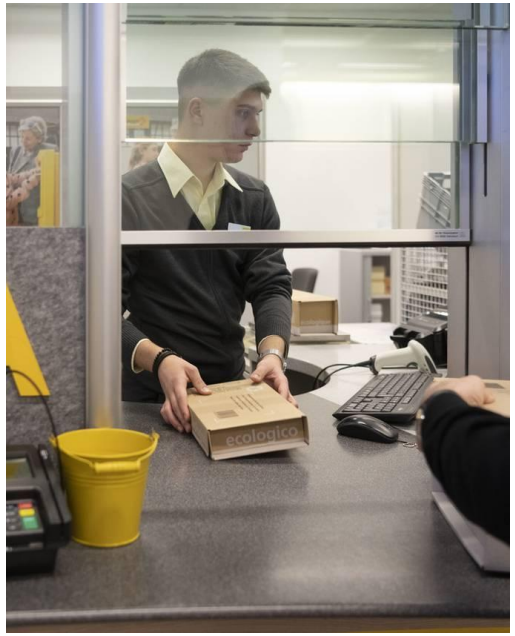
unicast

UDP

- Faster but not guaranteed transfers ("best effort")
- Typical Applications:
  - Live Streaming
  - Online Games
  - VoIP

unicast multicast broadcast

TCP vs UDP



UDP ~ Normaler A- oder B-Post Paket
~«Best Effort»

TCP ~ Eingeschriebener Paket
~ Ende-zu-Ende Bestätigung

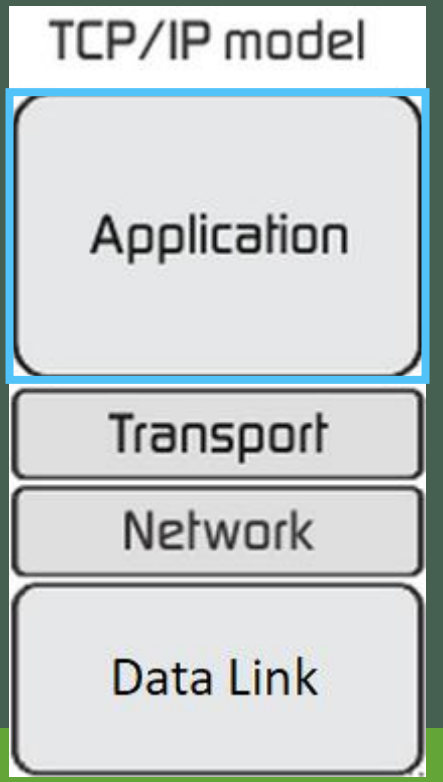
Was wird gebraucht für… Email?

Was wird gebraucht für… Web?

Was wird gebraucht für… IPTV?

