20CYS312 - Principles of Programming Languages
Exploring Programming Paradigms

Assignment-01

Presented by Ram Surya Suresh Kumar
CB.EN.U4CYS21060
TIFAC-CORE in Cyber Security
Amrita Vishwa Vidyapeetham, Coimbatore Campus

Feb 2024

Outline

1. Meta-Programming
2. Groovy
3. Scripting
4. Shell
5. Comparison and Discussions
6. Comparison and Discussions
7. Bibliography

Meta-Programming

Metaprogramming is writing programs that operate on other programs and as well operate on themselves.

Programs operating on other programs: Writing:

- Compilers,
- Assemblers,
- Interpreters,
- Linkers,
- Loaders,
- Debuggers, and
- Profilers

Programs operating on themselves:

- Introspection - If a program simply looks at and reports on itself, we call this introspection .
- Reflection - If the program also modifies itself, we call this reflection

Groovy

In Groovy, it's possible to perform metaprogramming at both runtime and compile-time.

Run time programming:

Runtime metaprogramming enables us to alter the existing properties and methods of a class. Also, we can attach new properties and methods; all at runtime.

- *propertyMissing*
- methodMissing
- ExpandoMetaClass
- Extensions

Compile-time Metaprogramming:

Using specific annotations, we can effortlessly alter the class structure at compile-time. In other words, **we can use annotations to modify the abstract syntax tree of the class at the compilation**.

- @ToString
- @TupleConstructor
- @EqualsAndHashCode
- @Canonical
- @AutoClone
- Logging Support With *@Log*

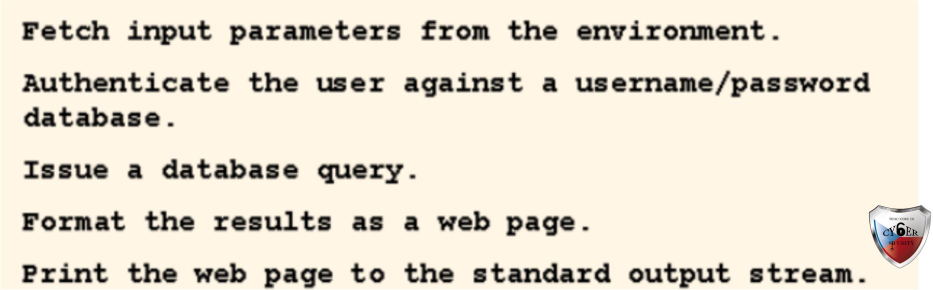
Scripting

Scripting is a programming paradigm that is characterized by the use of scripts, which are sequences of commands written in a scripting language. The scripting paradigm is often associated with interpreted languages and environments.

Some common characteristics and principles associated with the scripting paradigm:

- Interpreted Execution
- Dynamic Typing
- Rapid Development
- Scripting Languages

Example: Structure of a script in web development



```
Fetch input parameters from the environment.

Authenticate the user against a username/password
database.

Issue a database query.

Format the results as a web page.

Print the web page to the standard output stream.
```

Shell

A shell script is a text file that contains a sequence of commands for a UNIX-based operating system. It is called a shell script because it combines a sequence of commands, that would otherwise have to be typed into the keyboard one at a time, into a single script.

The shell is the operating system's command-line interface (CLI) and interpreter for the set of commands that are used to communicate with the system.

A shell script is usually created for command sequences in which a user has a need to use repeatedly in order to save time. Like other programs, the shell script can contain parameters, comments and subcommands that the shell must follow.

Examples of shell script applications:

- Automating the code compiling process.
- Running a program or creating a program environment.
- Completing batch
- Manipulating files.
- Linking existing programs together.
- Executing routine backups.
- Monitoring a system

Comparison and Discussions

Differences:

Execution Model:

▪ Metaprogramming: Often involves compile-time metaprogramming, where code modifications occur during the compilation phase. Runtime metaprogramming modifies code structures during program execution. ▪ Scripting: Generally involves interpreted execution, where scripts are executed line by line at runtime. There is typically no separate compilation step.

Use Cases:

▪ Metaprogramming: Commonly used in the development of libraries, frameworks, and code generation tools. It allows for the creation of flexible and generic solutions. ▪ Scripting: Commonly used for automation, system administration, quick prototyping, and tasks involving sequential execution of commands.

Language Features:

▪ Metaprogramming: Often involves language features like macros, template metaprogramming, or reflective capabilities that allow the manipulation of code structures. ▪ Scripting: Typically characterized by dynamic typing, high-level abstractions, and features that simplify common tasks, such as file I/O, string manipulation, and process execution.

Comparison and Discussions

Similarities:

1. Dynamic Nature: Both metaprogramming and scripting often involve dynamic features. Metaprogramming allows the modification of a program's structure at runtime, and scripting languages are often dynamically typed and interpreted.
2. Flexibility and Expressiveness: Both paradigms aim to provide flexibility and expressiveness to developers. Metaprogramming allows for code generation and modification, while scripting languages often have high-level abstractions and concise syntax for quick development.
3. Reflection: Both paradigms frequently make use of reflection. Metaprogramming often relies on reflective features to inspect and modify the program structure at runtime. Scripting languages may use reflection for dynamic runtime behavior

Bibiliography

https://www.baeldung.com/groovy-metaprogrammingcanonical
https://www.jdoodle.com/execute-groovy-online/
https://www.techtarget.com/searchdatacenter/definition/shell-script