Schulungsportfolio

plausibolo®
alles plausibel erklärt

Das Schulungsportfolio von *plausibolo* umfasst viele Themen aus dem IT-Bereich mit einem Schwerpunkt auf Open Source.

Einsatz ist im Raum DACH und in NL und BE möglich, darüber hinaus nach Absprache. Alle Kurse können auf Deutsch und Englisch gehalten werden.

Holger Jakobs, Mülheimer Str. 133, 51469 Bergisch Gladbach
Telefon +49 2202 817157 oder +49 178 9759012

Web: http://plausibolo.de Mail: jakobs@plausibolo.de

Alle Kurse stellen lediglich Vorschläge dar und können nach Kundenwunsch bezüglich Inhalt und Umfang individualisiert werden. Bei den meisten Schulungsthemen ist Holger Jakobs selbst der Trainer, bei einigen werden hochqualifizierte Kollegen eingesetzt, selbstverständlich in Absprache mit Ihnen als Kunde.

Auch für individuelle Beratung bei der Auswahl oder dem Einsatz von Softwaresystemen steht das Team von plausibolo.de zur Verfügung.

Als Schulungsbegleitmaterial werden neben aktuellen Fachbüchern, die jeweils empfohlen werden, auch eigene Unterlagen eingesetzt. Während des Trainings kommen Projektion des Bildschirminhalts, Flipchart und – sofern vorhanden – auch Whiteboards zum Einsatz.

Sie werden zufrieden sein, wenn Sie auf die über 25-jährige Erfahrung in der IT-Bildung bauen. Ich freue mich auf Ihren Auftrag.

Inhaltsverzeichnis

Themenbereich Datenbanken

SQL

SQL ist die „lingua franca“ der relationalen Datenbanken, die für alle unternehmenskritischen Daten, die nach Integrität verlangen, nach wie vor die erste bzw. einzige Wahl sind. Allerdings hat diese „lingua franca“ doch so einige Dialekte, so dass es manchmal schwer fällt, von einem zum anderen Datenbanksystem zu wechseln.

Im Kurs werden die zugrunde liegenden Ideen von SQL erläutert, der ISO SQL Standard vorgestellt, und anhand vieler Beispiele wird gezeigt, wie die Dialekte der bedeutenden SQL-Datenbanksysteme sich zum Standard verhalten. Auch werden die einzelnen Datentypen mit ihren Einsatzfeldern und Möglichkeiten vorgestellt.

Ein Randthema ist die Abgrenzung der Einsatzfelder gegenüber NoSQL-Datenbanken, die ja durchaus ihre Existenzberechtigung haben.

Datenbankentwurf

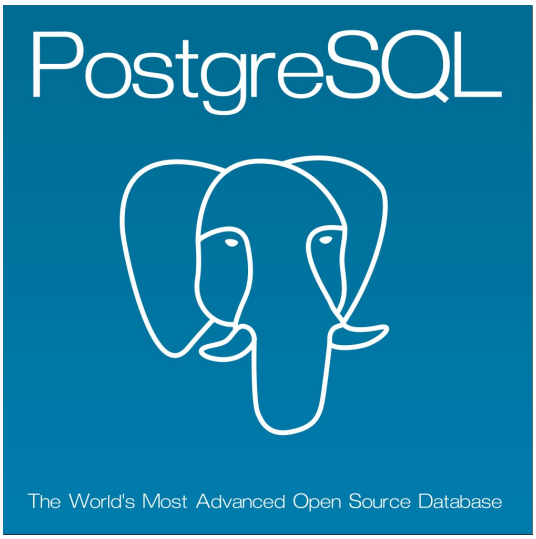
Einfach mal so ein paar Tabellen runterschreiben ist keine besonders gute Idee. Für den Datenbankentwurf muss man sich schon etwas Zeit nehmen und auch wissen, wie es richtig geht. Der Kurs erläutert das Entity-Relationship-Modell und wie aus einem solchen Modell die Kommandos zur Erzeugung der Tabellen abgeleitet werden. Die Tabellen werden gleich mit den notwendigen Constraints versehen, um die Integrität der Daten zu sichern.