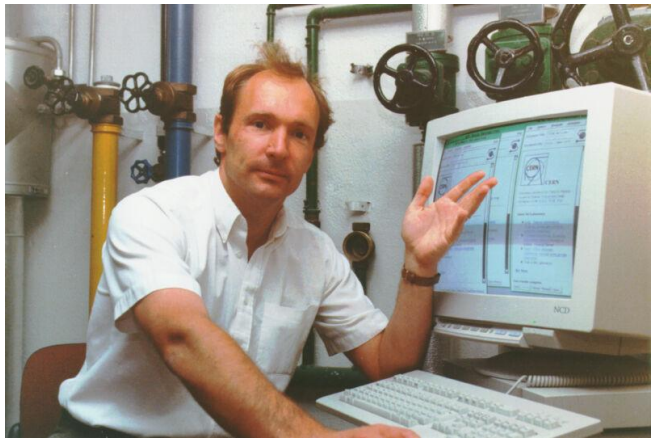
Web

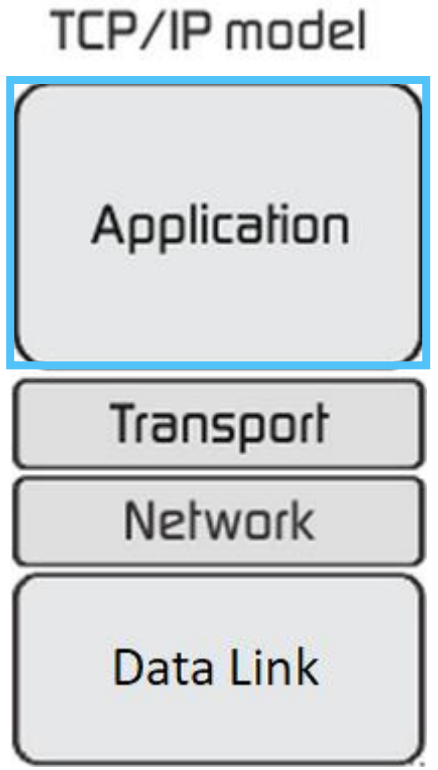

world wide web


TCP/IP model
Application
Transport
Network
Data Link

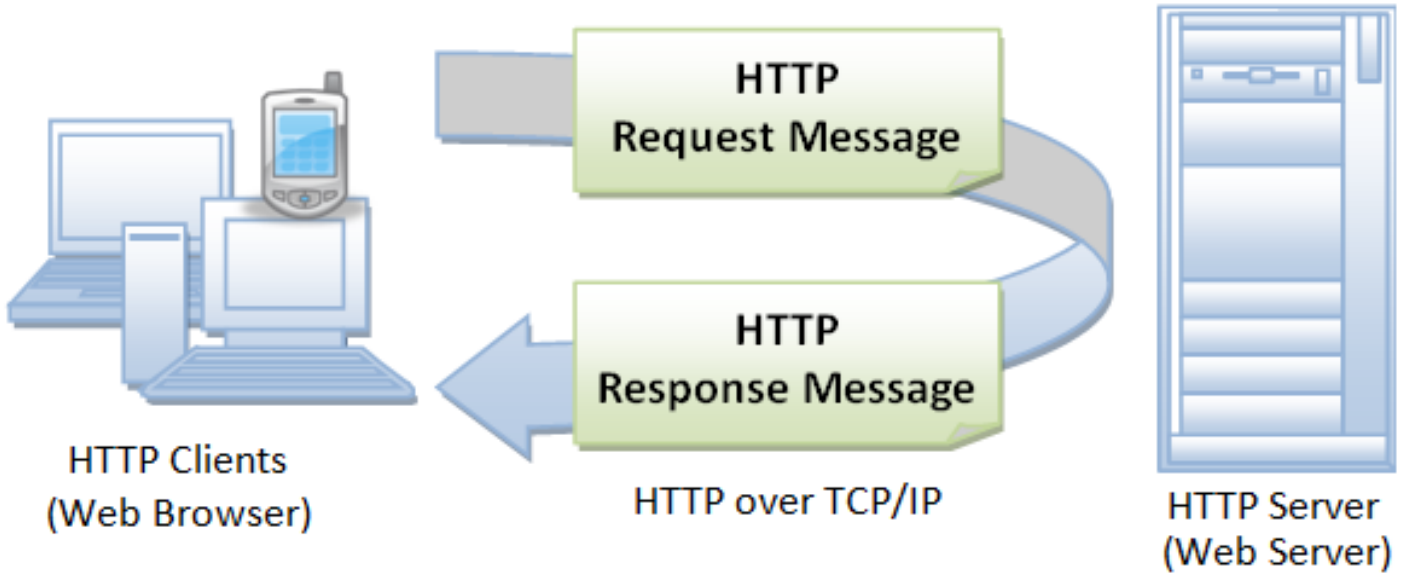
Das World-Wide-Web

Erst mit der Erfindung des **World Wide Web (WWW)** im Jahre 1989 wurde das Internet, welches damals schon 20 Jahre alt war, weltberühmt und entwickelte sich zu dem was wir heute kennen.





