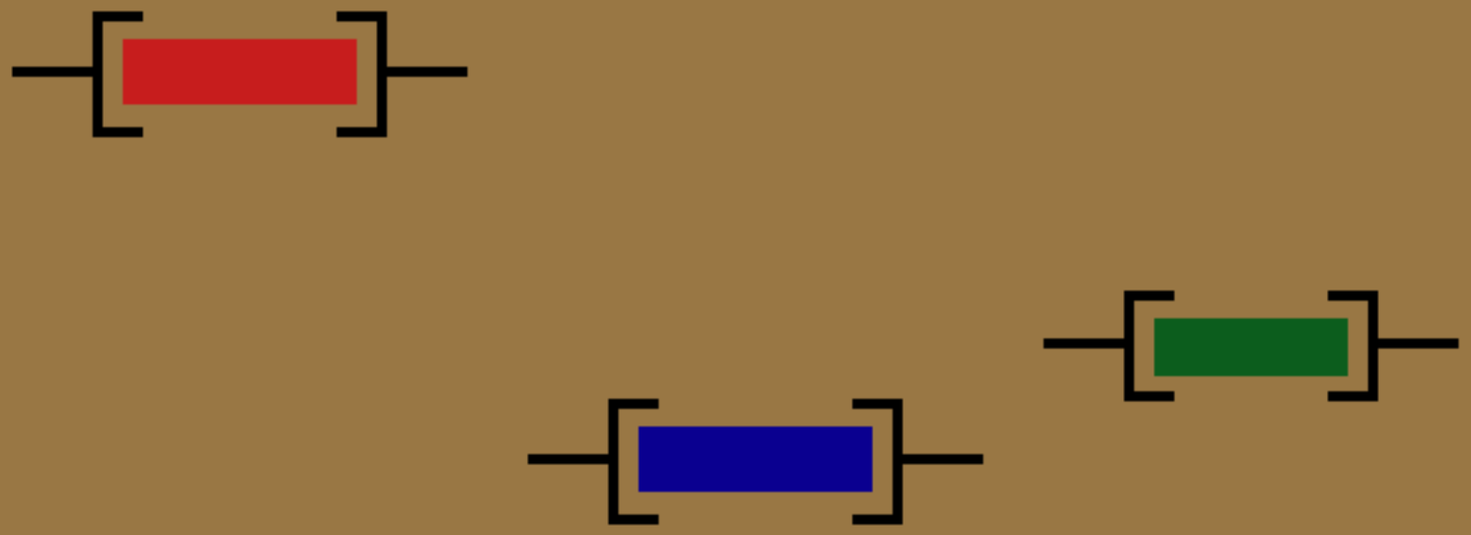


Script-Schnittstellen für compilierte Software

Josef Spillner <josef@kuarepoti-dju.net>

04.03.2006

Chemnitzer Linux-Tage

Worum geht es hier?

Es gibt Programmiersprachen, die compiliert werden.
* beispielsweise C oder C++
* traditionell sehr performant
* aber auch architekturabhängig

Es gibt auch interpretierte Sprachen.
* beispielsweise Java, Python oder PHP
* heutzutage nahezu immer bytecode-interpretiert

Es gibt zwei Arten der Verknüpfung.
* Erweitern (z.B. Sprachmodule in C schreiben)
* Einbinden (z.B. Skripte schreiben und in C-Code verwenden)

Unser Opfer

Grubby, der Chatbot.

Was macht ein Chatbot?
* Verbindet sich zu einem Netzwerk (GGZ, IRC, SILC, ...)
* liest empfangene Nachrichten
* reagiert eventuell darauf

Grubby ist modular aufgebaut.
* Jedes Modul (in C geschrieben) reagiert auf bestimmte Eingaben.
* Spezielles Modul 'embed' als Multiplexer für Skripte.

Programmiersprachen

Grubby spricht die folgenden Sprachen:
* Python
* Perl
* Ruby
* Tcl

Nicht weiter betrachtet:
* PHP
* Java
* JavaScript/ECMAscript (Rhino, KJSEmbed)
* Einbindungssprachen: Lua, SIOD, Io, ...