Wie werden Tabellen normalisiert? Bis zu welchem Grad ist Normalisierung sinnvoll? Welche Performance-Vorteile kann eine kontrollierte Redundanz bringen? Wie sichert man die Integrität der Daten trotz gewisser Redundanz? Welche Eigenschaften muss ein Datenbanksystem mitbringen, um hier die gewünschte Leistung zu erbringen, ohne dass man gleich zu Replikation und anderen aufwendigen Mitteln greifen muss?

Auf Wunsch werden auch Werkzeuge wie pgModeler oder SQL Power Architect vorgestellt.

PostgreSQL

PostgreSQL ist eines der bedeutendsten Datenbanksysteme und wird von großen Unternehmen wie „Instagram“, der Wochenzeitung „Die Zeit“ und dem Versandhaus „Zalando“ sowie kleineren Unternehmen wie „Online-Druck.biz“ erfolgreich eingesetzt. Im Gegensatz zu bekannteren Systemen ist es ein wirklich freies, herstellerungebundenes System, sehr nah am ISO SQL Standard, dabei extrem leistungsfähig und flexibel. Auch jüngste Entwicklungen wie die Speicherung in XML- und JSON-Formaten wurden schon umgesetzt.



- **Einführung in PostgreSQL**
  Konzepte wie MVCC und ACID, Standardnähe und Abweichungen vom bzw. Ergänzungen zum Standard, Datentypen einschließlich Arrays, Klassen und Vererbung, Unterschiede zu anderen SQL-Datenbanken, besondere Vorteile von PostgreSQL, auch gegenüber NoSQL-Datenbanken.
- **Administration von PostgreSQL auf Linux-Systemen**
  Installation, Benutzerverwaltung, Optimierung, Indexierung, Replikation, Datensicherung, Updates.
- **Datenbankentwurf und Umsetzung**
  Entwurf eines Entity-Relationship-Modells, Übertragung des Modells in Tabellen, Optimierung der Tabellen, Erstellen der Datenbank in PostgreSQL, Sicherung der Plausibilität und Integrität der Daten, Erstellen von Views, Prozeduren und Trigger, Vergabe von Berechtigungen.
- **Anwendungsentwicklung mit PostgreSQL**
  Datenzugriff aus diversen Programmiersprachen heraus (nach Kundenwunsch), Transaktionsmanagement mit Isolation Levels und ihrer Anwendung, Verwendung von Savepoints, Maximierung der Parallelität von Abläufen.
- **Migration von anderen Datenbanksystemen**
  Wie überträgt man Datenstrukturen und Dateninhalte von einem Datenbanksystem zum anderen? Welche Unterschiede im SQL-Dialekt müssen hierbei beachtet werden? Wo lauern Fallen? Welche Verbesserungen können erwartet werden?

Oracle

Oracle gehört zu den Schwergewichten im Markt und stellt ein Datenbanksystem für große und größte Unternehmen her. Der Umgang mit Oracle gestaltet sich nicht immer einfach, denn obwohl Oracle Mitglied im Standardisierungsgremium ist, folgt Oracle an einer ganzen Reihe von Stellen nicht dem ISO-Standard.

Wie unterscheiden sich die Oracle-Datentypen vom ISO-Standard und was ergibt sich daraus für den Administrator und den Anwendungsentwickler? Was ist bei der Programmierung zu beachten? Welche Workarounds gibt es für die Begrenzungen, die Oracle enthält?

Themen zu Oracle siehe oben unter PostgreSQL (von Einführung bis Migration).

