

Die Shell

- Die **Shell** ist ein einfaches Interface, um dem Betriebssystem Kommandos zu übermitteln
 - Andere Interfaces verwenden Grafische Elemente (z.B. Windows Explorer)
- Kommandos werden beim sogenannten **Prompt** eingetippt
 - der Prompt ist einfach der Beginn der Zeile
 - enthält oft Informationen wie z.B. das Working Directory, die Uhrzeit, etc.
 - ein Kommando besteht aus dem Namen, Parametern (flags) und Argumenten

```
[prompt]> <command> <flags> <args>
```

Files / Dateien

- Ein Computer speichert alle Informationen in Einheiten gespeichert, die man **Files** bzw. **Dateien** nennt.
- Files können sein
 - Programme
 - Texte
 - Web-Seiten
 - Bilder
 - Tonaufnahmen
 - Filme
 - etc.

File-Name und File-Typ

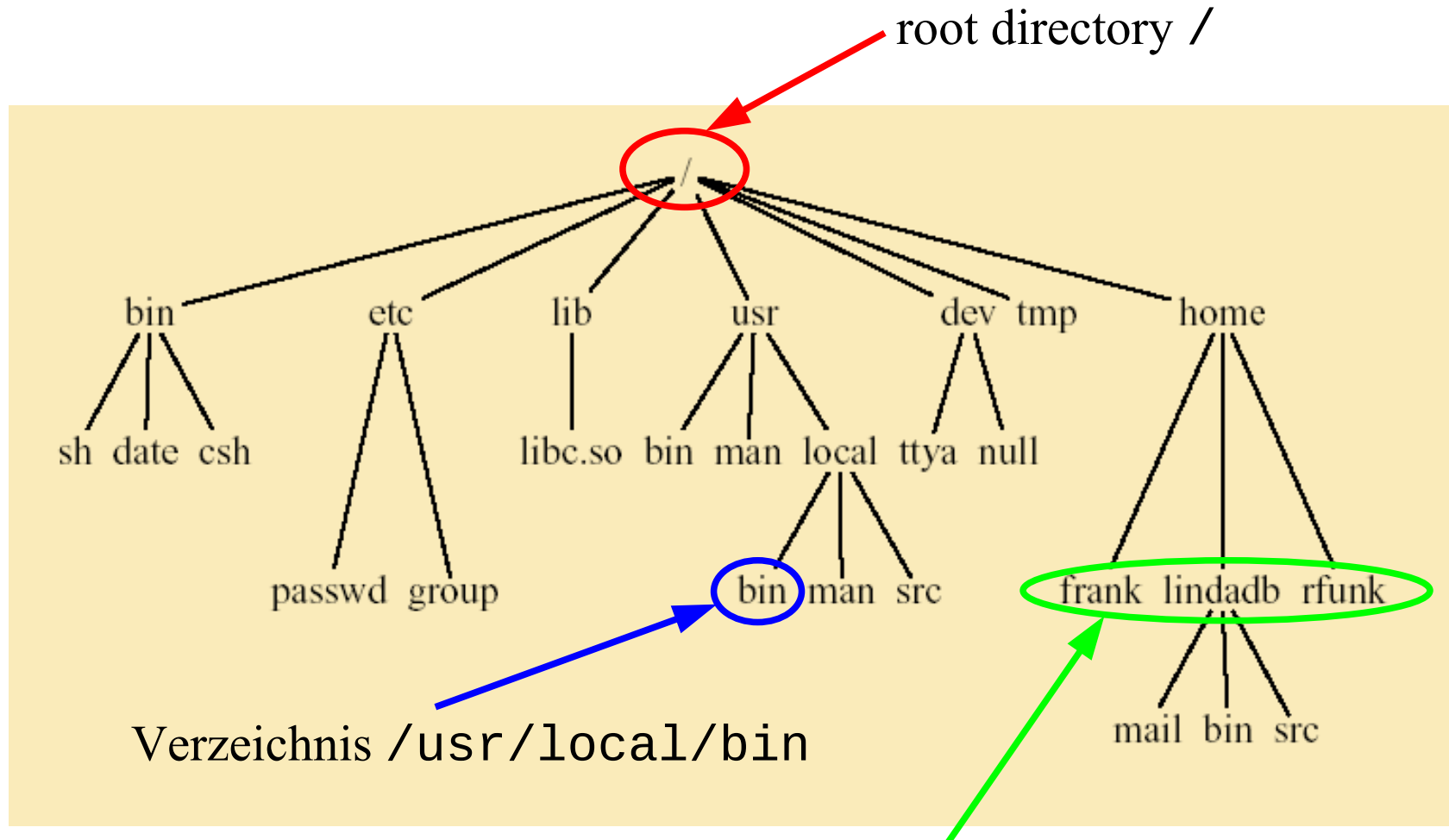
- Files werden unter einem Namen abgespeichert, und können mit diesem Namen später wieder abgerufen werden
- Ein File-Name besteht aus 2 Bestand-Teilen:
`name.extension`
- Der **Name** ist frei wählbar
 - üblicherweise ohne Leer- und Sonderzeichen
- Die **Extension** ist prinzipiell frei wählbar, gibt aber üblicherweise den Typ des Files an, z.B:
 - `einladung.txt`: Text-Datei
 - `Internationale.mp3`: Musik-Datei im MP3-Format
 - `KarelJ.java`: Java Programm
 - `Quadrat.task`: KarelJ task

