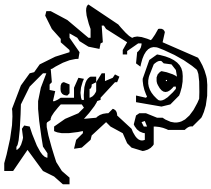
ONEFISH

Programming Language



Joey Fortuna
jf2279@columbia.edu
COMSW4115
White Paper
Summer 2006

Introduction

The Onefish programming language and environment is designed to solve a number of problems in K-12 programming practices, which is to say: kid programmers. Although much reference online is made to languages like LOGO (a dialect of LISP originally developed as a teaching language developed at MIT[1]), Alice (a drag-and-drop educational environment for creating 3-D graphics[2]), and KPL ("kids programming language" which uses gaming as a hook to draw kids into the world of development[3]), there is very little focus on programming as a diversion in and of itself. Alice comes closest to this paradigm, and as a result we borrow heavily on the work of the team at Carnegie Mellon.

Computer programming is an essentially creative act, and the chief attraction to software development (at least to us) is the extent to which one can manipulate one's own environment – similar to the feeling one gets when working with Play-Doh™. At this stage in human development, programming as an act of self-expression should rightfully take its place alongside other creative childhood exploits like wokring with Play-Doh™, coloring with crayons and throwing rocks at your sister's head.

Onefish

> Onefish: a simple, object-oriented, interpreted and interactive, architecture-neutral, cute, cuddly language.

Taking direction (as suggested) from the Java whitepaper, we'll not only be writing in the first person plural, we will also be describing Onefish in the context of a sequence of buzzwords. Here's what we mean by those buzzwords:

Simple

Onefish is designed to be used by children and childish adults at the very dawn of their interest in computers. You will not be able to invert (or even multiply) matrices with Onefish, but you will be able to calculate eigenvectors. Just kidding. The language is designed to provide the user with the ability to create and manipulate animated shapes, with accompanying sounds. In addition, each Onefish program is designed to be able to connect to another running program so that the shapes can communicate with one another (see Network-Aware, below). The emphasis of the language is simplicity, both of syntax and sematics and in terms of the reduced scope of the language's capabilities. However, once it achieves widespread popularity, we have no doubt that enthusiasts will extend

[1] http://en.wikipedia.org/wiki/Logo_programming_language
[2] http://www.alice.org/
[3] http://www.kidsprogramminglanguage.com/

Onefish with libraries that will enable one to control network systems and financial services mainframes with animated shapes and accompanying sounds.

Object-Oriented

Using Onefish's object-oriented capabilities developers will be able to create "classes" of shapes that can be re-used in multiple programs. Future, unimagined shapes can inherit from these classes.

Interpreted and Interactive

Rather than osterizing Onefish source into assembly code or machine language, we've chosen to implement a purely interpreted environment. This will have the advantage (as mentioned in an earlier lecture) of providing rapid response to updated code, as well as giving the developer the type of instant feedback she might expect from Play-Doh™ or throwing rocks back at her brother.

Architecture-Neutral

Onefish is designed to be executed within a Java environment, as a result, we are piggybacking on Java's own architectural neutrality. Simply put, if a machine can run Java, then it can run Onefish.

Cute

As our goal is to provide an experience of software development that is its own diversion, we will strive to create an IDE and runtime environment that is as user-friendly as possible. For instance, rather than spewing out arcane and toxic-seeming stack traces when errors are encountered, will endeavor to provide the user with just the right mixture of encouragement and guidance that will set them back on the right track. In addition, since Onefish will be an interpreted language, the user will be able to see the effects of their programming immediately and from within the development environment.

Cuddly

Borrowing both from the work of the developers of Alice and from concepts devised in Neil Stephenson's Diamond Age (or A Young Lady's Illustrated Primer) – a book which described an incredibly complicated nanodevice that served as an ever-changing machine that adapted to the needs of the young heroine, Nell -- we hope to create a language and environment that will serve as the type of companion for a child that a good book, or a worn teddy bear would be. We will strive to make the language as intuitive and involving as possible,

making use of rhymes, for example, wherever the syntax allows. In this, we will draw (not only from the above sources) but also from people like Lewis Carroll and Theodor Geisell (Dr. Seuss).

Summary

The Onefish programming language provides an engaging and entertaining introduction to the world of computer programming. Certainly, efforts at software development in general soar beyond the scope of what one can achieve with Onefish (both in terms of complexity and efficacy). But Onefish applications are designed to be self-created entertainment and to introduce budding developers to the world where creativity and technology intersect – which is precisely the point at which innovation is born.