SQLite

SQLite ist zweifellos das Datenbanksystem mit der höchsten Anzahl von Installationen weltweit, denn es ist in jedem Firefox-Browser, in jedem iOS- und jedem Android-Smartphone vorhanden. Ursprünglich als Datenbank für das Tcl/Tk-System (siehe Seite 5) entwickelt, hat es sich rasant verbreitet wegen seiner geringen Größe, der Einbettbarkeit und guten Leistungsfähigkeit.

In welchen Anwendungsszenarios ist SQLite ein geeignetes Werkzeug? Wo sind die Grenzen? Kann ich mit SQLite auch eine Server-Datenbank ersetzen?

Im Kurs werden die zugrunde liegenden Konzepte von SQLite erläutert, die Unterschiede zu anderen SQL-Dialekten dargestellt und anhand praktischer Beispiele geeignete Werkzeuge vorgestellt. Der Zugriff auf SQLite kann aus fast allen Programmiersprachen heraus erfolgen.

Auf Wunsch wird die Verwendung von SQLite innerhalb von Android erläutert, denn Android hat eine ganz eigene Schnittstelle zu SQLite.

Themenbereich Android

Android ist das am weitesten verbreitete Betriebssystem für Smartphones, mit dem eine breite Palette von Geräten angeboten wird. Der Anwender hat die Wahl zwischen preiswerten und hochwertigen, eleganten und robusten, kleinen und großen Smartphones, aber auch Tablets in vielen Formaten, die von diversen Herstellern angeboten werden.

Die Grundprinzipien sind gleich geblieben, aber seit den Anfängen mit Android 1.6 hat sich viel getan, sowohl in technischer wie auch in optischer Hinsicht.

Android für Anwender

Wer zum ersten Mal ein Smartphone verwendet, ist von der Vielzahl der Einstellungen und Möglichkeiten überrascht und erschlagen. Welche Dienste sollte man verwenden? Wie integriert man vorhandene E-Mail-Accounts? Oder steigt man komplett auf GMail um? Wie funktioniert das mit den Terminen und der Kommunikation zum Desktop-Kalender und zum Tablet? Wie synchronisiert man Adressbücher und Aufgabenlisten? Wie trennt man Privates von Geschäftlichem? Wie mache ich eine Datensicherung? Wie verhindere ich unbefugten Zugriff? Brauche ich einen Virenscanner für mein Smartphone? Welche Apps sind wirklich empfehlenswert?

Anhand von konkreten Fragestellungen werden im Kurs die ersten Schritte in die Smartphone-Welt begleitet und Lösungen aufgezeigt. Nicht alle Android-Smartphones sind komplett gleich, weil die Hersteller teilweise das Standard-Android verändert oder erweitert haben. Auch unterscheiden sie die 2.x Versionen von den 4.x Versionen.

Android für Entwickler

Auch mit vorhandenen Java-Kenntnissen kommt man erst einmal nicht weit. Obwohl Android-Apps mit Java geschrieben werden, so muss man eine ganze Menge an neuen Konzepten verinnerlichen, bevor man eine App lauffähig hinbekommt. Im Kurs wird die Architektur von Android einschließlich des Zusammenspiels der Bestandteile erläutert. Anschließend wird ein Entwicklungssystem aufgesetzt, wahlweise mit Eclipse, welches noch als Standard gilt, oder mit dem Newcomer Android Studio. Neben dem Emulator kann man die Apps auch gleich auf echter Hardware testen, aber die Geräte müssen dem Entwicklungssystem bekannt gemacht und dafür eingerichtet werden.

Der Lebenszyklus einer Android-App ist anders als der von Konsolenprogrammen, Servern und Swing-Anwendungen, so dass man hier umdenken muss. Es wird viel mit Threads gearbeitet, die miteinander kommunizieren. Apps bestehen aus sichtbaren Komponenten, den sogenannten Activities, aber auch unsichtbaren Komponenten, nämlich Services, Content Providern und Broadcast Receivern.

