Javascript

Dynamic type-checking:
- Wert bestimmt Datentyp, nicht Variable
- Ermöglicht sehr flexibles Entwickeln

Weakly typed:
- Funktionen sind typ-ungebunden
- Objekte lassen sich dynamisch erweitern
- Truthy & Falsy Variablen

Node.js

Node.js ist eine JavaScript Laufzeit-Umgebung basierend auf der V8-Engine von Chrome, welche die Programme just-in-time, also zur Laufzeit, kompiliert.
In Node.js geschriebene Programme laufen single-threaded, was eine problemlose Verteilung über mehrere Systeme ermöglicht und dadurch Anwendungen einfach skalierbar macht.
Der Event-Loop, welcher die Single-Thread-Architektur von Node.js ermöglicht, delegiert Aufgaben an externe Daten-Lieferanten oder Rechenprozesse und greift erst bei entsprechendem Callback wieder. So können I/O-Aufgaben erledigt werden ohne den Haupt-Prozess aufzuhalten.

Asynchron vs Synchron

Zum Vergleich wird 100 mal eine 10 MB grosse Datei ausgelesen und ihr Anfang ausgegeben.

Synchron (Laufzeit: 0m0.668s):

```
var fs = require('fs');
for(var i=0; i < 100; i++){
	var content = fs.readFileSync("10.mb");
	console.log("File " + i + " Read √: "+content[0]);
}
```

Asynchron (Laufzeit: 0m0.366s):

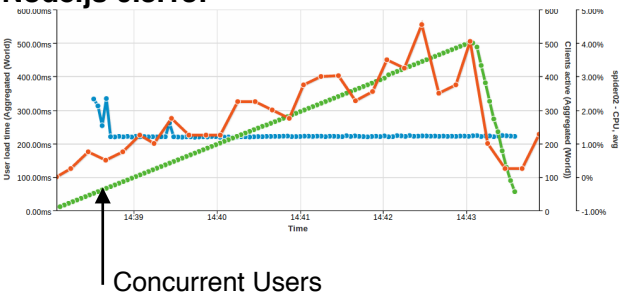
```
var fs = require('fs');
for(var i=0; i < 100; i++){
	(function(i){
		fs.readFile('10.mb', function(err, buf) {
			console.log("File " + i + " Read: "+buf[0]);
		})
	})(i);
}
```

Vergleich Node.js und PHP

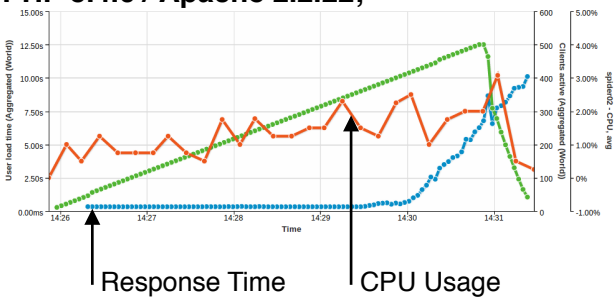
Quelle: http://blog.loadimpact.com/2013/02/01/node-js-vs-php-using-load-impact-to-visualize-node-js-efficency/

Getestet wurden 0-500 Anfragen, stetig erhöht über einen Zeitraum von 5 Minuten. Pro Anfrage wurden 50 Datensätze aus einer MySQL-Datenbank gelesen und zurückgegeben.

Node.js 0.8.18:



PHP 5.4.6 / Apache 2.2.22;



package.json

```
{
  "name": "Appoint",
  "description": "Web based appointment manager",
  "author": "Luis Ackermann <luis.ackermann@unibas.ch>",
  "dependencies": {
    "body-parser": "^1.9.3",
    "express": ">= 4.x.x",
    "moment": "^2.8.4",
    "mongoskin": "^1.4.4"
  }
}
```

appoint.js

```
var express = require('express'),
       bodyParser = require('body-parser'),
       moment = require('moment'),
       mongoskin = require('mongoskin');

var app = express();
var db = mongoskin.db('mongodb://@localhost:27017/appoint', {safe:true})

app.use(express.static('public'));
app.use(bodyParser.json());

var appointments = db.collection('appointment');

app.get('/appointments', function(req, res){
       var startDate = moment().startOf('week').toDate();
       var endDate = moment().add(2,'weeks').endOf('week').toDate();
       appointments.find({date: {$gte: startDate, $lt: endDate}} ,{sort: [['time',
1]]}).toArray(function(e, results){
              if (e) return next(e)
              res.send(results)
       })
});

app.post('/appointment', function(req, res){
       req.body.date = new Date(req.body.date);
       appointments.insert(req.body, {}, function(e, results){
              if (e) return next(e)
              res.send(results)
       })
});

app.put('/appointment/:id', function(req, res, next) {
       appointments.updateById(req.params.id, {$set:req.body}, {safe:true, multi:false},
function(e, result){
              if (e) return next(e)
              res.send(result?{message:'OK'}:{message:'ERR'})
       })
})

app.delete('/appointment/:id', function(req, res, next) {
       appointments.removeById(req.params.id, function(e, result){
              if (e) return next(e)
              res.send(result?{message:'OK'}:{message:'ERR'})
       })
})
app.listen(3000);
```