Echtzeitkartographie

Stefan Maihack Dipl. Ing. (FH)

01.10.2016

Einführung in die Echtzeit-Kartographie

- Dozent: Stefan Maihack Dipl. Ing. (FH); Lehrbeauftragter der FH-Karlsruhe
  → Email: stefan@maihack.de (Wichtig!!!)
  → Webseite: http://www.maihack.de
  → Angaben zu meiner Person
- Inhalt des Praktikums:
  ➔ Realisierung eines interaktiven, webbasierten Echtzeit-Informationssystem, bei dem Echtzeitdaten z.B. aktuelle Wetter- oder Verkehrsdaten eines beliebigen Landes dargestellt werden.
- Gruppenaufteilung und Themenverteilung:
  Themen: Klimadaten, Verkehrsdaten, Daten zur Gesundheit (Epidemien), Flüchtlingsmigrationsdaten, UFO-Sichtungen
- Andere Vorlesungszeit: Tag und Uhrzeit??? ➔ Samstag alle 14 Tage
- Kommunikation:
  Jeder Student und jede Studentin sendet bitte an die Emailadresse stefan@maihack.de eine Nachricht mit dem Betreff „**ECHTZEIT-KARTOGRAPHIE**“ und als Inhalt der Email bitte den Namen, die Matrikelnummer, und das Praktikumsthema, sowie die Gruppenbezeichnung angeben.
- Vorstellung der Studienarbeit → nächste Folien

Einführung in die Echtzeit-Kartographie

- Begriffe:
- **Echtzeit**, *Echtzeitbetrieb*, E real-time processing, in der Informationstechnologie eine Form der Datenverarbeitung, die von dem Nutzer als Antwortzeit eines DV-Systems als realistisch und verzögerungsfrei empfunden wird. Zur graphischen Präsentation dynamischer Vorgänge ist ein Echtzeitrendering notwendig, um z. B. Filme, Animationen oder Simulationen abspielen zu können oder Bewegungen in virtuellen Räumen (Virtual Reality) wiederzugeben.

  Beispiele:
  → aktuelle Wetterdaten
  → Verkehrsdaten
  → Flugzeugpositionen
  → Schiffspositionen
  → etc.
- **Real-Time-Webapplications**: Auslieferung . von Informationen in einem angemessenen Zeitraum
  Beispiele:
  → kollaborative Websites
  → Internet-Support
  → Spiele (auf der Basis von HTML 5)
  → Finanzanwendungen
  → Server-Monitoring-Anwendungen im Web
  → Kartografische Anwendungen

Einführung in die Echtzeit-Kartographie

Einige Beispiele

- http://www.marinetraffic.com/de/ais/home/centerx:-13/centery:57/zoom:7
- http://www.flightradar24.com/51,9/7
- http://tracker.geops.ch/?z=14&s=1&x=-8235743.4976&y=4971840.9526&l=transport
- http://www.whatsthere.co/

- http://www.spiegel.de/netzwelt/web/global-trend-trecker-15-echtzeit-animationen-und-live-tracker-a-1038974.html
- http://trendsmap.com/#
- http://cybermap.kaspersky.com/
- http://emojitracker.com/

Einführung in die Echtzeit-Kartographie

- Datenbeschaffung:

