OST
Ostschweizer
Fachhochschule

Web Development Fundamentals

Web Services

Prof. Dr. Matthias Baldauf

School of Management / Institute for Information and Process Management

Web Services

- Machine-to-machine communication
- Calling functions on remote computer or data exchange
- Identification of a web service via unique URL
- Interface description (input parameters, return format, etc.)
- Independent of concrete programming language

- Various implementation options for "service-oriented architectures" (SOA):
  - SOAP: protocol for the exchange of XML-based messages (quite complex)
  - **REST: simpler alternative based on web standards (and easy to use with JavaScript!)**

RESTful Services

- «Representational State Transfer»
- Identification of service through URL, communication via HTTP
- HTTP method specifies desired operation

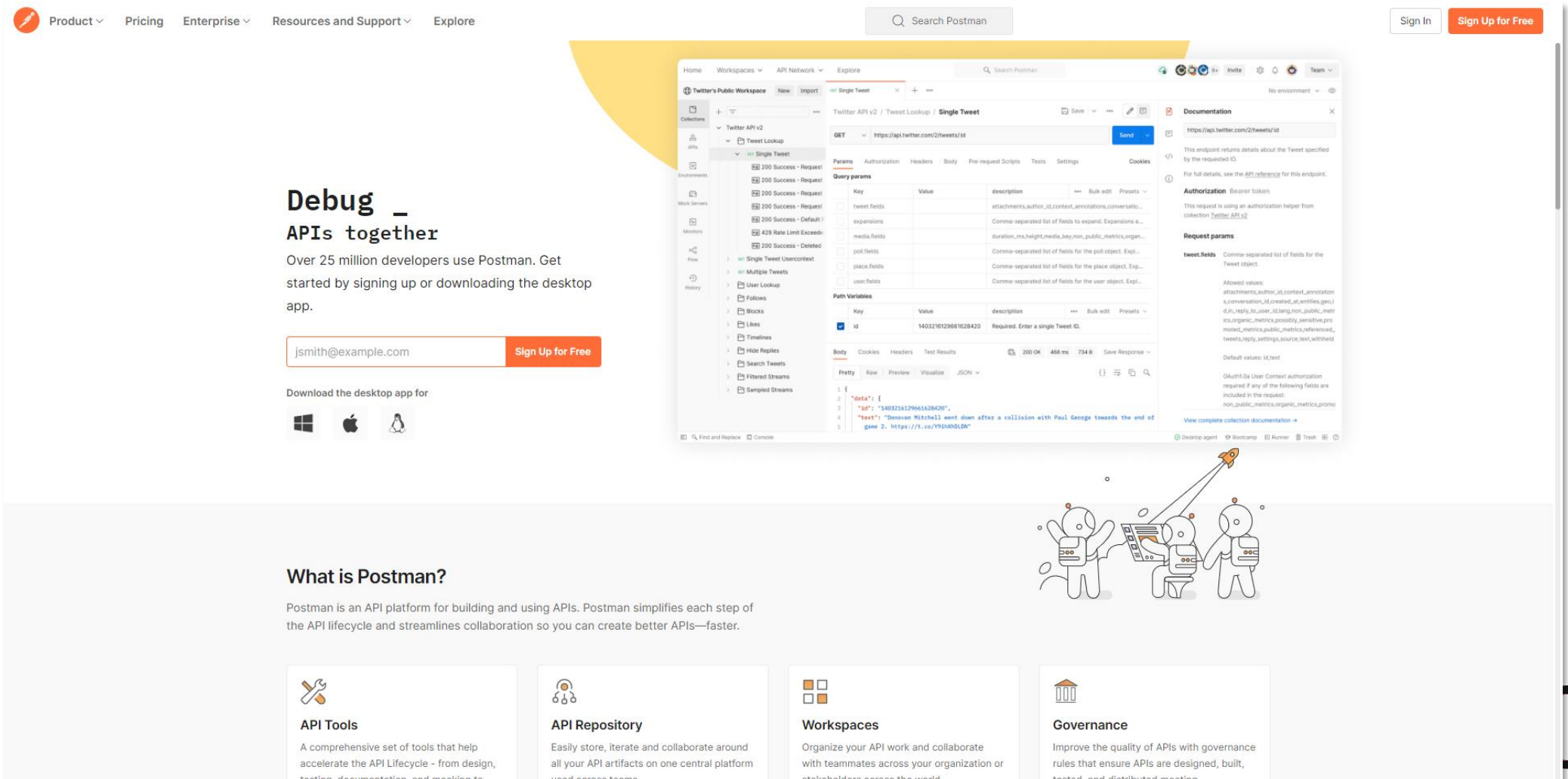
HTTP method (example)	Examples
GET	Retrieves information about a resource
POST	Inserts a new resource
PUT	Modifies an existing resource
DELETE	Deletes a resource

- Data mostly in JSON format
- Use of HTTP response codes, see https://developer.mozilla.org/en-US/docs/Web/HTTP/Status

