

FORMS - EIN- UND AUSGABE VON DATEN

MMWP2024 - LV09

INHALTSVERZEICHNIS

- Organisation
- Formulare
- Übermittlungsverfahren
- Validierungen

INHALTSSCHWERPUNKTE

- Ein- und Ausgabe von Daten in Browsern
- Gängige Datenübermittlungsmöglichkeiten
- Datenvalidierung und Asynchronität

VORAUSSETZUNG

Der Ausgangspunkt dieser Vorlesungsreihe ist das Wissen über funktionsweise und Struktur von:

- DOM-Tree
- HTML5 Elemente
- Events und Selektoren von Javascript

ZIELE

Vorstellung von:

- Formular Elementen
- POST und GET Verfahren
- Datenübermittlung über Ajax
- Live-Beispielen

FORMULARE

- Dienen der Interaktion mit Nutzenden
- Dienen der Erfassung und Übermittlung von Benutzerinformationen
- Anwendungen in: Suchmasken, Benutzername- und Passwortabfrage, Gästebücher, Online-Shops, Social Media, etc.

FORMULARBEREICH DEFINIEREN

- `<form> ... </form>`
- Notwendig, wenn die Formulardaten (über den definierten Weg) an einen Server gesendet werden sollen
- Darf in allen Elementen vorkommen, die Flusselemente als Inhalt erlauben (außer form)
- Erlaubte Inhalte: beliebig viele Flusselemente (außer form)

<FORM> ELEMENT ATTRIBUTE - 1

- action: definiert, an welche URI die Werte der Eingabeelemente gesendet werden (wenn nicht gesetzt, wird die aktuell geladene Seite genutzt)
- accept-charset: Liste von unterstützten Zeichensätzen
- autocomplete [on, off]: aktiviert automatische Vervollständigung aller enthaltenen Eingabeelemente
- enctype: definiert Codierung des Medientyps

<FORM> ELEMENT ATTRIBUTE - 2

- name: Identifizierungsname des Formulars
- novalidate: deaktiviert Prüfung der Eingabeelemente vor dem absenden
- target: bestimmt den Fensternamen des Verweisziels

<FORM> ELEMENT ACTIONS - 1

- method: HTTP-Methode, mit der Eingabedaten gesendet werden
- GET:
 - Die Daten werden, getrennt durch ein Fragezeichen, an die URI angehängen (sind in der Adresszeile des Browsers sichtbar)
 - `URI?name1=wert1&name2=wert2`
 - Geeignet, wenn durch den Request nur andere Daten als Antwort empfangen werden
 - Ungeeignet bei sensiblen Daten
- POST:
 - Geeignet, um große Datenmengen (mehrere MByte) zu übertragen
 - Daten werden im Header-(Body) der Anfrage übertragen
 - Geeignete Möglichkeit, wenn Daten auf dem Server verändert werden sollen, oder Logindaten (Passwörter) übertragen werden sollen
 - Ungeeignet um Aktionen nochmals auszuführen oder zu speichern (z.B. Suchen in einer Suchmaschine)

<FORM> ELEMENT ACTIONS - 2

- HEAD:
 - Weist den Server an, lediglich einen HTTP-Header wie bei GET zu senden, aber ohne den Dokumentinhalt
 - Geeignete Möglichkeit, um Server ping zu testen und unterstützte Formate abzufragen
- PUT:
 - Direktes Hochladen von Daten auf den Webserver, ohne dass ein Skript zur Datenverarbeitung aufgerufen wird
 - Nur bei Webservern für WebDAV oder RESTful Web Services verfügbar (asynchrones Hochladen)
 - Geeignete Möglichkeit, um bestehende Daten zu aktualisieren


<FORM> ELEMENT ACTIONS - 3

- DELETE:
 - Löscht angegebene Ressource auf dem Server
 - Nur bei Webservern für WebDAV oder RESTful Web Services verfügbar
 - Geeignete Möglichkeit, um bestehende Daten zu löschen
- TRACE:
 - Liefert die Anfrage so zurück, wie der Server sie empfangen hat (zum debuggen der Verbindung)
 - Wird häufig von Servern aus Sicherheitsgründen blockiert