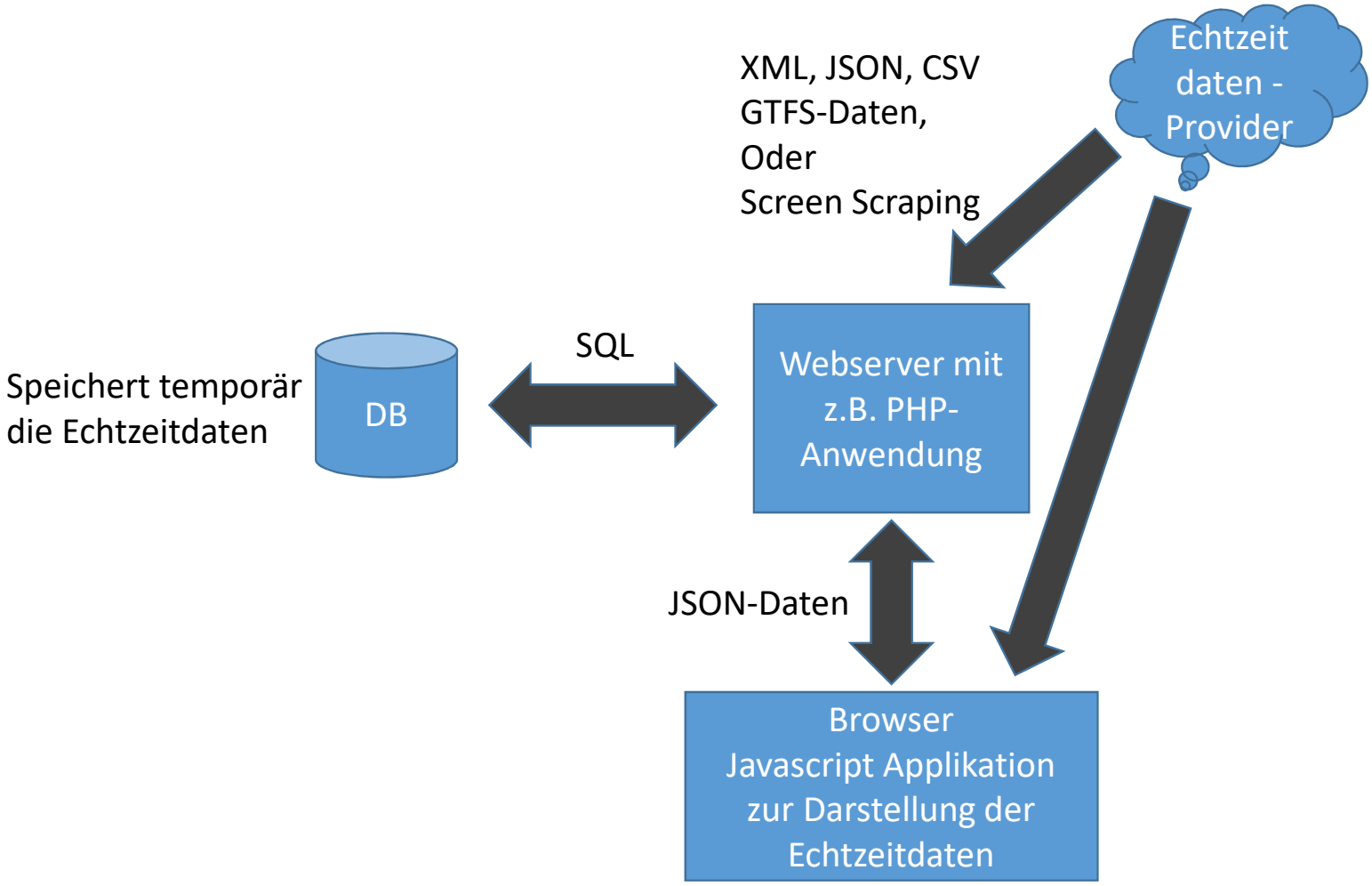
1. Mittels URL wird die Ressourcen bei einem Web- oder FTP-Server eines Providers angefragt.
2. Der Provider verarbeitet diese Anfrage und liefert das Ergebnis z.B. als XML- CSV- Datei oder JSON (GeoJSON)-Dokument (oder man muss sogar Screen Scraping machen).

- Problem: „Client-Server- und Server-Client-Verbindungen mit geringer Latenz“

  → Der Client fragt eine Ressource an, der Server antwortet und liefert. Der Server hat jedoch keinerlei Möglichkeit mit dem Client aktiv zu kommunizieren (keine bidirektionale Kommunikation).

- Mögliche Lösung: Pollingverfahren, WebSocket-API des W3C und PUSH (COMET) ➔ NÄCHSTE WOCHE MEHR DAZU!

Einführung in die Echtzeit-Kartographie



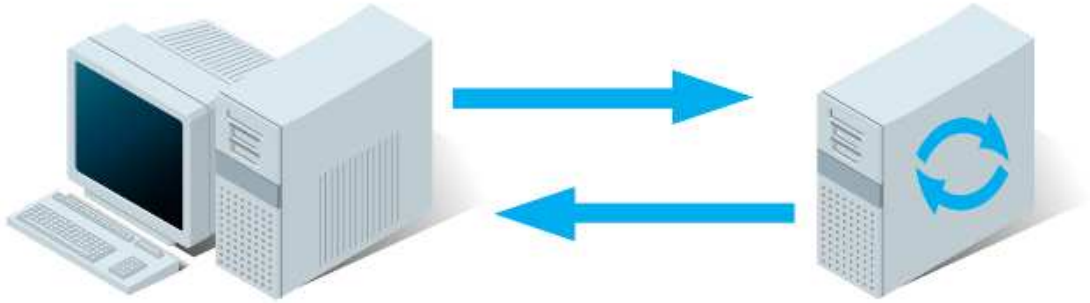
Einführung in die Echtzeit-Kartographie

- Aufgabenschwerpunkte

1. Beschaffung von sich kontinuierlich ändernden Daten (z.B. mittels WebSocket-Verbindung, Polling etc.)
2. Säubern der Daten und ablegen der Daten in eine Datenbank, falls notwendig
3. Datenumwandlung in ein nutzbares Datenaustauschformat (z.B. JSON/GeoJSON)
4. Erstellen einer Applikation zur kartographischen Darstellung der Daten (z.B. mittels Googlemaps API V3 oder Leaflet API)

