APIS

VORLESUNG AM SAE INSTITUTE

DOZENT: BJOERN ZAPADLO

ABOUT ME

Bjoern Zapadlo
Konstanz
34 Jahre

Teamlead / Senior Software Developer HolidayCheck AG

Informatik Studium 1999 - 2002
3 Agenturen in Stuttgart

HolidayCheck International Websites / new Framework
Neckermann / Thomas Cook

Dozent an der SAE, Dualen Hochschule Stuttgart,
Hochschule Furtwangen

PHP, Java, Scala, Javasccript, CSS, Html,MySQL,
MongoDB, Elasticsearch, ...

CONTACT ME

bjoern.zapadlo@gmail.com
http://www.zapadlo.de

@BjoeZap

https://www.xing.com/profile/Bjoern_Zapadlo
http://de.linkedin.com/pub/bjoern-zapadlo/36/889/1a5

Facebook
Google+

HOLIDAYCHECK AG

Größtes deutsches Meinungsportal für Reise und Urlaub
Vermittlung von Reisen
Sitz in der Schweiz, direkt am Bodensee
Börsennotiert über Tomorrow Focus AG
Existiert seit 1999
Ausgründungen in mehreren europäischen Ländern
Über 300 Mitarbeiter

HTTP://WWW.HOLIDAYCHECK.DE/JOBS

JETZT ABER SCHLUSS MIT DER WERBUNG...

UND IHR?

Name

Erwartungen

Wünsche

AGENDA

1. Definition - API
2. Daten
3. Kommunikation
4. OAuth
5. Die eigene API
6. Twitter
7. Facebook
8. GoogleMaps
9. OpenStreetMap
10. Amazon
11. Misc / Q&A

1. DEFINITIONEN

WAS IST EINE API?

API... WTF?

Active Pharmaceutical Ingredient: englische Bezeichnung für einen Wirkstoff in einem Medikament

Adaptive Planungsintelligenz: mathematische Methoden für industrielle Planungsaufgaben

African Plants Initiative: ein Projekt zur Förderung botanisch-taxonomischer Arbeit in Afrika

anonima petroli italiana SpA: eine italienische Ölgesellschaft

Arbeiterkommunistische Partei Irans: eine iranische Exilpartei

Api: ein Berg im Westen Nepals

...

APPLICATION PROGRAMMING INTERFACE

ENGLISCH FÜR PROGRAMMIERSCHNITTSTELLE IN DER INFORMATIK

DEFINITION

Eine Programmierschnittstelle ist ein Programmteil, der von einem Softwaresystem anderen Programmen zur Anbindung an das System zur Verfügung gestellt wird. Oft wird dafür die Abkürzung API (engl. application programming interface, dt. „Schnittstelle zur Anwendungsprogrammierung“) verwendet. Im Gegensatz zu einer Binärschnittstelle (ABI) definiert eine API nur die Programmanbindung auf Quelltextebene.

FUNKTIONSORIENTIERT

Funktionsorientierte Programmierschnittstellen kennen nur Funktionen mit oder ohne Rückgabewert als Mittel der Kommunikation. Dabei wird fast immer das Konzept der Handles verwendet. Man ruft eine Funktion auf und bekommt ein Handle zurück. Mit diesem Handle lassen sich weitere Funktionen aufrufen, bis abschließend das Handle wieder geschlossen werden muss. Das BIOS eines Personal Computer ist die älteste Programmierschnittstelle für diesen Rechnertyp.

DATEIORIENTIERT

Dateiorientierte Programmierschnittstellen werden über die normalen Dateisystemaufrufe open, read, write und close angesprochen. Sollen Daten an ein Objekt gesendet werden, werden diese mit write geschrieben, sollen welche empfangen werden, werden sie mit read gelesen. Unter UNIX ist dieses Prinzip bei der Ansteuerung von Gerätetreibern weit verbreitet.

OBJEKTORIENTIERT

Objektorientierte Programmierschnittstellen verwenden Schnittstellenzeiger und sind damit deutlich flexibler als die funktionsorientierten Programmierschnittstellen. Häufig wird eine Typbibliothek mitgegeben.

