HTML, MATLAB and Delphi

The field of computing has many programming languages and each language was designed to accomplish specific tasks. The problems can be so different so the languages developed to solve these tasks can have many differences. In this research paper I will look at three languages meant for solving various problems and widely used in computing and. HTML is one of the most heavily used markup languages in the world today, it is the way to mark up pages for websites in the Internet and local networks. MATLAB is a scientific language mainly targeted for use in research. Delphi is a commercial language used for prototyping and rapid application development.

Markup languages are not generally considered programming languages because inability to produce truly executable code. Instead, markup languages define the rules for document mark up and layout, like defining document elements and placing them in the right location for visual output. Such markup languages are used by many people everyday when printing documents or browsing the web. HTML (Hyper Text Mark Language) is the most popular text markup language used in today world. It is heavily used on the internet to present all kinds of information.

HTML does not have variables, functions and can not execute any code, instead HTML defines the look of the document when outputted to screen. HTML documents are stored as text, and markup is interpreted on the fly when requested. HTML requires an interpreter, called browser in order to display the document. Browsers are available for desktop and mobile environments, most browsers are utilizing GUI in order to display documents, but some browsers can operate from console.

The main construct of HTML is called a tag, tag is defined as text enclosed in the angled braces (“<” and “>”). HTML has a set of tags that can be used in the document, such as “<html>”, “<body>”, “<p>”, etc. Many tags in HTML work in pairs, and for each opening tag, there is a closing tag, each closing tag is defined by a slash “/” following the angled brace. All the tags mentioned above are paired tags that must be associated with a closing tags, “</html>”, “</body>”, “</p>”, somewhere in the document. Tags, or pairs of tags can be placed in between the opening and closing tags, creating a structure that can have many layers of tags nested into each other. Tags in HTML are not displayed to user and are used to create the markup and layout rules. Any text not enclosed in angled braces will be outputted to screen.

HTML has a number of versions, the most recent one is an HTML 5, which is still currently in development [1]. Although HTML has specifications for different versions of its standards, it is up to the parsers to enforce the standards and currently parsers do not support all features of HTML 5, and can even produce different results when the document is written in older HTML 4 standard. With HTML 5 not being officially certified yet, many of its features are considered experimental even though they have gained a widespread use in the past few years. HTML parser are generally not strict parsers, meaning that they still try to parse document and produce the output even if the document contains HTML errors. Lets consider an example:
In this example we have an HTML error, generally refereed as tag soup. In particular we see that some tags are mismatched. The <strong> is opened within the <p> tag, but is closed outside of it, violating nesting of HTML tags. Such document looks like HTML, but it contains the error and violates HTML syntax. Most Browsers try to recover from HTML errors, and still display the document as best as possible given the errors. Firefox will try to correct the mistake by assuming that the coder forgot to put missing closing and opening tags, and the document actually displayed by Firefox will have the following code:

```
<!DOCTYPE html>
<html>
<head>
<title>Example 1</title>
</head>
<body>
<p>
This is a tag soup <strong>example</strong>
</p>
</strong>
<p>
In this example child tags closes after the parent tag
</p>
</body>
</html>
```

Some of the problem of incorrectly rendered documents in different browsers can actually stem from such error correction done differently in different browsers or version of a browser. Other page rendering discrepancies originate from not following HTML standards by some browsers, and finally, some pages might be rendered incorrectly due to the lack of support of certain new features. When developing a document in HTML, it is important to consider many browsers and check for correct rendering in all popular browsers.

MATLAB is widely used in research and engineering[6]. MATLAB strengths are numerical computing, matrix manipulations and ability to interface with other languages and programming environements. One of MATLAB's key features is matrix manipulation, hence the name MATrix LABoratory. MATLAB is an interpreted language, with the interpreted being imbedded into the
MATLAB software. MATLAB programs are stored as source code in plain text and can be executed in any environment which supports MATLAB software and interpreter. It is currently available for Windows, Linux and Mac OS[2], but there are mobile application that can execute MATLAB programs distributively over the internet. The language is an imperative, procedural with OOP support. Code can be written in the procedural style, but can be intermixed with object oriented approach as well.