Einführung in die Echtzeit-Kartographie

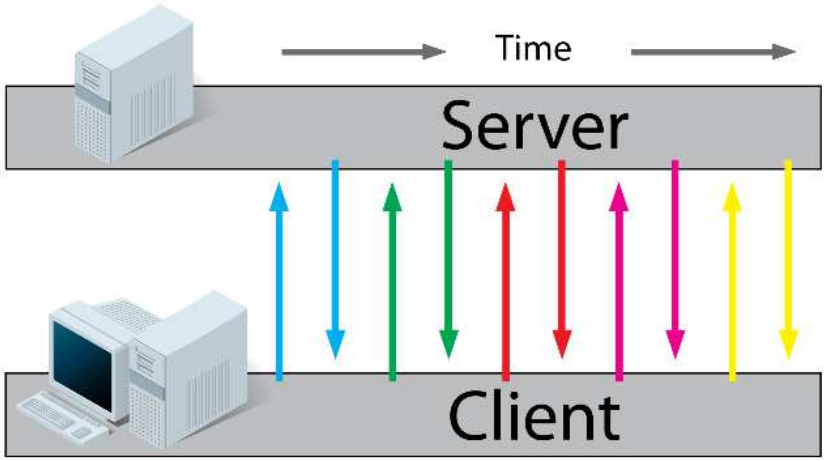
Datenübermittlung - Eine normale HTTP-Verbindung



1. Ein Client macht einen Request an einen Server
2. Der Server bearbeitet die Anfrage
3. Der Server sendet sie Antwort zum Client

Einführung in die Echtzeit-Kartographie

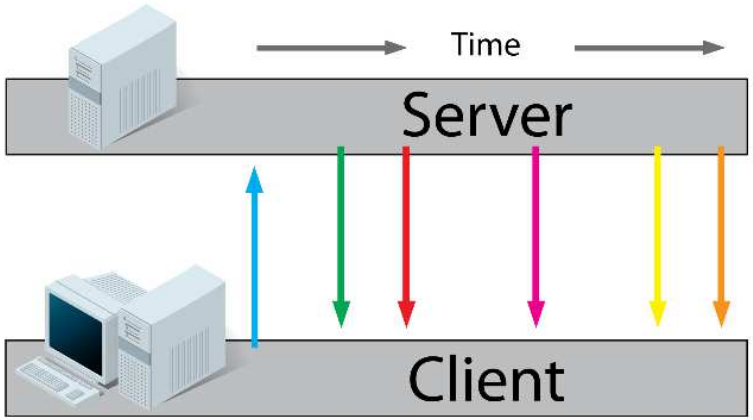
Datenübermittlung - AJAX Pollingverfahren



1. Ein Client macht einen Request an einen Server (ähnlich normalem HTTP-Request)
2. Die anfragende Webseite führt Javascript aus, welches immer wieder ein File in Intervallen vom Server empfängt.
3. Der Server bearbeitet die Anfrage und sendet diese wieder zurück an den Client.

Einführung in die Echtzeit-Kartographie

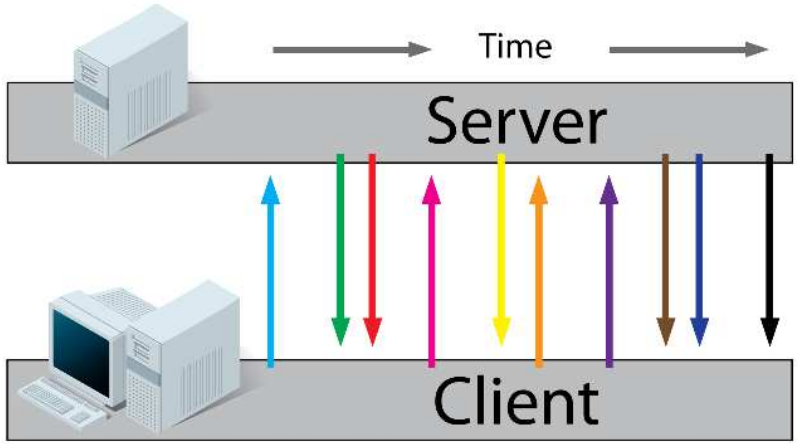
Datenübermittlung - HTML5 Server sendet Ereignisse (SSE) / EventSource



1. Ein Client macht einen Request an einen Server (ähnlich normalem HTTP-Request).
2. Die anfragende Webseite führt Javascript aus, welches eine Verbindung zum Server aufbaut.
3. Der Server sendet ein Ereignis (Event) zum Client, welches neue Informationen enthält.
   →Real-Time Traffic vom Server zum Client möglich (wird meistens benötigt).
   →Man verwendet einen Server, der eine Ereignisschleife verwendet.
   →Es ist nicht möglich sich mit einem Server zu verbinden, der eine andere Domäne als der Client hat.