Die Oberflächengestaltung von Activities geschieht mittels XML-Dokumenten, kann aber auch von GUI-Designern unterstützt werden. Wie werden die Bildschirmkomponenten, hier Views genannt, von Java aus angesprochen? Wie wird auf Bildschirmereignisse reagiert?

Wie erstellt man Layouts, die auf kleinen und großen Bildschirmen, auf Smartphones und Tablets, im Hoch- und Querformat gut funktionieren? Wie kann man die Arbeit hierfür – zumindest während der ersten Schritte – reduzieren?

Im Vergleich zu Desktops und Notebooks haben Smartphones und Tablets meist viel mehr Sensoren eingebaut, die von Android aus angesprochen werden können. Hierbei sind insbesondere der GPS-Empfänger, der Lagesensor, Beschleunigungssensor und der Kompass zu nennen. Manchmal sind auch Barometer und Thermometer vorhanden. All diese können auf sehr stromsparende Weise angesprochen werden.

Android-Apps geben oft multimediale Inhalte wieder. Wie werden Sound und Video in Apps eingebaut? Wie nimmt man Fotos und Videos aus einer App heraus auf? Wie greift man auf Portale wie Youtube zu?

Wegen des GPS-Empfängers sind standortbasierte Anwendungen sehr beliebt. Wie integriert man GoogleMaps in Anwendungen? Welche Möglichkeiten zur Standortbestimmung gibt es sonst noch? Wie werden Ereignisse ausgelöst, wenn man nach Hause, ins Büro oder ins Auto kommt bzw. diese Orte verlässt?

Wie werden bei Android Daten gespeichert? Welche Daten werden auf welche Art gespeichert und warum? Was geschieht auf dem Gerät, was in der Cloud? Wie kommuniziert man mit Servern?

Kurse zur Android-Entwicklung können nach Absprache zusammengestellt werden. In einem einwöchigen Kurs wird man nicht alle relevanten Themen in ausreichender Tiefe behandeln können, sondern muss Schwerpunkte setzen.

Themenbereich Dynamische Sprachen

Tool Command Language

Die Tool Command Language, kurz Tcl, ist nach Ansicht ihrer Kenner das *bestgehütete offene Geheimnis* in der Software-Industrie. Ihre Einfachheit, Flexibilität und Prägnanz sind unübertroffen. Es existieren sehr ausgereifte Interpreter mit und ohne GUI, bei Mac OS X gehört Tcl zum Lieferumfang, für Windows und gibt es Pakete von ActiveState, bei den meisten Linux-Distributionen ist Tcl in den Repositories vorhanden. Für die „großen“ Unix-Systeme AIX, HP-UX und Solaris bietet ActiveState kostenpflichtige Versionen an für den Fall, dass man den offenen Quellcode nicht selbst compilieren möchte.

- **Einführungskurs Tcl**
    - Ein erstes Programm
    - Einbindung des Interpreter-Aufrufs, Anwendung ausführbar bzw. im Dateimanager anklickbar machen
    - Installation von Tcl/Tk auf Linux und Windows
    - Erläuterung der "einfachsten Sprachsyntax der Welt“: 12 Regeln
    - Einfache Variablen: Zeichenketten, Ganzzahlen, Fließkommazahlen
    - Zeichenkettenoperationen und Mustervergleiche
    - Datenstruktur Liste, Listen-Verarbeitungskommandos
    - Datenstruktur Array, Array-Verarbeitungskommandos

- Datenstruktur Dictionary, Dictionary-Verarbeitungskommandos
- Ablaufstruktur Bedingung (if/then/else, switch)
- Ablaufstruktur Schleife (while, for, foreach)
- Arbeiten mit Datumswerten und Ausgabeformatierung
- Prozeduren (Funktionen), Parameterübergabe und Wertrückgabe
- Packages, eigene Bibliotheken erstellen
- Arbeiten mit Dateien
- Einführung in die Ausnahmebehandlung mit Exceptions
- Die Tcl-Standardbibliothek: ein Überblick
- Einbindung des Tk für grafische Oberflächen
- Einfache Widgets und ihre Anordnung auf GUIs