MATLAB being research oriented offers some interesting features, such as multiple variables in a function return, ability to interface with many other programming languages, new line as a delimiter, semicolon as an optional functional delimiter and others. Lets consider the following example:

```matlab
function [x, y] foo (a)
    x = 2 * a;
    if x > 10
        y = a + 5;
    else
        y = a * 5
    end
end
```

The function definition “function [x, y] foo (a)” specifies that this particular function will return two values defined by variable x and y within the function. This function accepts only one parameter. The values will be returned inline, and the programmer is responsible for catching them. Also, as seen from example, most language structures in MATLAB must end with an “end” keyword. MATLAB does not require delimiters to be present after the statements as the new line character serves as the delimiter, but most statements still end with a semicolon as a delimiter, because the commission of the semicolon causes the MATLAB to print the result of the statement to screen[5]. Such feature is very useful for debugging MATLAB code, but it is not the preferred way of outputting results. In the example above, if a is 5, the “y = a * 5” line will produce the output of “y = 25” to screen.

MATLAB programs can interface with programs written in other languages, for example there are way to invoke code written in C, C++ and FORTRAN. In addition programmer can work with Java right in the MATLAB code, without having to write a separate function to handle invocation of other programs like it is done for C/C++ and FORTRAN integration. The following code illustrates how Java can be used in MATLAB:

```matlab
xmlObj = xmlread('file.xml')
children = xmlObj.getChildNodes;
children = children.item(0);
```

In this code xmlread function of MATLAB returns java object the represents the entire xml file as a tree structure. GetChildNodes method returns the array of JAVA objects that represent sub-trees of the xml file. Please note the MATLAB syntax used to access java, in particular getChildNodes method call does not require parentheses to be used at the end, yet it works with parentheses as well. Such seamless integration if Java into MATLAB also brings some confusion. For example, Java uses zero-based indexes for arrays and other iterative structures, while MATLAB has one-based indexes. As such the programer must remember when working with Java objects and adjust the iteration accordingly.

MATLAB languages does not have explicit variable declarations, and variable type is inferred from the data assigned to the variable. As such coercion is widespread, and MATLAB even does
type conversions between Java data types and internal data types. Such type conversion is required to happen at some point since, java is a strongly typed language and any data entering into Java program form MATLAB needs to be converted to appropriate Java data type.

Matrix manipulation is one of the key features of MATLAB. A programmer can define a matrix almost like a regular variables with matrix elements being enclosed in the square brackets:

```plaintext
myMatrix = [4 5 7; 3 8 2];
```

This code will creates a 2x3 matrix with appropriate values; we can perform various algebraic operation on matrices just like on the regular variables, except the programmer must remember about the rules of matrix manipulations. For example, the code below will produce an “Inner matrix dimensions must agree” error:

```plaintext
myMatrix = [4 5 7; 3 8 2];
myOtherMatrix = [1 2 3; 9 8 7];
ans = myMatrix * myOtherMatrix;
```

MATLAB allows programmer to do a by element operations. For example, if I want one element of one matrix to be multiplied by a corresponding element of another matrix, I can use the code similar to the previous example, but instead of using multiplication operator, I can use by-element multiplication operator, which has dot (“.”) symbol prefixed to the operator[5]:

```plaintext
myMatrix = [4 5 7; 3 8 2];
myOtherMatrix = [1 2 3; 9 8 7];
ans = myMatrix .* myOtherMatrix;
```

The result of the expressions above will be a 2x3 matrix:

```plaintext
ans = [4    10    21;
       27    64    14]
```

Aside from matrices which for obvious reasons support only numeric data types, MATLAB has cell arrays which can hold non-numeric types as well, such as strings.