PROTOKOLLORIENTIERT

Protokollorientierte Programmierschnittstellen sind unabhängig vom Betriebssystem und der Computerhardware. Allerdings muss das Protokoll stets neu implementiert werden. Um diesen Aufwand zu minimieren, wird die protokollorientierte Schnittstelle durch eine funktions- oder interfaceorientierte Schnittstelle gekapselt. Man kann hier weiterhin zwischen allgemeinen (z. B. SOAP) und anwendungsspezifischen (z. B. SMTP) Protokollen unterscheiden.

WOZU NE API?

ZUGÄNGLICH MACHEN VON FUNKTIONALITÄT ÜBER EINEN STANDARDISIERTEN WEG

WELCHE APIS KENNT IHR?

TWITTER JAVA ECLIPSE

YOUTUBE GOOGLE MAPS

AMAZON WINDOWS PHP

PYTHON ZEND FRAMEWORK

OPEN STREETMAP ADOBE AIR

GOOGLE GEARS FACEBOOK

ANDROID FLASH JQUERY

"INTERNE" APIS

Jede Programmiersprache und jedes Framework besitzen eine Dokumentation, in welcher der Zugriff auf Funktionen der Programmiersprache / des Frameworks erklärt werden.

In Objekt-orientierten Programmiersprachen (z.B. Java, PHP) nennt man das API.

"EXTERNE" APIS

Gerade im Bereich des Internets werden die „externe“ API's (Google Maps, Flickr, Facebook, ...) immer wichtiger. Hierbei handelt es sich entweder um Webservices (SOAP / Rest) oder um externe Bibliotheken, die eingebunden werden. Hierbei sind vor allem Bibliotheken für Javascript von Interesse.

Hierbei wird meist ein sog. API-Key benötigt.

API KEY

An application programming interface key (API key) is a code passed in by computer programs calling an API (application programming interface) to identify the calling program, its developer, or its user to the Web site. API keys are used to track and control how the API is being used, for example to prevent malicious use or abuse of the API (as defined perhaps by terms of service).

The API key often acts as both a unique identifier and a secret token for authentication, and will generally have a set of access rights on the API associated with it.

API keys can be based on the UUID system (e.g. 550e8400-e29b-41d4-a716-446655440000) to ensure they will be unique to each user.

2. DATEN

WELCHE DATENAUSTAUSCHFORMATE KENNT IHR?

Text

XML

Html

JSON

Binärdaten

TEXT

Einfachstes Format

Quasi schon antik ;)

Strukturierung schwierig => CSV

Kaum overhead

Encoding

XML

Strukturierter Text

Viel overhead

Eigene Tags möglich

Muss bestimmten Regeln entsprechen

Lange Zeite der quasi Standard

XML - BEISPIEL

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<verzeichnis>
     <titel>Wikipedia Städteverzeichnis</titel>
     <eintrag>
          <stichwort>Genf</stichwort>
          <eintragstext>Genf ist der Sitz von ...</eintragstext>
     </eintrag>
     <eintrag>
          <stichwort>Köln</stichwort>
          <eintragstext>Köln ist eine Stadt, die ...</eintragstext>
     </eintrag>
</verzeichnis>
```

JSON

Javascript Object Notation

Kurzschreibweise für Javascript Objekte

Sehr einfach

Kaum overhead

Native Unterstützung in vielen Programmiersprachen

JSON - BEISPIEL

