Elixir Programming

Course outline

Module 1: Introduction to Elixir

Module 1: Introduction to Elixir is an introductory course to the Elixir programming language. It covers the basics of the language, including data types, functions, and control structures. It also introduces the Elixir toolchain and provides an overview of the language's features and capabilities. This module is designed to give students a solid foundation in the language and prepare them for further exploration of Elixir.

Lessons

- Overview of Elixir and its Benefits
- Setting up an Elixir Development Environment
- Elixir Syntax Basics
- Working with Variables and Data Types
- Control Flow and Pattern Matching
- Working with Functions
- Working with Modules and Structs
- Working with Processes and Concurrency
- Working with Ecto and Databases
- Working with Phoenix and Web Applications

After completing this module, students will be able to:

- Understand the basic syntax and data types of Elixir
- Create and manipulate basic data structures such as lists, tuples, and maps
- Use pattern matching to deconstruct data structures
- Utilize the functional programming features of Elixir such as recursion, higher-order functions, and tail-call optimization

Module 2: Working with Data Types

Module 2 of the Elixir Programming course focuses on working with data types. Students will learn how to create and manipulate data types such as strings, integers, floats, and lists. They will also learn how to use pattern matching to extract data from data structures and how to use the pipe operator to chain functions together. Finally, students will learn how to use the Enum module to work with collections of data.

Lessons

- Introduction to Data Types in Elixir
- Working with Strings
- Working with Numbers
- Working with Atoms
- Working with Tuples
- Working with Lists
- Working with Maps
- Working with Binaries
- Working with Structs
- Working with Date and Time Types

After completing this module, students will be able to:

- Understand the different data types available in Elixir, such as integers, floats, strings, atoms, and tuples.
- Utilize pattern matching to extract data from data structures.
- Create and manipulate data structures such as lists, maps, and structs.
- Use the Enum and Stream modules to work with collections of data.

Module 3: Pattern Matching

Module 3 of the Elixir Programming course focuses on pattern matching, a powerful tool for writing concise and expressive code. Students will learn how to use pattern matching to deconstruct data structures, assign variables, and create powerful functions. They will also explore the use of guards and the pin operator to further refine their pattern matching skills.

Lessons

- Introduction to Pattern Matching
- Pattern Matching Syntax
- Pattern Matching with Variables
- Pattern Matching with Tuples
- Pattern Matching with Lists
- Pattern Matching with Maps
- Pattern Matching with Structs
- Pattern Matching with Binaries
- Pattern Matching with Regular Expressions
- Pattern Matching with Guards
- Pattern Matching with Function Heads
- Pattern Matching with Captures
- Pattern Matching with Conditional Clauses
- Pattern Matching with Multiple Clauses
- Pattern Matching with Macros
- Pattern Matching Performance Considerations

After completing this module, students will be able to:

- Understand the concept of pattern matching and how it is used in Elixir programming.
- Utilize pattern matching to create more concise and efficient code.
- Identify and use the different types of pattern matching in Elixir programming.
- Implement pattern matching to solve complex problems in Elixir programming.

Module 4: Control Flow

Module 4 of the Elixir Programming course covers the fundamentals of control flow in Elixir. It covers topics such as conditionals, loops, and pattern matching. It also covers how to use these concepts to create more complex programs. This module provides a solid foundation for students to build upon as they progress through the course.

Lessons

- Introduction to Control Flow in Elixir
- Conditional Statements in Elixir
- Pattern Matching in Elixir
- Loops and Iteration in Elixir
- Working with Recursion in Elixir
- Working with Guards in Elixir
- Working with Macros in Elixir
- Working with Exceptions in Elixir
- Working with Streams in Elixir
- Working with Tasks in Elixir

After completing this module, students will be able to:

- Understand the basic control flow structures in Elixir, such as if/else, case, and cond.
- Utilize pattern matching to control the flow of execution in Elixir programs.
- Implement recursion to solve problems in Elixir.
- Use guards to add additional conditions to control flow statements.

Module 5: Modules and Functions

Module 5 of the Elixir Programming course covers the fundamentals of modules and functions in Elixir. It covers topics such as creating and using modules, defining functions, and using pattern matching to create more powerful functions. It also covers topics such as guards, anonymous functions, and recursion. This module provides a solid foundation for writing efficient and maintainable Elixir code.

Lessons

- Introduction to Modules
- Defining and Using Functions
- Pattern Matching
- Anonymous Functions
- Recursion
- Capturing and Passing Functions

- Working with Modules
- Organizing Code with Modules
- Using Macros
- Working with Protocols

After completing this module, students will be able to:

- Understand the concept of modules and how to create them in Elixir
- Utilize functions to create reusable code
- Implement pattern matching to make code more concise and efficient
- Utilize guards to control the flow of execution in functions

Module 6: Working with Strings and Binaries

Module 6 of the Elixir Programming course focuses on working with strings and binaries. Students will learn how to manipulate strings and binaries, as well as how to use pattern matching to extract data from strings and binaries. Additionally, students will learn how to use the String module to perform various string operations, such as splitting, joining, and searching. Finally, students will learn how to use the binary module to encode and decode data.