<FORM> ELEMENT ACTIONS - 4

- OPTIONS:
 - Liefert eine Liste der vom Server unterstützten Methoden und Merkmale
 - Z.B. unterstützte Protokolle, Datenformate, En/De-Codierungen und Authentifizierungsmethoden
- CONNECT:
 - Von Proxyservern implementierte Methode, die SSL-Tunnel zur Verfügung stellen können
 - Tunneln der Anfrage über einen Proxy, der verschlüsseltes Packet (HTTPS), weitergibt
 - Meist über HTTP/1.2-1.3
- [FORM Element](#)


FORMULAR SUPPORT

Form attribute  - LS

Usage % of all users 96.52%

Global

Attribute for associating input and submit buttons with a form.

Current aligned Usage relative Date relative Filtered All 

Chrome	Edge *	Safari	Firefox	Opera	IE	Chrome for Android	Safari on iOS *	Samsung Internet	Opera Mini *	Opera Mobile *	UC Browser for Android	Android Browser *	Firefox for Android	QQ Browser	Baidu Browser	KaiO Brows
		3.1-4														
4-9	12-15	5	2-3.6				3.2-4.3					2.1-2.3				
10-129	16-129	5.1-17.6	4-131	10-113	6-10		5-17.6	4-24		12-12.1		3-4.4.4				2.5
130	130	18.0	132	114	11	130	18.0	25	all	80	15.5	130	130	14.9	13.52	3.1
131-133		18.1-TP	133-135				18.1									

Quelle

FORMULAR - EINGABE ELEMENTE

allgemeine Attribute:

- readonly:
 - Der vorgeschlagene Wert kann nicht verändert werden
 - Veränderung dieses Attributs über JavaScript möglich
- disabled:
 - Eingabefeld soll weder benutzt noch seine Daten mit übertragen werden
 - Wird vom Browser üblicherweise ausgegraut dargestellt
 - [Input readonly](#)
- name:
 - Gibt dem Formularelement einen eindeutigen Namen
 - Wird zusammen mit dem Wert des Eingabefeldes an den Webserver übertragen

FORMULAR - EINGABE ELEMENTE - 2

allgemeine Attribute:

- value: Vorbelegungswert des Eingabeelements
- placeholder: gibt einen Beispieltext für das Eingabefeld an, der bei der ersten Eingabe verschwindet
- autofocus: setzt den Cursor beim Laden der Seite automatisch auf das Eingabefeld
- autocomplete: zeigt dem Browser, ob ein Eingabefeld automatisch ausgefüllt werden soll
- inputmode: hilft Browsern mit Bildschirmtastatur die passende Belegung zu wählen
- [Weitere Informationen](#)

FORMULAR - EINGABE ELEMENTE - 3

allgemeine Attribute:

- required:
 - Erzwingt die Eingabe in das Eingabefeld, bevor das Formular abgesendet werden kann
 - Durch CSS Pseudoklassen :valid und :invalid lassen sich Elemente mit ungültigen Daten hervorheben (neu in CSS3-2023)
- pattern:
 - Gibt einen regulären Ausdruck an, dem der Eintrag des Eingabefeldes entsprechen muss (neu in CSS3-2022)
- [Validierung](#)

FORMULAR - EINGABE ELEMENTE - 3

```
<input type="button">  
<input type="checkbox">  
<input type="color">  
<input type="date">  
<input type="datetime-local">  
<input type="email">  
<input type="file">  
<input type="hidden">  
<input type="image">  
<input type="month">  
<input type="number">  
<input type="password">  
<input type="radio">  
<input type="range">
```

Input typen

EINZEILIGES TEXTEINGABEFELD

- `<input type="text">` zur Eingabe kurzer einzeliger Texte
- Attribute:
 - `size`: Anzahl der angezeigten Zeichen (besser: CSS-Angabe `width`)
 - `maxlength`: maximale Feldlänge in Zeichen