HTTP Request / Response Verfahren



Zugriff mit Python auf eine Webseite

- Wir öffnen die Webseite mit einem "Socket"
- client_socket = TCP Verbindung zu einem anderen Computer
- name = Domänenname im Domain Name System
- TCP port 80 = "Standard-Port" für Web
- while: Schleife für den Empfang – bis chunk leer ist
- Dann drucken der empfangenen Daten

```python
import socket
name = "www.hurni.org"
port = 80

# Funktion zum Abrufen der Webseite
def get_website():

    # Erstellen eines TCP-Sockets
    client_socket = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
    client_socket.connect((name, port))
    # HTTP-GET-Anfrage erstellen
    request = f"GET / HTTP/1.0\r\nHost: {name}\r\n\r\n"
    client_socket.sendall(request.encode('utf-8'))

    # Daten vom Server empfangen (Byte-weise)
    response = b""
    while True:
        chunk = client_socket.recv(4096) # Empfange maximal 4096 Bytes
        if not chunk:
            break # Keine Daten mehr
        response += chunk

    # Verbindung schliessen
    client_socket.close()

    # Ausgabe der empfangenen Daten (dekodiert)
    print("--- EMPFANGENE DATEN VOM SERVER ---")
    print(response.decode('latin-1'))

get_website() # Funktion aufrufen, um die Webseite abzurufen
```

Erhaltene Antwort vom Web-Server

- **Status Code (200 OK):** Die angeforderte Ressource (die Webseite) wurde erfolgreich gefunden und wird übertragen.
- **Server-Information:** Die Webseite wird von einem Apache/2.4.58 (Ubuntu)-Server bereitgestellt.
- **Zeichenkodierung (UTF-8):** Sowohl der HTTP-Header (Content-Type: text/html; charset=UTF-8) als auch das HTML-Dokument (<meta charset="UTF-8">) weisen den Browser an, den Inhalt als UTF-8 zu interpretieren.
- **HTML-Struktur:** Man sieht den Beginn eines gültigen HTML5-Dokuments mit den Tags <!DOCTYPE html>, <html>, <head>, <title> und dem Beginn des <style>-Blocks

Protocol Version

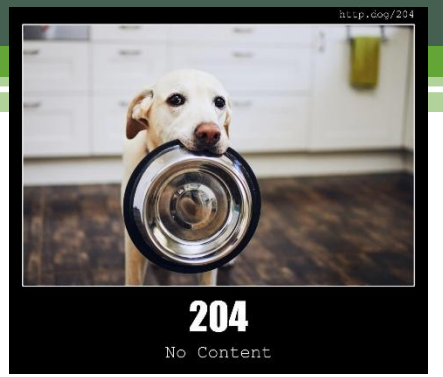
Status Code

```
HTTP/1.1 200 OK
Date: Mon, 08 Dec 2025 09:35:12 GMT
Server: Apache/2.4.58 (Ubuntu)
Vary: Accept-Encoding
Access-Control-Allow-Origin: *
Content-Security-Policy: frame-ancestors *;
Content-Length: 4726
Connection: close
Content-Type: text/html; charset=UTF-8
```

Status Headers

```
<!DOCTYPE html>
<html lang="de">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Hurni's Bastelecke</title>
    <style>
        body {
            font-family: 'Segoe UI', Tahoma, Geneva, Verdana, sans-serif;
            background-color: #f4f4f9;
            color: #333;
            display: flex;
            justify-content: center;
            align-items: center;
            height: 100vh;
            margin: 0;
            text-align: center;
        }
```

Status Codes



HTTP Status Codes

1xx Informational

100	Continue
101	Switching protocols
102	Processing
103	Early Hints

2xx Succesful

200	OK
201	Created
202	Accepted
203	Non-Authoritative
204	No Content
205	Reset Content
206	Partial Content
207	Multi-Status
208	Already Reported

3xx Redirection

301	Moved Permanently
302	Found
303	See Other
304	Not Modified

4xx Client Error

400	Bad Request
401	Unauthorized
402	Payment required
403	Forbidden
404	Not Found
405	Method Not Allowed
406	Not Acceptable
409	Conflict
413	Payload Too Large
429	Too Many Requests

