

Groovy Programming Language

Duration : 5 days

Groovy Getting Started

- Download Groovy
- Setting up Groovy
- Installing Groovy
- groovyc — the Groovy compiler
- groovysh — the Groovy command -like shell
- groovyConsole — the Groovy Swing console
- IDE integration

Groovy Big Picture

- What is Groovy
- Groovy vs Java Features
- Why Groovy?
- Groovy language fundamentals
- Variables and Types
- Static and dynamic typing
- HelloWorld Groovy Program
- Compiling using groovy compiler
- Running Groovy Program

Groovy Language Spec

Syntax

- Comments
 - Single line comment
 - Multiline comment
 - GroovyDoc comment
 - Shebang line
- Keywords
- Identifiers
 - Normal identifiers
 - Quoted identifiers
- Strings
 - Single quoted string
 - String concatenation

- Triple single quoted string
 - Escaping special characters
 - Unicode escape sequence
- Double quoted string
 - String interpolation
 - Special case of interpolating closure expressions
 - Interoperability with Java
 - GString and String hashCodes
- Triple double quoted string
- Slashy string
- Dollar slashy string
- String summary table
- Characters
- Numbers
 - Integral literals
 - Alternative non-base 10 representations
- Binary literal
- Octal literal
- Hexadecimal literal
 - Decimal literals
 - Underscore in literals
 - Number type suffixes
 - Math operations
 - The case of the division operator
 - The case of the power operator
- Booleans
- Lists
- Arrays
- Maps

Operators

- Arithmetic operators
- Normal arithmetic operators
- Unary operators
- Assignment arithmetic operators
- Relational operators
- Logical operators
- Precedence
- Short-circuiting
- Bitwise operators
- Conditional operators
- Not operator
- Ternary operator
- Elvis operator
- Object operators

- Safe navigation operator
- Direct field access operator
- Method pointer operator
- Regular expression operators
- Pattern operator
- Find operator
- Match operator
- Other operators
- Spread operator
- Spreading method arguments
- Spread list elements
- Spread map elements
- Range operator
- Spaceship operator
- Subscript operator
- Membership operator
- Identity operator
- Coercion operator
- Diamond operator
- Call operator
- Operator precedence
- Operator overloading

Program Structure

- Package names
- Imports
- Default imports
- Simple import
- Star import
- Static import
- Static import aliasing
- Static star import
- Import aliasing
- Scripts versus classes
- public static void main vs script
- Script class
- Methods
- Variables

Object Orient nation and Implementation in Groovy

- Types
 - Primitive types
 - Class

- Normal class
 - Inner class
- Anonymous inner class
 - Abstract class
 - Interface
 - Constructors
 - Positional argument constructor
 - Named argument constructor
 - Methods
 - Method definition
 - Named arguments
 - Default arguments
 - Varargs
 - Method selection algorithm
 - Exception declaration
 - Fields and properties
 - Fields
 - Properties
 - Annotation
 - Annotation definition
 - Annotation placement
 - Annotation member values
 - Retention policy
 - Closure annotation parameters
 - Meta-annotations
- Declaring meta-annotations
- Behavior of meta-annotations
- Meta-annotation parameters
- Custom annotation processors
 - Inheritance
 - Generics
- Traits
 - Methods
 - Public methods
 - Abstract methods
 - Private methods
 - The meaning of this
 - Interfaces
 - Properties
 - Fields

- Private fields
- Public fields
- Composition of behaviors
- Overriding default methods
- Extending traits
 - Simple inheritance
 - Multiple inheritance
- Duck typing and traits
 - Dynamic code
 - Dynamic methods in a trait
- Multiple inheritance conflicts
 - Default conflict resolution
 - User conflict resolution
- Runtime implementation of traits
 - Implementing a trait at runtime
 - Implementing multiple traits at once
- Chaining behavior
 - Semantics of super inside a trait
- Advanced features
 - SAM type coercion
 - Differences with Java 8 default methods
- Differences with mixins
- Static methods, properties and fields
- Inheritance of state gotchas
- Self types
 - Type constraints on traits
 - The @SelfType annotation
- Limitations
 - Compatibility with AST transformations
 - Prefix and postfix operations

