Zusammenfassung Tag 16

Worum geht es in diesem Abschnitt

- **Internationalisierung**
  - Programm so designt das es mit mehreren Sprachen umgehen kann
  - Häufig als i18n bezeichnet
- **Lokalisierung**
  - Übersetzungsdateien / Programm in andere Sprache testen
  - Häufig als l10n

- **Locales (LC_ALL, LC_*,…)**
  - Dateien, hinterlegt z.B. abkürzung Wochentage, Trennung Zahlen, usw…
- **Zeitzone des Systems unter Linux**
  - Benutzer können unterschiedliche Zeitzonen haben
- **Kodierungen von Dateien**
  - UTF-8, ASCII, Unicode, ISO-8859
  - Ist wichtig unter anderem für Umlaute

Locales und LC_ALL

- ***date***
  - gibt das Datum aus
- ***LC_ALL="en_US.utf-8" date***
  - gibt das Datum mit dem Local en_US.utf-8 aus
  - Zuerst die Sprache, In Großbuchstaben das Land, zuletzt Kodierungen
- **Unter Ubuntu sind nicht alle Locals vorhanden.**
  Es müssen einige aktiviert werden um sie zu benutzen
  - Unter CentOS sind alle locals bereits aktiviert
- **/usr/share/i18n/SUPPORTED**
  - Alle locals hinterlegt die generiert werden können
- ***sudo locale-gen "bg_BG.UTF-8"***
  - generiert das local bg_BG mit der Codierung UTF-8
  - danach *sudo update-locale* zum Aktualisieren der locals

LC_Time, LC_NUMERIC, date und printf

- ***env | grep "LC"***
  - gibt alle Umgebungsvariablen mit dem namen LC aus
- ***export LC_Time="bg_BG.utf-8"***
  - überschreibt die Umgebungsvariable für LC_Time
  - Datum wird dann in der Bulgarischen Schreibweise ausgegeben

- ***printf***
  - Befehl zum ausgeben
  - selbiger Befehl wie bei c (selbe Parameter)
- ***printf "%.2f" 100000***
  - gibt die Zahl mit 2 Nachkommastellen aus
- ***printf "%'.2f" 10000***
  - gibt folgendes aus 100.00,00
- ***LC_ALL="en_US.utf-8" printf "%'.2f" 100000***
  - gibt folgendes in US Schreibweise aus:
    100,00.00
  - Umgebungsvariable LC_ALL wird für den Befehl auf en_US.utf-8 gesetzt
- **LC_NUMERIC**
  - Umgebungsvariable für Formatierung von Zahlen
- ***export LC_NUMERIC="en_US.utf-8"***
  - setzt die Umgebungsvariable dauerhaft auf en_US.utf-8

Wie funktionieren Locales intern

- **/etc/locale.gen**
  - liste mit verfügbaren locals
  - durch auskommentieren lassen sich die locals aktivieren
    (nicht empfehlenswert, besser übers tool aktivieren)
- ***locale -ck LC_TIME***
  - gibt die Konfiguration für LC_TIME aus
- ***locale -ck LC_NUMERIC***
  - gibt die Konfiguration für LC_NUMERIC aus

Wie werden Locale standardmässig gesetzt (Ubuntu)

- **Datei für Umgebungsvariable / locals**
  - /etc/enviroment
  - /etc/locale
  - /etc/default/locale

- ***sudo update-locale LC_TIME="bg_BG.utf-8"***
  - aktualisiert die Umgebungsvariable von LC_TIME
  - wird dann in der entsprechenden Datei geändert
- **~/.pam_enviroments**
  - pam ()
  - Datei mit Variablen für Benutzer
  - überschreibt ggf locals
  - *nano ~/.pam_enviroments*
    - zum editieren der Datei
  - Wird von der Gui benutzt
- **~/.bashrc**
  - *nano ~/.bashrc*
  - *export LC_TIME="bg_BG.utf-8"*
  - setzt die Variable
  - nur für die Shell

Wie werden Locale standardmäßig gesetzt (CentOS)

- ***LC_TIME="en_US.utf-8" date***
  - setzt die LC_TIME temporär auf en_US
- ***localectl status***
  - gibt die system locale aus
- ***localectl set-locale LANG=de_DE.utf8 LC_TIME=bg_BG.utf8 LC_NUMERIC=en_US.utf8***
  - setzt die Umgebungsvariablen auf die entsprechende Werte
- **Es reicht die LANG variable zu setzten**
  - andere LC_ Variablen werden dieser angepasst
- **ggf Neustart notwendig**
- **/etc/locale.conf**
  - hier werden locals gespeichert
  - beim login wird /etc/profile.d/lang.sh ausgeführt
    dort wird locale.conf geladen
- **Für jeweiligen Benutzer locale Datei im Homeverzeichnis anlegen**
  - *nano ~/.i18n*
    - zum anlegen der Datei
  - dort LC_ Variable reinschreiben
  - *chmod +x ~/.i18n*

