# HGL: Hypertext Generation Language

Yan Koyfman – Goup Leader (ysk2106@columbia.edu)
Shruti Gandhi (slg2109@columbia.edu)
Timothy Kaczynski (tdk2102@columbia.edu)
Edward Mezarina (eem65@columbia.edu)

COMS W4115: Programming Languages and Translators
Department of Computer Science
Columbia University

27 September, 2005

## Introduction

There was a time when a person with an idea or an opinion had to convince a publisher that the content they had created was worth the time and money required to distribute it. The Internet has made it possible for anybody with a personal computer to publish just about whatever they want to an incredibly large and diverse community of people, at little to no cost. Unfortunately, the languages that describe to a computer how to display that content have little to nothing in common with the languages used for general purpose computing. HGL allows a programmer to display his or her content using familiar syntax and semantics.

HTML, as its name suggests, uses a markup language where content is static and buried in a sea of tags which make it difficult to edit. Languages such as PHP provide more flexibility and a structure which more closely resembles C or Java, but still require the programmer to have an understanding of how HTML works. HGL is a PHP-style language which abstracts the markup language of HTML inside built-in data types, so the programmer can concentrate on their content rather than how it will be presented.

HGL is geared towards the experienced C or Java programmer who desires a simple and familiar way to create web pages which contain static and a limited amount of dynamic content. HGL, when compiled, produces a mix of HTML and CSS which can then be directly published to the Internet.

## Goals

The main goal of HGL is to remove the dependency on HTML from languages such as PHP. HGL is a language which resembles PHP in many ways. When used for web development, a drawback of PHP is the requirements to embed HTML tags directly inside the PHP code. While PHP provides the ability to easily turn static content into dynamic content, it relies on HTML or CSS to handle presentation logic. HTML is a language which has little to nothing in common with recent high level languages for general purpose computing, such as C and Java. This requires someone who is already skilled in general purpose computing and programming to learn yet another language. HGL contains built-in types to abstract the presentation logic of HTML from the developer, allowing them to describe their content using familiar syntax and semantics.

PHP is an incredibly powerful language in and of itself. There are also an extensive list of built-in and third-party libraries available to add features such as database and EIS access. HGL does not attempt to duplicate the breadth of PHP. Instead, HGL defines a subset of the PHP language, and adds features that provide the presentation logic which PHP lacks.

## Main Language Features

### Syntax

The syntax of HGL resembles that of PHP, with some modifications. Variable names are preceded by a dollar sign ($), but must be declared before use. Some features present in PHP are not, however, permitted in HGL; for example, variable interpolation in double quoted strings is not available.

**Data Types**

Support for standard types, such as integer, floating point, boolean and strings, are available.  In addition, built-in objects to represent HTML constructs such as a page, paragraph, table, ordered and unordered list are supported, along with several settable attributes per object.  All variables must be declared before use, along with their type. Arrays of at least one dimension of all objects are possible.  Dynamic arrays, however, are not available; array sizes must also be declared before use.  Arrays can be indexed only with integers, not strings.

**Flow Control**

Standard mechanisms for controlling if/else statements are available.  The body of loops is delimited by curly braces, { and }.

**Looping**

HGL allows 'while' and 'for' type looping, but will not support 'foreach'.

**Operators**

The following arithmetic operators are permitted:  + - / *
In addition, a special string concatenation operator, | (pipe), is available.  The period (.) operator is used to access attributes and methods on HTML objects.  Standard boolean operators for use in if/else statements are present as well:  < > <= >= == != || &&

**Scoping**

Symbols are scoped at either file (global) level or the function level, depending on where they are declared.

**Comments**

Embedded comments are prefixed with two forward slash characters (//), which comments the remainder of the line.

**Example**

This HGL code will generate a simple web page with three red paragraphs and an unordered list:

```
// Define simple variables
int   i, red;
string          str;

// Declare HTML objects; several are built in to the language
page  pg1;
```

```
paragraph   pp[3];
ulist ul;

// Set values using assignment operator
$i    = 0;
$red  = 0xff0000;
$str  = "World";

// Loop through paragraph array to set its attributes
while ($i<3) {
      $pp[$i].color = $red;
      $pp[$i].text  = "Hello, " | $str | " " | $i;

      // Add each paragraph to the page object
      $pg1.addElement($pp[$i]);
}

// Add a list item and append to page object
$ul.addItem("List item");
$pg1.addElement($ul);

// Print out the HTML associated with the page object
print ($pg1);
```

Would generate HTML & CSS similar to the following:

```
<html>
<head>
<style type="text/css">
<!--
      p.paragraph { color: 0xff0000 }
-->
</style>
</head>

<body>

<p class=paragraph>Hello, World 1</p>
<p class=paragraph>Hello, World 2</p>
<p class=paragraph>Hello, World 3</p>

<ul>
      <li>List item</li>
</ul>

</body>
</html>
```