Directories / Verzeichnis

Folder / Ordner

- Um bei vielen Dateien nicht den Überblick zu verlieren organisiert man sie in sogenannte **Directories** oder **Verzeichnisse** bzw. **Ordner** oder **Folder**
- Ein Directory enthält
 - ein oder mehrere Files
 - ein oder mehrere Directories→ Hierarchische Struktur
- Der **Pfad** gibt den Weg durch diese hierarchische Struktur an
 - Einzelne Komponenten werden üblicherweise durch / (Unix/Linux) oder \ (DOS/Windows) getrennt

Hierarchische Directories



Verzeichnis `/usr/local/bin`

Home-Directories `/home/name`
bzw. `~name`

Spezielle Verzeichnisse

- **Root-Directory** /
 - das oberste Directory
 - alle anderen Directories sind Unter-Directories des Roots
- **Home-Directory** ~
 - Ihr eigenes Directory
 - Ihre eigenen Dateien sollten im Home-Directory bzw. in von Ihnen angelegten Unter-directories Ihres Homes angelegt werden
- **Working Directory** .
 - Das Directory, in dem Sie sich gerade befinden
- **Über-Directory** ..
 - Das Directory über Ihnen

Navigation in Directories

- `cd`
 - Change *Directory* (or your home directory if unspecified)
 - Syntax: `cd <directory>`
 - Examples:
 - `cd Uebung1` (Unterverzeichnis des Working Dirs)
 - `cd ../Uebung1` (Unterverzeichnis des Über Dirs)
 - `cd ~` (Home-Directory)
 - `cd ~/Uebung1` (Unterverzeichnis des Home Dirs)
- `mkdir`
 - *MaKe DIRectory*
 - Syntax: `mkdir <directory>`
 - Example: `mkdir Uebung2`

Inhalt eines Directories ansehen

- `ls <Pfad>`
 - LiStet den Inhalt des angegebenen Pfads (oder des working directory wenn kein Pfad angegeben wird)
 - **Example:** `ls Uebung1`
- `pwd`
 - *Print Working Directory*
 - gibt den Pfad für das Directory aus, in dem Sie sich gerade befinden

Kopieren bzw. Verschieben

- cp
 - *CoPiert* ein File
 - das Original bleibt erhalten
 - **Syntax:** `cp <source> <destination>`
 - **Example:** `cp Quadrat.task Quadrat.task.bak`
- mv
 - *MoVes* oder verschiebt ein File
 - wie cp, aber das Original wird gelöscht!
 - **Syntax:** `mv <source> <destination>`
 - **Examples:**
 - `mv Quadrat.task Quadrat.task.bak`
 - `mv Quadrat.task Uebung1/`

Löschen

- `rm`
 - *ReMove*
 - **Syntax:** `rm [<options>] <files>`
 - **Example:** `rm Quadrat.task`
- `rmdir`
 - *ReMove DIRectory*
 - das Directory muß vorher geleert werden!
 - (mit `rm` bzw. `rmdir`)
 - **Syntax:** `rmdir <directory>`
 - **Example:** `rmdir Uebung1`