Sichere Strings und Speicher in C

Pablo Yánez Trujillo

Poolmanager Team
Universität Freiburg

9. August, 2006

- Bibliotheken sind eine Ansammlung von Funktionen, die andere Programmierer dabei helfen, nicht das Rad täglich neu zu erfinden
- Jedem Entwickler steht es frei, seine Bibliothek so weiterzugeben, wie es ihm passt
- In der Unix Welt hat sich aber das GNU Build System etabliert, weil es viele Vorteile bringt
- Das GNU Build System ist eine Ansammlung von Skripten, die versuchen die Einstellungen und installierte Programme herauszufinden, die nötig sind, um das Paket zu bauen, unabhänhig davon, ob man unter GNU/Linux, Solaris, BSD, Windows, usw. arbeitet

- Auf der Kursseite befindet sich eine Projektvorlage für das GNU Build System unter `tag03/projekt_vorlage`
- Ich werde nur eine ganz kleine Einführung in das GNU Build System machen
- Das GNU Build System besteht aus vielen Tools: libtoolize, automake, autoconf, autoheader, usw.
- Anhand unsere Bibliothek werden wir lernen, wie man schnell mit diesen Tool ein Paket baut

- Neben den Dateien auf `projekt_vorlage` gibt es mehrere Dateien, die man selber anlegen muss, weil sie je nach Projekt angepasst werden müssen
- `configure.ac`: Durch diese Datei beschreibt wie, was das `configure` Skript testen soll. Sie wird von `autoconf` gelesen
- `Makefile.am`: Diese Datei beschreibt, wie die `Makefile` aussehen soll. Sie wird von `automake` gelesen
- Meistens hat man ein Verzeichnis `src`, wo der Quellcode gespeichert ist. Bei Bibliotheken hat man üblicherweise kein `src` sondern `projekt_name`

Projekt sc_strings

- Wir werden ein kleines Projekt schreiben, welches die Arbeit mit C-Strings erleichtert
- Wir werden eine "Kopie" der GString Klasse der Glib (Gtk+) machen
- Projektname: sc_strings
- Unter `tag03/sc_strings` befindet sich bereits ein wenig Quellcode

- Ziel der Bibliothek: Benutzung des dynamischen Speichers für die Strings, so dass es Zeichenkette gespeichert werden, nur dann wenn es genug Speicher reserviert wurde
- Eine passende Struktur, sowie 2 Funktionen: `sc_strings_new` und `sc_strings_free`
- Mit dem GNU GCC Compiler muss man mit der Linker Option `-lsc_strings` linken, wenn man unsere bibliothek benutzt

sc_strings Struktur

sc_strings Struktur

```
typedef struct {
    char *str; /* Zeiger auf die dyn.  Zeichenkette */
    size_t len; /* Länge der Zeichenkette */
    size_t size;
    /* Anzahl Bytes, die bereits reserviert sind */
} sc_ strings;
```

sc_strings_new Funktion

sc_strings_new Funktion

```
sc_strings *sc_strings_new(const char *init);
```

- Erzeugt ein neues sc_strings Objekt aus dem dynamischen Speicher mit der Zeichenkette init
- Wenn es keinen freien Speicher gibt, so wird NULL zurückgeliefert

sc strings free Funktion

sc strings free Funktion

```
char *sc_strings_free(sc_strings *string, int content);
```

- Gibt den Speicher frei: den Speicher des `sc_strings` Objekts und der Zeichenektte

Ziele: was sollen wir schreiben?

- Wir werden nur ein Paar Funktionen schreiben
- Es sollten Funktione für folgende Ziele geben:
  - Umwandlung von C-String in `sc_strings`
  - Strings anhängen
  - Löschen Teilstrings
  - Ersetzen eines Teilstrings durch ein String
  - "trim" Funktion
  - Einfügen eines Teilstrings an einer bestimmten Stelle
  - .... überlegt euch was schönes und spiel weiter damit, wenn ihr Zeit habt

Vielen Dank für die Aufmerksamkeit beim "besten Kurs" des SommerCampus 2006