Columbia University
Programming Languages and Translators
Spring 2007
Professor Edwards

Semester Project Proposal

Steve Henderson
Levi Lister
Abe Skolnik
Wei Teng

Executive Summary

Two modes:

- Text: simplified Python.
- GUI: HyperCard for the Java age.

Introduction

The name of our language is “SLAWscript” (Steve, Levi, Abe, and Wei’s scripting language). SLAWscript is a general-purpose (yet simple) scripting language, designed to enable the easy production of either text (i.e. command-line environment) applications or GUI applications that are made of dialog boxes, optionally with custom graphics. Amongst other possible uses, it will allow for quickly programming and deploying interactive training, tutorial, and survey applications.

SLAWscript is modeled on Python, but on a smaller scale. SLAWscript has no arrays, classes, or objects. At this time, only the three standard UNIX-like channels (stderr, stdin, and stdout) are accessible; that is to say, files cannot be opened and used. Also, SLAWscript is not strict about the use of leading spacing.

Fundamental Need

The Java Swing Application Programming Interface (“API”) provides a rich set of Graphical User Interface (“GUI”) widgets that can create useful and intuitive GUI interfaces. Because such interfaces are based on Java, they are highly portable and deployable and thus are desirable for mixed-OS, mixed-CPU-architecture communities. However, Swing requires considerable Java knowledge and experience to create a functional GUI application. This can prevent a non-Swing-savvy researcher, engineer, or other programmer from experiencing the benefits of Swing. For example, a seventh-grade music teacher may want to quickly design a GUI-based tutorial for reviewing musical notation. He or she envisions a series of windows where the student reviews and demonstrates knowledge of musical notation. However, although the teacher is highly familiar with the content of the tutorial, he or she has no idea how to implement it. He or she doesn’t have the time to learn Swing (or another language’s GUI library), and needs a custom solution with per-student question generation and answer processing that cannot be performed (or cannot be easily performed) with common presentation tools such as Microsoft PowerPoint.

We are developing SLAWscript to address this need. SLAWscript allows a designer to use a simple scripting language to develop a program that may use a series of customizable dialog boxes that interact with an end-user. SLAWscript allows a programmer to focus primarily on the content of the application and to use SLAWscript’s program control logic to specify how the application should behave based on user input. The more complicated aspects of Swing applications (component layout, event handling, etc.) are handled by the SLAWscript implementation.

Key Features

- Conventional

  SLAWscript attempts to use conventional notation where possible, as limited by the expressive abilities of ASCII. For example, the bar symbol ('|') is used to both begin and end an operator which returns either the absolute value (for numeric operands) or the string length (for string operands).

- Dynamic

  In SLAWscript, variables don't need to be declared, and they are allowed to contain different data types at different points in time.

- Flexible

  In SLAWscript, the addition operator can take either a number or string as either of its parameters, and intelligently decides whether to perform arithmetic addition or string concatenation. The multiplication operator is similarly flexible, and intelligently decides whether to perform arithmetic multiplication or string multiplication (i.e. multiplying 3 by "Hi" produces "HiHiHi"). In general, wherever a number is required, a variable containing a string containing an appropriate number may be used instead. (The primary exception: "assert" statements.) This allows for easier use of user-entered numbers in SLAWscript programs. For example, if the user entered "3" in response to a prompt, and that string is stored in a variable called "input", then the expression (10–input) yields the number seven.

- Interpreted

  Our implementation of SLAWscript is an interpreter, which facilitates rapid development.

- Intuitive

  SLAWscript is designed to use the English language as a basis whenever it is helpful to do so; for example, to copy the data from a variable named 'a' to a variable named 'b', simply use the command: "copy a to b".

- Portable and architecture-neutral

  Our implementation of SLAWscript is based on Java, which gets us "for free" the advantages that it should be able to run correctly on many different operating systems and CPU types.

- Reduced ambiguity

  In SLAWscript, the equals sign means only one thing: test for equality. Copying data unchanged from one variable to another can be done with the "copy" verb, and general-purpose assignment can be accomplished with the "set" verb.

Representative program

One representative program is an exam preparation assistant for a course in the humanities, such as a history course. Many of these tests require memorization of large amounts of information. SLAWscript can easily be used to create a program to act as an interactive practice exam. This practice exam would involve a series of text prompts or dialog boxes that display practice questions, prompting for student input after each question. SLAWscript's control logic allows the test designer to then branch and evolve the exam based on the student's input. For example, if the student answers incorrectly, hints can be presented to aid in memorization. Or, if the student is mastering the questions corresponding to a certain level of difficulty, the test can provide more difficult questions, thus adapting to the individual student.

A study guide for a typical Art Humanities course is already in the early-prototype stage. The plan for this program is for it to act as a set of interactive flash cards by (for each available picture) displaying a picture of a painting and presenting appropriate multiple-choice questions, and then responding appropriately depending on whether the user answers correctly or incorrectly.

Examples of Syntax

```
set a to 9              # this is how we “load” a literal value
set a to a+1            # this is how we increment a variable
copy a to b             # this is how we copy from one variable to another
put b to stdout         # this is how we “print”
put b+"\n" to stdout    # this is how we “println”

# the CLI way to ask a question and get the user’s answer

put "What is your favorite color?" to stdout
get color from stdin

# the GUI way to do it

get color from string dialog with message "What is your favorite color?"
```

Future Expansion

If time permits, we will program and provide a “bundling” utility which will package together the SLAWscript interpreter, a program written in SLAWscript, and (optionally) picture files needed for the SLAWscript program into a single Java Archive (jar) file. In this way, SLAWscript programmers may create royalty-free distributable SLAWscript applications that can be started simply by entering a single command, for example “java -jar ArtHumanitiesStudyGuide.jar”, or by double-clicking the jar file in graphical environments that support running Java jar-based programs in that way.