Lessons

- Introduction to Strings and Binaries
- Working with Character Encodings
- Pattern Matching with Regular Expressions
- Working with Binary Data
- Manipulating Strings and Binaries
- Working with Unicode
- Performance Optimization for Strings and Binaries
- Working with Binary Files
- Working with Text Files
- Working with Binary Protocols

After completing this module, students will be able to:

- Understand the basic concepts of strings and binaries in Elixir.
- Manipulate strings and binaries using the various functions and operators available in Elixir.
- Create and use regular expressions to search and replace strings and binaries.
- Utilize the pattern matching capabilities of Elixir to work with strings and binaries.

Module 7: Working with Lists and Tuples

Module 7 of the Elixir Programming course covers working with lists and tuples. Students will learn how to create, manipulate, and use lists and tuples in their programs. They will also learn how to use pattern matching to extract data from lists and tuples, and how to use list comprehensions to create new lists. Finally, they will learn how to use the Enum module to perform common operations on lists and tuples.

Lessons

- Introduction to Lists and Tuples
- Creating and Manipulating Lists and Tuples
- Working with List Comprehensions
- Accessing Elements in Lists and Tuples
- Iterating over Lists and Tuples
- Sorting Lists and Tuples
- Merging and Splitting Lists and Tuples
- Pattern Matching with Lists and Tuples
- Working with Recursive Lists and Tuples
- Performance Considerations for Lists and Tuples

After completing this module, students will be able to:

- Create and manipulate lists and tuples in Elixir.
- Understand the differences between lists and tuples.
- Utilize pattern matching to access elements of lists and tuples.
- Use list and tuple functions to manipulate data.

Module 8: Working with Maps

Module 8 of the Elixir Programming course focuses on working with maps. It covers topics such as creating maps, accessing and updating values, merging maps, and using the Map module. It also covers pattern matching with maps and using the MapSet module. This module provides a comprehensive overview of working with maps in Elixir.

Lessons

- Introduction to Maps
- Creating and Accessing Maps
- Updating and Deleting Map Elements
- Iterating over Maps
- Pattern Matching with Maps
- Working with Map Keys
- Working with Map Values
- Merging Maps
- Comparing Maps
- Using Maps with Structs

After completing this module, students will be able to:

- Understand the fundamentals of working with maps in Elixir.
- Create and manipulate maps using the Elixir language.
- Utilize the various functions and operators available for working with maps.
- Implement maps in their own Elixir programs to store and retrieve data.

Module 9: Working with Structs

Module 9 of the Elixir Programming course focuses on working with Structs. Structs are a data type that allow you to store data in a structured way. This module will cover topics such as creating Structs, accessing and updating Structs, and using Structs in pattern matching. Additionally, this module will discuss the advantages and disadvantages of using Structs.

Lessons

- Introduction to Structs
- Creating Structs
- Accessing Structs
- Updating Structs
- Comparing Structs
- Pattern Matching with Structs
- Working with Maps and Structs
- Working with Lists and Structs
- Working with Tuples and Structs
- Working with Binaries and Structs
- Working with Keywords and Structs
- Working with Atoms and Structs
- Working with Functions and Structs
- Working with Modules and Structs
- Working with Processes and Structs
- Working with Exceptions and Structs
- Working with Macros and Structs
- Working with Protocols and Structs
- Working with Typespecs and Structs
- Working with Behaviours and Structs

After completing this module, students will be able to:

- Understand the concept of structs and how to create them in Elixir
- Utilize structs to store and access data in an efficient manner
- Create custom structs to represent complex data structures
- Leverage structs to create more organized and maintainable code

Module 10: Working with Processes

Module 10 of the Elixir Programming course focuses on working with processes. It covers topics such as spawning processes, sending and receiving messages, and linking processes. It also covers the basics of supervision trees and how to use them to manage processes. Finally, it covers the basics of distributed programming with Elixir.

Lessons

- Understanding Processes in Elixir
- Spawning and Linking Processes
- Managing Processes with Supervisors
- Working with Agents
- Using Tasks for Asynchronous Processing
- Working with GenServers
- Using GenStage for Data Flow
- Working with Streams
- Using Registry for Process Discovery
- Using Dynamic Supervisors for Fault Tolerance

After completing this module, students will be able to:

- Understand the fundamentals of Elixir processes and how they are used to build concurrent applications.
- Utilize the Elixir Process module to create and manage processes.
- Implement message passing between processes to communicate and coordinate tasks.
- Utilize the Supervisor module to create a fault-tolerant system.