- **Fortgeschrittene Tcl-Programmierung**
  - Datenstrukturen: Listenoperationen, Array-Operationen, Dictionary-Operationen, Iterieren über Listen, Arrays und Dictionaries, Speichern, Übertragen, Wiederherstellen von Datenstrukturen (Serialisierung, Deserialisierung)
  - Default-Parameter, Funktionen mit variabel langen Parameterlisten, Parameterübergabe von Arrays, Listen und Dictionaries
  - Objektorientierung mit TclOO
  - Exceptions: Abfangen von Fehlersituationen, Loggen von Fehlermeldungen
  - Packages und ihre Verwendung (package require, pkgIndex.tcl)
  - Namespaces
  - Registry-Zugriff unter MS Windows
  - Ausführen externer Kommandos
  - Arbeiten mit SQLite: Tabellen anlegen, Daten einfügen, abfragen, ändern, löschen
  - Anbindung an andere Datenbanksysteme (PostgreSQL, Oracle, ODBC)

- **Fortgeschrittene Tk-Programmierung**
  - Grafische Benutzeroberflächen (GUI): Standard-Widgets (Bildschirmelemente): Button, Label, Entry, Text, Radiobutton, Checkbutton, Frame, Canvas, Toplevel, MessageBox, Image, Scrollbar
  - Geometry Manager (pack, grid, place)
  - Widget-Bibliotheken mit MegaWidgets (BWidgets, [incr Tk], tablelist)

- **Special: Netzwerkprogrammierung mit Tcl**
  - Einsatz von Tcl in der Netzwerkprogrammierung
  - Eventsteuerung (keine Erfindung von JavaScript, sondern seit Jahrzehnten erfolgreich in Tcl eingesetzt)
  - Verwendung von Standardprotokollen (Mail, Web, FTP usw.)
  - Verwendung von TCP und UDP
  - Verwendung von DNS, LDAP und SNMP
  - Einsatz von Threads zusätzlich zur Eventsteuerung
  - Tcl-Anwendungen als Webserver
  - Absicherung von TCP-Verbindungen mittels SSL

JavaScript

JavaScript hat seit seinem ersten Auftauchen als LiveScript im Jahre 1995 eine rasante Entwicklung durchgemacht. Sein Potenzial wurde sehr lange Zeit nicht erkannt, wobei die ersten, ziemlich langsamen und auch unsicheren Interpreter hierzu ihren Teil beigesteuert haben. Mittlerweile ist JavaScript in vielen Produkten zu Hause, so in PDF, in Widgets, in Webbrowsern und Smartphone-Betriebssystemen und auch unter dem Namen ECMAScript standardisiert. Seit einigen Jahren existieren auch Implementationen für die serverseitige Programmierung, und zwar mit zunehmendem Erfolg, besonders hervorzuheben ist hier NodeJS.

NodeJS basiert auf dem JavaScript-Interpreter V8 von Google, der genauso wie die aktuellen JavaScript-Engine von Mozilla namens SpiderMonkey, sehr leistungsfähig ist. Außerdem gibt es noch andere Entwicklungen wie Nashorn von Oracle, welches auf der Java Virtual Machine aufsetzt und in Java8 die Engine Rhino aus dem Mozilla-Projekt ersetzt.