Einführung in die Echtzeit-Kartographie

Datenübermittlung - COMET (PUSH) – Long Polling



- Comet ist eine Sammlung von HTML5-Techniken, die Streaming und Long-Polling für Echtzeitanwendungen bereitstellen.
  →http://en.wikipedia.org/wiki/Comet_%28programming%29
  →http://www.ibm.com/developerworks/web/library/wa-reverseajax1/index.html

1. Ein Client macht einen Request an einen Server (ähnlich normalem HTTP-Request).
2. Die anfragende Webseite führt Javascript aus, welches immer ein File vom Server empfängt.
3. Der Server antwortet aber nicht immer unmittelbar mit den Informationen, sondern wartet bis neue Informationen vorliegen.
4. Falls neue Informationen dem Server vorliegen, so schickt er diese an den Client.
5. Nachdem der Client die neuen Informationen erhalten hat, sendet er eine neue Anfrage an den Server.

Einführung in die Echtzeit-Kartographie

Datenübermittlung - COMET (PUSH) – Ein Beispiel

```
1  <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.1//EN" "http://www.w3.org/TR/xhtml11/DTD/xhtml11.dtd">
2  <html xmlns="http://www.w3.org/1999/xhtml">
3  <head>
4    <title>Comet Beispiel 1</title>
5    <meta http-equiv="Content-Type" content="text/html; charset=utf-8" />
6    <script type="text/javascript">
7
8    function chat(txt) {
9      document.getElementById("chat").innerHTML += txt;
10   }
11   </script>
12
13 </head>
14 <body>
15
16   <div id="chat"></div>
17
18   <?php
19   for($i=0; $i<10; $i++) {
20
21  // Datenbankabfrage hier einbauen!!!
22
23   echo '<script type="text/javascript">';
24   echo 'chat("<p>server is still alive at '.date('Y-m-d H:i:s').'</p>");';
25   echo '</script>';
26
27 // sende die Ausgabe zum Browser
28   flush();
29
30 // warte 0.2 Sekunden um den Server zu entlasten
31   usleep(200000);
32
33 }  // Ende der Schleife
34
35 ?>
36
37
38 <script type="text/javascript">
39   chat("<p>server is done</p>");
40 </script>
41
42 </body>
43 </html>
```

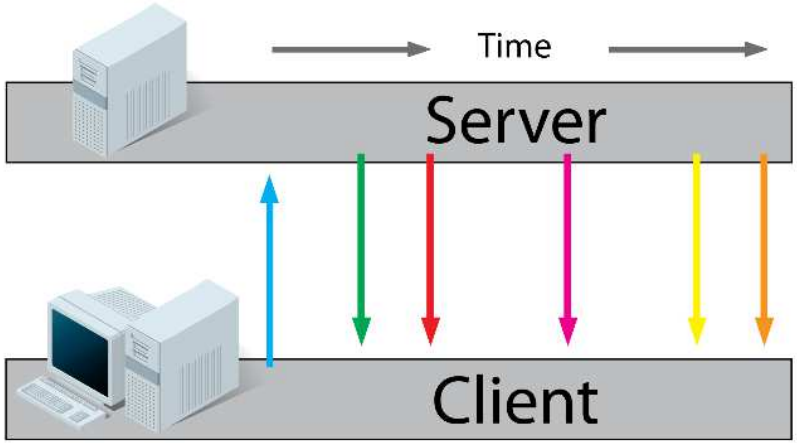
- In der Zeile #21 kann dann die Datenbankabfrage eingebaut werden.