Ex. Customer management in REST style

- GET http://www.example.com/customers/12345
- GET http://www.example.com/customers/12345/orders
- POST http://www.example.com/customers
- PUT http://www.example.com/customers/12345
- DELETE http://www.example.com/customers/12345

Postman

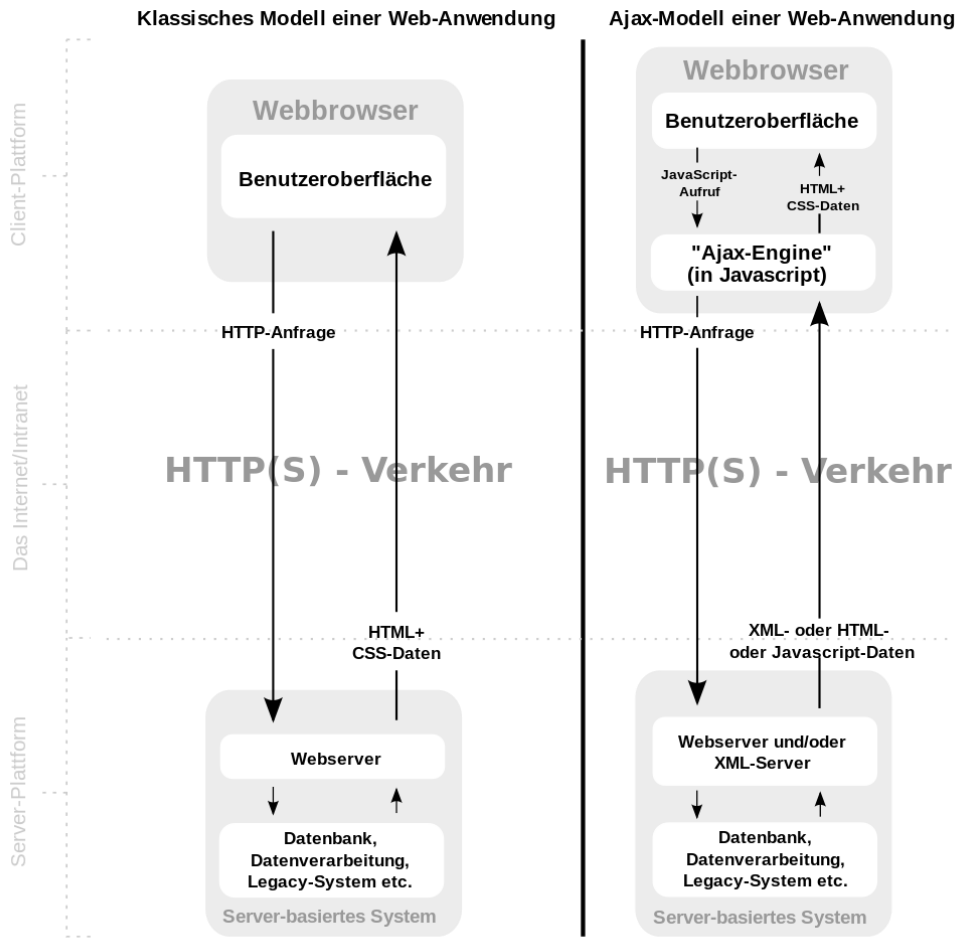

Product
Pricing
Enterprise
Resources and Support
Explore
Search Postman
Sign In
Sign Up for Free
Debug _
APIs together
Over 25 million developers use Postman. Get started by signing up or downloading the desktop app.
jsmith@example.com
Sign Up for Free
Download the desktop app for
What is Postman?
Postman is an API platform for building and using APIs. Postman simplifies each step of the API lifecycle and streamlines collaboration so you can create better APIs—faster.
API Tools
A comprehensive set of tools that help accelerate the API Lifecycle - from design,
API Repository
Easily store, iterate and collaborate around all your API artifacts on one central platform
Workspaces
Organize your API work and collaborate with teammates across your organization or
Governance
Improve the quality of APIs with governance rules that ensure APIs are designed, built,

Exercise

- Install Postman
- Choose a sample REST service
  - https://restcountries.com
  - https://exchangerate.host
  - http://jservice.io
  - https://api.chucknorris.io/
- Call selected service with Postman

Web service calls in JavaScript: Ajax

- Ajax: **Asynchronous JavaScript and XML**
- Web services over HTTP within a web page without reloading the page itself.
- "XML" in the name not quite correct anymore, because JSON is often used as data format



OST

Advantages and disadvantages of Ajax?

