

Ruby & ROR Course Curriculum

Part I

Ruby Language Skills and Techniques

- a) **Running Ruby**
 - Command-line Ruby
 - The Interactive Ruby (irb) console
 - Objects, variables, and methods
- b) **Basic object-orientation concepts**
 - Naming conventions
 - Variable assignment
 - Local vs. instance variables
 - Life of different variables in methods
 - Method-calling semantics
- c) **Method arguments and return values**
- d) **Classes and modules**
 - Instantiating classes
 - Polymorphism in ruby
 - Mixing in modules
- e) **Built-in classes**
 - String
 - Array
 - Hash
 - Symbol
 - Numerics
- f) **Blocks and iterators**
 - Blocks vs. methods vs. lambdas
 - Collection iteration
 - Single-object iteration
- g) **Exceptions**
 - Built-in exceptions
 - Writing your own exception classes
 - Exceptions in Rails

Part II

ACTION CONTROLLER

- a) **Controller Actions and View Templates**
- b) **Basics of controller/view interaction**
 - Creating controllers and view with "generate"
 - Default rendering rules
 - Shared controller/template instance variables
 - Separating controller logic from view specifics
- c) **HTML templating with Embedded Ruby (ERB)**
 - Rules of ERb processing
 - Layouts
 - Using master and partial templates
- d) **Fine-tuning controllers and view**
- e) **Controller filters**
- f) **Redirecting control**
- g) **Default and custom view helper methods**
- h) **Writing and processing HTML forms in Rails**
 - Using form helper methods
 - "Magic" field initialization from instance variables
 - Accessing CGI data through the "params" hash
- i) **Ajax calls and RJS (Ruby/JavaScript) templates**
 - Basic DOM updating with Ajax
 - Using RJS for composite Ajax calls
 - Render different formats
- j) **Routing**
 - Purpose of routes
 - Writing routes for the rails app
 - Types of routes and usage

ACTIVE RECORD

a) Model Design and Database Management

b) Domain modeling for Rails and Ruby

c) Describing the "what" of the application

d) Creating ActiveRecord models with "generate"

- Object-relational mapping with ActiveRecord
- ActiveRecord models and Ruby classes

e) ActiveRecord associations

- The Association Hierarchy
- One-to-Many Relationships
- The belongs to Association
- The has many Association
- Many-to-Many Relationships
- Has and belongs to many
- Has many through
- One-to-One Associations

f) Migrations

- Creating Migrations
- Naming Migrations
- Migration Pitfalls
- Migration API

g) Create table(name, options)

- Defining Columns
- Column Type Mappings
- Column Options
- Decimal Precision
- Column Type Gotchas
- Custom Data Types
- "Magic" Timestamp Columns
- Macro-Style Methods
- Relationship Declarations

h) Convention over Configuration

- Pluralization
- Setting Names Manually
- Legacy Naming Schemes

i) Defining Attributes

- Default Attribute Values
- Serialized Attributes

- Logging
- Default Query Caching in Controllers

j) Updating

- Updating by Condition
- Updating a Particular Instance
- Updating Specific Attributes
- Convenience Updaters
- Controlling Access to Attributes
- Deleting and Destroying

k) Session Management in Rails

l) Advanced Finding

- Conditions
- Ordering of Find Results
- Limit and Offset
- Select Option
- From Option
- Group By Option
- Locking Option
- Joining and Including Associations
- Read Only

m) Validations

- The Simple Declarative Validations
- Validates acceptance of
- Error Message
- The accept Option
- Validates associated
- Validates confirmation of
- Validates each
- Validates inclusion of and validates exclusion of
- Validates existence of
- Validates format of
- Validates length of
- Validates numericality of
- Validates presence of

HELPERS

- ActiveRecordHelper
- Reporting Validation Errors
- Error message on(object, method, prepend text , append text , css class "formError")
- Error messages for (params)
- Automatic Form Creation Form (name, options)
- Input (name, method, options)

- Customizing the Way Validation Errors Are Highlighted
- AssetTagHelper
- Head Helpers
- Auto discovery link tag(type = rss, urloptions = {}, tagoptions = {})
- Image path(source)
- Image tag(source, options = {})
- Javascript include tag (sources)
- Javascript path(source)
- Stylesheet link tag(sources)
- Stylesheet path(source)
- Assert the MIME Content Type of a Response (or Other Header Values)
- Assert Rendering of a Particular Template
- Assert Redirection to a Specified URL
- Assert Setting of Flash Messages
- Assert Database Changes Resulting from Actions
- Assert Validity of a Model
- Asserting View Output

ACTION MAILER BASIC

❖ Testing in Rails

a) Purpose of testing

b) Fixtures

- Accessing Fixture Records from Tests
- Dynamic Fixture Data
- Using Fixture Data in Development Mode
- Generating Fixtures from Development Data
- Fixtures Options

❖ Unit Testing

a) Assertions

- Basic Assertions
- Assert and deny
- Assert block
- Assert empty
- Assert equal and assert not equal
- Assert in delta and assert in epsilon
- Assert include
- Assert instance of
- Assert kind of
- Assert match and assert no match
- Assert nil and assert not nil
- Assert same and assert not same
- Assert raise and assert nothing raised
- Assert respond to

b) Functional testing

- Assert That Variables Were Assigned Properly for Use by Templates
- Assert the HTTP Status Code of a Response and Its MIME Content Type

c) Rails Integration Tests

- The Integration Test API
- Assert redirected to(options = {}, message = nil)
- Assert response(type, message = nil)
- Assert template(expected = nil, message = nil)