Einführung in die Echtzeit-Kartographie

Datenübermittlung - COMET (PUSH) – Ein zweites Beispiel – Comet & AJAX

Einführung in die Echtzeit-Kartographie

Datenübermittlung - Web Socket API



1. Ein Client macht einen Request an einen Server (ähnlich normalem HTTP-Request).
2. Die anfragende Webseite führt Javascript aus, welches eine Verbindung zum Server aufbaut.
3. Der Server also auch der Client können nun Daten senden, sobald neue Daten zur Verfügung stehen
   → Real-Time Traffic vom Server zum Client möglich (wird meistens benötigt).
   → Man verwendet einen Server, der eine Ereignisschleife verwendet.
   → Mit WebSockets ist es möglich auf einen Server einer anderen Domain zu zugreifen.
   → Es ist außerdem möglich WebSocketserver von 3. Herstellern zu verwenden, z.B. Pusher oder andere. Es ist somit nur notwendig die Clientseite zu implementieren, was sehr einfach ist.
   → http://code.tutsplus.com/tutorials/start-using-html5-websockets-today--net-13270

Einführung in die Echtzeit-Kartographie

Datenübermittlung - Web Socket API – Erste Schritte

- Öffnen einer WebSocket-Verbindung:

```
var connection = new WebSocket
(´ws://html5rocks.websocket.org/echo´, [´soap´, ´xmpp´]);
```

- Nachdem die Verbindung geöffnet ist, können Daten an den Server gesendet werden:

```
connection.onopen = function {
  connection.send(‚Ping‘); // Sendet die Nachricht „Ping“ an den Server
};
```

- Fehler behandeln:

```
connection.onerror 0 function (error) {
  console.log(´WebSocket Error ´ + error);
};
```

- Meldungen vom Server behandeln:

```
connection.onmessage = function (e) {
  console.log(´Server: ´ + e.data);
};
```

Einführung in die Echtzeit-Kartographie

Datenübermittlung - Web Socket API – Erste Schritte

- Mit dem Server kommunizieren:

```
// Senden eines Strings
connection.send(´meine Mitteilung´);

// Senden von canvas Bilderdaten als ArrayBuffer
var img = canvas_context.getImageData(0, 0, 400, 320);
var binary = new Unit8Array(img.data.length);
for (var i = 0; i < img.data.length; i++) {
  binary[i] = img.data[i];
}
connection.send(binary.buffer);

// Senden eines Files mittels Blob
var file = document.querySelector(ínput [type=„file“]´).files[0];
connection.send(file);

//Meldungen vom Server behandeln:
connection.onmessage = function (e) {
  console.log(´Server: ´ + e.data);
};
```

Einführung in die Echtzeit-Kartographie

Datenübermittlung - Web Socket API – Erste Schritte

- Binär-Frames können im Blob- oder ArrayBuffer-Format empfangen werden. BinaryType muss entweder auf „blob“ oder „arrayBuffer“ gesetzt werden (Standard ist Blob).

```
// Setzten von binaryType um die erhaltenen Binärdaten als ´blob´ oder ´arraybuffer´ zu akzeptieren.
Connection.binaryType = ´arraybuffer´;
connection.onmessage = function(e) {
  console.log(e.data.byteLength); // ArrayBuffer Objekt falls binär
};
```

- WebSocket besitzen Erweiterungen, damit können Frames z.B. komprimiert werden.

```
// Bestimmung akzeptierter Erweiterungen
console.log(connection.extensions);
```

Einführung in die Echtzeit-Kartographie

Datenübermittlung - Web Socket API – Erste Schritte

- Erstes vollständiges Websocket-Programm websocket1.php

```
1. //window.alert("websSocket-Verbindung mit ws://echo.websocket.org");
2.
3. // WebSocket-Objekt wird instanziiert
4.  var ws = new WebSocket('ws://echo.websocket.org/');

5.  ws.onopen = function() {
6.    console.log('WebSocket-Verbindung aufgebaut.');
7.    ws.send ('Hi WebSocket Endpoint!');
8.    console.log('Uebertragene Nachricht: Hallo WebSocket Endpoint!');
9. };
10.
11.  ws.onmessage = function (message) {
12.    console.log ('Der Server sagt: ' + message.data);
13.    console.log(ws.extensions);
14.    meldung = message.data;
15.    window.alert(meldung);
16.    ws.close();
17.    };
18.
19.  ws.onclose = function(event) {
20.    console.log('Der WebSocket wurde geschlossen oder nicht aufgebaut .');
21.  };
22.
23.  ws.onerror = function(event) {
24.    console.log('Mit dem WebSocket ist etwas schief gelaufen!');
25.    console.log('Fehlermeldung: ' + event.reason + '(' + event.code + ')');
26.  };
```

Einführung in die Echtzeit-Kartographie

Datenaustauschformate – JSON (JavaScript Object Notation)

- JSON (ausgesprochen wie der amerikanische Name „Jason“) – Alternatives Austauschformat zu XML
- JOSN stellt selbst gültigen Javascript-Code dar.
- Zugriff auf einzelne Eigenschaften kann über normalen Attributzugriff erfolgen.
- Es lassen sich nahezu beliebige Datenstrukturen darstellen (Großer Vorteil!!!)