5xx Server Error

500	Internal Server Error
501	Not Implemented
502	Bad Gateway
503	Service Unavailable
504	Gateway Timeout
505	HTTP Version Not Supported
506	Variant Also Negotiates
507	Insufficient Storage
508	Loop Detected
510	Not Extended
511	Network Authentication Required

Aufgabe: Speichern der empfangenen Datei

- **Markiert** allen Inhalt von <!DOCTYPE html> bis </html>
- Speichert diesen Inhalt unter einer Datei lokal mit Endung ".html" – z.b. file.html
- Öffnet die Datei mit einem Browser (Doppelklick)

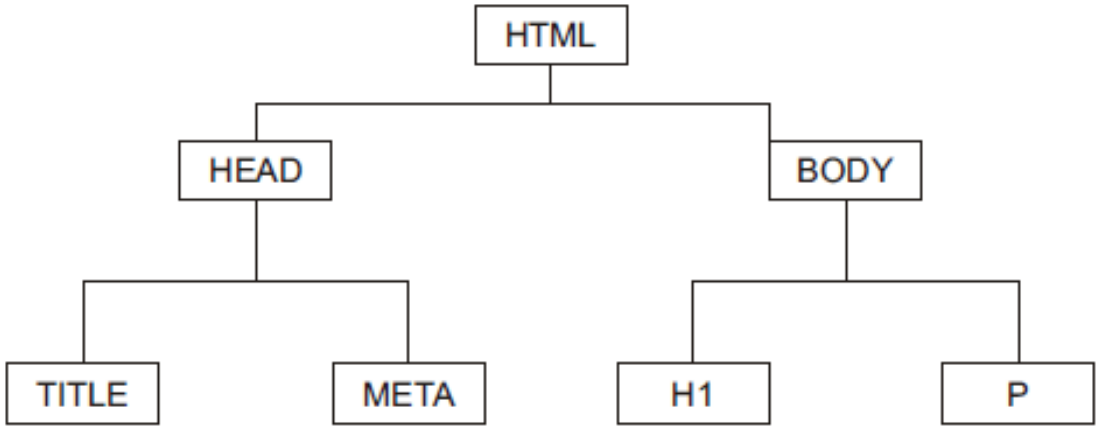
```
<!DOCTYPE html>
<html lang="de">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-
   scale=1.0">
    <title>Hurni's Bastelecke</title>
    <style>
        HIER IST CSS CODE
    </style>
</head>
<body>
    <div class="container">
        <h1>Hurni's Bastelecke</h1>
        <p>Spielwiese fuer diverse Projekte!</p>
    </div>

    <script>
        HIER IST JAVASCRIPT
    </script>
</body>
</html>
```

Hyper Text Markup Language (HTML)

- HTML ist eine hierarchisch strukturierte Seitenbeschreibungssprache. In HTML werden das Layout (style) und der Inhalt (content) von Webseiten beschrieben.
- Die Informationen werden in einem Baum, dem Dokumentenbaum abgelegt.
- Ein einfaches Beispiel für so einen Dokumentenbaum ist in der folgenden Abbildung zu sehen. In der HTML-Datei wird der Baum im Klartext anhand von sogenannten HTML-Tags aufgebaut.

HTML Head und Body

```
<html>
    <head>
        <title>Titel der Seite</title>
    </head>
    <body>
        <h1>Dies ist eine Webseite</h1>
        <p>Wirklich! Es ist eine!</p>
    </body>
</html>
```



Unsere erste Webseite

- Wir benutzen den Online-Webseiten Editor von OneCompiler.com

  Link: https://onecompiler.com/html

- Wir speichern diese Struktur unter dem index.html tab

HTML – alles ist definiert durch sogenannte Tags!

