

E-Business und digitale Prozesse

Übung

API und Datenverarbeitung:

Beispiel Open Transport und JavaScript

Prof. Dr. Thomas Myrach
Universität Bern
Institut für Wirtschaftsinformatik
Abteilung Informationsmanagement

Stand: 16. September 2022

- Sie kennen die grundsätzlichen Möglichkeiten des Datenaustausches zwischen Anwendungssystemen und wie sich diese im World Wide Web darstellen.
- Sie wissen wie Dateien im World Wide Web aufgerufen und verarbeitet werden.
- Sie wissen, wie Daten im Internet über Programmierschnittstellen (API's) abgerufen werden können.
- Sie können sehr einfache Web-Applikationen mit der Skriptsprache JavaScript realisieren.
- Sie können eine Web-API im Rahmen einer einfachen Web-Applikation nutzen.

Agenda: Übersicht

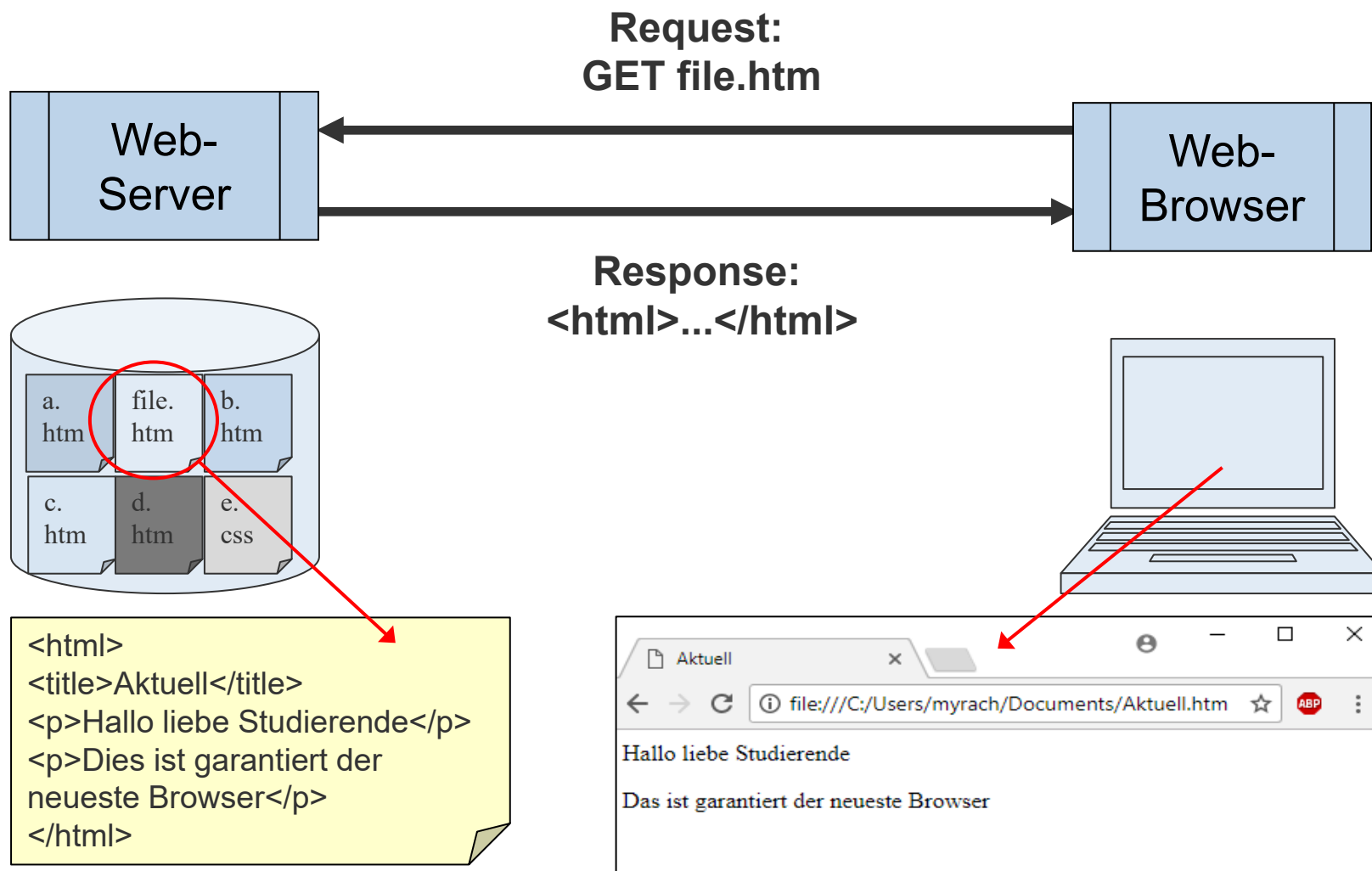
Datenaustausch über das Internet

Grundlagen JavaScript

Verarbeitung übertragener Daten in JavaScript

Datenaustausch im Internet

Aufruf von Ressourcen über einen Web-Server



Datenaustausch im Internet

Verarbeitung der Ressourcen im Web-Browser

- Ein Web-Browser kann vor allem das HTML-Datenformat interpretieren und entsprechend anzeigen.
- Ein Web-Browser kann auch andere Datenformate verarbeiten und anzeigen, insbesondere diverse Bildformate.
- Weitere Optionen:
 - Anzeige und Verarbeitung der Dateien über spezielle Browser-Plugins.
 - Vor allem: PDF-Dateien.
 - Download der Dateien und Verarbeitung mit anderen Programmen.