Objektart	Format	Beispiel
String	Text innerhalb von Anführungszeichen	„text“
Zahlenwert	Folge von 0 bis 9, optional mit Dezimalpunkt und/oder Exponent	1.6
Boolscher Wert	True oder false	True
Liste/Array	Aufgelistete Werte innerhalb von eckigen Klammern durch Komma getrennt	[wert1, wert2, …]
Objekt/“assoziativer Array“	Index/Wert-Zuordnung mithilfe eines Doppelpunktes innerhalb von geschweiften Klammern, durch Komma getrennt.	{„attname1“: wert1, „attname2“:wert2, …}

Einführung in die Echtzeit-Kartographie

Datenaustauschformate – JSON vs XML

- JSON-Format
- ```
  {"employees":[
     {"firstName":"John", "lastName":"Doe"},
     {"firstName":"Anna", "lastName":"Smith"},
     {"firstName":"Peter", "lastName":"Jones"}
  ]}
  ```

- XML-Format
- ```
  <employees>
     <employee>
        <firstName>John</firstName> <lastName>Doe</lastName>
     </employee>
     <employee>
        <firstName>Anna</firstName> <lastName>Smith</lastName>
     </employee>
     <employee>
        <firstName>Peter</firstName> <lastName>Jones</lastName>
     </employee>
  </employees>
  ```

Einführung in die Echtzeit-Kartographie

Datenaustauschformate – JSON (JavaScript Object Notation) - HTML

- JSON-Beispiel in HTML

```
{
        „json_ist_toll“: true,
        „namen“: [
                „Fabian“,
                „Stefan“,
                „Simon“,
                „Joerg“,
                „Bettina“,
                „Uwe“,
                „Mark“,
                „Franz“,
                „Heinz“
        ],
        „nummer“: 7
}
```

- Wird JSON-Code dekodiert, erhält man ein Objekt mit den Attributen „json_ist_toll“ (boolscher Wert: true), „namen“ (mit einer Liste von Namen) und „nummer“ (Zahlenwert: 7).

Einführung in die Echtzeit-Kartographie

Datenaustauschformate – JSON (JavaScript Object Notation) - Javascript

- Einfaches Umwandeln der Daten in ein Javascript-Objekt:

```
var jsonData = ´{„foo“: „bar“}´;                    // JSON-Daten
var jsonVar = eval(´(´ + jsonData + ´)´);           // Umwandeln in ein Javascript-Objekt
```

- Immer ein Framework (z.B. von json.org) einsetzen, dass Javascript-Objekte aus den JSON-Daten erstellt. Grund: Ausführen von gefährlichem Javascriptcode.

- JSON-Daten können mittels Framework durch JSON.parse(jsonData) geladen und mithilfe von JSON.stringify(jsonObject) dekodiert werden.

Einführung in die Echtzeit-Kartographie

Datenaustauschformate – JSON (JavaScript Object Notation) - Javascript

- **Script, das direkt JSON-Daten von einem Server lädt und verarbeitet**

```
$.getJSON('./js/daten2.json', function(data) {
        var output="<ul>";

        for (var i in data.users) {
            output+="<li>" + data.users[i].firstName + " " + data.users[i].lastName + "--" + data.users[i].joined.month+"</li>";

        }

        output+="</ul>";
        document.getElementById("placeholder").innerHTML=output;
                    for (ii=0; ii<=1; ii++){
                      alert(ii + ' ' + data.users[ii].firstName + ' ' + data.users[ii].lastName);
                    }
  });
```

Einführung in die Echtzeit-Kartographie

Datenaustauschformate – JSON (JavaScript Object Notation) - PHP

- Seit PHP 5.2.0 ist das JSON-Format ein fester Bestandteil.
- JSON-Daten mit PHP verarbeiten: Ausgabe des kompletten assoziativen Arrays

```
PHP
<?php
$namen = array(
       ´Fabian´          => 20,
       ´Stefan´          => 65,
       ´Simon´           => 18,
       ´Joerg´           => 60,
       ´Bettina´         => 12,
       ´Uwe´             => 18,
       ´Mark´            => 15,
       ´Franz´           => 65,
       ´Heinz´           => 90
);
echo json_encode($namen);
?>
```

- Ausgabe des Wertes vom Schlüssel „Fabian“:

```
echo json_encode($namen["Fabian"]);
```

