

# PRG500 : Ruby Programming

**Code :** PRG500

**Duration :** 3 days

**Category :** Scripting

## Audience :

This course is intended for experienced developers who want to learn the Ruby scripting language.

## Prerequisites :

Knowledge and experience with programming in another programming language, such as C, C++, C#, Visual Basic, Java or Perl is required.

## Realization :

The theory is discussed on the basis of presentation slides and is interspersed with practical exercises. Illustrative demos provide further clarification of the concepts. The course material is in English.



**Ruby Programming**



## Contents :

In this course, participants learn to program in Ruby using the syntax and language constructs of that language. After an introduction about the background of Ruby, the installation and the way in which code can run, the variables, data types and control flow of Ruby are discussed. Attention is also paid to methods and parameter passing in Ruby and to object oriented aspects such as classes, objects, inheritance and polymorphism. The handling of errors through exception handling is discussed as well. The course continues with a discussion of partitioning code into modules, modules as namespaces and mixins as the use of modules from the standard library. Also closures such as blocks, lambdas and Procs are discussed and Ruby meta programming is treated with introspection, Open Classes and dynamic method invocation. Finally writing RubyGems is on the course schedule and an overview of the capabilities of the Ruby Standard Library is given.

### Module 1 : Ruby Intro

- What is Ruby?
- Ruby Timeline
- Ruby Object Orientation
- Ruby Installation
- Interactive Ruby
- Ruby Execution
- Loading Ruby Code
- Ruby Naming Conventions
- Executing External Programs
- Ruby Blocks
- Ruby Resources

### Module 2 : Variables and Data Types

- Numbers and Big Numbers
- Strings and String Literals
- String Interpolation
- Arrays
- Hash
- Range
- RegExp
- Struct
- Types of Variables
- Ruby Naming Conventions
- Constants
- Local Variables
- Instance Variables
- Class Variables
- Global Variables
- Pre-defined Variables

### Module 3 : Control Flow

- Statements
- Assignment
- Assignment operators
- Conditionals
- Multiple Selection
- while Loop
- until Loop
- for Loop
- each Iteration
- Aithmetic operators
- Comparison operators
- Ruby truth
- Equality
- Logical Operators

### Module 4 : Methods and Parameters

- Method Definitions
- Invoking Methods
- Methods and Parenthesis
- Return values
- Default value argument
- Variable Argument List
- Array Argument
- Hash Argument
- Methods with Code Block
- Method with Bang
- Aliasing Methods

### Module 5 : Classes and Objects

- Classes and Objects
- Example Class and Object
- Objects in Ruby
- Ruby Classes
- Class Syntax
- Creating Objects
- Object Initialization
- Getters and Setters
- Attribute Accessors
- Current Object
- Class Variables
- Class Methods
- Method Visibility
- Singleton Methods
- Inheritance
- Overriding
- Method Lookup
- Duck Typing
- Operator Overloading

### Module 6 : Exception Handling

- Error Handling
- Exception Handling
- Raising Exceptions
- Handling Exceptions
- Exception Class
- Exception Class Hierarchy
- Typed Exception Handling
- Ensure Block
- Retry Command
- Exception Handling
- Throw and Catch
- Raising Exceptions
- User Defined Exceptions

### Module 7 : Modules

- Modules
- Module Importing
- Files without Namespacing
- Modules for Namespacing
- Namespaces
- Mixins
- Mixin Example
- Include versus Extend
- Mixins and Inheritance Chain
- Modules versus Classes
- Comparable Module
- Enumerable Module

### Module 8 : Closures

- What are Closures?
- Benefits of Closures
- Lambdas
- Lambdas with Parameters
- Procs
- Proc and Lambda
- Procs versus Lambdas
- Proc as Argument and Return
- Proc Objects as Closure
- What are Blocks?
- Representing Blocks
- Calling Blocks with Yield
- Passing Arguments
- Ampersand Operator
- From Proc to Block

### Module 9 : Meta Programming

- What is Introspection?
- Introspection Code
- What is Meta Programming?
- Classes are Open
- Class Definition are Executable
- Receivers
- Classes are Objects
- Dynamic Method Invocation
- Method Missing
- Invoking method\_missing
- define\_method
- Hooks
- Evaluating Code
- Dynamic Typing

## **Module 10 : Ruby Gems**

- What are Gems?
- About RubyGems
- Creating Gems
- Hello World Gem
- gemspec file
- Installing and Using the Gem
- Publish the Gem
- Grabbing the Gem
- Adding Code
- Multiple Files
- Rake
- Rake Demo
- Writing Tests
- Simple Test
- Documenting Code
- Using Rdoc
- What is Bundler?
- Using Bundler

## **Module 11 : Ruby Standard Library**

- Standard Library Overview
- Files
- Accessing Files
- File Open Modes
- Reading and Writing
- Directories
- Date and Time
- XML Access
- DOM Parsing
- SAX Parsing
- MultiThreading