EINZEILIGES SUCHFELD

- `<input type="search">` entspricht einzeiligem Texteingabefeld
- Manche Browser ergänzen ein Lupensymbol oder einen Knopf zum Löschen des Eingabefeldes

PASSWORT-EINGABEFELD

- `<input type="password">` ähnliche Eigenschaften wie einzeliges Texteingabefeld
- Eingegebene Zeichen werden durch Platzhalter (Punkte, Sterne) dargestellt
- Browser kann die Eingabe speichern, um diese beim nächsten Mal wieder anzubieten
- Übertragung erfolgt nur verschlüsselt, wenn der Server HTTPS anbietet!

TELEFONNUMMER-EINGABEFELD

- `<input type="tel">` entspricht einzeiligem Texteingabefeld
- Bildschirmtastaturen können ein numerisches Tastenfeld darstellen (Mobilgeräte oder Bildschirmtastaturen)

WEBADRESSEN-EINGABEFELD

- `<input type="url">` entspricht einzeiligem Texteingabefeld
- Während der Eingabe wird auf das Format einer validen Internetadresse geprüft
- Bildschirmtastaturen können eine Taste `.com` anzeigen (Mobilgeräte oder Bildschirmtastaturen)

E-MAIL-EINGABEFELD

- `<input type="email">` Eingabe wird auf Format einer E-Mail-Adresse geprüft
- Bildschirmtastaturen können eine Taste mit @-Symbol anzeigen (Mobilgeräte oder Bildschirmtastaturen)
- **Input Text**

ZAHLEN-EINGABEFELD

- `<input type="number">` entspricht einzeiligem Texteingabefeld
- Bildschirmtastaturen können ein numerisches Tastenfeld darstellen (Mobilgeräte oder Bildschirmtastaturen)
- Attribute:
 - max: Maximalwert
 - min: Minimalwert
 - step: legt fest, in welchen Stufen die Werte eingegeben werden dürfen


SCHIEBEREGLER


- `<input type="range">` ermöglicht grafische Eingabe von Werten über Schieberegler
- Bildschirmtastaturen können ein numerisches Tastenfeld darstellen
- Attribute:
 - max: Maximalwert (Standard: 100)
 - min: Minimalwert (Standard: 0)
 - step: legt fest, in welchen Stufen die Werte eingegeben werden dürfen
 - orient: vertical stellt einen vertikalen Schieberegler dar (alternativ in CSS: `transform: rotate(270deg)`)
 - [Input Number](#)

DATUMS-EINGABEFELDER


- `<input type="date">` Browser können eine Kalenderdarstellung zur Eingabe anbieten
- Bildschirmtastaturen können ein numerisches Tastenfeld darstellen
- Datums-Eingabefeld mit Uhrzeit in der Zeitzone des Nutzers:
 - `<input type="datetime-local">`
 - Browser können eine Kalenderdarstellung zur Eingabe anbieten
 - Mittlerweile von aktuellen Browser unterstützt








DATE TIME LOCALE

HTML element: input: type="datetime-local" 

Usage % of all users  ?

Global 94.72%

Current aligned Usage relative Date relative Filtered All 

Chrome	Edge *	Safari	Firefox	Opera	IE  *	Chrome for Android	Safari on iOS *	Samsung Internet	Opera Mini *	Opera Mobile *	UC Browser for Android	Android Browser *	Firefox for Android	QQ Browser	Baidu Browser	KaIO Brows
4-19		3.1-14	2-92	10.1			3.2-4.3					2.1-4.3				
20-129	12-129	14.1-17.6 	93-131 	11.5-113	6-10		5-17.6	4-24		12-12.1		4.4-4.4.4				2.5
130	130	18.0 	132 	114	11	130	18.0	25	all	80	15.5	130	130	14.9	13.52	3.1
131-133		18.1-TP 	133-135 				18.1									