 - Alle möglichen Formate.

Download von Dateien

Beispiel: Bundesamt für Statistik

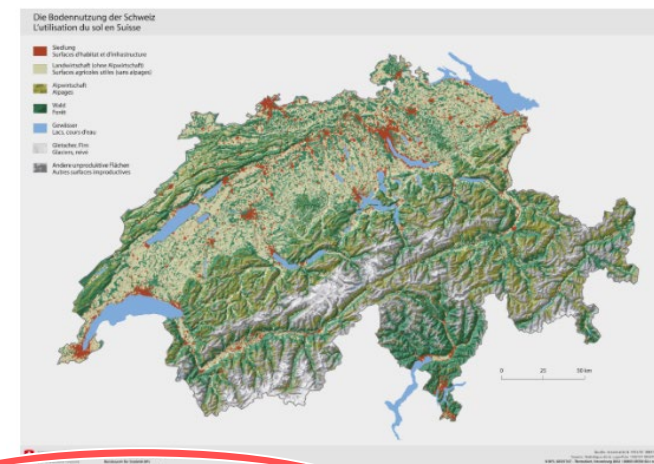
Ständige Wohnbevölkerung ab 15 Jahren nach Geburtsort der Eltern



[Download Tabelle](#)
(XLS, 43 kB)

Typ	Tabelle
Veröffentlicht am	31.01.2017
Dargestellter Zeitraum	2015
Herausgeber	Bundesamt für Statistik
Thema	Bevölkerung
BFS-Nummer	su-d-40.02.01.05.05-2015
Sprache	DE
Weitere Sprachen	FR, IT
Erhebung(en)	Strukturerhebung

Die Bodennutzung der Schweiz



[Download Karte \(PNG, 2 MB\)](#)
[Download Karte \(PDF, 3 MB\)](#)

Typ	Karte
Veröffentlicht am	01.01.2002
Herausgeber	Bundesamt für Statistik
Raumgliederung	Schweiz / Schweiz
Dargestellter Zeitraum	1.1.2002
Thema	Raum und Umwelt
BFS-Nummer	KM05-00354-02-c-suis-2002-df
Sprache	DE/FR
Erhebung(en)	Arealstatistik der Schweiz
Copyright	© BFS/OFS/UST/FSO

Abruf von Daten

API und Internet

- Ein Application Programming Interface (API) bzw. eine Programmierschnittstelle ermöglicht zwei Softwareprogrammen miteinander zu kommunizieren.
- Web-APIs (REST):
 - Web-URL werden über ein einfaches HTTP-Protokoll aufgerufen.
 - Ergebnisse können (als Text) in verschiedenen Formaten zurückgegeben werden.
- Web-Services (SOAP):
 - Die Kommunikation (Frage, Antwort) erfolgt typischerweise über das SOAP-Protokoll in einem XML-Format.
 - Für jeden Web-Service bestimmt eine Schnittstellenbeschreibung in einem maschinenlesbarem Format, wie mit dem Webservice zu interagieren ist.