```
{
    "titel": "Wikipedia Städteverzeichnis",
    "eintrag": [
        {
            "stichwort": "Genf",
            "eintragstext": "Genf ist der Sitz von ..."
        },
        {
            "stichwort": "Köln",
            "eintragstext": "Köln ist eine Stadt, die ..."
        }
    ]
}
```

UND WAS IST JETZT DAS BESTE?

Es kommt darauf an... ;)

Text bei einfachen Funktionalitäten

JSON mittlerweile quasi Standard

3. KOMMUNIKATION

WARUM BRAUCHT MAN EINEN STANDARD?

VERSTEHT EIN NORWEGER EINEN GRIECHEN? ;)

Kommunikation unabhängig von der Programmiersprache soll ermöglicht werden

WEBSERVICES

Das World Wide Web Consortium definiert die Bereitstellung eines Webservices als Unterstützung zur Zusammenarbeit zwischen verschiedenen Anwendungsprogrammen, die auf unterschiedlichen Plattformen und/oder Frameworks betrieben werden.

Ein Webservice oder Webdienst ist eine Software-Anwendung, die mit einem Uniform Resource Identifier (URI) eindeutig identifizierbar ist und deren Schnittstelle als XML-Artefakt definiert, beschrieben und gefunden werden kann. Ein Webservice unterstützt die direkte Interaktion mit anderen Software-Agenten unter Verwendung XML-basierter Nachrichten durch den Austausch über internetbasierte Protokolle.

WTF...?

Schnittstelle via Netzwerk ansprechbar

Externe Funktionalität

Kommunikation meist via XML, es gibt aber auch andere Formate

SOA in modernen Systemen

SOAP

SOAP (ursprünglich für Simple Object Access Protocol) ist ein Netzwerkprotokoll, mit dessen Hilfe Daten zwischen Systemen ausgetauscht und Remote Procedure Calls durchgeführt werden können. SOAP stützt sich auf XML zur Repräsentation der Daten und auf Internet-Protokolle der Transport- und Anwendungsschicht (vgl. TCP/IP-Referenzmodell) zur Übertragung der Nachrichten. Die gängigste Kombination ist SOAP über HTTP und TCP.

Die Abkürzung SOAP wird offiziell seit Version 1.2 nicht mehr als Akronym gebraucht, da es erstens (subjektiv) keineswegs einfach (Simple) ist und zweitens nicht nur dem Zugriff auf Objekte (Object Access) dient.

SOAP

Meist entspricht ein Serviceendpunkt einer URI, die mehrere Services anbietet. Diese werden per WSDL bekannt gegeben.

Die Web Services Description Language (WSDL) ist eine plattform-, programmiersprachen- und protokollunabhängige Beschreibungssprache für Netzwerkdienste (Webservices) zum Austausch von Nachrichten auf Basis von XML.

SOAP NACHRICHT