- **Einführungskurs JavaScript**
  - Sprachelemente, Syntaxregeln
  - Variablen, Variablentypen (Zahlen, Strings)
  - Ablaufstruktur Bedingung (if/else, switch)
  - Ablaufstruktur Schleife (while, for, foreach)
  - Funktionen, Parameterübergabe und Wertrückgabe
  - Objekte
  - Arrays
  - Standardbibliothek (z. B. Datumswerte und Ausgabeformatierung)
- **JavaScript im Browser**
  - Einbinden von JavaScript in Webseiten
  - Kommunikation zwischen HTML-Elementen und JavaScript
  - Ereignissteuerung
  - Document Object Model, Ein- und Ausblenden von Teilen des Dokuments
  - Erzeugung neuer Elemente im Dokumentenbaum
  - Setzen und Löschen von Cookies
  - Verwenden des lokalen Speichers
- **Fortgeschrittene JavaScript-Programmierung**
  - JavaScript-Objekte herstellen
  - Erstellen neuer Objekte basierend auf Prototypen
  - Verwenden von ausgewählten Design Patterns (Singleton, Factory)
  - Anwenden von Bibliotheken wie jQuery und zepto
  - Anwenden von Frameworks wie jQuery Mobile und AngularJS
  - Data Binding mit AngularJS und Firebase
- **Serverseitiges JavaScript**
  - Einführung in NodeJS
  - Einführung in Meteor, welches wiederum NodeJS verwendet
  - Vorteile einer einheitlichen Sprache auf Server und Client
  - Data Binding von Meteor an MongoDB als persistentem Speicher

Themenbereich Klassische Programmiersprachen

C und C++

C und C++ sind die am häufigsten verwendeten Programmiersprachen, wenn das Ziel lautet, ausführbaren Maschinencode zu erzeugen. Sie sind nach ISO standardisiert und auf quasi allen Plattformen verfügbar. C++ ist neben ihrem Vorfahren C die Basis fast aller heute verwendeten Software, die meisten großen Softwarepakete sind damit geschrieben, seien es Datenbankserver, Office-Pakete, Grafikbibliotheken oder Interpreter für dynamische Sprachen und Bytecode. Aber auch aus der individuellen Anwendungsentwicklung sind C und C++ nicht wegzudenken. Der von C und C++ erzeugte Code ist maschinennah und wird daher sehr schnell ausgeführt, weil keine Interpretationsschicht mehr zwischen Code und Prozessor liegt.

- **Einführungskurs C und C++**
    - Aufbau eines Programms, Compilation und Aufruf
    - Variablentypen und ihre interne Darstellung
    - Konsolen-Ein- und Ausgabe in C und in C++, Formatierung
    - Ablaufstruktur Bedingung (if/else, switch)
    - Ablaufstruktur Schleife (while, for, foreach)
    - Funktionen, Parameterübergabe und Wertrückgabe
    - C++-Funktions-Overloading und Referenzparameter
    - C++-Klassen und -Objekte
    - Arrays
    - Standardbibliothek (z. B. Datumswerte und Ausgabeformatierung)
- **Fortgeschrittene Programmierung mit C**
    - Dateioperationen mit C, Erzeugen, Öffnen, Schließen, Datei-Modus, Direktzugriffsdateien, Text-/Binärdateien
    - Netzwerkprogrammierung mit C
    - Erzeugen von externen Funktionen, Headerfiles, Object Files
    - Erzeugen von statischen und dynamischen Bibliotheken
    - Verwenden der Standardbibliothek
    - Verwenden von Systemaufrufen
- **Fortgeschrittene Programmierung mit C++**
    - Dateioperationen mit C++, Erzeugen, Öffnen, Schließen, Datei-Modus, Manipulatoren, Direktzugriffsdateien, Text-/Binärdateien
    - Netzwerkprogrammierung mit C++
    - Die Standardklasse string
    - Eigene Klassen
    - Vererbung, Mehrfach-Vererbung
    - Konstruktoren, Destruktoren
    - virtuelle, rein virtuelle Methoden und abstrakte Klassen
    - Overloading von Operatoren
    - Containerklassen (Vector, List, Map, Stack usw.)
    - Templates und Namensbereiche

- Ausnahmebehandlung
- **Neuigkeiten in C++11**: u. a. Move-Semantik, Move-Konstruktor, Element-Initialisierungslisten, reguläre Ausdrücke, Threads, smart Pointer, auto

