Theorieteil

1 Open Source Software

1.1 Definition

Open Source bezeichnet im allgemeinen Software, deren Quellcode frei zugänglich ist und weiterverarbeitet werden darf. Die Open Source Initiative[1] (OSI) wendet den Begriff Open Source auf all die Software an, deren Lizenzverträge den folgenden drei charakteristischen Merkmalen entspricht:

Die Software (d. h. der Programmcode) liegt in einer verständlichen Form da: Meistens wird nur die Source gliefert und der Benutzer muss das Programm noch selber kompilieren.

Die Software darf beliebig kopiert, verbreitet und genutzt werden: Für Open-Source-Software gibt es keine Nutzungsbeschränkungen. Weder bezüglich der Anzahl der Benutzer, noch bezüglich der Anzahl der Installationen. Mit der Vervielfältigung und der Verbreitung von Open-Source-Software sind auch keine Zahlungsverpflichtungen gegen einen Lizenzgeber verbunden.

Die Software darf verändert und in der veränderten Form weitergegeben werden: Durch den offengelegten Quelltext ist Verändern ohne weiteren Aufwand für jeden möglich. Weitergabe der Software soll ohne Lizenzgebühren möglich sein. Open-Source-Software lebt förmlich von der aktiven Beteiligung der Anwender an der Entwicklung. So bietet sich Open-Source-Software zum Lernen, Mitmachen und Verbessern an.

1.2 Ideologie und Lizenzen

Die OSI beschreibt die Idee von Open Source folgendermassen:

> The basic idea behind open source is very simple: When programmers can read, redistribute, and modify the source code for a piece of software, the software evolves. People improve it, people adapt it, people fix bugs. And this can happen at a speed that, if one is used to the slow pace of conventional software development, seems astonishing.
>
> We in the open source community have learned that this rapid evolutionary process produces better software than the traditional closed model, in which only a very few programmers can see the source and everybody else must blindly use an opaque block of bits.

So einfach und schön diese Definitionen auch tönen, so verworren ist die Wirklichkeit: Viele Entwickler nennen ihre Software Open Source obwohl sie den drei Merkmalen nicht vollständig entprechen. Weiter muss man unterscheiden zwischen Open Source Sofware, Freier Software und Freeware Software. Die folgende Tabelle soll etwas Klarheit bringen:

[1] http://www.opensource.org/

Licence family: Liberty policy [1]	Licence family: Copyleft [2]	Licence family: Copyright	License name	Download, and use software at no charge	Sources available	Use, Modify or Distribute for free or just for a fee for copy.	Use, Modify, Distribute for a fee or with sources not provided.	Examples
Free								
	Non Copylefted							
		Non Copyrighted	**Public Domain software**	You can [5]	Yes	You can [5]	You can [6]	Xwindow
		Copyrighted to author	**MIT License**	You can	Not always	You can	You can	
			BSD Licence	You can	Not always	You can [7]	You can [7]	BSD Unix
			Apache License	You can	Not always	You can [8]	You can [8]	Apache
	Copylefted	Copyrighted to author	**LGPL**	You can	Yes	You can [3]	You can't [9]	OpenVRML
			GPL	You can	Yes	You can [3]	You can't [4]	Most of GNU softwares
Semi Free Software		Copyrighted to author	**Semi-Free Software**	Yes if you are individual and for non profit purpose only	No	You can't [4]	You can't [4]	PGP
Proprietary Software		Copyrighted to author	**Freeware** (price=0)	You can	No	You can't [4]	You can't [4]	Sonique
			Shareware (price>0)	Only for evaluation purpose	No	You can't [4]	You can't [4]	Most of ASP Softwares, Paint Shop Pro...
			Commercial licence	You can't [4]	No	You can't [4]	You can't [4]	Windows, Office...

Licence chart updated on 2002-09-21

(1) Don't forget that "free" software refers to "liberty" to use and distribute it. So don't use "free software" for a program which price is null.
(2) Copyleft means that changes and distributions can be made with no additionnal restrictions. So a gratis software must be kept gratis.
(3) Sources must be provided
(4) You can if author give its authorization
(5) Everyone who modify a non copylefted and non copyrighted software can use its own licence for his version. So some modified/distributed versions may be Proprietary Softwares.
(6) Your modified version can become a Proprietary Software if you want
(7) Name of authors/contributors can't be used to endorse or promote products derived from the software.
(8) A modified version can't be called with its original name. Name of authors/contributors can't be used to endorse or promote products derived from the software.
(9) You can link a LGPLed library into a commercial program but you must allowusers to use another version of this library.