JSON - JavaScript Object Notation

Compact data format in easy to read text form for data exchange

Adapted from object notation in JavaScript:

- Data is stored in name/value pairs

  ```
  "firstname": "Matthias"
  ```

- Data are separated by commas

  ```
  "firstname": "Matthias", "familyname": "Baldauf"
  ```

- Curved brackets indicate objects

  ```
  "lecturer" : { "firstname": "Matthias", "familyname": "Baldauf" }
  ```

- Square brackets indicate arrays

  ```
  "modules" : [1, 2, 3]
  ```

JSON data types

Data type	Examples
String	`"WBDG is my favorite module!"`
Number	`123`, `1.23`
Boolean value	`true`, `false`
Null value	`null`
Array	`["St.Gallen", "Rapperswil", "Buchs"]`
Object	`{"id": 1, "name": "WBDG", "lecturer": "Baldauf"}`

```
{
  "id": 1,
  "modulename": "WBDG",
  "year": 2023,
  "semester": "FS",
  "description": "WBDG is part of the Business Software Development major...",
  "lecturer": {
    "id": 12,
    "familyname": "Baldauf",
    "firstname": "Matthias",
    "degree": "Dr."
  },
  "students": [
    {
      "id": 23,
      "familyname": "Mustermann",
      "firstname": "Max"
    },
    { ... }
  ]
}
```

Useful JSON tools:
https://jsonlint.com/
https://jsonformatter.org/

OST

Working with JSON in JavaScript

Access to values via key, either with dot or bracket notation

```
let jsonObject = { "id":1, "modulename": "WBDG", "students": [12, 23, 34, 45] };

// two ways to access JSON elements
console.log(jsonObject.id); // Points notation
console.log(jsonObject["modulename"]); // Bracket notation

console.log(jsonObject.students[0]); // Access array
```

Working with JSON in JavaScript

Loop through items of an array or JSON object

```
jsonObject.students.forEach(function (item, index) { // iterate over all array elements
  console.log(index + ": " + item);
});

jsonObject.map((item, index) => { // iterate over all object elements
  console.log(index + ": " + item);
})
```

Working with JSON in JavaScript

- Important: Distinguish between JSON object and its text representation!
- `JSON.parse` turns a string into a JSON object, `JSON.stringify` vice versa

```
let jsonString = '{ "id":1, "modulename": "SWI1" }'; // plain string, no object!
console.log(jsonString);

let jsonObjectFromString = JSON.parse(jsonString);
console.log(jsonObjectFromString);

let jsonStringFromObject = JSON.stringify(jsonObjectFromString);
console.log(jsonStringFromObject);
```

JSON example

```
{
  "id": 1,
  "modulename": "WBDG",
  "year": 2023,
  "semester": "FS",
  "description": "WBDG is part of the Business Software Development major...",
  "lecturer": {
    "id": 12,
    "familyname": "Baldauf",
    "firstname": "Matthias",
    "degree": "Dr."
  },
  "students": [
    {
      "id": 23,
      "familyname": "Mustermann",
      "firstname": "Max"
    }
  ]
}
```

Exercise: How to access

- year
- lecturer‘s last name
- first student of the list

OST

Ajax calls with `fetch`

- Function `fetch`, call with URL of the service and various parameters
- Callbacks to define behavior on success/failure of the service call.

```
let requestOptions = {
  method: "GET"
  // various others possible
};

fetch("<SERVICE_URL>", requestOptions)
  .then(response => response.json()) // Interpret response as JSON
  .then(data => console.log("success", data)) // success case
  .catch(error => console.log("error", error));
```

https://developer.mozilla.org/en-US/docs/Web/API/Fetch_API
https://wiki.selfhtml.org/wiki/JavaScript/Window/fetch

OST

Ajax calls with `fetch`

- Other variants

```
fetch("<SERVICE_URL>", requestOptions)
  .then(function (response) {                    // write your own function here ...
    //...
    return response.json();
  })
  .then(data => handleData(data))                // or define a function to process the data
  .catch(error => console.log("error", error));

function handleData(data) {
  console.log(JSON.stringify(data));
}
```

https://developer.mozilla.org/en-US/docs/Web/API/Fetch_API
https://wiki.selfhtml.org/wiki/JavaScript/Window/fetch

OST

Ajax calls with fetch

- The fetch() method returns a promise
  - Either use .then and .catch as in previous slides or
  - it is possible to use async / await

```
let test_function = async function () {
  let response = await fetch(<API_URL>);
  let data = await response.json();
  return data;
}
```

→ The code "waits" for the resolution of the promise before it continues

Exercise

- Choose a sample service
  - https://restcountries.com
  - https://exchangerate.host
  - https://api.chucknorris.io/
- Call selected service with `fetch`
- Output some relevant parts of the response

Transfer form data

```
let form = document.getElementById("myform");
// the form element with id myform

let data = new FormData(form);
// FormData creates a key/value list of all input elements with name attribute!

let requestOptions = {
  method: "POST",        // transferring data mostly done with POST method
  body: data
};

// call fetch with these requestOptions
```


OST

Exercise

- Create a simple form (e.g., with one text field and one selection menu) in HTML
- Add a button for submitting the form data via POST using the `fetch` method
- Destination address: https://matthiasbaldauf.com/wbdg23/submitdatajson (returns all passed values in JSON format)

- Additional task:
  - How to access the HTTP response code in the callback function?