



7437 - EDI und E-Business Standards

Praktikumsaufgabe 01:
Programmierung kleiner
„Organisationshilfen“ rund um die EAN



Die Aufgabe im größeren Kontext



- Kontext
 - Als Handelspartner (Lieferant, Händler) müssen Sie GLNs, GTINs und NVEs vergeben und/oder überprüfen.
 - Bei der Erzeugung von EDI-Daten werden an verschiedenen Stellen Seriennummern benötigt, die jeweils nur einmal vergeben werden dürfen.
- Zweck der Übung
 - Schaffung einfacher Werkzeuge mit Kommandozeilen-Interface zur Lösung der o.g. Aufgaben
 - Grundlage für spätere Praktikumsaufgaben



ean_util: GTIN, GLN, NVE prüfen und vergeben



- Schreiben Sie ein Kommandozeilen-Programm „**ean_util**“ mit zwei Betriebsarten:
 - Prüfsummenziffer nachrechnen
 - `% ean_util string`
 - RC = 0 falls Prüfsumme ok, -1 falls nicht
 - Meldung "Pruefsummenfehler" (RC = 1) oder "Argument unzuessaessig" (RC = -1) nach stderr im Fehlerfall, etwa wenn *string* nicht nur Ziffern enthält
 - Prüfsummenziffer ermitteln, ggf. links mit Nullen auffüllen
 - `% ean_util -l n [string]`
 - RC = 0 falls ok, -1 sonst (z.B. wenn *string* zu lang oder unzulässig)
 - Auf stdout: *string* um Prüfziffer ergänzt und ggf. links auf *n* Stellen mit Nullen aufgefüllt. Typische Werte für *n*: 13 (GTIN), 18 (NVE)
 - Pipe mode*: Eingabe von stdin lesen, falls nicht als Argument übergeben



numgen: Einfacher Nummerngenerator



- Nummerngenerator:
 - Um NVE o.ä. lückenlos und eindeutig vergeben zu können, benötigen wir einen Nummerngenerator. Entwickeln Sie „numgen“ wie folgt:
 - `% numgen [nrkreis]`
 - --> RC = 0 falls ok, -1 falls *nrkreis* nicht existiert
 - --> Rückgabe-String als Zeile auf stdout (natürliche Zahl)
 - Persistenz
 - Führen Sie eine Datei `~/numgen` ein
 - Darin befinden sich *key/value*-Paare (je eins pro Zeile) der Art `nrkreis = naechster_wert`
 - Jeder Aufruf von "numgen" liefert den zu *nrkreis* hinterlegten Wert und inkrementiert diesen in der Datei.
 - Locking
 - Damit "numgen" parallel von mehreren Anwendungen benutzt werden kann, sollte ein *file locking* auf `~/numgen` zum Einsatz kommen!
 - Initialisierung
 - `~/numgen` enthalte anfangs die Zeile "nve = 1" (nve = Default für *nrkreis*)



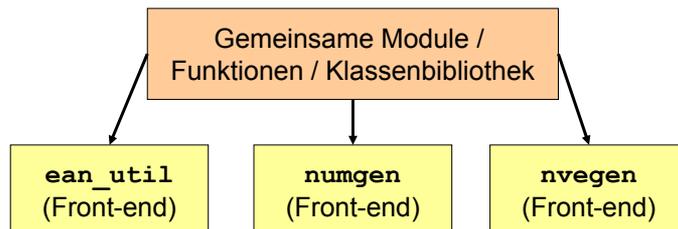
- NVE-Generator:
 - Schreiben Sie ein Programm "nvegen", das pro Aufruf eine neue NVE liefert:

```
% nvegen GLN
```

 - RC = 0 falls ok, -1 bei internen Fehlern (z.B. numgen nicht vorhanden)
 - Rückgabe-String als Zeile auf stdout (NVE)
 - Vorgehen:
 - Erstellen & testen Sie zunächst „numgen“ und „ean_util“ !
 - „nvegen“ lässt sich leicht durch Kombination dieser beiden Werkzeuge wie folgt erzeugen:
 - Rufen Sie einfach „numgen“ für die nächste Seriennummer auf
 - Konkatenieren Sie diese Nummer, die benötigten Anteile der übergebenen GLN sowie das NVE-Präfix in der richtigen Reihenfolge
 - Rufen Sie damit „ean_util -l 18“ auf, um die noch fehlende Prüfziffer zu ergänzen.



- Festlegung der Implementierungssprache
 - Skriptsprachen wie Perl, Ruby, Python sind für diese Aufgaben ideal.
 - Empfehlung: **Ruby** wählen
 - Spätere EDI-Bibliothek gibt es nur für Ruby
 - Modern, konsequent objekt-orientiert, in Fernost weit verbreitet
- Modulares Vorgehen empfohlen:





- Abgabefrist
 - Abgabeschluss ist **Donnerstag, der 28.4.05 12:00 Uhr**
- Was abgeben?
 - **ean_util***, **numgen***, **nvegen***, **eanlib*** (z.B.: ean_util.c, numgen.rb)
- Bedingungen
 - Teambildung bereits empfohlen, jedoch:
 - Durchführung (noch) pro Person – nicht: pro Gruppe!
 - Abgabe daher auch: pro Person