Input Datetime

WOCHEN-EINGABEFELD

- `<input type="week">` Kombination von Wochenangabe mit Jahreszahl
- Browser können eine Kalenderdarstellung zur Eingabe anbieten
- Attribute:
 - `list`: Angabe mehrerer Optionen in Verbindung mit `<datalist>`
 - `value`: String im Format `YYYYw-WW` mit `YYYY` als Jahreszahl und `WW` als Wochennummer (bis 52 bzw. 53)

DATUM UND ZEITEINGABEFELDER

- Datums-Eingabefeld: `<input type="month">`
Kombination von Monatsangabe mit Jahreszahl
- Uhrzeit-Eingabefeld: `<input type="time">` Browser können individuelle Darstellung zur Eingabe anbieten
- Browser können eine Kalenderdarstellung zur Eingabe anbieten
- **Input time**

VERSTECKTE FELDER

- `<input type="hidden">` Eingabefelder, die dem Nutzer nicht angezeigt werden
- Daten dieser Felder werden mit an Server übertragen
- Nützlich für Bot-Erkennung bei automatisierter Eingabe von Formularen
- Übertragung von Tokens für die Authentifizierung
- `Input hidden`

DATEI-UPLOAD-FELD - 1

- `<input type="file">` Dateien vom lokalen Gerät an den Webserver übertragen
- Funktioniert nur mit POST-Methode
- Im Formular muss Attribut `enctype="multipart/form-data"` notiert werden, damit nicht nur der Name, sondern auch die Datei mit übertragen wird
- Maximale Dateigröße wird im Webserver konfiguriert
- Webserver speichert empfangene Dateien nur temporär bis zur Verarbeitung der aktuellen Anfrage, das Verarbeitungsskript des Servers muss sich sofort um die Sicherung der Dateien kümmern

DATEI-UPLOAD-FELD - 2

Attribute

- accept: kommagetrennte Liste von MIME-Typen (können auch Wildcards enthalten)
- multiple: ermöglicht die Auswahl mehrerer gleichzeitig zu sendender Dateien, dabei sollte das Attribut name mit eckigen Klammern [] enden
- **Input upload**

FARBAUSWAHL-ELEMENT

- `<input type="color">` nach Farbauswahl enthält das Attribut `value` den sechsstelligen hexadezimalen Farbcode mit führendem `#`
- [Input color](#)

RADIO-BUTTONS

- `<input type="radio">` erlauben die gleichzeitige Auswahl einer von zur Verfügung stehenden Optionen
- Attribute:
 - `name`: Buttons mit gleicher Belegung dieses Attributs bilden eine Auswahlgruppe
 - `value`: legt den zum Button gehörigen Wert fest, der zusammen mit dem Wert von `name` übertragen wird
 - `checked`: zur Vorselektierung des Buttons beim Laden der Seite
- [Input radio](#)

CHECKBOXEN

- `<input type="checkbox">` erlauben die gleichzeitige Auswahl von beliebig vielen Optionen (Multiple Choice)
- Attribute:
 - name: Buttons mit gleicher Belegung dieses Attributs bilden eine Auswahlgruppe, dabei muss der Name auf eckige Klammern [] enden
- Input Checkbox
- Switches bei Mobilgeräten

BUTTONS - 1

Attribute:

- form: Zuordnung zur ID des Formulars, zu dem der Button gehören soll
- formaction: überschreibt die im action-Attribut des zugehörigen Formulars angegebene URL
- formmethod: überschreibt die im method-Attribut des zugehörigen Formulars angegebene Sendemethode
- formenctype: überschreibt die im enctype-Attribut des zugehörigen Formulars angegebene Codierung

BUTTONS - 2

Attribute:

- Auslösebutton
 - `<button type="button">` ehemals `<input type="button">` löst clientseitige Aktionen aus
 - durch `addEventListener` lässt sich eine JavaScript-Funktion zum Button zuweisen, die beim Anklicken ausgelöst wird
- Absendebutton
 - `<button type="submit">` ehemals `<input type="submit">` sendet das zugehörige Formular zum Webserver
 - durch `addEventListener` lässt sich eine JavaScript-Funktion zum Button zuweisen, die beim Anklicken ausgelöst wird
- Resetbutton
 - `<button type="reset">` ehemals `<input type="reset">` setzt die Werte des zugehörigen Formulars zurück
- Input Button