Somit wird klar: Freie Software sagt nichts über ihre Lizenz aus, so kann die Software gratis sein, aber keinen Source mitbringen oder sogar nur für eine kurze Frist (Probezeit) kostenfrei sein. Software kann aber auch etwas kosten und trotzdem Open Source sein. Falls du Open Source Software schreibst und auf anderen Software Bibliotheken aufbauen willst, musst du auf jeden Fall die Lizenzbedingungen genauer anschauen.

1.3 Open Source Community

Die Open Source Community ist gross und stellt eine Vielzahl von Produkten her. Das wohl bekannteste Beispiel von OS Software ist Linux mit all seinen verschiedenen Distributionen und Applikationen. Die Community beschränkt sich aber nicht nur auf Linux, sie ist sehr bestrebt die offene Software auch für kommerzielle Plattformen wie Windows bereitzustellen. Viele Produkte wie z.B. OpenOffice sind Klone von kommerzieller Software, die ihren Vorbildern zum verwechseln Ähnlich werden. Diese Ähnlichkeit kann so stark sein, dass die Lizenzinhaber der Originalsoftware Klage gegen die Software-Community einreichen wegen Copyrightverletzungen. Sogar Blizzard hat den WarCraft 2 Klon FreeCraft verklagt, das Projekt wurde eingestellt.

1.4 Open Source Tools

Wenn an einem Open Source Projekt gearbeitet wird, dann ist es oft von Vorteil auch Open Source Software dazu zu benützen. Hier eine Liste der bekanntesten Tools. Die Liste erhebt keinen Anspruch auf Vollständigkeit.

- Integrated Development Environments (IDE)

  KDeveloper: (Linux) Vielleicht die beste IDE von allen, Subversion, CTAGS, Doxygen unterstützung.
  `http://www.kdevelop.org`

XEmacs (Linux, Windows, MacOSX) Die traditionelle und für Neulinge etwas umständliche IDE. Viele Plugins erhältlich, die für die Entwicklung von umfangreicher Software auch empfehlenswert ist (z.B. ECB) `http://www.xemacs.org/`

Dev-C++ Ist eine einfache und vollständige IDE, die v.a. auf Windows benutzt wird: `http://www.bloodshed.net/devcpp.html`

- Code Management

  Subversion: (Linux, Windows, MacOSX) Ein Version Control System der neusten Generation: `http://subversion.tigris.org`, für Windows gibt es einen speziellen grafischen Client `http://tortoisesvn.tigris.org/`.

  CVS: (Linux, Windows, MacOSX) Vorgänger von Subversion, immer noch viel gebraucht: `http://www.nongnu.org/cvs/`, spezieller Windows GUI basierten CVS Client: `http://www.wincvs.org/`.

  Many Others: Hier ein guter Vergleich der Verschiedenen VCSs: `http://better-scm.berlios.de/comparison/comparison.html`.

- Kommunikation

  Mailman: Ein Server Dienst, der mit dem lokal Mailprogram (z.B. Postfix) zusammen arbeitet und mailing listen managed. `http://www.gnu.org/software/mailman/`

  Internet Relay Chat: Ein Chatprotokol, dass von vielen verschiedenen Clients implementiert wurde: `http://www.mirc.com/`, `http://www.xchat.org/`.

  Wiki: Eine im WWW verfügbare Seitensammlung, die von den Benutzern nicht nur gelesen, sondern auch online geändert werden kann. Es gibt viele verschiedene Umsetungen, Orxonox verwendet Trac: `http://www.edgewall.com/trac/`

- Open Source Software Projekte

  Sourceforge: Weltweit die grösste Ansammlung von Open Source Software (Registered Projects 06.11.2005: 105,395). `www.sourceforge.net`.

  Freshmeat: Wohl das zweitgrösste Open Source Software Portal: `http://freshmeat.net/`.

- Some Open Source Games

  XBlast Ein Bomberman Klon `http://xblast.sourceforge.net/`.

  FreeCNC: Ein Command&Conquer Klon: `http://holarse.wue.de/freecnc/index.php`.

  SuperTux: Ein jump'n'run ähnliches Spiel: `http://supertux.berlios.de/`.

  Duke3D: Ein Klon des Duke Nukem 3D, sehr buggy: `http://icculus.org/projects/duke3d/`.

  Doom Legacy: Ein Klon von Doom: `http://legacy.newdoom.com/`.

Linux Game Tome: Eine Ansammlung von vielen Linux Games.
`http://www.happypenguin.org/`.