```
<?xml version="1.0"?>
<s:Envelope xmlns:s="http://www.w3.org/2003/05/soap-envelope">
   <s:Header>
       <m:RequestID xmlns:m="http://www.lecture-db.de/soap">a3f5c109b</m:
   </s:Header>
   <s:Body>
       <m:DbResponse xmlns:m="http://www.lecture-db.de/soap">
           <m:title value="DOM, SAX und SOAP">
               <m:Choice value="1">Arbeitsbericht Informatik</m:Choice>
               <m:Choice value="2">Seminar XML und Datenbanken</m:Choice>
           </m:title>
       </m:DbResponse>
   </s:Body>
</s:Envelope>
```

REST

Representational State Transfer (mit dem Akronym REST) bezeichnet ein Programmierparadigma für Webanwendungen. Es gibt keine explizite Norm, daher gehen die Vorstellungen, was REST ist, auseinander. I m Grunde bezeichnet REST die Idee, dass eine URL genau einen Seiteninhalt als Ergebnis einer serverseitigen Aktion (etwa das Anzeigen einer Trefferliste nach einer Suche) darstellt, wie es der Internetstandard HTTP für statische Inhalte (Permalink) bereits vorsieht, ein Ziel, das für dynamisch erzeugte Seiten mitunter jedoch zusätzlichen Aufwand erfordert.

PRINZIPIEN VON REST

ADRESSIERBARKEIT

Jeder Dienst, den ein REST-konformer Server zur Verfügung stellt, hat eine eindeutige Adresse, den Uniform Resource Identifier (URI). Diese „Straße und Hausnummer im Netz“ durch einen URI stellt sicher, dass das Angebot eines Webservices auf einfache, standardisierte Art und Weise einer Vielzahl von Anwendungen (Clients) zur Verfügung steht. Eine konsistente Adressierbarkeit erleichtert es außerdem, einen Webservice als Teil eines Mashups weiterzuverwenden.

UNTERSCHIEDLICHE REPRÄSENTATIONEN

Die unter einer Adresse zugänglichen Dienste können unterschiedliche Darstellungsformen (Repräsentationen) haben. Ein REST-konformer Server kann je nachdem, was die Anwendung anfordert, verschiedene Repräsentationen ausliefern (z. B. HTML, JSON oder XML). Damit kann der Standard-Webbrowser über die gleiche URI-Adresse sowohl den Dienst, als auch die Beschreibung oder Dokumentation abrufen. Es wird nicht der interne Datenbankeintrag ausgeliefert, sondern das je nach Anfrage evtl. in eine bestimmte Kodierung oder Sprachversion übertragene Dokument.

ZUSTANDSLOSIGKEIT

REST setzt auf ein zustandsloses Client-Server-Protokoll. Dabei enthält jede HTTP-Botschaft alle Informationen, die notwendig sind, um die Nachricht zu verstehen. Deshalb muss weder der Server noch die Anwendung Zustandsinformationen zwischen zwei Nachrichten speichern. Eine derartig strikte Trennung der Zuständigkeiten zwischen Client und Server führt dazu, dass ein REST-konformer Webservice als zustandslos (stateless) bezeichnet werden kann: Jede Anfrage eines Clients an den Server ist in dem Sinne in sich geschlossen, als dass sie sämtliche Informationen über den Anwendungszustand beinhaltet, die vom Server für die Verarbeitung der Anfrage benötigt werden.

