

Klein aber oho - eigene FLOSS-Projekte gründen und pflegen

Meike Reichle
meike@a10i.net

Universität Hildesheim

Chemnitzer Linux-Tage

1. März 2008

Wer ist die Zielgruppe?



- Nur der Autor selbst? Motivation?



Generelles

Infrastruktur

Code

Dokumentation

Lizenzen

Werbung

THINK
BIG!

Wer ist die Zielgruppe?



- Nur der Autor selbst? Motivation?
- **Kentnisstand:** Anfänger, Umsteiger, Profis, ...
- **Berufsgruppe:** Studenten, Architekten, Rechtsanwälte, Graphiker, Übersetzer, Journalisten, ...
- **Anwendungsgebiet:** Beruflich, Alltag, Spaß, ...
- **Ausstattung:** Hardware, Leistungsfähigkeit



Die Zielgruppe



- Was sind ihre Interessen?
- Wie kann sie erreicht werden?
- Was motiviert meine Zielgruppe?
- Wie kann ich von der Zielgruppe erreicht werden?



Bevor es los geht ...



- Es ist immer besser **mit**zumachen als **neu**zumachen – FLOSS ist ein Mannschaftssport!



Bevor es los geht ...



- Es ist immer besser **mit**zumachen als **neu**zumachen – FLOSS ist ein Mannschaftssport!

Wenn es schon ähnliche Projekte gibt ...

- Was will ich anders machen?
- Reicht vielleicht auch ein Fork?
- Können einzelne Teile übernommen werden?
- Was kann ich mir sonst noch anschauen?
- Wo liegen die **entscheidenden** Unterschiede?



Webseite / Infrastruktur



- Verfügbare Plattformen:
Sourceforge, BerliOS, Savannah, ...
- Zielgruppengerechte Gestaltung!
- Worum geht's hier? Screenshots, Feature Liste, FAQ
- Downloads
 - Leicht zu finden und zu benutzen!
 - Fertige Pakete sind attraktiver
 - Nicht jeder hat eine DSL Flatrate



Webseite / Infrastruktur



- Aktuell – Datum der letzten Änderung
- Leichter Kontakt – auch der Nutzer untereinander
 - Kontaktmöglichkeiten
 - Wiki/Mailingliste
- Bug Tracking System? Bugzilla, RT, Mantis, Wiki ...
- Mehrsprachig?

- Und schließlich ... † R.I.P



Code



- Lesbar!
- Code-Konventionen
 - Kommentare, Variablen- und Funktionsnamen sprechend und falls möglich in Englisch
 - Ansprechende und übliche Formatierungen
 - Kurze Funktionen
- Modular – Erleichtert die gemeinsame Entwicklung



Code



- Versioniert – Quellcode per RCS verwalten und Downloads immer mit Versionsnummer versehen
 - CVS
 - Subversion
 - Andere (git, Mercurial, bzt)
- Saubere Behandlung von Ausnahmen
- README (welche Datei tut was? Welche Abhängigkeiten gibt es?) und Changelog beilegen
- **Team Maintenance ist immer besser!**



Dokumentation



- **Viel! Überall! Lesbar!**

Generelles

Infrastruktur

Code



Dokumentation

Lizenzen

Werbung

THINK
BIG!

Dokumentation



- **Viel! Überall! Lesbar!**
- Im Code
- In der Software
 - man/info page
 - Auf der Kommandozeile: `foobar --help`
 - In der Gui (evt. browser, yelp o.ä. verwenden)
- Auf der Webseite, aber auch zum Herunterladen in verschiedenen Formaten



Dokumentation



- **Die tollste Software nutzt nichts wenn ich sie nicht benutzen kann**
- Installation und ggf. notwendige Konfigurationen
- Funktionsumfang, Parameter, Optionen, ...
- Howto
- FAQ

- Doku schreiben ist keine lästige Pflichtübung!



Lizenzen



- Eine Lizenz **MUSS** dabei sein
- Was ist "frei" und wer sagt das?
- Welche Rechte sollen vorbehalten sein?
- Wo soll die Software rein?
- Am besten gleiche Lizenz für Code und Dokumentation
- **Keine eigenen Lizenzen schreiben!!**



Existierende Lizenzen



Wo immer möglich bitte existierende Lizenzen nutzen!

- GPL
 - Am weitesten verbreitet – gute Kompatibilität
 - "Virale Lizenz"
 - Auch für Graphiken, Musik, etc. geeignet, da „Source“ hier breit definiert
 - Achtung! Verschiedene Versionen
- MIT/BSD – (quasi) alles erlaubt
- CC-SA – Mit Attribution (DFSG-frei ab 3.0)
- Achtung! Public Domain existiert in Deutschland nicht!



”Werbung”



- **Nicht nerven!**
- Projekt in entsprechenden Verzeichnissen listen lassen (Freshmeat, Free Software Directory, ...)
- Projekt in Distributionen bringen – Selbst paketieren, Debian: RFP Bug, ...
- Mail an MLs (Kein Spam!)
- Projekt auf Events vorstellen
- Blog etc.



THINK BIG!



... und morgen die ganze Welt!!

- Internationalisation
 - Keine hardcodierten Strings, Dateipfade, etc.
 - PO Files und gettext
 - Bei der Gestaltung von GUIs bedenken
- Porting auf andere OS/Architekturen/Geräte?
 - Trennung von Funktionalität und Design
 - Kein ASM einbauen, keine Bit-Breiten voraussetzen, BE/LE beachten ...
- Robuste Infrastruktur
- "Bus-Faktor"



3 Dinge ...



1. Zusammen ist besser als allein!
2. Guter Code und **gute Dokumentation!**
3. Keine eigenen Lizenzen!





Ich würde mich freuen,
von Euren Projekten zu hören!

Fragen? Kommentare? Eigene Tips?

meike@a10i.net