GRAFISCHER ABSENDEBUTTON

- `<input type="image">` Absendeknopf, der mit einem Bild unterlegt ist
- zusätzlich zu den Formulardaten werden die Koordinaten des Klicks zum Server übertragen in der Form `name.x` und `name.y`
- Browser können eine Kalenderdarstellung zur Eingabe anbieten
- Attribute:
 - `src`: URL der unterlegten Grafik
 - `width`: Breite der Grafik
 - `height`: Höhe der Grafik
 - `alt`: Alternativtext, falls die Grafik nicht geladen werden kann
- [Input Button](#)

MEHRZEILIGES TEXT-EINGABEFELD

- `<textarea></textarea>` dient der Eingabe von mehrzeiligen Texten
- Größe des Textfeldes lässt sich vom Nutzer verändern
- Attribute:
 - `rows`: bestimmt die Anzahl der angezeigten Zeilen
 - `cols`: bestimmt die Anzahl der angezeigten Spalten
 - `wrap [hard, soft]`: legt fest, ob Text umgebrochen werden soll
- **Input Text area**

AUSWAHLLISTEN - 1

- `<select> ... </select>` enthält beliebige Anzahl an Auswahloptionen (`option`), aus denen der Nutzer wählen kann
- Attribute:
 - `size`: legt Anzahl der Angezeigten Zeilen der Auswahlliste fest (Standardwert 1 erzeugt Dropdown-Liste)
 - `multiple`: ermöglicht die gleichzeitige Auswahl mehrerer Einträge, dabei muss dem `name`-Attribut ein Arrayname (mit abschließenden eckigen Klammern) zugewiesen werden

• `<input type="text">`

AUSWAHLLISTEN - 1

- `<option>...</option>` definiert jeweils einen auswählbaren Eintrag der Auswahlliste
- Attribute:
 - `selected`: Vorauswahl des Eintrages
 - `value`: gibt den Wert an, der bei gewähltem Eintrag anstelle des angezeigten Eintragstextes übertragen wird
- `Input select options`

VERSCHACHTELTE AUSWAHLLISTEN

- `<optgroup><option>... </option><option>...</option></optgroup>`
- Auswahllemente sollen in hierarchisch verschachtelten Menüstrukturen dargestellt werden
- In vielen Browsern werden die Auswahllemente jedoch nur gruppiert unter den Hauptoptionen angezeigt
- Nur eine Verschachtelungsebene erlaubt
- `Input selection optgroup`

DATENLISTEN

- `<datalist><option>...</option><option>...</option></datalist>`
- erlaubt die unterstützte Eingabe in ein Textfeld durch eine vordefinierte Liste von Einträgen
- dem list-Attribut des input-Elements wird dabei die id der datalist zugewiesen, welche die zu nutzenden Elemente enthält
- [Input selection datalist](#)

FORTSCHRITTSANZEIGE

- `<progress></progress>` stellt den Fortschritt einer Aktion dar
- Attribute:
 - `max`: gibt an, wie viele Schritte maximal möglich sind (Standardwert: 1)
 - `value`: Anzahl der abgearbeiteten Schritte (sollte kleiner als `max` sein)
- **Input Progression**

AUSGABEELEMENT

- `<output>...</output>` dient zur Ausgabe des Ergebnisses einer Benutzereingabe oder Berechnung
- Attribute:
 - `for`: gibt die Identifikation des Elements an, auf welches sich dieses Element beziehen soll
- **Input Output**

GRUPPIERUNG

- `<fieldset> ... </fieldset>` erlaubt die Zusammenfassung mehrerer Eingabeelemente zu einer Gruppe
- Attribute:
 - `disabled`: deaktiviert die Eingabeelemente und graut sie aus
 - `form`: erlaubt die Zuordnung zu einem Formular mit der angegebenen Identifikation
 - `name`: Identifizierungsname des Elementes
 - Ein `legend`-Element, um eine Überschrift der Gruppe zu definieren
 - beliebig viele Fluss-Elemente
- [Input Fieldset](#)