LANG, LOCALE, etc

- **$LANG**
  - setzt alle LC Variablen
  - ist eine LC Variable spezifisch gesetzt
    wird der Standard wert überschrieben
- **LC_ALL**
  - überschreibt alle LC und LANG Variablen
- ***yum langinstall fr_FR***
  - installiert die Übersetzung Französisch
- ***LANGUAGE="fr:de:en_US:en" man***
  - ruft das Programm man auf
  - falls Französisch vorhanden ist wird es ausgegeben
  - wenn Französisch nicht vorhanden ist wird es in Deutsch ausgegeben
  - wenn Deutsch nicht vorhanden ist wird es in Englisch ausgegeben

- Unter einer GUI können Sprachen mit dem Programm Sprachen installiert werden

Zeitzonen verändern (CentOS)

- ***timedatectl status***
  - gibt alle Informationen zur Zeit aus. (Zeitzonen usw)
- ***timedatectl list-timezones***
  - gibt eine liste aller Zeitzonen aus
- ***tzselect***
  - Programm zur einfacheren Auswahl von Zeitzonen
- ***timedatectl set-timezone Europe/Berlin***
  - ändert die Zeitzone auf Berlin
- **/usr/share/zoneinfo/**
  - hier liegen Informationen zu den Zeitzonen

Zeitzonen verändern (Ubuntu)

- **/usr/share/zoneinfo/**
  - hier liegen Information zu den Zeitzonen
- **/etc/timezone und /etc/localtime müssen angepasst werden**
- ***sudo dpkg-reconfigure tzdata***
  - Befehl zur neu Konfiguration von tzdata
  - einfaches Menü um Zeitzone auszuwählen

TZ und tzselect

- **Umgebungsvariable TZ**
  - hier kann eine Zeitzone für einen Benutzer reingeschrieben werden
- ***export TZ="Asia/Bangkok"***
  - setzt die Umgebungsvariable TZ

Kodierungen (Teil1) – ASCII, ISO-8859

- **Wird zur richtigen Darstellung von Zeichen benötigt**
- **ASCII**
  - Standard seit 17.Juni 1963
  - von der American Standards Association (ASA)
    heutzutage: ANSI
  - 1 Zeichen besteht aus 7 bit
    2^7 = 128 mögliche Zeichen
  - Folgende Zeichen sind enthalten

```
!"#$%&'()*+,-./0123456789:;<=>?
@ABCDEFGHIJKLMNOPQRSTUVWXYZ[\]^_
`abcdefghijklmnopqrstuvwxyz{|}~
```

  - besondere Zeichen wie z.B Umlaute können mit ASCII nicht dargestellt werden

- **ISO-8859-1 (LATIN-1)**
  - 1 Zeichen besteht aus 8 bit
    2^8 = 256 mögliche Zeichen
  - Unterstützung von ASCII Zeichen
    zusätzlich alle europäischen Schriftzeichen
    (ä, ü, ö, …)
- **ISO-8859-7**
  - ähnlich wie ISO-8859-1
  - andere Sonderzeichen
  - keine ä, ü , ö
    dafür Ω , usw…
- **Wir müssen wissen wie eine Datei codiert ist um diese auch richtig öffnen zu können**

Kodierungen (Teil2) – Unicode, UTF-8

- **Standard mit dem Ziel alle Schriftzeichen abzudecken**
- **Aufgeteilt in 17 ”Unicode-Ebenen“**
  - jede Ebene enthält bis zu 2^16 Schriftzeichen
    65.536 mögliche Schriftzeichen
- **Bisher 6 Ebenen belegt**
- **UNICODE, UTF-32**
  - Unicode braucht 32bit pro Zeichen
  - 2^32 verschiedene Schriftzeichen
  - 4 mal so viel Speicherplatz wird benötigt wie bei ISO-8859-1
- **UTF-8**
  - Schriftzeichen haben Variable länge
  - ASCII Zeichen weiterhin 1 Byte (8bit)
  - Besondere Zeichen verwenden mehr bits
    - Ü benötigt 2 Byte (16bit)
  - Einfache Schriftzeichen brauchen weniger Speicherplatz
  - Flexible Länge ermöglicht ausreichend Platz für neue Zeichen
  - Beim Berechnen der Länge eines Textes

kann nicht mehr die Dateigröße verwendet werden
Schriftzeichen sind ja unterschiedlich lang

Kodierungen umwandeln (ivonc)

- ***sudo apt-get install kate***
  - Editor mit Kodierungsauswahl
- **Unten rechts im Editor kann die Kodierung ausgewählt werden**

- ***iconv -f ISO-8859-1 -t UTF-8 umlaute.txt***
  - wandelte die Datei umlaute.txt von ISO-8859-1 in UTF-8
  - man muss die Kodierung die die Datei besitzt wissen
- ***iconv -f ISO-8859-1 -t UTF-8 umlaute.txt -o umlaute-utf8.txt***
  - wandelte die Datei umlaute.txt von ISO-8859-1 in UTF-8
    und speichert diese unter umlaute-utf8.txt ab.

Nützliche Befehle:

clear	Bereinigt die Konsole
strg+c	Beendet ein Programm / unterbricht einen Befehl
cat	Erzeugt eine Ausgabe z.B. von einer Datei
nano	Einfacher Editor zum bearbeiten von Dateien
commandname –help	Öffnet meistens die Hilfe eines Programm
man commandname	Öffnet das Manual eines Programm falls vorhanden