Closures

- Syntax
 - Defining a closure
 - Closures as an object
 - Calling a closure
- Parameters
 - Normal parameters
 - Implicit parameter
 - Varargs

- Delegation strategy
 - Groovy closures vs lambda expressions
 - Owner, delegate and this
 - The meaning of this
 - Owner of a closure
 - Delegate of a closure
 - Delegation strategy
- Closures in GStrings
- Closure coercion
- Functional programming
 - Currying
 - Left currying
 - Right currying
 - Index based currying
 - Memoization
 - Composition
 - Trampoline
 - Method pointers

Semantics

- Statements
 - Variable definition
 - Variable assignment
 - Multiple assignment
 - Overflow and Underflow
 - Object destructuring with multiple assignment
 - Control structures
 - Conditional structures
- if / else
- switch / case
 - Looping structures
- Classic for loop
- for in loop
- while loop
 - Exception handling
 - try / catch / finally
 - Multi-catch
 - Power assertion
 - Labeled statements
- Expressions

- GPath expressions
 - Object navigation
 - Expression Deconstruction
 - GPath for XML navigation
- Promotion and coercion
 - Number promotion
 - Closure to type coercion
 - Assigning a closure to a SAM type
 - Calling a method accepting a SAM type with a closure
 - Closure to arbitrary type coercion
 - Map to type coercion
 - String to enum coercion
 - Custom type coercion
 - Class literals vs variables and the as operator
- Optionality
 - Optional parentheses
 - Optional semicolons
 - Optional return keyword
 - Optional public keyword
- The Groovy Truth
 - Boolean expressions
 - Collections and Arrays
 - Matchers
 - Iterators and Enumerations
 - Maps
 - Strings
 - Numbers
 - Object References
 - Customizing the truth with asBoolean() methods
- Typing
 - Optional typing
 - Static type checking
 - The @TypeChecked annotation
- Activating type checking at compile time
- Skipping sections
 - Type checking assignments
 - List and map constructors
 - Method resolution
 - Type inference
- Principles

- Variables vs fields in type inference
- Collection literal type inference
- Least upper bound
- instanceof inference
- Flow typing
- Advanced type inference
 - Closures and type inference
- Return type inference
- Parameter type inference
- Explicit closure parameters
- Parameters inferred from single-abstract method types
- The @ClosureParams annotation
- @DelegatesTo
 - Static compilation
 - Dynamic vs static
 - The @CompileStatic annotation
 - Key benefits
- Type checking extensions
 - Writing a type checking extension
 - Towards a smarter type checker
 - The extensions attribute
 - A DSL for type checking
 - Type checking extensions API
- AST
- Events
 - Working with extensions
- Support classes
- Class nodes
- Helping the type checker
- Throwing an error
- isXXXExpression
- Virtual methods
- Scoping
- Other useful methods
 - Advanced type checking extensions
 - Precompiled type checking extensions
 - Using @Grab in a type checking extension
 - Sharing or packaging type checking extensions
 - Global type checking extensions

- Type checking extensions and @CompileStatic
- Mixed mode compilation
- Transforming the AST in an extension

Runtime and compile-time metaprogramming

- Runtime metaprogramming
- Compile-time metaprogramming
- The Grape dependency manager
- Unit Testing in Groovy

DSL-Domain Specific languages

- Command Chains
- Operator Overloading
- Script base classes
- Adding Properties to Numbers
- @DelegatesTo
- Compilation customizers
- Custom type checking extensions
- Builders

Exception handling try / catch / finally

- Multi-catch
- Power assertion
- Labeled statements
- Expressions
- GPath expressions
- Object navigation
- Expression Deconstruction
- GPath for XML navigation
- Promotion and coercion
- Number promotion
- Closure to type coercion
- Assigning a closure to a SAM type
- Calling a method accepting a SAM type with a closure