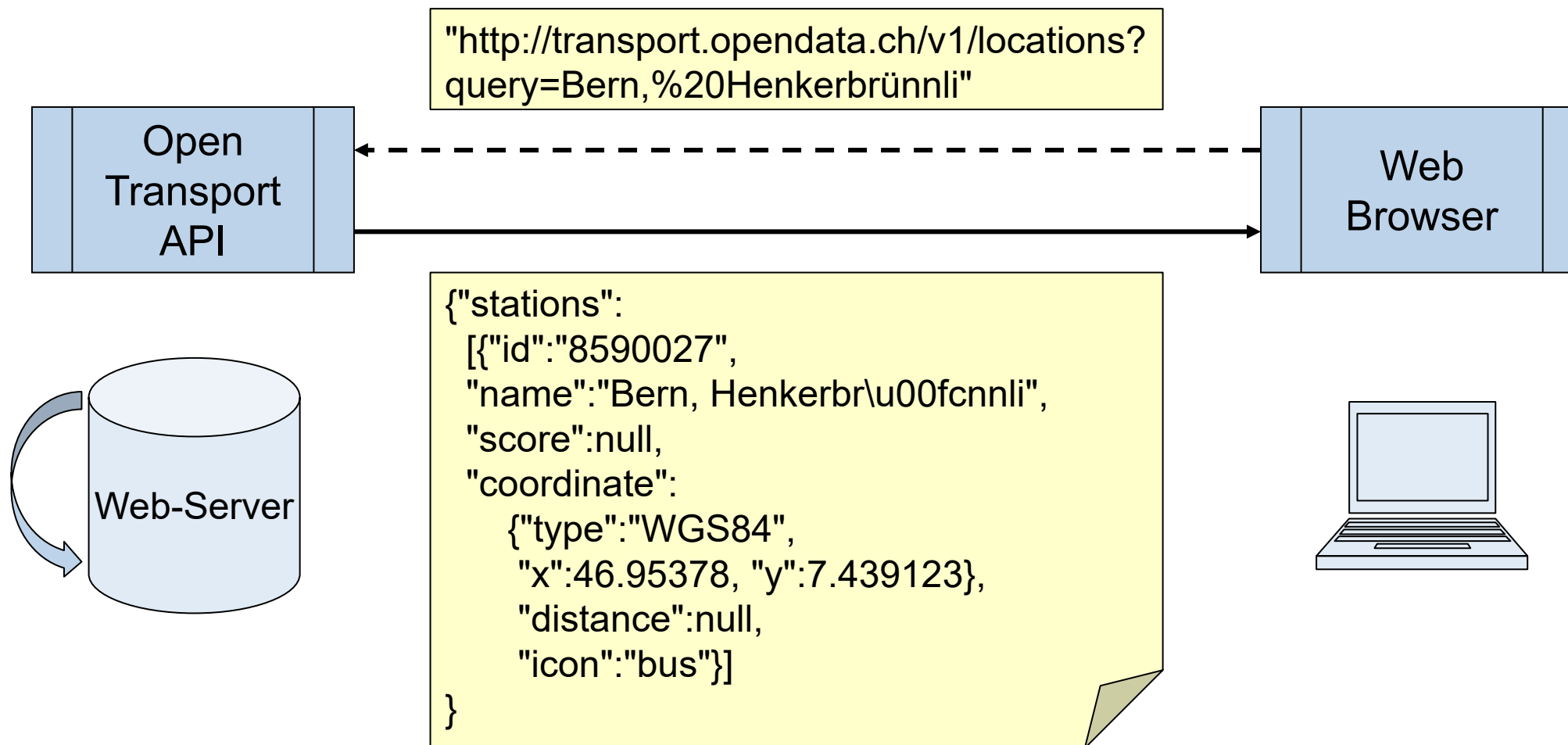
Abruf von Daten

Beispiel Web-API: Swiss Open Transport API

- Die Transport API erlaubt Entwicklern, die Zeitpläne öffentliche Transportunternehmen in ihren jeweiligen Anwendungen zu nutzen.
- Über eine URI wird der Web-Dienst aufgerufen und erklärt:
 - Aufruf: <http://transport.opendata.ch> ...
- Für Datenabfragen werden an die URI die angebotenen Ressourcen (plus Parameter) angefügt:
 - .../v1/locations? ...
 - .../v1/connections? ...
 - .../v1/stationboard? ...
- Als Ergebnis werden Daten im JSON-Format zurückgegeben.
 - Diese werden ohne weiteres im Web-Browser angezeigt.
 - Sie können aber auch im Rahmen von (Skript-) Programmen weiterverarbeitet werden.

Synchrone Datenschnittstelle

Open Transport Abfrage nach Stationskoordinaten



Synchrone Datenschnittstelle

Verarbeitung der empfangenen JSON-Daten

- Ein Web-Browser kann JSON-Daten nicht ohne weiteres verarbeiten.
- Da es sich bei JSON um ein Textformat handelt, werden sie im Web-Browser als textueller Inhalt behandelt.
- Da keine HTML-Tags vorliegen, werden die Daten als unstrukturierter Text angezeigt.
- Um die Daten gezielt zu verarbeiten, müssen sie im Rahmen einer Skriptsprache umgewandelt werden.

```
{"stations": [{"id": "8590027", "name": "Bern, Henkerbr\u00fclli", "score": null, "coordinate": {"type": "WGS84", "x": 46.95378, "y": 7.439123}, "distance": null, "icon": "bus"}]}
```

