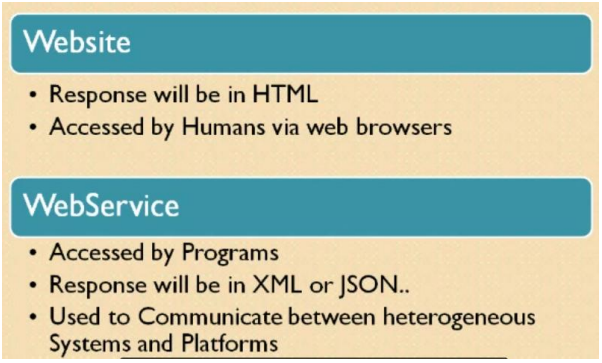
API, REST, Webservice,

Web-Services sind der Grundpfeiler moderner verteilter Anwendungen. Mit Hilfe von Web-Services schaffet man **Schnittstellen**, die es prinzipiell beliebigen Clients erlauben, Dienste der Server-Anwendung in Anspruch zu nehmen.

Unterschied Website und Webservice:



API:

Ein API stellt Entwicklern Daten zur Verfügung und nimmt Daten entgegen. Es steht für „application programming interface“. Es kann

- Daten in einer Datenbank ablegen, und Endpunkte bereitstellen nämlich
- zum Lesen,
- Anlegen,
- Bearbeiten und
- Löschen.

Der Vorteil von APIs: als deren Nutzer muss man sich nicht damit auseinandersetzen, wo die Daten gespeichert werden und wie die Datenbank dahinter aussieht. Man schickt JSON an HTTP(s)-Endpunkte und bekommt JSON-Daten zurück. Wie die Endpunkte heißen und wie die Anfrage und die Antwort aussehen, beschreibt die API-Dokumentation. Die wird hier nun beschrieben:

Kein vernünftiger Programmierer wird heute direkt über manuell zusammengebaute Nachrichten mit einem Web-Service kommunizieren. **Stattdessen verwendet man APIs**, die die gesamte Kommunikation zwischen Client und Server transparent durchführen. Ob für diese Kommunikation dann JSON, XML oder ein völlig anderes Format eingesetzt wird, spielt dabei keine Rolle. Tatsächlich gibt es APIs, bei denen das Nachrichtenprotokoll beliebig ausgetauscht werden kann, ohne dass sich für den Programmierer etwas ändert.

Aktuell unterscheidet man insbesondere **zwei Arten von Web-Services**. Und zwar

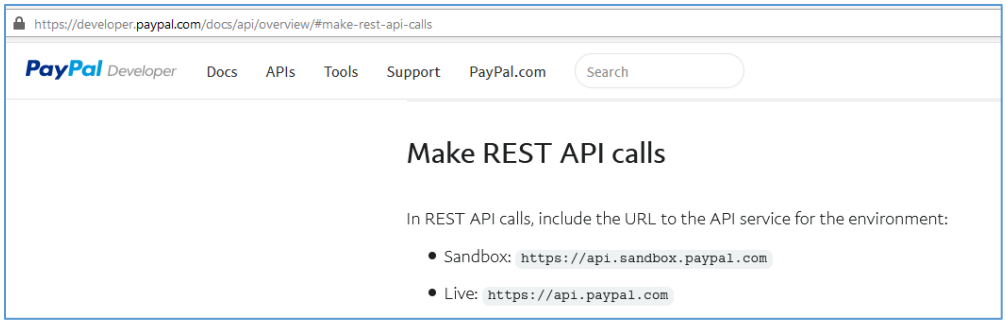
- **SOAP und**
- **REST.**

Beide haben gemeinsam, dass zwei Parteien miteinander kommunizieren. Ein Client und ein Server.

Bei einem auf SOAP basierten Web-Service kann der Datenaustausch zwischen Server und Client nur mithilfe von XML-Daten geschehen wohingegen wir bei einem REST Web-Service bei der Auswahl des Datenformats im Wesentlichen frei sind.

Viel wichtiger ist aber REST.

Beispiel: Eine API ist wie eine URL, die einen JSON-Response zurückgibt. Hier für Paypal, um Paypal im eignen Webshop anbieten zu können.



Representational State Transfer (REST)

REST ist kein Web-Service-Protokoll. REST ist weder eine konkrete Technologie noch ein offizieller Standard. Es handelt sich vielmehr um einen **Softwarearchitekturstil**, bestehend aus Leitsätzen und bewährten Praktiken für netzwerkbasierte Systeme.