Hyper Text Markup Language (HTML)

- Das HTML-Grundgerüst besteht aus den Tags, die minimal vorhanden sein müssen, um eine HTML-Seite darzustellen. Dies ist zuerst das alles umschliessende html-tag. Mit **<html>** fängt jede Webseite an und mit **</html>** hört auch jede Webseite auf.

- Innerhalb des html-tags sind die zwei Unterbereiche mit **<head>** und **<body>** bezeichnet. Dabei ist **<head>** der Container für alle Metadaten betreffs der Webseite. Im **<body>** sind alle darzustellenden Informationen enthalten.

- Bei den Metadaten ist der **<title>** vorgeschrieben. Dieses Feld enthält den Text, den ein Browser in der Titelzeile seines Fensters anzeigen soll.

Farbe

- Innerhalb des Grundgerüstes kann die Hintergrundfarbe für die HTML-Datei gewählt werden. Das **bgcolor**-Attribut im **<body>**-tag legt diese Farbe für das ganze Dokument fest:

```
<body bgcolor="green">
...
</body>
```

- Die Werte für dieses Attribut können definierte Farbnamen oder mit Hexadezimalzahlen angegebene Farbwerte (RGB-Modell!) angegeben werden.
- Für die Strukturierung von Text stehen ähnliche Mittel zur Auswahl, wie sie eine einfache Text-Verarbeitungsprogramme (z.B. Word) zur Verfügung stellt.
- Normaler Fliesstext wird in Absätzen (paragraph) geschrieben. Jeder Absatz wird dabei von einem <p>-tag umschlossen.

Text

Um Fliesstext weiter strukturieren zu können stellt HTML noch abgestufte Arten von Überschriften bereit. Diese Tags heissen **<h1>** bis **<h6>** in Reihenfolge ihrer Wertigkeit. So ist **<h1>** eine "grössere" Überschrift als **<h2>** usw.

```
<html>
   <head>
      <title>Titel der Seite</title>
   </head>
   <body>
      <h1>HTML-Grundlagen</h1>
        <p>HTML ist eine hierarchisch strukturierte Seitenbeschreibungssprache. In HTML werden das Layout ... </p>
      <h2>Das HTML-Grundgeruest</h2>
        <p>Das HTML-Grundgeruest besteht aus den Tags, die vorhanden sein muessen, um eine HTML-Seite darzustellen... </p>
      <h1>Auszeichnung von Text</h1>
        <p>Fuer die Strukturierung von Text stehen aehnliche Mittel zur Auswahl, wie sie eine einfache Textverarbeitung...</p>
      <h2>Fliesstext</h2>
        <p>Normaler Fliesstext wird in Absaetzen (paragraph) geschrieben. Jeder Absatz wird dabei ... </p>
     </body>
</html>
```

Bilder einfügen

Das Einbinden von Bildern ist in HTML vergleichsweise einfach. Das **<img>**-tag, das für die Einbindung von Grafiken zuständig ist, kann einfach an der Stelle, an welcher die Grafik erscheinen soll, im laufenden HTML-Text eingebunden werden.