ÜBERSCHRIFT FÜR FORMULARBEREICH

- `<legend> ... </legend>` darf nur als erstes Element in einem fieldset-Element vorkommen
- Attribute:
 - beliebig viele Stil-Elemente
 - Überschriften
- Form Element Beschreibung

FORMULAR-ELEMENTENBESCHRIFTUNGEN

- `<label>...</label>` durch Anklicken der Beschriftung wird das dazugehörige Eingabeelement aktiviert (Vergrößerung der aktiven Eingabeelementfläche)
- Beschriftung kann dadurch zugefügt werden, dass das Formularelement innerhalb der label-Tags steht, oder durch Nutzung des for-Attributs
- Attribute:
 - for: gibt die Identifikation des Formularelements an, auf die sich die Beschriftung beziehen soll
 - form: gibt die Identifikation des Formulars an, zu dem die Beschriftung gehören soll
- [Form Element Beschreibung](#)

ÜBERMITTLUNGSVERFAHREN

HTTP Method	CRUD	Collection Resource (e.g. /users)	Single Resource (e.g. /users/123)
POST	Create	201 (Created), 'Location' header with link to /users/{id} containing new ID	Avoid using POST on a single resource
GET	Read	200 (OK), list of users. Use pagination, sorting, and filtering to navigate big lists	200 (OK), single user. 404 (Not Found), if ID not found or invalid
PUT	Update/Replace	405 (Method not allowed), unless you want to update every resource in the entire collection of resource	200 (OK) or 204 (No Content). Use 404 (Not Found), if ID is not found or invalid
PATCH	Partial Update/Modify	405 (Method not allowed), unless you want to modify the collection itself	200 (OK) or 204 (No Content). Use 404 (Not Found), if ID is not found or invalid
DELETE	Delete	405 (Method not allowed), unless you want to delete the whole collection — use with caution	200 (OK). 404 (Not Found), if ID not found or invalid

Quelle und Status Codes

GET

Get Beispiel

POST

Post Beispiel

PUT

Put Beispiel

Anfrage:

PUT /new.html HTTP/1.1

Host: example.com

Content-type: text/html

Content-length: 16

`<p>New File</p>`

Antwort:

HTTP/1.1 201 Created

Content-Location: /new.html

OPTIONS

Options Beispiel

Anfrage:

OPTIONS / HTTP/2

Host: example.org

User-Agent: curl/8.7.1

Accept: */*

Antwort:

HTTP/1.1 204 No Content

Allow: OPTIONS, GET, HEAD, POST

Cache-Control: max-age=604800

Date: Thu, 13 Oct 2016 11:45:00 GMT

Server: EOS (lax004/2813)

HEAD

Head Beispiel

Anfrage:

HEAD / HTTP/1.1

Host: example.com

User-Agent: curl/8.6.0

Accept: */*

Antwort:

HTTP/1.1 200 OK

Content-Type: text/html; charset=UTF-8

Date: Wed, 04 Sep 2024 10:33:11 GMT

Content-Length: 1234567

DELETE

Delete Beispiel

Anfrage:

```
DELETE /file.html HTTP/1.1
```

```
Host: example.com
```

Antwort:

```
HTTP/1.1 200 OK
```

```
Content-Type: text/html; charset=UTF-8
```

```
Date: Fri, 21 Jun 2024 14:18:33 GMT
```

```
Content-Length: 1234
```

```
<html>
```

```
  <body>
```

