



Praktikum zu LV 7328 - Ruby: **Übung 08**

Projektarbeit, Teil 2
Ausgabeformatierung,
Umgang mit Datenbäumen



Organisatorisches



- Arbeitsverzeichnis:
`~/lv/ruby/08/`
- Dateinamen:
`08-registry.rb` # Von 07-registry.rb kopieren,
erweitern & abgeben
- Werkzeuge:
`ruby` # Der Interpreter
`emacs` # mit Ruby-Mode. Auch X-Emacs ok
`scite, irb, ri` # Optionale Tools, wie üblich
mit Ruby-Mode
`komodo` # IDE für Ruby u.a. Skriptsprachen
- Vorlagen:
`08-registry.rb` # Nur bei Bedarf (Dozenten fragen)



- Ablauf
 - Aufgabe 08 ist der zweite Teil einer "Projektarbeit"
 - Eine weitere Aufgabe könnte noch folgen.
 - Quereinstieg ist möglich, da unabhängige Ziele verfolgt werden und jeweils Zwischenlösungen der früheren Aufgaben zur Verfügung stehen werden.
 - Derzeitiger Planungsstand:
 - 07: Scanner, mit regulären Ausdrücken
 - 08: Versch. Scripting, u.a. Datenbäume
 - xx: Ein GUI-Browser für den Datenbaum
- Gegenstand / Ziel des Projekts:
 - Export/Import-Dateien der Windows Registry einlesen und in ein Datenmodell übertragen.
- Hinweis zur Abgabe:
 - Es gibt 2 Punkte für P08. Abgabe erst am 5.1.07, gemeinsam mit P09



Hintergrund, Details



- Siehe gleichnamige Abschnitte der Aufgabe 07.



- Vorlage kopieren
 - Sie können mit Ihrer Datei "07-registry.rb" weiterarbeiten, indem Sie diese unter "08-registry.rb" in den neuen Ordner kopieren.
 - Falls Sie keine befriedigende Lösung bei Aufgabe 07 erreichten, steht Ihnen eine Vorlage unter "08-registry.rb" im Dozentenverzeichnis zur Verfügung.
 - Ihre Aufgabe besteht i.w. in der Erweiterung dieser Datei.
- Eingabedaten
 - Für Ihre Tests verwenden Sie dieselben Dateien "klein.reg" und "alles.reg.bz2" wie in Aufgabe 07 bzw. deren mit `iconv`, `dos2unix` und `bunzip2` bzw. `bzcat` angepassten Versionen "klein-utf8.reg" und "alles-utf8.reg.bz2".
 - Erneut gilt: Testen Sie erst mit der großen Datei, wenn alles mit der kleinen gelingt!



A: Ausgabe der eingelesenen Daten

- Implementieren Sie Methoden `to_s` für die Klassen
`RegType::Hex`, `RegType::DWord`, `RegType::String`,
`RegNode`
- Geben Sie am Programmende jeden Knoten dann einfach mittels "puts" aus (welches implizit `RegNode#to_s` verwendet), indem Sie über jeden Knoten von "root" iterieren - analog zum Zählen der Knoten zuvor.
- Testen Sie das Ergebnis:

```
$ 08-registry klein-utf8.reg > tmp1  
$ diff tmp1 klein-utf8.reg | more
```
- Die Daten sollten inhaltlich übereinstimmen.
- Vereinfachung: Die Ausgabe langer Hex-Ketten in einer Zeile wird akzeptiert.



B (*, opt.): Ausgabe der eingelesenen Daten, Forts.

- Erreichen Sie nun exakte Übereinstimmung bei "diff"
- Ausnahme:
Die Kopfzeilen "Windows Registry Editor ..." + Leerzeile dürfen fehlen.

C (*, opt.): Große Testdatei

- 1) Gelingt der vorher/nachher-Vergleich auch hier?
- 2) Bestimmen Sie die Laufzeiten
 - für den scan-Teil und
 - für den Ausgabeteil des Programms
bei Anwendung auf "alles-utf8.reg"



D: Entwickeln Sie die Methode "RegNode#add" weiter

- Sie soll nun einen echten Datenbaum erzeugen, der die Struktur der Pfade widerspiegelt.
 - Optional: Optimieren Sie Ihren "add"-Algorithmus, sodass er "alles-rtf8.reg" innerhalb weniger Minuten verarbeitet.
- Hinweise
 - Generell: Sie können die vorgegebenen Klassen auch erweitern und umgestalten, wenn Ihr Lösungsweg dies erfordert - die Vorlage ist als Hilfe gedacht, nicht als "Beengung".
 - Zu A: Beim Rekonstruieren des Hex-Formats haben sich String#unpack und Array#collect bewährt!
 - Zu D: Ergänzen Sie bei Bedarf weitere Methoden zu RegNode, wenn dadurch "add" einfacher zu implementieren ist.



Anhang: Registry-Ausschnitt + Teilbaum



[HKEY_LOCAL_MACHINE\SYSTEM]

[HKEY_LOCAL_MACHINE\SYSTEM\ControlSet001]

[HKEY_LOCAL_MACHINE\SYSTEM\ControlSet001\Control]

"CurrentUser"="USERNAME"

"WaitToKillServiceTimeout"="20000"

"SystemStartOptions"="FASTDETECT NOEXECUTE=OPTIN"

"SystemBootDevice"="multi(0)disk(0)rdisk(0)partition(1)"

[HKEY_LOCAL_MACHINE\SYSTEM\ControlSet001\Control\AGP]

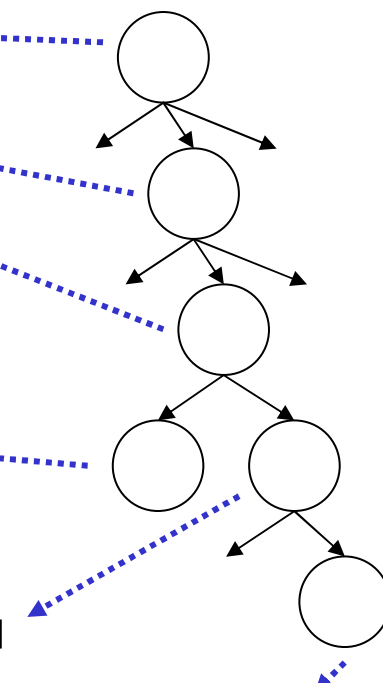
"102B0520"=hex:80,00,00,00,00,00,00,00

"102B0521"=hex:80,00,00,00,00,00,00,00

[HKEY_LOCAL_MACHINE\SYSTEM\ControlSet001\Control\Arbiters]

[HKEY_LOCAL_MACHINE\SYSTEM\ControlSet001\Control\Arbiters\AllocationOrder]

"Pci"=hex(a):48,02,00,00,00,00,00,00,00,00,00,00,00,00,00,00,00,00,00,00,00,00,00,00



Ihr Datenbaum soll wie hier am Ausschnitt gezeigt die logische Pfadstruktur der Registry-Einträge nachbilden. Die Logik dazu ist in die Methode `Regnode::add` einzubauen.