Das REST-Paradigma entwickelte sich aus dem 1994 von Roy Fielding entworfenen HTTP Object Model. Fielding entwickelte seine Idee von einem einheitlichen Konzept über die Jahre weiter, bis er 2000 den REST-Architekturstil im Rahmen seiner Dissertation veröffentlichte. Fielding war zuvor auch an der Spezifikation des Hypertext-Transfer-Protokolls (HTTP) beteiligt.

RESTful APIs werden von vielen Websites eingesetzt, unter anderen von Google, Amazon, Twitter und LinkedIn.

Tatsächlich handelt es sich bei REST um ein Architekturmodell für verteilte Anwendungen. Der bekannteste Vertreter des REST-Architekturmodells ist das **World Wide Web**. Ausschlaggebend für die „RESTfulness“ (als **RESTful** bezeichnet man eine Anwendung, die den Prinzipien des REST-Modells folgt.) einer Architektur ist dabei das Vorhandensein von Ressourcen,

- die über eine einheitliche Syntax (URLs) adressierbar sind, sowie einheitlicher Operationen, die auf diese Ressourcen angewendet werden können.

Obwohl dieses Prinzip allen Websites zugrunde liegt, ist es insbesondere im Zusammengang mit Web-Services interessant.

Die technische Grundlage des REST-Modells ist das http-Protokoll.

REST nutzt HTTP für eine zustandslose Client-Server-Kommunikation, d. h., ein Client sendet Anfragen (Requests) an einen Server, der diese bearbeitet und dann Antworten (Responses)

zurücksendet. Man unterscheidet zwischen einem REST-Server, der Ressourcen bereitstellt, und REST-Clients, die auf diese Ressourcen zugreifen. Dazu besitzt jede Ressource eine ID, die in der Regel **als URI** (Uniform Resource Identifier) modelliert ist.

Zur Adressierung eines REST-Service (auch Ressource genannt) dient eine URI, die aus Server und Port sowie einem Basispfad und einem Pfad der Ressource besteht:

http://server:port/basePath/resourcePath/

Bei einer **RESTful API** handelt es sich um eine Programmierschnittstelle, die HTTP-Anfragen verwendet, um per GET, PUT, POST und DELETE auf Daten zuzugreifen.

Damit wird der darunter registrierte REST-Service angesprochen, der die gewünschte Aktion ausführt. Es werden z. B. neue Datensätze angelegt oder Informationen zu bestehenden Datensätzen abgefragt. Die Kommunikation basiert auf http und stützt sich vor allem auf die vier Operationen POST, GET, PUT und DELETE:

- POST – erzeugt neue Daten, d.h. eine neue Ressource.
- GET – definiert einen Lesezugriff auf eine oder mehrere Ressourcen. GET ist das „Arbeitspferd" des World Wide Web. Mehr als 95% aller Interaktionen im WWW haben lesenden Charakter.
- PUT – Verändert eine existierende Ressource. Falls diese noch nicht existiert, kann eine Ressource auch neu erzeugt werden.
- DELETE – löscht eine Ressource.

Beispiele für mögliche Endpunkte:

Endpunkt	HTTP-Methode	Funktion
/magazines	GET	gibt alle Magazine zurück
/magazines	POST	erstellt ein Magazin
/magazines/{id}	GET	gibt das Magazin mit der angegebenen ID zurück
/magazines/{id}	PUT	aktualisiert den kompletten Datensatz eines Magazins
/magazines/{id}	PATCH	aktualisiert Teile des Datensatzes eines Magazins
/magazines/{id}	DELETE	löscht den Datensatz eines Magazins

Möchte man den Server also beispielsweise anweisen, etwas zu löschen, so könnte man – statt wie gewohnt – einer POST- oder GET-Anfrage auch eine DELETE-Anfrage benutzen. Doch dann fragt sich, wie man dem Server mitteilen soll, was er zu löschen hat. Auch hierfür hat das HTTP-Protokoll (oder vielmehr das Web) bereits die Antwort: Jeder „Ressource", die der Server verwaltet, wird eine eigene URL zugeordnet.

