



Diese Wiki Seite sieht etwas verhaselt aus, da sie in erster Linie im Präsentationsmodus funktionieren muß.

Browser im Kiosk Modus: * Safari: keine Ahnung * Firefox recht gut per Plugin:

<https://addons.mozilla.org/de/firefox/addon/1568> * Opera von Haus aus = Parser mit Ragel

komplizierte Grammatiken und rattenschnelles XML

Vortrag am 22. September 2010

If you graph these numbers, patterns emerge. [Max Cohen, PI](#)

== Warum Parser bauen?

trotz [XML](#), [JSON](#), [YAML](#) etc.: * eigene maßgeschneiderte Protokolle, * 3rd Party Libs nicht vorhanden, * nicht gut genug oder * Lizenzprobleme. == Was spricht gegen Ad-Hoc Parsing (a.k.a. Gefrickel)? * der Parser sollte korrekt sein, * Wartbarkeit, * [BNF Grammatik](#) ist meist schon da, * fast immer schlechter als ein humorloser Zustandsautomat. == Quelltext Impression z.B. die [XML Spec](#) sagt:

```
document      ::= prolog element Misc*
prolog        ::= XMLDecl? Misc* (doctypedecl Misc*)?
XMLDecl       ::= '<?xml' VersionInfo EncodingDecl? SDDDecl? S? '?>'
VersionInfo   ::= S 'version' Eq ( '"' VersionNum '"' | "'" VersionNum "'" )
```

[Ragel](#) will:

```
VersionInfo    = S 'version' Eq ( '"' VersionNum '"' | "'" VersionNum "'" );
XMLDecl        = '<?xml' VersionInfo EncodingDecl? SDDDecl? S? '?>';
prolog         = XMLDecl? Misc*;
document       = prolog elem_query Misc*;
```

und natürlich noch ein paar Zaubersprüche rundrum.

== Wie komme ich drauf? * Experiment ob SVG ([Path Syntax!](#)) als Speicherformat für [piccolo2d.org](#), eine [Java 2D Scenegraph API](#) taugt. Dann könnten Designer die Screens direkt mit Illustrator oder Inkscape bauen. Ergebnis: evtl., * iPhone App soll ca. 500K einfaches XML schneller als [NSXMLParser](#) parsen (14 Sec.). Ergebnis: 0.3 Sec. == Anschlappen: Die Bausteine * die [Grammatik \(BNF-artig\)](#) * „Actions“ bzw. Callbacks in die eigene Logik * Treiber Code um das ganze loszutreten * [ragel Kommandozeile per Makefile](#)

- Zustandsautomat in C compilieren
- Zustandsautomat via [Graphviz](#) visualisieren

* Installation

```
sudo port install ragel
```

== Beispiel: SVG Path Parser

* [Grammatik beim W3C](#) * [Ragel Input](#) * [Ragel Output \(Objective C\)](#) * [Ragel Output \(Zustandsautomat PDF\)](#)

http://gist.github.com/579192#file_path_parser.rl

== Beispiel: XML Parser

* [Grammatik beim W3C](#) * [RelaxNG Schema](#) * [Ragel Input](#) * [Ragel Output \(Objective C\)](#) * [Ragel Output \(Zustandsautomat PDF\)](#)

== Ausblick: Zustandsautomaten * [UTU Hub State](#)



== Vielen Dank

für Eure Aufmerksamkeit.



Feedback willkommen an [Marcus Rohrmoser](#)

Die Folien zum Nachlesen gibt's hier:

http://mro.name/go/cocoaheads_ragel



[CocoaHeads](#), [Ragel](#), [Parser](#), [XCode](#)

From:

<https://wiki.mro.name/> - 

Permanent link:

<https://wiki.mro.name/cocoaheads/ragel?rev=1288173162>

Last update: **2010/10/27 11:52**

