Examination Generation Grading Language (EGGL)

COMS W4115: Programming Languages and Translators
Project Proposal

Gordon Hew (CVN) (gh2242@columbia.edu)

Introduction

Tests have been used as a measure to gauge a person's mastery over a particular skill. They have taken many forms depending on the nature of the skill that is being assessed. As most students can attest, the majority of examinations follow a basic question and answer format. In general, instructors create these tests in a WYSWIG text editor such as Word and grade a student's answers by hand. This process is error prone and requires a large amount of human effort. This is largely due to instructors using tools that are not designed for creating tests.

The Examination Generation Grading Language (EGGL) seeks to provide a simple language that can easily facilitate the creation and scoring of computerized exams. Such a language would be beneficial to the academic community in that it would allow instructors to quickly write an interactive exam that can be scored instantly. Such a language can allow more sophisticated users to create reusable features such as selecting questions at random from a pool, adaptively selecting questions based difficulty, or dynamically generating new questions.

Language Features

EGGL is a procedural programming language that supports basic data types, control flow, operations, and user defined routines.

File Declaration

Each EGGL file is defined as a single Exam. Code that resides outside of a function is executed during runtime.

Data Types

EGGL supports basic data types such as

- string
- integer
- floating point values (i.e. double)
- array
- map

Control Flow

EGGL supports basic if-then-else conditional logic. It will also support while and for loops.

Built-in Operations

EGGL supports basic arithmetic functions such as:

- +: addition
- -: subtraction
- *: multiplication
- /: division
- ==: equality check
- !=: negates equality check

Additional supported built-ins include:

- size: Finds the size of an array or string.
- in: Checks if an element is in an array.
- randomInteger(x,y): ouputs a random integer. x and y are optional parameters and define the range of the random value.

Input/Output:

- print: Prints a string to standard out.

EGGL specific built-ins:

- prompt: The actual text of the question being asked. Only one can exist in a question function.
- choices: The choices that a user can present. If not present, it is assumed that the choice is free form text. Only one can exist in a question function.
- answer: The correct answer to the question. Only one can exist in a question function.

Functions

EGGL supports the creation of functions that can be used for reusable computation. EGGL also supports the following custom function types:

- question: This function is used to define Questions that compose an exam. Additionally, questions can be nested within other question functions to create a multi-part question.
- grade: Only a single grade function can be defined in an exam in order to support custom grade calculation. If left undefined, the grade reported is a percentile of questions answered correctly.

Code Examples

The below code snippet defines a simple exam with two questions.

```
Declare SampleExam.eggl

// custom function declaration
func yesNoChoices(){
      return [ “yes”, “no” ];
}

// question definition
func question question1(){
      prompt( “Is the sky blue?” );
      choices( @yesNoChoices );
      answer( “yes” );
}
```

```
func question question2() {
      prompt( “Who is the largest car maker?” );
      answer( “Toyota” );
}

question1();
question2();
```

Sample output:

```
1. Is the sky blue?
a. yes
b. no
> a
2. Who is the largest car maker?
> Honda
50% of questions answered correctly.
```