Beispiel:
möchte man also beispielsweise den Kunden mit der Nummer 94213 aus der Datenbank löschen, so könnte man dem Server eine DELETE-Anfrage an die URL

http://www.mustersoft.de/kunden/94213

senden. Möchte man stattdessen nur die persönlichen Daten des Kunden aktualisieren, so würde man eine PUT-Anfrage starten. Allerdings wäre es damit nicht getan, denn die neuen Kundendaten müssen dem Server ja in irgendeiner Form mitgeteilt werden. Hierzu macht REST keine genauen Vorschriften. In der Praxis verwendet man allerdings häufig XML. Prinzipiell kann man aber auch HTML oder jedes beliebige andere Format wählen. Da REST hier keine besonderen Anforderungen stellt, muss natürlich gewährleistet sein, dass Client und Server dieselbe Sprache sprechen bzw. verstehen. Das REST-Modell zeichnet sich vor allem dadurch aus, dass es leicht zu verstehen ist und auf Konzepten aufbaut, die sich immer wiederholen. Web-Services, die diesem Muster folgen, werden deshalb immer beliebter. So bietet beispielsweise Amazon seine WebServices inzwischen sowohl in einer SOAP- als auch in einer REST-Variante an.

Die Sammlung aller REST-Services auf einem Server nennt man REST API.

Frameworks für das Implementieren von REST-Schnittstellen:

Es existieren viele Frameworks.

- Restify Plattform: JavaScript/Node.js mcavage.me/node-restify
- Slim Plattform: PHP www.slimframework.com
- Laravel https://laravel.com/ ein mächtiges PHP-Framework
- Lumen Micro-Framework, von Laravel abstammend https://lumen.laravel.com/

Quellen:

Stefan Tilkov in: REST und http; dpunkt-Verlag, Heidelberg, 2015

Manuel Ottlik, in: Schnittstellen-Erklärer, c´t Magazin Heft 5 2020, S.136-139 (www.ct.de/yg72)
Manuel Ottlik, in RESTregal, c´t Magazin Heft 2, 2010, S. 154-159, (www.ct.de/y3hx)

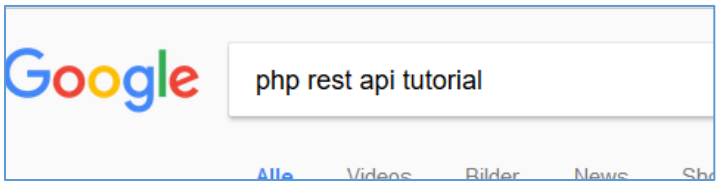
https://www.youtube.com/watch?v=gYlj8-0O-Y4

Links:

- 2020.Juni: REST-API in PHP in 3 Teilen, Start, single get und Post, update und delete:aus 2018-Traversy Media – KEIN Login
  https://www.youtube.com/watch?v=OEWXbpUMODk
  https://www.youtube.com/watch?v=-nq4UbD0NT8
  https://www.youtube.com/watch?v=tG2U18EmIu4
- 2020.Juni: REST-API in PHP with „ProgrammingKnowledge“ 1:43:35 alle in einem:
  https://www.youtube.com/watch?v=dlGtSoigdB0
-
- 2020.Juni slim v3, ab 2. Login und Signup
  https://www.youtube.com/watch?v=XdiDxls2Ut4 - Slim start
  https://www.youtube.com/watch?v=OPs7-6Wk3DE
  https://www.youtube.com/watch?v=eMGIC5mbVyQ
-
- SLIM3 Restful API mit PHP und MySQL – mit INSERT INTO….
  https://www.youtube.com/watch?v=DHUxnUX7Y2Y
-
- Erklärung API:
  https://www.youtube.com/watch?v=KLe2lCEy-Xw

- Zugriff auf Github und „users“ – REST:
- https://www.youtube.com/watch?v=Q-BpqyOT3a8

https://poe-php.de/tutorial/rest-einfuehrung-in-die-api-erstellung



https://www.leaseweb.com/labs/2015/10/creating-a-simple-rest-api-in-php/

dazu gehört zum Download: https://github.com/mevdschee/php-crud-api

Mit Slim: http://mfg.fhstp.ac.at/development/erstellung-eines-einfachen-rest-api-backends-mit-php/

https://www.youtube.com/watch?v=DHUxnUX7Y2Y

deutsch!!! https://www.youtube.com/watch?v=nULAxphFrCA

englisch: https://www.youtube.com/watch?v=CmBm59V-SOE

Erklärung REST API und RESTful Web Service: https://www.youtube.com/watch?v=LooL6_chvN4

https://www.youtube.com/watch?v=RTjd1nwvlj4

https://www.youtube.com/watch?v=wbB3lVyUvAM

https://www.youtube.com/watch?v=hMxGhHNOkCU