Zustandslosigkeit in der hier beschriebenen Form wirkt sich begünstigend auf die Skalierbarkeit eines Webservices aus. Beispielsweise können eingehende Anfragen im Zuge der Lastverteilung unkompliziert auf beliebige Maschinen verteilt werden: Da jeder Request in sich geschlossen ist und Anwendungsinformationen somit ausschließlich auf der Seite des Clients vorgehalten werden, ist auf der Seite des Servers keine Sitzungsverwaltung erforderlich. In der Praxis nutzen jedoch viele HTTP-basierte Anwendungen Cookies und andere Techniken, um Zustandsinformationen zu behalten.

VERWENDUNG VON HYPERMEDIA

Sowohl für Anwendungsinformationen als auch für Zustandsveränderungen werden Hypermedia benutzt: Repräsentationen in einem REST-System sind typischerweise im HTML- oder XML-Format, welche sowohl Informationen als auch Links zu anderen Ressourcen enthalten. Deshalb ist es oftmals möglich, von einer Ressource zu einer anderen zu navigieren, indem man einfach Verknüpfungen folgt, ohne dass dafür Registrierungsdatenbanken oder ähnliche Infrastrukturen erforderlich sind. Diese Verknüpfung von Ressourcen innerhalb einer REST-Architektur wird auch als Verbindungshaftigkeit bezeichnet.

OPERATIONEN

REST-konforme Dienste müssen einige Operationen vorsehen, die auf alle Dienste (Ressourcen genannt) angewendet werden können. HTTP, zum Beispiel, hat eine klar definierte Menge von Operationen, darunter GET, POST, PUT und DELETE.

HTTP schreibt vor, dass GET „sicher“ (englisch safe) sein muss, was bedeutet, dass diese Methode nur Informationen beschafft und keine sonstigen Effekte verursacht. Die Methoden GET, HEAD, PUT und DELETE müssen laut HTTP-Spezifikation idempotent sein, was in diesem Zusammenhang bedeutet, dass das mehrfache Absenden der gleichen Anforderung sich nicht anders auswirkt als ein einzelner Aufruf.

HTTP / REST OPERATIONEN

GET: fordert die angegebene Ressource vom Server an.

POST: fügt eine neue (Sub-)Ressource unterhalb der angegebenen Ressource ein. Da die neue Ressource noch keine URI besitzt, adressiert den URI die übergeordnete Ressource. Als Ergebnis wird der neue Ressourcenlink dem Client zurückgegeben. POST kann im weiteren Sinne auch dazu verwendet werden Operationen abzubilden, die von keiner anderen Methode abgedeckt werden.

PUT: die angegebene Ressource wird angelegt oder dazu geändert. Üblicherweise ohne Seiteneffekte.

DELETE: löscht die angegebene Ressource.

HEAD: fordert Metadaten zu einer Ressource vom Server an.

OPTIONS: prüft, welche Methoden auf einer Ressource zur Verfügung stehen.

PATCH: ein Teil der angegeben Ressource wird geändert, Seiteneffekte erlaubt.

UND WAS JETZT BENUTZEN?

REST

SO FUNKTIONIERT DAS INTERNET

4. OAUTH

WAS IST OAUTH?

OAuth ist ein offenes Protokoll, das eine standardisierte, sichere API-Autorisierung für Desktop-, Web- und Mobile-Applikationen erlaubt. Es wurde von Blaine Cook und Chris Messina initiiert.