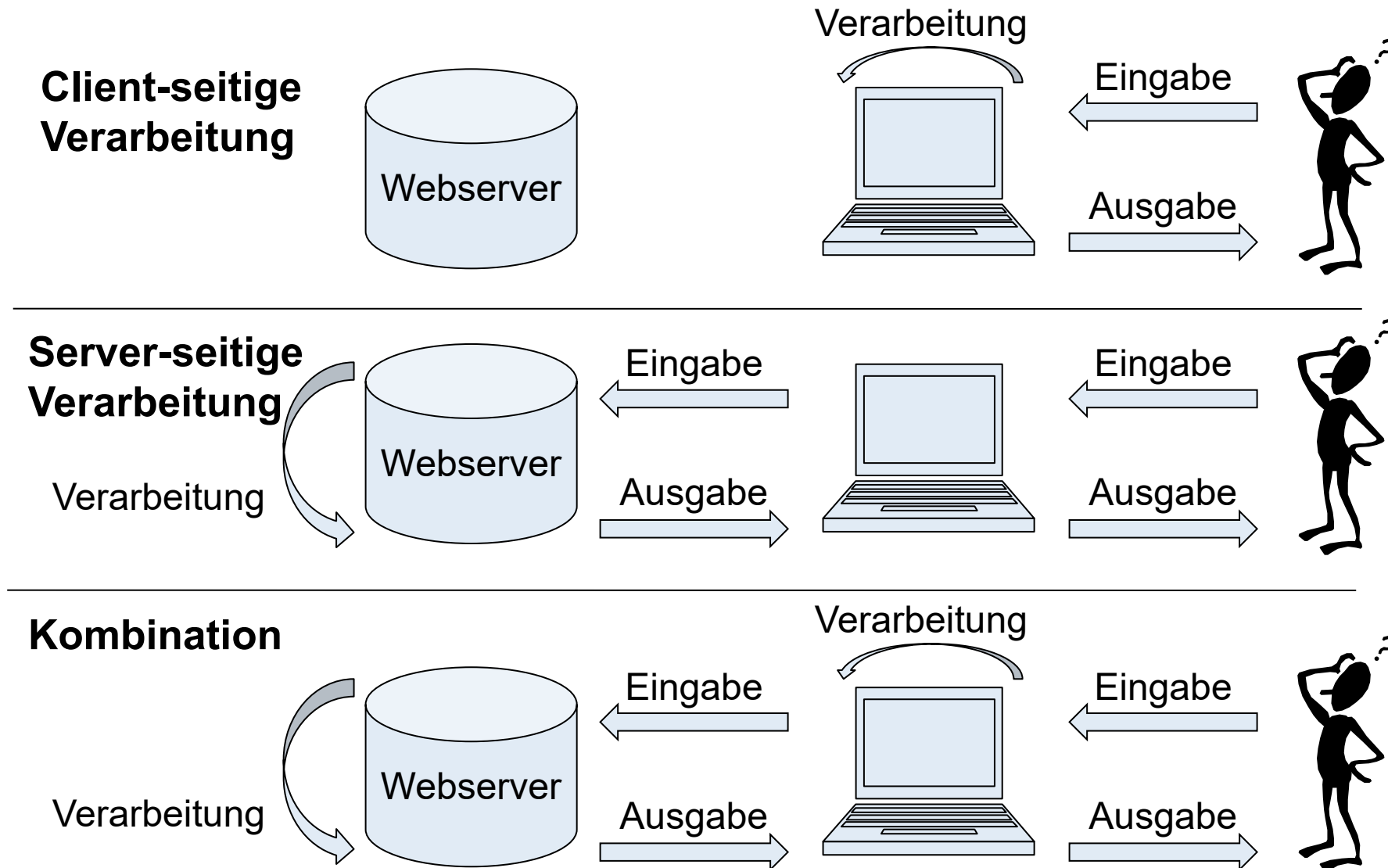
Agenda: Übersicht

Datenaustausch über das Internet

Grundlagen JavaScript

Verarbeitung übertragener Daten in JavaScript

Client- und Server-seitige Interaktion



Beispiel JavaScript

JavaScript-Integration

- Anweisungen einer Client-basierten Skriptsprache müssen als Inhalt eines SCRIPT-Elements erfolgen.
 - Programmcode innerhalb des HTML-Dokuments:


```
<script type="text/javascript">  
    ... Programmanweisungen  
</script>
```
 - Aufruf als externe Datei:

```
<script src="dateiname.js"/>
```
- Anweisungen von Skriptsprachen können zu unterschiedlichen Zeiten zur Ausführung gelangen:
 - beim Laden;
 - beim Vorliegen eines Ereignisses.

Beispiel JavaScript

Objekte

- JavaScript weist Merkmale von objektorientierten Programmiersprachen auf.
- Objekte können einerseits Daten aufweisen (Properties), andererseits auch Methoden (Methods), um das Objekt zu manipulieren.

Object	Properties	Methods
	<code>car.name = Fiat</code> <code>car.model = 500</code> <code>car.weight = 850kg</code> <code>car.color = white</code>	<code>car.start()</code> <code>car.drive()</code> <code>car.brake()</code> <code>car.stop()</code>