Module 11: Working with Agents

Module 11 of the Elixir Programming course focuses on working with Agents. Agents are a powerful tool for managing state in Elixir, allowing you to store and update data in a safe and concurrent way. In this module, you will learn how to create and use Agents, as well as how to use them to manage state in your applications. You will also learn how to use Agents to coordinate tasks between processes, and how to use them to create distributed systems.

Lessons

- Understanding the Agent Model
- Creating Agents in Elixir
- Working with Agent State
- Updating Agents with Functions
- Using Agents for Asynchronous Processing
- Working with Agent Mailboxes
- Agent Supervision Strategies
- Agent Performance Optimization
- Integrating Agents with OTP
- Advanced Agent Techniques

After completing this module, students will be able to:

- Understand the concept of Agents and how to use them in Elixir programming.
- Create and manage Agent processes in Elixir.
- Utilize Agent functions to perform asynchronous tasks.
- Implement Agent-based solutions to solve complex problems.

Module 12: Working with ETS

Module 12 of the Elixir Programming course covers the use of the ETSmodule library, which provides an efficient way to store and retrieve data in Elixir. It covers topics such as creating and managing ETS tables, using the ETS API, and using ETS for distributed applications.

Lessons

- Introduction to ETS
- Understanding ETS Tables
- Working with ETS Tables
- Creating and Deleting ETS Tables
- Accessing ETS Tables
- Working with ETS Keys
- Working with ETS Values
- Using ETS for Performance Optimization
- Advanced ETS Techniques
- Troubleshooting ETS Issues

After completing this module, students will be able to:

- Understand the fundamentals of the Elixir programming language
- Utilize the ETS module to store and retrieve data from Elixir applications
- Create and manage ETS tables to store and retrieve data efficiently
- Implement ETS tables in Elixir applications to improve performance and scalability

Module 13: Working with OTP

Module 13 of the Elixir Programming course covers the use of OTP (Open Telecom Platform) for building robust, fault-tolerant applications. It covers topics such as the OTP principles, supervisors, applications, and gen_server. It also covers how to use OTP to build distributed applications and how to use OTP to monitor and debug applications.

Lessons

- Introduction to OTP
- Understanding the OTP Behaviour
- Building OTP Applications
- Working with OTP Supervisors
- Using OTP Gen Servers
- Working with OTP Gen Events
- Implementing OTP Gen FSM
- Working with OTP Gen Workers
- Using OTP Release Handling
- Understanding OTP Application Architecture

After completing this module, students will be able to:

- Understand the concept of OTP and its components
- Create and use OTP behaviours such as GenServer, Supervisor, and Application
- Implement fault-tolerance and self-healing strategies using OTP
- Utilize OTP to build distributed, fault-tolerant, and scalable applications

Module 14: Working with Phoenix

Module 14 of the Elixir Programming course covers the Phoenixmodule, a web development framework for Elixir. It covers topics such as creating a Phoenix application, routing, controllers, views, templates, and more. It also covers topics such as authentication, authorization, and testing. This module provides a comprehensive overview of the Phoenixmodule and its features, allowing students to create powerful web applications with Elixir.

Lessons

- Introduction to Phoenix
- Setting up a Phoenix Project
- Working with Controllers and Views
- Working with Models and Ecto
- Working with Routes and Endpoints
- Working with Templates and Layouts
- Working with Channels and LiveViews
- Working with Plugins and Addons
- Working with Authentication and Authorization
- Working with Testing and Debugging
- Deploying a Phoenix Application
- Integrating with External Services
- Performance Tuning and Optimization
- Best Practices for Phoenix Development

After completing this module, students will be able to:

- Understand the fundamentals of the Phoenix web framework and its components.
- Develop web applications using the Phoenix framework.
- Utilize the Ecto library to interact with databases.
- Implement authentication and authorization using the Guardian library.

Module 15: Working with Ecto

Module 15 of the Elixir Programming course focuses on working with Ectomodule, a library for managing data in Elixir applications. It covers topics such as creating and configuring repositories, defining schemas, and querying data. Additionally, students will learn how to use Ectomodule to interact with databases and other external services.

Lessons

- Introduction to Ecto
- Working with Ecto Queries
- Understanding Ecto Schemas
- Using Ecto Migrations
- Working with Ecto Associations
- Using Ecto Repositories
- Working with Ecto Changesets
- Debugging Ecto Queries
- Performance Tuning with Ecto
- Integrating Ecto with Phoenix Framework

After completing this module, students will be able to:

- Understand the fundamentals of Ecto and its role in Elixir programming.
- Create and manage databases using Ecto.
- Utilize Ecto to query and manipulate data.
- Implement Ecto in Elixir applications to interact with databases.