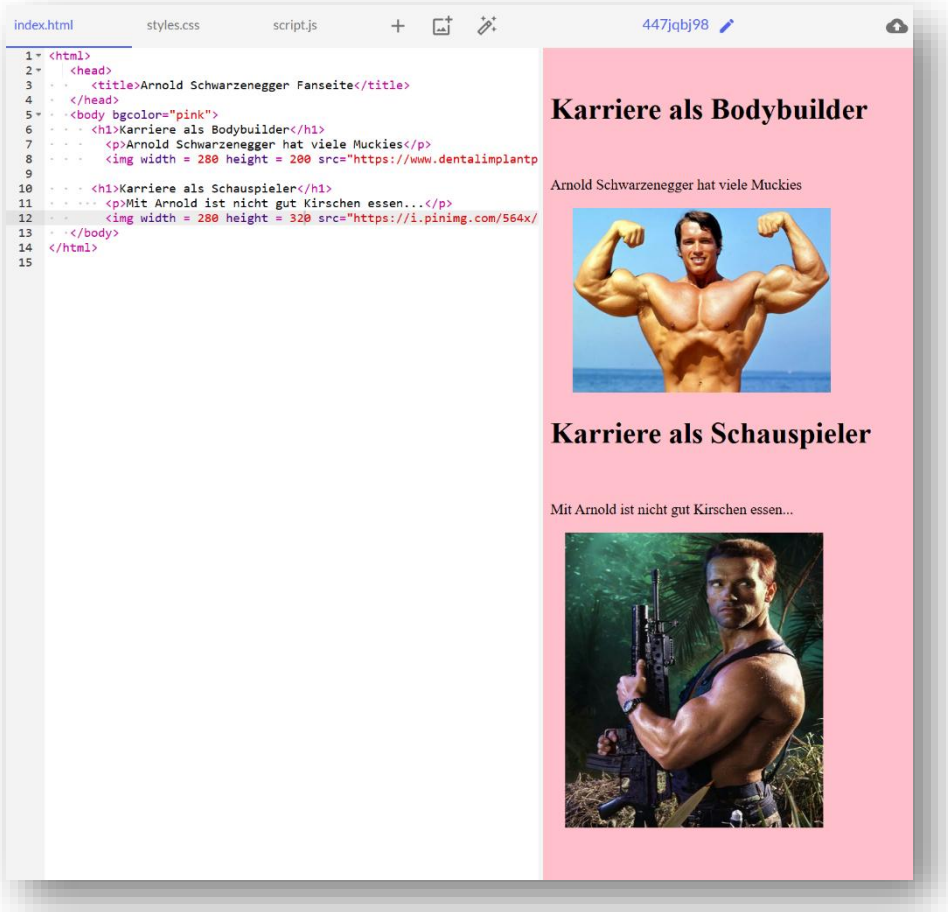
```
<img src="https://kssursee.lu.ch/-/media/KSSursee/Bilder/Schule/Schuelerorganisation/Schuelerforum.JPG" alt="Schulforum"/>
```

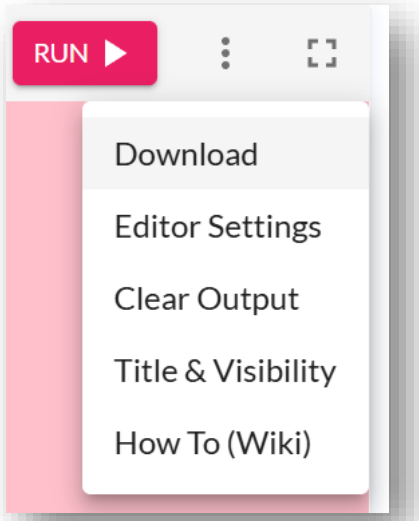
- Pflichtattribute **src** und **alt**.
  - Mit **src** wird der Speicherort der Grafikdatei angegeben. Diese kann lokal sein oder auf eine andere Seite verweisen – wie hier oben
  - **alt** steht für „alternative information“. Der Alternativtext wird vom Browser angezeigt, wenn ein Bild nicht geladen werden kann. Die Gründe für ein nicht geladenes Bild sind vielfältig. So kann ein Benutzer das Laden von Bildern ganz abgeschaltet haben (Anti-Phishing-Filter oder einfach nur wegen einer schmalbandigen Internetverbindung). Oder der Benutzer verwendet einen Textbasierten Browser
  - **width** und **height** passt die Grösse des Bildes an – Angabe in Pixel (z.B. width="500")

Aufgabe: gestaltet eure erste Webseite !