Letztere oft für spezielle Aufgaben besser geeignet!

Ansatz der Einbindung

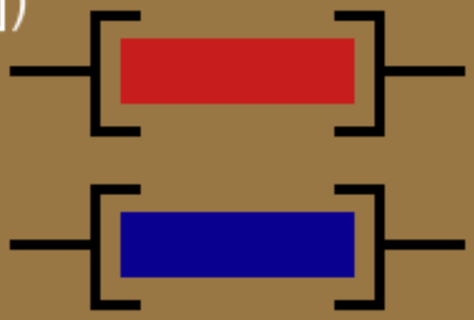
Vorbereitung:
* Erkennung der entsprechenden Bibliothek
* Einbinden der Header-Datei und linken

Ablauf:
* Interpreter initialisieren
* Skript laden und im Cache halten
* C-String in sprachspezifisches String-Objekt umwandeln
* String-Objekt in Laufzeitumgebung 'einpflanzen'
* Skript ausführen
* Antwort auslesen und wieder zurückwandeln

Beispiel: Python

Autokonfiszierung:

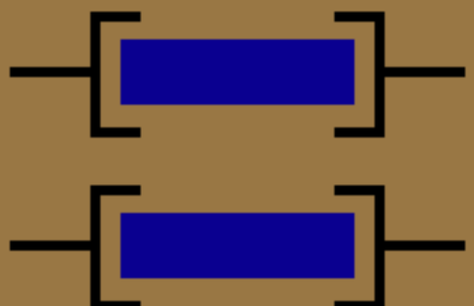
* ...
  AC_SUBST(python_includes)
  AC_SUBST(python_libraries)
  AC_DEFINE([EMBED_PYTHON], 1, [Define to use embedded Python])



Quelltext:

* ...
  #ifdef EMBED_PYTHON
  /* hier die sprachspezifischen Angaben */
  #endif
  ...

* #include <Python.h> /* Python-Definitionen einbinden */
  ...
  PyObject *pxDict; /* Globale Hash-Tabelle */



Beispiel: Python (2)

Quelltext, Ausführung:

```
* ...
  Py_InitializeEx(0); /* Ab Python 2.4*/
  ...
  f = fopen(script, "r"); /* Datei öffnen... */
  PyRun_SimpleFile(f, script); /* ...und ausführen */
  fclose(f);
  ...
  pValue = PyDict_GetItemString(pxDict, "answer"); /* Variable auslesen */
  char *foo = PyString_AsString(pValue); /* ...und von Python nach C konvertieren */
  ...
  Py_Finalize();
```

Live-Demo mit Grubby

Grubby: joke!
* Joke-Skript eingebunden über 'embed'-Modul
* Funktionalität: liest einen Fortune-Spruch aus

Weiterer Ansatz: Auswertung einzelner Blöcke

Bekannt aus der Webentwicklung: HTML-Dateien mit {PHP, JSP, ...}-Tags
* Einschränkungen: Eine Sprache pro Datei, ein Modul pro Sprache

Leichte Modifikation von 'embed': mod_script für Apache

Aus
  PyRun_SimpleFile(dateiname)
wird
  PyRun_SimpleString(quelltext)
etc.

Ablauf:
* Finden von XML-PI-Knoten
* Auffangen der Ausgaben: Umbiegen von Dateideskriptoren

Beispieldatei test.xml

```
<?xml version="1.0">
<html>
<head>
<title>Testseite</title>
</head>
<body>

<?python
import time
print time.ctime()
?>

</body>
</html>
```

Abschluss und Hinweise

Grubby: http://www.ggzgamingzone.org/backend/grubby/
* für C++-Spieleserver auch Scripting Host

mod_script: http://www.kuarepoti-dju.net/download/

Literatur:
* Programming Ruby:
  Kapitel Extending Ruby
* Python-Dokumentation:
  Extending and Embedding the Python Interpreter
* Perl-Manpages:
  man perlembed
* Tcl-Manpages:
  man Tcl_EvalFile