Einführung in die Echtzeit-Kartographie

Datenaustauschformate – JSON (JavaScript Object Notation) - PHP

- Mit json_encode() können alle Datentypen nach JSON konvertiert werden (vorausgesetzt UTF-8 codiert).
- Klassische Arrays (mit aufsteigendem numerischem Index), werden in JSON-Arrays (mit eckigen Klammern) umgewandelt.
- Assoziative Arrays werden in Objekte umgewandelt.
- Beispiel:
  json_encode(mixed $value, int $option = 0);

  json_enode(array(„apples“ => true, „bananas“ => null));
  //Ergibt: {„apples“:true, „bananas“:null}

Einführung in die Echtzeit-Kartographie

Datenaustauschformate – JSON (JavaScript Object Notation) - PHP

- Übertragen von Objekten mittels JSON. Beispiel:

```
firstname = „Thomas“;
lastname = „Müller“;

json_encode($user);
//Ergibt: {„firstname“:“Thomas“, „lastname“:“Müller“}

$user->birthdate = new DateTime();

/* Ergibt:
{
„firstname“:“Thomas“,
„lastname“:“Müller“,
„birthdate“: {
   „date“:“2012-06-06 08:46:10“,
  „timezone_type“:3,
  „timezone“:“Europe/Berlin“
  }
}

json_encode($user);
?>
```

Einführung in die Echtzeit-Kartographie

Datenaustauschformate – JSON (JavaScript Object Notation) - PHP

- Zusammenfassung der JSON-Funktionen:
- json_decode Dekodiert eine JSON-Zeichenkette
  mixed json_decode (string $json [, bool $assoc = false [, int $depth = 512 [, int $options = 0]]])
  → json: Der zu decodierende String (nur UTF-8)
  → assoc: Wenn TRUE, werden zurückgegebene Objekte in assoziative Arrays konvertiert.
  → depth: Benutzerspezifische Verschachtelungstiefe.
  → options: Bitmaske mit JSON-Dekodieroptionen. Derzeit JSON_BIGINT_AS_STRING unterstützt (standardmäßig werden große Ganzzahlen in Fließkommazahlen umgewandelt.
- json_encode Gibt die Zeichenkette zurück, die die JSON-Darstellung von „value“ beinhaltet.
  string json_encode (mixed $value [, int $option = 0 [, int $depth = 512]])
  → value: Der zu kodierende „value“ kann von jedem Typ außer Ressource sein.
  → options: Bitmaske bestehend aus „JSON_HEX_QUOT, JSON_HEX_TAG, JSON_HEX_AMP etc.
  → depth: Setzt die maximal Tiefe. Muss größer sein als 0.
  Der Rückgabewert gibt einen JSON-kodierten String zurück. Im Fehlerfall wird FALSE zurückgegeben.
- json_last_error_msg Gibt die Fehlermeldung des letzten Aufrufs von json_encode oder json_decode zurück.
  String json_last_error_msg (void)
- json_last_error Gibt den letzten aufgetretenen Fehler (falls vorhanden) zurück.
  int json_lst_error (void)
  Gibt einen Integerwert zurück, der Wert kann eine der folgenden Konstanten sein: JSON_ERROR_NONE (Kein Fehler aufgetreten; JSON_ERROR_DEPTH (Die maximale Suchtiefe wurde überschritten); JSON_ERROR_STATE_MISMATCH (Ungültiges oder missgestaltetes JSON); JSON_ERROR_SYNTAX (Syntaxfehler)

Einführung in die Echtzeit-Kartographie

Datenaustauschformate – JSON (JavaScript Object Notation) – PHP-Beispiele

- PHP-Coding – JSON_DECODE

```
<?php
$json = '{"a":1,"b":2,"c":3,"d":4,"e":5}';

var_dump(json_decode($json));
var_dump(json_decode($json, true));

?>
```

- Das oben gezeigte Beispiel erzeugt folgende Ausgabe:

```
object(stdClass)#1 (5) {
    ["a"] => int(1)
    ["b"] => int(2)
    ["c"] => int(3)
    ["d"] => int(4)
    ["e"] => int(5)
}

array(5) {
    ["a"] => int(1)
    ["b"] => int(2)
    ["c"] => int(3)
    ["d"] => int(4)
    ["e"] => int(5)
}
```

Einführung in die Echtzeit-Kartographie

Datenaustauschformate – JSON (JavaScript Object Notation) – PHP-Beispiele

Einführung in die Echtzeit-Kartographie

Datenaustauschformate – JSON (JavaScript Object Notation)

- Zeigen der Beispiele:

- Speichern und Ausgeben einer Userliste: JSON1.PHP
- Speichern und Ausgeben von Googlemaps-Daten: JSON2.PHP
- Verarbeiten von JSON-Daten mit PHP: JSON3.PHP
- **Datenerhebung mit PHP und Ausgabe mit Javascript: JSON4.PHP**
- Lesen und Verarbeiten von JSON-Daten eines externen Servers: JSON5.PHP

