Visual Pro Tutorial A

A beginners guide to Visual Scripting

VisualPro: A Lightweight; Visual Scripting Tool

Tutorial: Classes and Objects

Authors:
Edward Patch
Student Number: 1801492

Supervisor:
Mike Dacey

16 February 2022

Contents

1 Learning Objectives

The following learning objectives are as follows:

1. Understanding the VisualPro application.
2. Understanding the terminologies of classes, objects and methods.
3. Writing several classes in a Visual Scripting environment.

2 Introduction

2.1 What is VisualPro?

VisualPro aims to create a lightweight Visual Scripting tool that encourages individuals to the development field. VisualPro enables users to create code structures to help develop ideas into reality.
Note: VisualPro does not have logical programming tools in its current implementation.

2.2 What will the tutorial cover?

The tutorial covers VisualPro Environment of how to use the software, found in section 3, page 4 and terminology found in section 4, page 5. The tutorial teaches the basics of Object-Oriented and how to implement the structure within VisualPro, found in section 5, page 6.

Example 2.1: Object-Oriented Languages (or) Languages with Object-Oriented Features

C++, C# and Java. To view other object-oriented languages,
WordDisk - Languages with object-oriented features [1].

3 VisualPro Environment

3.1 Features

VisualPro offers a few features such as:

- Classes, Functions and Arguments, and Variables.
- Saving in Multiple Languages.
- Drag and Drop Elements and Text Areas.
- Property Windows to Control Arguments and Relationships.

3.2 Known Bugs

A couple of bugs include:-

- Arguments for functions are not currently available.
- Deleting containers does not mathematically reset the following location or move existing containers backwards.

3.3 Saving Progress

After completing a tutorial, select the language and press save.

Tip 3.1: Compatibility

As mentioned previously, not all languages support object-orientation. If a language is not compatible with a particular keyword, the code generator will ignore the selected syntax object.
Example Language: C is not supported.

4 Terminology

4.1 What terminologies exist within Object-Orintation programming?

4.2 What is a Class?

A class enables developers to containerise code that is referenced later as objects. Java language enables the demonstration of a typical Class declaration in an Object-Oriented Programming (OOP) environment.

Example 4.1: Code Example - C# and Java

An example syntax of declaring a class in C# and Java languages:-

```
public class Name {
    protected void Method_A() {}
    public void Method_B() {}
    private void Method_C() {}
}
```

Within languages similar to C# and Java, member tags enable scope-protection to classes, methods and variables with different purposes. The code snippet displays that the declaration of members, public, private, and protected members align with the class or method declaration. To understand classes and modifiers within C#, look at Classes in the C# Language- [2].

Example 4.2: Code Example - C++

```
class Name {
    protected:
        void Method() {}
    public:
        void Method() {}
    private:
        void Method() {}
};
```

C++ language uses member colon style with a new line. To understand C++ classes and modifers, look at Classes in C++ Language [3]
Note: - the modifiers of protected, public and private members mean the same as the previous example.

4.3 What is a Method?

Methods belong within a class that interacts with neighbour methods and variables, child class methods and variables or object methods and variables (dependent on the access modifier).

Unlike Class containers, Methods execute a 'script' that **can** communicate with inherited Variables and Members, usually using the *this* or *self* keyword dependent on the programming language.

Tip 4.1: Documentation Tip

When using a programming language, it is important not to remember every language's syntax. As programming languages are the same, 'not all size fits the one' tool, sometimes it is beneficial to try different languages to fix the same problem to see many different methods available. After learning the generic tools, progressing from one language to another becomes a relaxed and speedy experience. Refer to the programming language documentation for any uncertainties.

4.4 What is a Object?

When referencing a Class in another instance, it is important to remember that this is called an Object. Two examples of objects demonstrates the meaning, with a reference of what an Object is to enable further study on this terminology.

Example 4.3: Object Demonstration (C# and Java Language)

Java:

```
public class Program {
    public static void Main(String[] args) {

    }
}
```

C#:

```
class Program {
    static void Main(string[] args) {

    }
}
```

5 Object-Orientation

5.1 Exercise: Understanding the basics

5.2 Exercise: Trying out a Class

5.3 Exercise: Animal Types

5.4 Exercise: Vehicle Components

6 Keywords

- OOP - Object-Orinted Programming.

References

[1] Word Disk, "List of object-oriented programming languages - Wikipedia @ WordDisk," Mar. 2018. [Online]. Available: https://worddisk.com/wiki/List_of_object-oriented_programming_languages/

[2] Bill Wagner, "Classes," Sep. 2021. [Online]. Available: https://docs.microsoft.com/en-us/dotnet/csharp/fundamentals/types/classes

[3] cplusplus.com, "Classes - C++ Tutorials." [Online]. Available: https://www.cplusplus.com/doc/tutorial/classes/