Module 16: Working with Mix

Module 16 of the Elixir Programming course covers the Mixmodule, a tool for managing Elixir projects. It covers topics such as creating and running projects, configuring tasks, and using Mix to manage dependencies. It also covers how to use Mix to create and run tests, as well as how to use Mix to deploy applications.

Lessons

- Understanding the Mix Module
- Creating and Managing Projects with Mix
- Compiling and Running Elixir Code with Mix
- Configuring Mix Tasks
- Using Mix to Manage Dependencies
- Working with Mix Releases
- Debugging Elixir Code with Mix
- Testing Elixir Code with Mix
- Deploying Elixir Applications with Mix
- Automating Elixir Development with Mix

After completing this module, students will be able to:

- Understand the concept of Mix and its purpose in Elixir programming.
- Create and configure Mix projects.
- Utilize Mix tasks to compile, test, and run Elixir projects.
- Use Mix to manage dependencies and generate release packages.

Module 17: Working with ExUnit

Module 17 of the Elixir Programming course covers the ExUnit testing framework. It provides an introduction to ExUnit, including how to write tests, run tests, and debug tests. It also covers how to use ExUnit to test Elixir code, as well as how to use ExUnit to test Phoenix applications.

Lessons

- Introduction to ExUnit
- Writing Tests with ExUnit
- Assertions and Expectations
- Using Tags and Filters
- Working with Setup and Teardown
- Working with Contexts
- Working with Mocks and Stubs
- Working with Doctests
- Working with Timeouts
- Working with Captured Output
- Working with Exceptions
- Working with Code Coverage
- Working with Custom Assertions
- Working with Custom Reporters
- Working with Custom Formats
- Working with Custom Filters
- Working with Custom Contexts
- Working with Custom Doctests
- Working with Custom Timeouts
- Working with Custom Captured Output

After completing this module, students will be able to:

- Understand the fundamentals of ExUnit and how to use it to test Elixir code.
- Create and run ExUnit tests for Elixir code.
- Utilize ExUnit's built-in assertions and custom assertions to test Elixir code.
- Debug and troubleshoot ExUnit tests.

Module 18: Working with Logging

Module 18 of the Elixir Programming course covers the basics of working with Loggingmodule, a library for logging messages in Elixir. It covers topics such as setting up Loggingmodule, configuring log levels, and using the Logger module to log messages. It also covers advanced topics such as customizing log messages and using the Logger module to log errors.

Lessons

- Introduction to Logging in Elixir
- Configuring Logging in Elixir
- Understanding Logging Levels in Elixir
- Using Loggers in Elixir

- Logging to Files in Elixir
- Logging to Databases in Elixir
- Logging to External Services in Elixir
- Logging Performance Metrics in Elixir
- Logging Exceptions in Elixir
- Logging Best Practices in Elixir

After completing this module, students will be able to:

- Understand the different types of logging available in Elixir and how to use them.
- Implement logging in Elixir applications to track and debug errors.
- Utilize the Elixir Logger library to create custom loggers and format log messages.
- Analyze log data to identify and troubleshoot application issues.

Module 19: Working with Plug

Module 19 of the Elixir Programming course covers the use of Plugmodule, a library for building composable web applications in Elixir. It covers topics such as creating a Plug, using Plug.Conn, and using Plug.Router to create routes. It also covers how to use Plug with Phoenix and Ecto.

Lessons

- Overview of Plug and its Benefits
- Writing a Basic Plug
- Working with Plug Parameters
- Using Plug Adapters
- Working with Plug Routers
- Writing Custom Plug Middleware
- Debugging Plug Applications
- Testing Plug Applications
- Deploying Plug Applications
- Integrating Plug with Phoenix Framework

After completing this module, students will be able to:

- Understand the purpose and usage of the Plug module in Elixir programming.
- Create custom Plug modules to extend the functionality of an Elixir application.
- Utilize the Plug module to create web applications and APIs.
- Implement authentication and authorization using the Plug module.

Module 20: Working with Nerves

Module 20 of the Elixir Programming course focuses on working with Nerves, a framework for building embedded systems with Elixir. Students will learn how to create and configure a Nerves project, how to use the Nerves toolchain, and how to deploy and debug their applications. Additionally, students will explore the Nerves ecosystem and learn how to use the NervesHub platform to manage their devices.

Lessons

- Introduction to Nerves
- Setting up a Nerves Project
- Working with GPIOs and I2C
- Working with UARTs and SPI
- Working with Networking Protocols
- Working with System Services
- Working with Firmware Updates
- Working with Embedded Linux
- Working with Device Drivers
- Working with Embedded Controllers

After completing this module, students will be able to:

- Understand the fundamentals of the Nerves framework and its components.
- Develop and deploy applications to embedded devices using Nerves.
- Utilize the Nerves toolchain to configure and manage embedded devices.
- Create custom Nerves images and packages for specific applications.