Programming Paradigms

Styles of Programming

Programming – Extra Content

Last modified 19/04/16 by Richard Stern

aie
SPECIALIST EDUCATORS IN GAMES, ANIMATION & FILM VFX

Contents

Introduction

- Programming Paradigms are a way to classify different styles of languages.
- Some languages are strictly one style, others use multiple styles.
- Some paradigms are based on and extend other paradigms.

List of Paradigms

- There are many paradigms, common ones include:
  - Imperative vs Declarative.
  - Procedural vs Functional.
  - Object Oriented
  - Logic
  - Symbolic

- For an extensive list:
  - https://en.wikipedia.org/wiki/Programming_paradigm

What are side effects?

- A common difference between paradigms is whether they support side effects.

- A language supports side effects if:
  - A statement, function or procedure can change the program's state or operate on data that was not part of its input.

```
//No side effects
int Add(int a, int b)
{
    return a + b;
}
```

```
int value = 5;

//Has side effects
int Add(int a, int b)
{
    value++; //side effect
    return a + b;
}
```

Imperative vs Declarative

- Imperative is the most common paradigm.
- Imperative uses statements to change a program's state.
  - Step by step instructions for the computer to perform.
  - Describes *how* the program should operate.
  - Often contains loops, gotos and switch statements.
  - Allows side effects.
- Imperative examples:
  - Most procedural and OO languages including C++, C, Java, etc.
  - Machine code is imperative.
  - A cooking recipe is imperative (Step by step instructions.)

Imperative vs Declarative

- Declarative is the opposite of Imperative.
- Declarative describes the desired result:
  - Describe *what* the program should do, not *how* it should do it.
  - No side effects.
- Declarative Examples:
  - SQL, Regular Expressions, Prolog, Wolfram Language
  - A shopping list could be declarative. (Doesn't list where to shop, which isle each product is found in, or the process of purchasing items.)
- Declarative Links:
  - SQL example: http://www.w3schools.com/sql/sql_where.asp
  - Wolfram example: https://www.youtube.com/watch?v=_P9HqHVPeik

Procedural vs Functional

- Procedural is a form of Imperative programming.
  - Most Imperative languages are also Procedural.
- Procedural languages treat programs as a series of procedures which contain step by step instructions.
  - Has side effects.
- Procedures can be called in any order, any number of times.
- Examples:
  - C, Fortran, Pascal, BASIC

Procedural vs Functional

- Functional languages are often Declarative.
- Functional programming treats programs as a series of math functions.
  - The output of a function is purely dependent on the input.
  - No side effects.
  - No state changing.
  - No changeable variables.
- Examples:
  - F#, Haskell, Hope

```
let double x = x * 2
let quadruple x = double(double(x))

quadruple 2
```

```
//Output
8
```

Other Paradigms

- Object Oriented is Procedural and Imperative.
  - Encapsulates functions and data into objects.
  - E.g. C++, Python, Java, C#, PHP.
- Logic Paradigm is often Functional and Declarative.
  - Based on formal logic.
  - E.g. Prolog, Datalog.
- Symbolic is often Functional and Declarative.
  - The program can modify its own formulas as if they were data.
  - E.g. Wolfram Language, LISP.

Further Reading

- Programming Paradigms
  - https://en.wikipedia.org/wiki/Programming_paradigm
- List of languages by paradigm
  - https://en.wikipedia.org/wiki/List_of_programming_languages_by_type
- Comparison of multi-paradigm languages
  - https://en.wikipedia.org/wiki/Comparison_of_multi-paradigm_programming_languages
- Comparison of paradigms
  - https://en.wikipedia.org/wiki/Comparison_of_programming_paradigms