Beispiel JavaScript

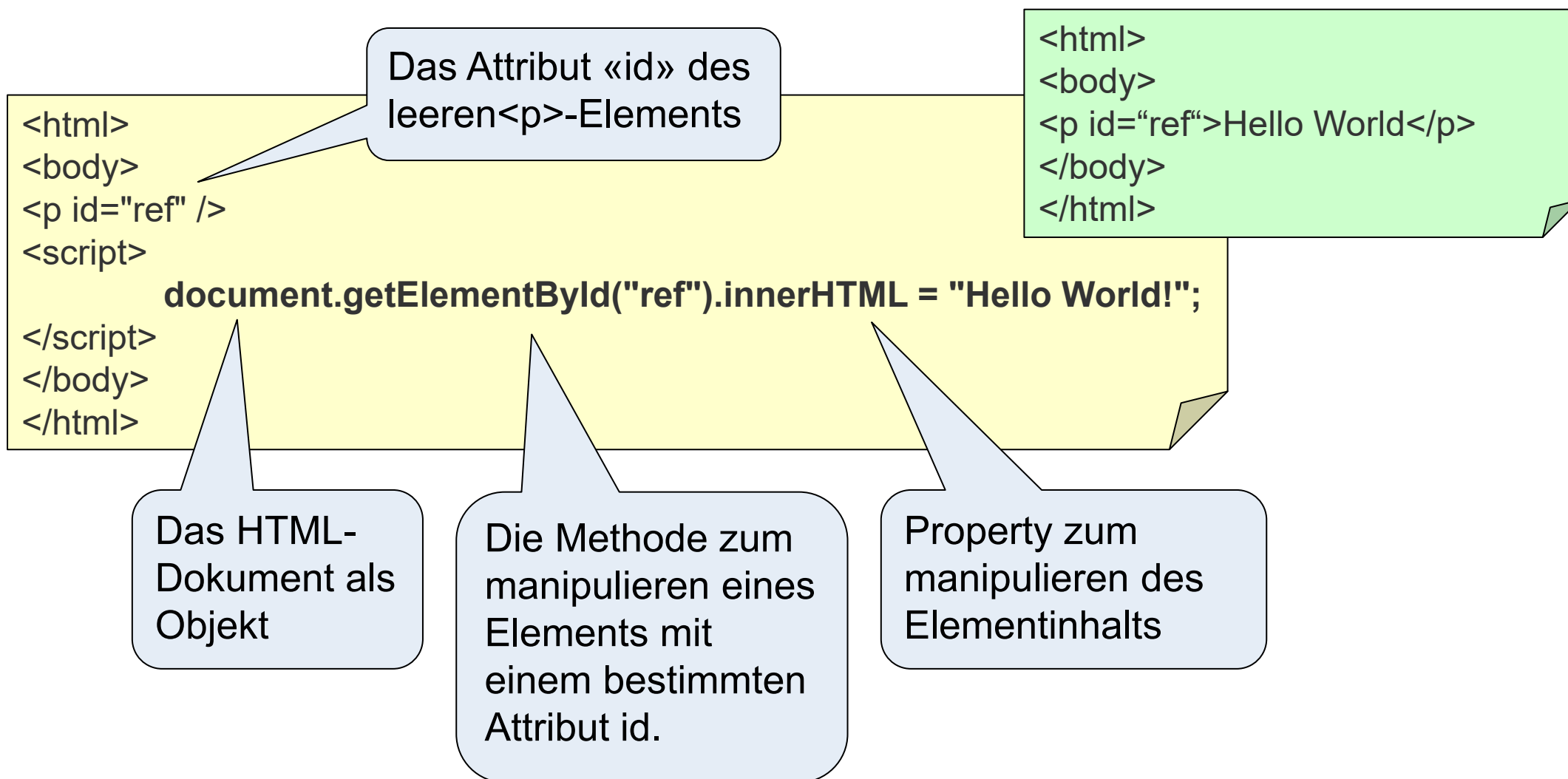
Objekt «document»

- Wenn ein HTML-Dokument in ein Browser-Fenster geladen wird, dann wird es zu einem `Document`-Objekt.
- Mit dem `Document`-Objekt können alle HTML-Elemente eines Dokuments angesprochen werden.

Methods	Description
<code>open()</code>	opens an HTML document to display the output
<code>close()</code>	closes an HTML document
<code>write()</code>	Writes HTML expressions or JavaScript code into an HTML document
<code>writeln()</code>	write a new line character after each HTML expressions or JavaScript Code
<code>getElementById()</code>	returns the reference of first element of an HTML document with the specified id.
<code>getElementByName()</code>	returns the reference of an element with the specified name
<code>getElementsByTagName()</code>	returns all elements with the specified tagname

Beispiel JavaScript

Skript zum Einfügen eines Elementinhalts



Beispiel JavaScript

Skript zum Anzeigen eines JavaScript-Objektwertes

```
<html>  
<body>  
<p id="ref" />  
<script>
```

Definieren eines
JSON-Textes.

```
    const Text="{\"stations\": [{\"id\":\"8590027\", \"name\":\"Bern, Henkerbr\u00fclli\",  
    \"score\":null, \"coordinate\": {\"type\":\"WGS84\", \"x\":7.46.95378, \"y\":7.439123},  
    \"distance\":null, \"icon\":\"bus\"}}]\"};  
    const myObj = JSON.parse(Text);  
    document.getElementById(\"ref\").innerHTML = myObj.stations[0].name;
```

```
</script>  
</body>  
</html>
```

Umwandeln des
JSON-Textes in ein
JavaScript-Objekt.