Ein Endbenutzer (User) kann mit Hilfe dieses Protokolls einer Anwendung (Consumer) den Zugriff auf seine Daten erlauben (Autorisierung), die von einer anderen Anwendung (Service) verwaltet werden, ohne alle Details seiner Zugangsberechtigung zur anderen Anwendung (Authentifizierung) preiszugeben. Der Endbenutzer kann so Dritte damit beauftragen und dazu autorisieren, sich von ihnen den Gebrauchswert von Anwendungen erhöhen zu lassen. Typischerweise wird dabei die Übermittlung von Passwörtern an Dritte vermieden.

BEGRIFFE

SERVICE PROVIDER

(deutsch: „Dienstanbieter“) ist eine Website oder ein Web Service, bei dem die Informationen liegen, auf die ein kontrollierter Zugriff erlaubt werden soll. Er hat die volle Kontrolle und Verantwortung für die Implementierung von OAuth. Der Service Provider muss kein Identity Provider (Anbieter für ein zentrales Identititätsmanagement; zum Beispiel OpenID) sein.

USER

(deutsch: „Benutzer“) ist der Eigentümer der Informationen. Ihm soll mit OAuth die Kontrolle über seine Informationen ermöglicht werden. OAuth ist so konzipiert, dass nur die manuelle Freigabe des Users den Zugriff auf seine Daten ermöglicht.

CONSUMER

(deutsch: „Nachfrager“ oder „Konsument“) ist eine Applikation, die Zugriff auf die Informationen eines Users erlangen möchte. Es kann eine Website, ein Desktopprogramm, eine Mobile-Applikation, eine Set-Top-Box und so weiter sein, die in jedem Fall Zugang zum Internet haben muss. Ein Consumer Developer ist der Entwickler der Consumer-Applikation, die mit dem Service Provider interagiert.

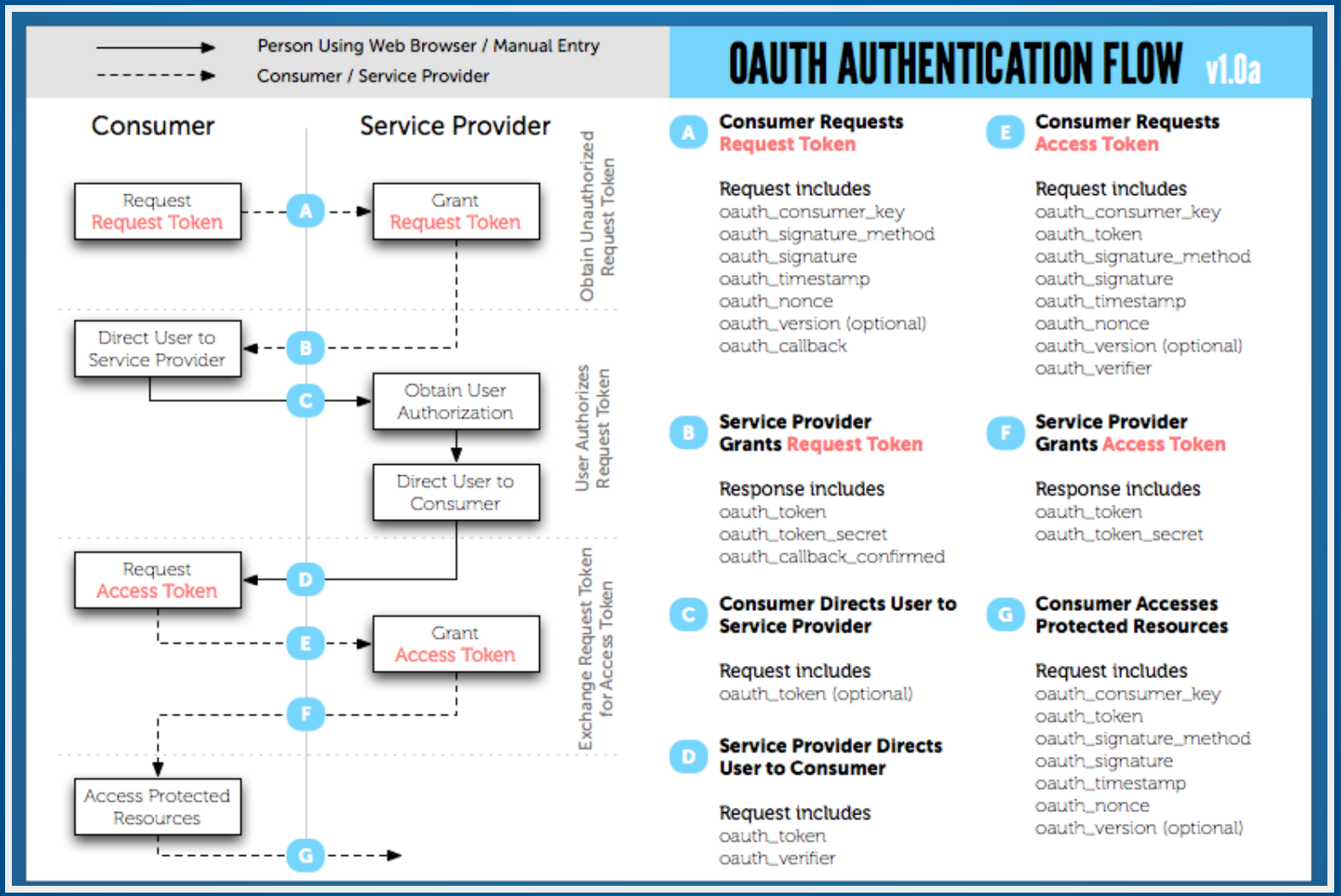
PROTECTED RESOURCES

(deutsch: „geschützte Ressourcen“) sind die Informationen des Users, auf die mit Hilfe von OAuth ein kontrollierter Zugriff erlaubt werden soll. Dabei kann es sich um Daten (Fotos, Dokumente, Adressen und so weiter), Aktivitäten (das Schreiben von Blogbeiträgen, der Transfer von Geld und so weiter) oder den geschützten Zugriff auf eine URL

TOKEN

werden an Stelle von Benutzername-Passwort-Kombinationen verwendet, um auf Ressourcen zuzugreifen. Ein Token ist meist eine Zeichenkette aus Buchstaben und Zahlen; Sonderzeichen können auch verwendet werden. Um es vor Missbrauch zu schützen soll es schwer zu erraten und passend zu einer Sicherheitsabfrage sein. OAuth unterscheidet zwischen Abfrage-Token und Zugangs-Token.

WIE FUNKTIONIERT OAUTH?



OAUTH 2.0

Einfacher

Mehrere Workflows (Javascript Application)

SSL

Regenerate Token

Aber noch in der Entwicklung

5. DIE EIGENE API

WANN / WARUM?

Eigenes Framework

Schnittstelle nach aussen

EINE GUTE API

Easy to learn

Easy to use, even without documentation

Hard to misuse

Easy to read and maintain code that uses it

Sufficiently powerful to satisfy requirements

Easy to extend

Appropriate to audience

UND WAS HEISST DAS?

Versionierung

Stabilität

Standards

Dokumentation

6. TWITTER

Twitter Dev Pages

Twitter Clients

7. FACEBOOK

Facebook Dev

8.
GOOGLEMAPS

Google Maps API

9. OSM (OPENSTREET MAP)

10. AMAZON

Affiliate API

11. MISC / Q&A

FURTHER INFO

Google

Wikipedia

Developer Resources Twitter / FB / ...

FRAGEN?

THE END

VIELEN DANK FÜR DIE AUFMERKSAMKEIT UND STIMMUNG!

Made with reveal.js