Java

Java gehört zu den beliebtesten Programmiersprachen überhaupt, für kaum eine gibt es so viele Erweiterungen und Bibliotheken. Sie ist auf Geräten aller Größenordnungen zu Hause, von der Armbanduhr bis zum Großrechner – wenn auch nicht in genau derselben Version. Die Programme werden von der Java Virtual Machine JVM ausgeführt, weshalb die Programme nur ein mal für alle Plattformen übersetzt zu werden brauchen. Einzige Voraussetzung für die Ausführung ist die Verfügbarkeit einer JVM für den Zielrechner. Eine Variante der JVM namens Dalvik wird im Betriebssystem Android eingesetzt.

- **Einführungskurs Java**
  - Aufbau eines Programms, Compilation und Aufruf
  - Variablentypen und ihre interne Darstellung
  - Konsolen-Ein- und Ausgabe, Formatierung
  - Ablaufstruktur Bedingung (if/else, switch)
  - Ablaufstruktur Schleife (while, for, foreach)
  - Funktionen, Parameterübergabe und Wertrückgabe
  - Funktions-Overloading und Referenzparameter
  - Arrays
  - Standardbibliothek (z. B. Date, GregorianCalendar, String)
- **Fortgeschrittene Programmierung mit Java**
  - Interfaces
  - Packages
  - Collection Classes, auch typsichere
  - Dateioperationen
  - Netzwerkprogrammierung
  - Objekthierarchien
  - Threads
  - jar-Files
  - Javadoc
- **GUI-Programmierung mit Java Swing**
  - Vergleich von AWT, Swing, SWT und JavaFX
  - Bildschirmelemente
  - pluggable Look and Feel
  - Layoutmanager von Oracle Java
  - Verwendung anderer Bibliotheken für das Layout
  - Bindung von Daten an komplexe GUI-Container wie JList

Themenbereich Linux

Linux ist das am häufigsten eingesetzte Betriebssystem überhaupt, auch wenn auf Desktops und Notebooks andere Systeme dominierend sind. Der Webservermarkt teilt sich derzeit zu ungefähr gleichen Teilen auf zwischen Linux, anderen Unix-artigen Betriebssystemen und Windows. Alle Android-Smartphones, die rund 80% des Smartphone-Weltmarkts ausmachen, laufen auf Linux. Von den Top 500 Supercomputern der Welt liefen im November 2013 über 96% auf Linux.

Von daher sind Linux-Kenntnisse heutzutage so gut wie unabdingbar, sobald man sich von der reinen Desktop-Umgebung mehr zum Kern der IT bewegen möchte.

Linux-Administration

- **Einführung in Linux**
  Umgang mit der Shell (bash), Linux-Dateisyteme, Dateitypen, Shell-Programmierung, Vorstellung einiger Distributionen (Ubuntu-Varianten, SuSE, Red Hat, Knoppix und andere Live-Systeme)
- **Administration von Linux**
  Benutzerverwaltung, Plattenorganisation, Anbinden von RAID- und NAS-Systemen, Datensicherung, Zugangsbeschränkungen, Automatisierung von Vorgängen
- **Linux im Netz**
  Einrichtung von Webserver, Mailserver (smtp, imap), Fileserver, Datenbankserver, individuelle Server, Überwachung übers Netz, Netz-Sicherheit durch Verschlüsselung
- **Vorbereitung auf die LPIC-1-Prüfung**
  Das Linux Professional Institute bietet Prüfungen an, die in der Fachwelt hoch gehandelt werden. Der Kurs bietet hierfür eine Vorbereitung an, die allerdings eigene, lange Erfahrung nicht ersetzt, sondern lediglich ergänzt. Es wird auf die Prüfungszentren verwiesen, wo anschließend die Prüfung abgelegt werden kann.