The last language I reviewed is Delphi. This language can be seen as a dialect of Object Pascal. The first version of Object Pascal was developed by Apple, but since than the further development has continued at a number of different companies. Delphi as it is used now is more than just a languages or compiler, it is rather a set of development tools such as IDE, compiler, debugger, etc bundled together, as such when one is referring to Delphi, it most often means the entire set of tools. For the purposes of this paper, I will refer to Delphi mainly as a compiler.

Delphi is an imperative strongly typed, object oriented language, based on the pascal syntax. As such many of the syntactic elements of pascal, such as “begin” and “end” blocks, variable declarations blocks, “:=” assignment operator and others are present in Delphi.

One interesting aspect of Delphi when it comes to the data types and variables are the way Delphi handles enumerated types. In additional to the traditional enumerations, there are so called “Sets”, which can store a set of values, up to 255, but each value can be set as “On” or “Off” [7]. A set variable can than be used in branching later on with evaluation condition depending on whether a values in the set is set “On” or “Off”. To better illustrate the concept of sets, lets look at the example:

```plaintext
var
    SmallNums : Set of 0..55;       // Set of the 56 set members
begin
    // We have a range of 0 to 55 values that we can set
    SmallNums := [3..12,23,30..32];  // Set some of the members on
```
if 12 in SmallNums
then ShowMessage('12 is in SmallNums')
else ShowMessage('12 is NOT in SmallNums');
if 13 in SmallNums
then ShowMessage('13 is in SmallNums')
else ShowMessage('13 is NOT in SmallNums')

This example will produce the following result:

12 is in SmallNums
13 is NOT in SmallNums

This is due to the fact that our sets did not have a value of 13 turned “On”, while the value of 12 was “On”. Sets can be used over integer in the range of 0 to 255, characters or enumerated types. Delphi also can allow programmer to define own data types with a different possible range of values, this can be achieved through a feature called “subranges” [8].

Delphi supports dynamic sized arrays, which means that the array size can change over time. Like Java Delphi allows for generic programming, a term used to describe a programming style in which algorithms are written with no data type specifications, and data types is specified later, when using such generic code.

Unlike many other modern languages, Delphi does not have a garbage collection routine provided, so it is a programmers responsibility to manage memory and enforce memory deallocation[3]. This is an interesting feature, as programs in Delphi can be compiled not only to Windows and Mac OS systems, but also to the Android platform, which inherently relies on Dalvik virtual machine that has built-in garbage collection. There are plans to expand Delphi compiler to support compilation of programs for Linux environment.

Delphi being imperative language has some support for functional programming[3], this is partially used to avoid potential memory leaks associated with creating new variables and objects and than having the need to deallocate the memory.

One of the main features of the Delphi as the entire set of development tools is its suitability for RAD, or Rapid Application Development. Delphi IDE has tools for GUI development, and prototype creation, which makes Delphi a fairly widespread language for making software prototypes or developing new software in limited time.

Let’s consider a few code examples of Delphi:

```delphi
var
    count : Integer;
begin
    For count := 1 to 5 do
        ShowMessageFmt('Count is now %d',[count]);
end;
```

In this example we can see the variable count being declared as integer, before that variable is used as the control variable in the for loop. The for loop sets the count variable to be 1 and sets the stopping condition for the variable to be equal to five. The “to” keyword means that the count will be incremented by one on each loop iteration. “downto” keyword can be used to decrement the control variable of the loop.

Delphi subroutines can be broken into two related but separate constructions[4]. Procedures, are subroutines that can take zero or more parameters and do not return a value back:
procedure ShowTime;              // A procedure with no parameters
begin
    // Display the current date and time
    ShowMessage('Date and time is '+DateTimeToStr(Now));
end;
    // Let us call this procedure
ShowTime;

Functions on the other side, can return values. Each function has a hidden variable “Result” to which the result value must be assigned in order for function to return it.

In this paper I have looked at three different languages, used for different tasks and in different environments. I have looked, on my opinion, at some of the most interesting aspects of these languages, which differentiate each of these languages from the competition in their respective fields.


