



= Radio x Pi

 :=  +  +  +  +  +  + 

Radio Rekorder mit lua,ruby,lighttpd,cron,streamripper auf einem Raspberry Pi.

Vortrag am 14. Februar 2013

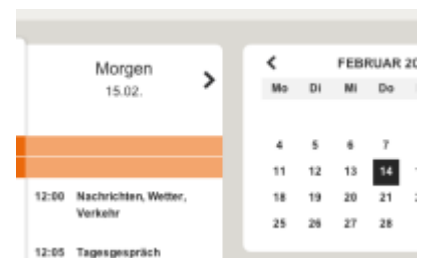
== Motivation: RSS Feeds meiner Lieblingssendungen



* Zündfunk * B2 Radiokrimi  * break beat action  == Amuse-Gueule

* Installation in 3min

== Scrape B2: Sendungsinfos



* BR Scraper (nokogiri/ruby) * Kalender * Sendezeiten * Sendungstexte + Bild * zum Schluß: meta nach stdout und broadcast-render.lua pipen!

 == Storage: keine DB, nur statische html Seiten

* pro Sendung eine html Datei, * Filesystem == Website == Datenbank * Dateinamen yyyy/mm/dd/HHMM abc.html * "meta" Tags angelehnt an Dublin Core PBMI



== Render + parse simple xhtml

* html ist selbst geschrieben, * also bekannt, * also einfach zu parsen == lua: wieso?

* extrem schneller Start der Runtime * extrem sparsame Runtime (RAM) * jedes Jahr eine neue

Sprache  == lua: wie sieht das aus?

* [lua BNF](#) * Beispiel: [broadcast-render.lua](#) * Beispiel: [broadcast_from_file](#) == lua: pattern matching

* [LPEG](#) == lua: html template engine

* [slt2](#) * Beispiel: [broadcast.slt2.html](#) * Beispiel: [broadcast_to_html](#)

== lua: war's das?

Bei weitem nicht:

* Metatables * OOP * Functional * C bridge. Beispiel: [msleep](#) == Publish: [lighttpd](#)

* cgi * redirects * http digest auth (mp3/[Privatkopie](#))



== Rip

* streamripper * taglib



== Vielen Dank

für Eure Aufmerksamkeit.



Feedback willkommen an [Marcus Rohrmoser](#)

Die Folien zum Nachlesen gibt's hier:



oder <http://wiki.mro.name/cocoaheads/>

[Vortrag](#), [CocoaHeads](#), [Raspi](#), [Radio](#), [Recorder](#)

From:

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

Permanent link:

https://wiki.mro.name/cocoaheads/radio_pi?rev=1360861804

Last update: **2013/02/14 18:10**