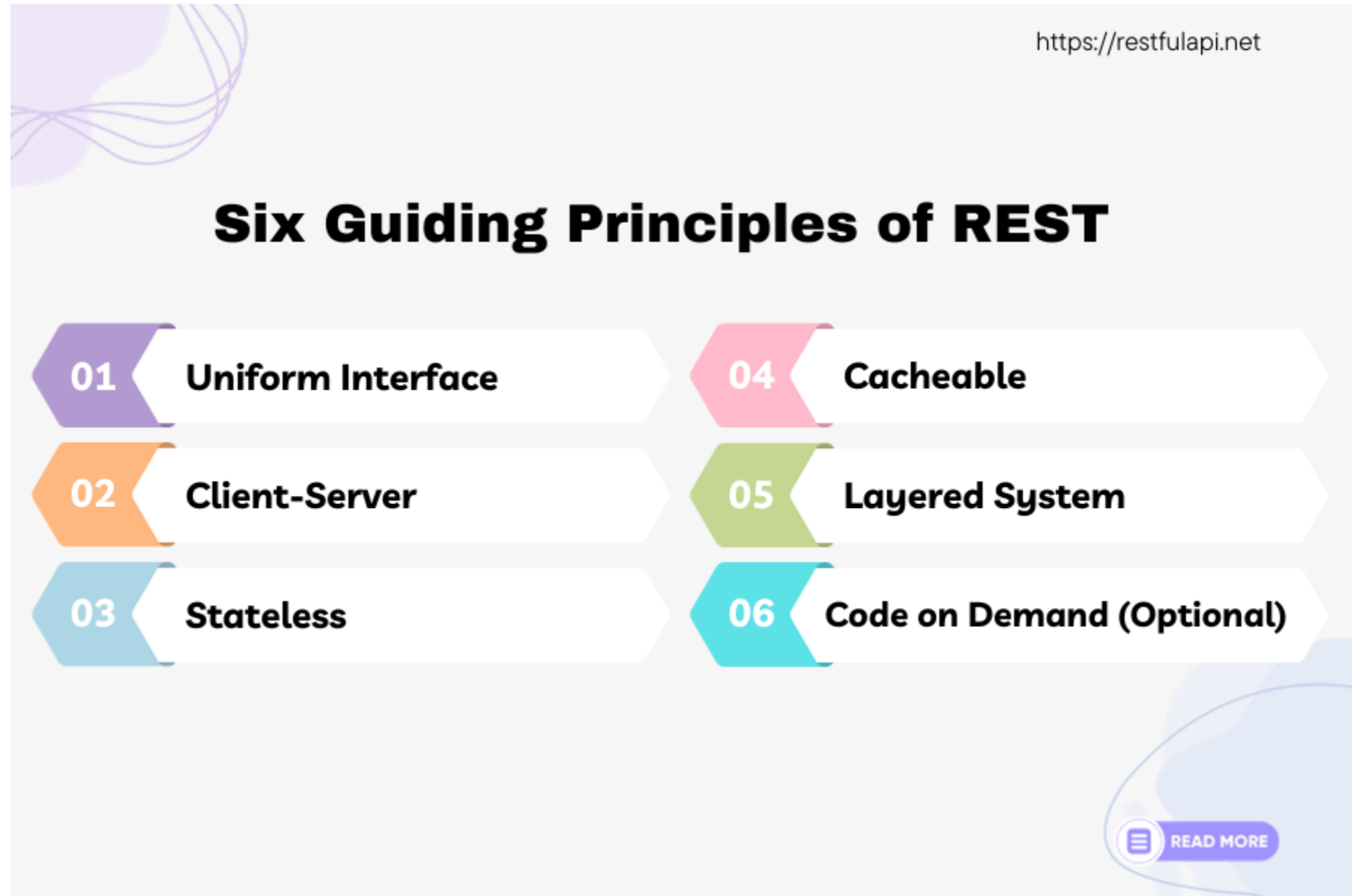
```
    <h1>File "file.html" deleted.</h1>
```

```
  </body>
```

REST-SERVICES

- REST ist ein Akronym für REpresentational State Transfer und ein Architekturstil für verteilte Hypermedia-Systeme
- Eine Web-API (oder ein Webdienst), die dem REST-Architekturstil entspricht, wird als REST-API (oder RESTful-API) bezeichnet
- Einzelne Ressourcen werden in Anfragen mithilfe von URIs identifiziert
- XML oder JSON Objekte für Datenübermittlung (Requests und Responses)
- Zustandsloser Dienst (Keine Sitzung oder zeitbasierte Zugriffe)
- Möglichkeit der Authentifizierung über API-KEY oder SSL-Zertifikate