- Nutzt wieder https://onecompiler.com/html
- Ändert Farbe und Text
- Fügt ein Bild ein (mit Link zu diesem Bild) – skaliert das Bild mit width und height

Aufgabe: gestaltet eure erste Webseite

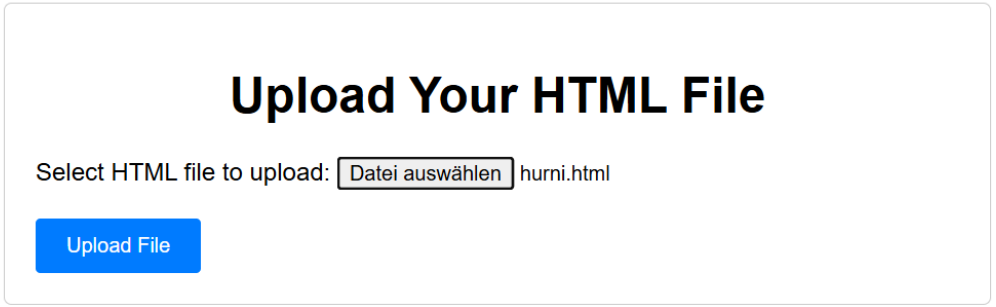




Speichert sie unter eurem Namen, ohne Leerzeichen

Parat zum Hochladen!

- Einfache Webseite zum Hochladen und betrachten im Web
- https://hurni.4lima.de/uploader/

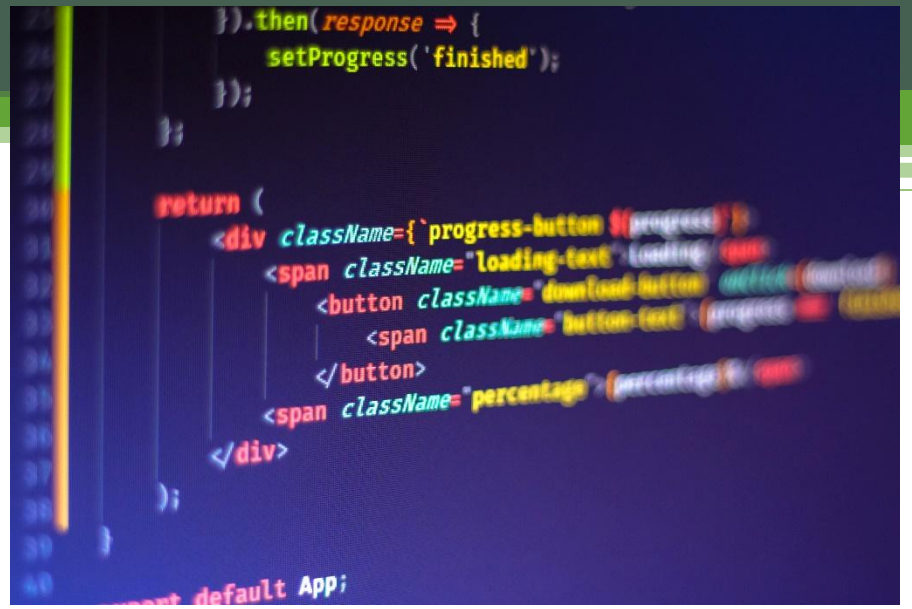


Cascading Style Sheet

Cascading Style Sheets (CSS)

- CSS ist eine Sprache für elektronische Dokumente. Zusammen mit HTML und JavaScript ist es eine der Kernsprachen des World Wide Webs
- Sie ist wird vom World Wide Web Consortium (W3C) laufend weiterentwickelt.
- Mit dieser Sprache kann man das optische Erscheinungsbild einer Webseite "zentral" steuern, ohne sämtliche Seiten einzeln anpassen zu müssen

Javascript


}).then(response => {
setProgress('finished');
});
};
return (
<div className={`progress-button
<span className="loading-text"
<button className="download-button"
<span className="button-text"
</button>
<span className="percentage"

);
}
export default App;

JavaScript

- JavaScript (kurz JS) ist eine Skriptsprache, die dafür entwickelt wurde, Benutzerinteraktionen auszuwerten, Inhalte zu verändern, nachzuladen oder zu generieren und so die Möglichkeiten von HTML zu erweitern.

- JavaScript wird hauptsächlich für die Entwicklung interaktiver Webseiten eingesetzt – aber längst nicht nur. Es wird mitunter auch verwendet, um Mikrocontroller zu programmieren