Einführung in die Echtzeit-Kartographie

Datenaustauschformate – GeoJSON

- Format zur Kodierung von einer Vielzahl räumlicher Daten.
- GeoJSON baut auf dem JSON-Standard auf.
- Ein GeoJSON-Objekt stellt eine Geometrie, eine Eigenschaft oder eine Sammlung von Eigenschaften dar.
- www.geojson.org
- Beispiel:

```
{
       „type“:          „Feature“,
       „geometry“:              {
                          „type“ : „“Point“,
                          „coordinates“ : [125.6, 10.1]
                        },
       „properties“:    {
                          „name“ : „Dinagat Islands“
                        }
}
```

Einführung in die Echtzeit-Kartographie

Datenaustauschformate – GeoJSON

- Folgende Geometrytypen werden supported:
  → Points
  → LineString
  → Poygon
  → Multipoint
  → MultilineString
  → MultiPolygon
- Geometrische Objekte mit zusätzlichen Eigenschaften, nennt man „Feature Objects“.
- Sets von Features nennt man „FeatureCollection“ Objekte.
- Es wird das WGS84-Koordinatensystem benutzt.
- Geodaten Deutschland im Geojson-Format: http://opendatalab.de/projects/geojson-utilities
  → Statistik Geburten
  → Statistik Sterbefälle
  → Statistik Arbeitsmarkt der Bundesagentur für Arbeit
  → Lohn- und Einkommensteuerstatistik
  → Tourismuszahlen
  → Straßenverkehrsunfälle
  → Baufertigstellungen
  → Kassenergebnisse der Gemeinden
  → Baugenehmigungen
  → Wanderungsstatistik

Einführung in die Echtzeit-Kartographie

Datenaustauschformate – GTFS (Generalized Format Feed Specification)

- GTFS Datenaustauschformat, um komplette Fahrpläne abzubilden.
- GTFS wird im Gegensatz zum deutsch Standard VDV-452 bei vielen nützlichen & innovativen Projekten verwendet.
- Siehe http://www.swu.de/privatkunden/swu-nahverkehr/gtfs-daten.htm
- GTFS ist ein von Google entwickeltes Format: http://developers.google.com/transit/gtfs
- Einige Feedfiles:
  - → agency.txt Transitbehörden
  - → stops.txt Punkte an denen Passagiere ein- und aussteigen können
  - → routes.txt Transitrouten
  - → trips.txt Ein Trip sind 2 oder mehrere Stops auf einer Strecke
  - → stop_times.txt Zeiten an denen das Gefährt Punkte erreicht hat
  - → calendar.txt Datumsangaben für Service ID´sdes wöchentlichen Fahrplans
  - → calendar_dates.txt Ausnahmen für Service ID´s
  - → fare_attributes.txt Fahrpreisinformationen
  - → fare_rules.txt Regeln für die Anwendung des Fahrpreises
  - → frequencies.txt Headway (Zeiten zwischen den Trips)
  - → transfer.txt Regeln für Verbindungen zwischen Transferpunkten zwischen Routen
  - → feed_info.txt Zusätzliche Informationen

Einführung in die Echtzeit-Kartographie

Datenaustauschformate – GTFS Realtime

- Seit 2011 gibt es das GTFS-Realtime Format
- Variante des GTFS für Echtzeitdaten von öffentlichen Nahrsverkehrsnetzen
- Zur Zeit verwenden folgende Städte dieses Format:

  → Boston
  → Portland
  → San Diego
  → San Francisco
  → Madrid
  → Turin

- Spezifikation unter: http://code.google.com/transit/realtime
- Zur Zeit werden 3 Datentypen unterstützt:

  → Trip Updates (Verschiebungen im Fahrplan)
  → Service Alerts (Wartungs & Bauarbeiten)
  → Vehicle Position (genaue Position von Bussen und Bahnen)

Einführung in die Echtzeit-Kartographie

Datenaustauschformate – JSON (JavaScript Object Notation)

Wichtige Links

- Einen Googlemaps Key beantragen und herunterladen:
  https://developers.google.com/maps/documentation/javascript/get-api-key
- Googlemaps API´s aktivieren
  https://console.developers.google.com/project/black-octagon-101208/apiui/apis/library
- Javascript Tutoriell
  http://www.kostenlose-javascripts.de/tutorials/
- Javascript Online-Syntaxchecker
  http://esprima.org/demo/validate.html
- PHP Tutoriell 1
  file:///D:/Eigene%20Dateien/Doku/SELFPHP_5.6.4/index.html
- PHP Tutoriell 2
  http://www.php-einfach.de/php-tutorial/php-array.php