- Game Programmierung Resourcen

Gamedev: Eine gute Seite mit vielen Tutorials und einem sehr aktiven Forum.
`www.gamedev.net`

Gamasutra: Eine sehr professionelle Entwicklerverbindung, ebenfalls sehr viele online Resourcen.
`www.gamasutra.com`

2 KDevelop

In diesem Praktikum wirst du lernen, wie man den KDevelop für Orxonox einrichtet. KDevelop ist die Linux IDE (Integrated Development Environment), die wir für die Orxonox Entwicklung verwendet haben. Falls du nicht KDevelop benutzen willst und dein eigenes IDE bevorzugst (z.B. weil du lieber Windows verwenden willst) dann versuche Orxonox auch dort einzurichten. Wir können dir aber wahrscheinlich nicht gross beim Einrichten helfen. Falls du einen eigenen IDE verwendest, schreib grad ein kleines Tutorial unter `https://www.orxonox.net/cgi-bin/trac.cgi/wiki/LinuxCodingTools`.

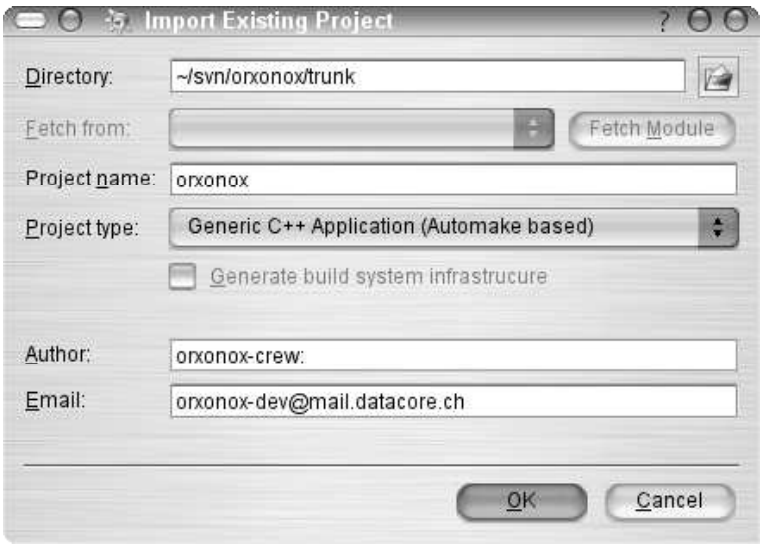
1. **Starten**: Startet kdevelop, entweder von der Kommandozeile. (Falls euch kdevelop gefällt könnt ihr auch einen Shortut erstellen.)

   ```
   kdevelop&
   ```

2. **Projekt einbinden**: Nun msst du das Projekt einlesen, was ganz einfach geht mit:

   ```
   Project -> Import Existing Project
   ```

   Weitere Einstellungen könnt ihr der unten angefügten Grafik entnehmen.



3. **Einstellungen** Stellt KDevelop so ein, wie es euch passt. Geht dabei nach dem Tutoial auf unserem Wiki vor:

   ```
   https://www.orxonox.net/cgi-bin/trac.cgi/wiki/LinuxCodingTools
   ```

Falls das Tutorial Fehler aufweisen sollte oder du findest, dass es noch nicht ganz komplett ist: dann logge einfach mit unter deinem Namen ein und korrigiere rsp. erweitere es einfach!

4. **Spielen**: Am besten lernst du den Editor kennen, indem du etwas damit rumspielst.

3 Subversion in Action

Das letzte mal hast du gelernt, wie man mit Subversion einfach den Inhalt des *Repository* herunterladen kann. Dieses mal wirst du die Änderungen der letzten Woche in euere lokale Version hinein laden müssen.

1. Update Wechsle in einer Shell in das Verzeichnis, welches du mit Subversion ausgecheckt habt, und gib dort folgenden Befehl ein:

   ```
   svn up
   ```

   Dies ladet die Änderungen runter, du siehst eine Liste der veränderter Dateien.

   In der ersten Spalte ist jeweils der Status des Files, in der zweiten das File selbst.

U	File updated
A	File hinzugefügt
D	File gelöscht
G	Änderungen erfolgreich mit lokalen Änderungen zusammengefügt
C	Überschneidungen im File vorhanden

   Um eine Zusammenstellung der Änderungen zu erzeugen gib folgendes Command in deiner Shell ein:

   ```
   svn log -r HEAD:5466
   ```

   Hier bedeuten: *log*: Log anzeigen. *-r*: von Revision *HEAD*: momentane Revision *5466*: ältere Revision (hier die Revision vom letzten Mittwoch.)