REST-SERVICE GUIDELINES



Quelle

SWAGGER - BEISPIEL

pet Everything about your Pets		Find out more ^
POST	/pet/{petId}/uploadImage uploads an image	🔒 ▼
POST	/pet Add a new pet to the store	🔒 ▼
PUT	/pet Update an existing pet	🔒 ▼
GET	/pet/findByStatus Finds Pets by status	🔒 ▼
GET	/pet/findByTags Finds Pets by tags	🔒 ▼
GET	/pet/{petId} Find pet by ID	🔒 ▼
POST	/pet/{petId} Updates a pet in the store with form data	🔒 ▼
DELETE	/pet/{petId} Deletes a pet	🔒 ▼
store Access to Petstore orders		^
GET	/store/inventory Returns pet inventories by status	🔒 ▼
POST	/store/order Place an order for a pet	▼
GET	/store/order/{orderId} Find purchase order by ID	▼
DELETE	/store/order/{orderId} Delete purchase order by ID	▼

SWAGGER - BEISPIEL

pet Everything about your Pets

[Find out more](#) ^

POST `/pet/{petId}/uploadImage` uploads an image



Parameters

Cancel

Name	Description
petId * required integer(\$int64) (path)	ID of pet to update <input type="text" value="petId"/>
additionalMetadata string (formData)	Additional data to pass to server <input type="text" value="additionalMetadata"/>
file file (formData)	file to upload <input type="text" value="Durchsuchen... Keine Dat...sgewählt."/>

Execute

Responses

Response content type **application/json** v

Code	Description
200	successful operation

Example Value | Model

```
{
  "code": 0,
  "type": "string",
  "message": "string"
}
```

FORMULAR BEISPIEL

```
1 <!-- https://dev.to/info_generalhazedawn_a3d/form-validation-using-javas
2 <!DOCTYPE html>
3 <html lang="en">
4 <head>
5     <meta charset="UTF-8">
6     <meta name="viewport" content="width=device-width, initial-scale=1.0
7     <title>Form Validation Example</title>
8 </head>
9 <body>
10     <h1>User Registration Form</h1>
11     <form id="registrationForm">
12         <label for="username">Username (required):</label>
13         <input type="text" id="username" name="username" required>
14         <br>
15
```

Quelle

FORM VALIDIERUNG HTML5

- Möglichkeit der HTML-Validierung über spezielle Input attribute
- Client-side validation
- Möglichkeit der erweiterten Client-side validierung über Javascript

JAVASCRIPT FORM VALIDIERUNG (BROWSERSEITIG)

```
1 document.getElementById('registrationForm').addEventListener('submit', f
2     event.preventDefault(); // Prevent form submission
3     const errorMessages = [];
4
5     // Validate Username
6     const username = document.getElementById('username').value;
7     if (username.trim() === '') {
8         errorMessages.push('Username is required.');
```

Quelle

SERVERSEITIGE VALIDIERUNG - PHP

PHP Daten Validierung

SERVERSEITIGE VALIDIERUNG - VUE

VEE-validate

SERVERSEITIGE VALIDIERUNG - ANGULAR

Angular 17 Form Validierung

SERVERSEITIGE VALIDIERUNG - ASPNET

ASP.NET Core Daten validierung

WEITERES

- [Mozilla Form Guide](#)
- [W3Schools HTML5 Elements](#)
- [Mozilla Form Tag](#)
- [Google Forms](#)

QUELLEN

- W3 Forms WIA (3.10.2024): [Quelle](#)
- Whatwg Form Tag specification (3.10.2024): [Quelle](#)
- W3Schools Form guide (3.10.2024): [Quelle](#)
- Selfhtml.org guidance (3.10.2024): [Quelle](#)
- W3.org Form standard (3.10.2024): [Quelle](#)

ABSPANN

Neuntes Level geschafft!

Fragen und Feedback?