Zuweisen eines
Objektwertes als
Elementinhalt.

Agenda: Übersicht

Datenaustausch über das Internet

Grundlagen JavaScript

Verarbeitung übertragener Daten in JavaScript

Beispiel JavaScript

Ajax

- Es existieren mehrere Methoden, mit denen JavaScript die von einem Server empfangenen Daten weiterverarbeiten kann.
- Ajax (Asynchronous JavaScript and XML) bezeichnet ein Konzept der asynchronen Datenübertragung zwischen einem Browser und dem Server.
- Dieses ermöglicht es, HTTP-Anfragen durchzuführen, während eine HTML-Seite angezeigt wird, und die Seite zu verändern, ohne sie komplett neu zu laden.
- Ein wesentliches Element dieser Datenübertragungsmethode ist das XMLHttpRequest-Objekt.
- Das XMLHttpRequest-Objekt ist Bestandteil vieler Browser und kann im Rahmen von Skript-Sprachen bearbeitet werden.

Beispiel JavaScript

Objekt «XMLHttpRequest»

- Das Objekt bietet einen einfachen Weg, Daten von einem URL zu erhalten.
- Trotz seines Namens kann man mit dem Objekt nicht nur XML laden und es unterstützt auch andere Protokolle als HTTP.

Methods	Description
open()	Initialisiert eine Anfrage.
send()	Sendet die Anfrage.
setRequestHeader()	Setzt den Wert eines HTTP Anfrage-Headers.
getResponseHeader()	Liefert den String mit dem Text des angegebenen Headers.
onload	Wurde als Event in XMLHttpRequest 2 aufgenommen, das reagiert, wenn eine Reponse vorliegt.

Beispiel JavaScript

Abfragen und Bearbeiten von JSON-Daten

1

```
const xmlhttp = new XMLHttpRequest();
```

1. Ein neues Objekt XMLHttpRequest wird angelegt.
2. Anfrage wird initialisiert und gesendet.

2

```
xmlhttp.open("GET", "http://transport.opendata.ch/v1/locations?query=Bern,%20Henkerbrännli");  
xmlhttp.send();
```

3

```
xmlhttp.onload = function() {  
  const myObj = JSON.parse(xmlhttp.responseText);  
  ...};
```

3. Eine eintreffende Antwort wird im Rahmen einer Funktion verarbeitet.

Beispiel JavaScript

Abfragen und Bearbeiten von JSON-Daten (Variante)

1

```
const xmlhttp = new XMLHttpRequest();  
xmlhttp.responseText = "json";
```

1. Ein neues Objekt XMLHttpRequest wird angelegt.
2. Anfrage wird initialisiert und gesendet.

2

```
xmlhttp.open("GET", "http://transport.opendata.ch/v1/locations?query=Bern,%20Henkerbrännli");  
xmlhttp.send();
```

3

```
xmlhttp.onload = function() {  
  const myObj = xmlhttp.response; ...};
```

3. Eine eintreffende Antwort wird im Rahmen einer Funktion verarbeitet.

- Der Transfer von Daten geschieht im Web über Downloads von Dateien oder über Abfragen von Programmierschnittstellen (API).
- Daten können unterschiedliche Datenformate haben, was ihre Weiterverarbeitung beeinflusst.
- Web-Applikationen können mit Skript-Sprachen wie JavaScript realisiert werden.
- JavaScript ist eine objektorientierte Programmiersprache, die von Web-Browsern üblicherweise unterstützt wird.
- Im Rahmen von JavaScript können Daten aus Web-API aufgenommen und verarbeitet werden.
- Dies ist eine Voraussetzung, um Web-Applikationen auf der Basis von im Web verfügbaren Daten zu entwickeln.

Ende der Lektion

Wie geht es weiter?



- Zu dieser Lektion stehen Übungsaufgaben zur Verfügung.
- Diese Aufgaben können in Ilias bearbeitet werden.
- Die Lösung der Aufgaben wird besprochen.