Bedeutung der HTTP-Methoden

- definiert in RFC 7231: "Hypertext Transfer Protocol (HTTP/1.1): Semantics and Content"
- durch eine Arbeitsgruppe der IETF
- Stand 2014

RFC 7231: definiert die Verarbeitung der Inhalte

```
Internet Engineering Task Force (IETF)                  R. Fielding, Ed.
Request for Comments: 7231                                         Adobe
Obsoletes: 2616                                          J. Reschke, Ed.
Updates: 2817                                                 greenbytes
Category: Standards Track                                      June 2014
ISSN: 2070-1721
```

Hypertext Transfer Protocol (HTTP/1.1): Semantics and Content

Abstract

The Hypertext Transfer Protocol (HTTP) is a stateless application-level protocol for distributed, collaborative, hypertext information systems. This document defines the semantics of HTTP/1.1 messages, as expressed by request methods, request header fields, response status codes, and response header fields, along with the payload of messages (metadata and body content) and mechanisms for content negotiation.

Methoden ("Verbs") (aus RFC 7231)

Method	Description	Sec.
GET	Transfer a current representation of the target resource.	4.3.1
HEAD	Same as GET, but only transfer the status line and header section.	4.3.2
POST	Perform resource-specific processing on the request payload.	4.3.3
PUT	Replace all current representations of the target resource with the request payload.	4.3.4
DELETE	Remove all current representations of the target resource.	4.3.5
CONNECT	Establish a tunnel to the server identified by the target resource.	4.3.6
OPTIONS	Describe the communication options for the target resource.	4.3.7
TRACE	Perform a message loop-back test along the path to the target resource.	4.3.8

All general-purpose servers MUST support the methods GET and HEAD.
All other methods are OPTIONAL.

"safe"-Methoden

Definition in RFC 7231:

- *"Request methods are considered "safe" if their defined semantics are essentially read-only; i.e., the client does not request, and does not expect, any state change on the origin server as a result of applying a safe method to a target resource. Likewise, reasonable use of a safe method is not expected to cause any harm, loss of property, or unusual burden on the origin server."*
- schließt serverinterne Verarbeitungen / Speicherungen nicht völlig aus!
- "safe"-Methoden sind: GET, HEAD, OPTIONS, ,TRACE

Konsequenz

GET-Anfragen der Art

```
GET xyz.de/webshop/?cmd=additem&key1=value1&key2=value2
```

um Daten auf dem Webserver neu einzutragen, sind unzulässig!

"idempotent"-Methoden

Definition in RFC 7231:

- *"A request method is considered "idempotent" if the intended effect on the server of multiple identical requests with that method is the same as the effect for a single such request."*
- Mehrfache Aufrufe müssen zum identischen Ergebnis führen
- "idempotent"-Methoden sind: PUT, DELETE und die "safe"-Methoden

Und wie muss man POST einschätzen?

- weder "safe"
    - erzeugt neue Daten auf dem Webserver
- noch "idempotent"
    - jeder weitere Aufruf erzeugt weitere Daten

RFC 7231 dazu:

```
The POST method requests that the target resource process the
representation enclosed in the request according to the resource's
own specific semantics.  For example, POST is used for the following
functions (among others):

o  Providing a block of data, such as the fields entered into an HTML
   form, to a data-handling process;
o  Posting a message to a bulletin board, newsgroup, mailing list,
   blog, or similar group of articles;
o  Creating a new resource that has yet to be identified by the
   origin server; and
o  Appending data to a resource's existing representation(s).
```