

**Introduction to Erlang Programming**

**Course Number:** ERL-100  
**Duration:** 3 days

**Overview**

This Introduction to Erlang training teaches attendees the fundamental concepts of Erlang, including its functional nature, concurrency, and error handling. This course helps attendees learn Erlang efficiently, stops them from making novice errors, ensures they are up to speed with the development workflow, and gets them thinking in the Erlang way.

**Prerequisites**

All students must have strong programming skills in another language.

**Materials**

All Erlang training students receive comprehensive courseware.

**Software Needed on Each Student PC**

* A complete, remote virtual environment is provided for training and is accessible via the Internet from any modern web browser.

**Objectives**

* Understand the basics of Erlang
* Read/write/design and observe Erlang programs
* Think concurrently and handle errors
* Leverage best development and design practices

**Outline**

* Introduction
* Why Erlang?
  + History of Erlang
  + Erlang and the BEAM
  + Who is using Erlang?
* The Shell, Types, and Constructs
  + The Shell
  + Data Types
  + Variables
  + Complex Data Structures
  + Pattern Matching
  + BIFs
  + Function Calls
  + Modules
* Sequential Programming
  + Conditional Evaluation
  + Defensive Programming and Guards
  + Recursion
* Sequential Error Handling
  + Run time errors
  + Try...catch, throw and catch
* Concurrent Erlang
  + Processes: Creating processes, registered processes and observer processes
  + Message passing, receiving messages and data in messages
* Process Design Patterns
  + Client Server Models
  + A Server Example
  + Finite State Machines
  + Event Managers
  + Supervisors
* Process Error Handling and Fault Tolerance
  + Links
  + Exit Signals
  + Definitions
  + Propagation Semantics
  + Monitors
  + Robust Systems
  + Error Handling Example
* Functional Programming Constructs
  + Funs
  + Higher Order Functions
  + List Comprehensions
* Maps and Records
  + Records
  + Records and the Shell
  + Maps
* Erlang Term Storage
  + ETS Tables
  + Handling Elements
  + Searching and Traversing
  + Match Specifications and Select
  + Other Issues
  + Observer Table Viewer
* Unit Testing
  + What is EUnit?
  + How to use EUnit
  + Testing Functional Code
  + Infrastructure
  + Fixtures
  + Testing Systems With State
  + Testing Concurrent Programs
* Good Design Practices: Style and Efficiency
  + Applications and Modules
  + Libraries
  + Return Values
  + Internal Data Structures
  + Concurrency
  + Conventions
* Conclusion