Ruby Cheat Sheet

1. Basic Syntax

Comments

- **Single-line Comment**: `# This is a comment`

- **Multi-line Comment**:

```
=begin
This is a
multi-line comment
=end
```

Basic Operators

- **Arithmetic**: `+`, `-`, `*`, `/`, `%`, `**`

- **Comparison**: `==`, `!=`, `>`, `<`, `>=`, `<=`, `<=>`

- **Logical**: `&&`, `||`, `!`

- **Assignment**: `=`, `+=`, `-=`, `*=`, `/=`, `%=`, `**=`

String Interpolation

- **Interpolation**: `"Hello, #{name}"`

---

2. Data Types

Numeric Types

- **Integer**: `123`, `-456`

- **Float**: `123.456`, `-456.789`

- **Complex**: `1+2i`

Boolean

- **TrueClass**: `true`

- **FalseClass**: `false`

Strings

- **Single-quoted**: `'Hello'`

- **Double-quoted**: `"Hello"`

- **Multiline**: `<<-HEREDOC ... HEREDOC`

Symbols
- **Immutable**: `:symbol`, `:"symbol with spaces"`

Arrays
- **Ordered Collection**: `[1, 2, 3]`, `%w(apple banana cherry)`

Hashes
- **Key-Value Pairs**: `{ key: "value" }`, `{ "key" => "value" }`

Ranges
- **Inclusive**: `1..10`

- **Exclusive**: `1...10`

Nil
- **Represents Nothing**: `nil`

---

3. Variables and Constants

Variables
- **Local**: `variable_name`

- **Instance**: `@variable_name`

- **Class**: `@@variable_name`

- **Global**: `$variable_name`

Constants
- **Uppercase**: `CONSTANT_NAME`

---

4. Control Structures

Conditional Statements
- **If-Else**:

```
if condition
  # code
elsif another_condition
  # code
```

```
else
  # code
end
```

- **Ternary**: `condition ? true_value : false_value`
- **Case**:

```
case variable
when value1
  # code
when value2
  # code
else
  # code
end
```

Loops

- **While**:

```
while condition
  # code
end
```

- **Until**:

```
until condition
  # code
end
```

- **For**:

```
for i in 1..10
  # code
end
```

- **Each**:

```
(1..10).each do |i|
  # code
end
```

- **Times**:

```
10.times do |i|
  # code
end
```

Break, Next, and Redo

- **Break**: Exit the loop.

- **Next**: Skip to the next iteration.

- **Redo**: Repeat the current iteration.

---

5. Methods and Blocks

Defining Methods

- **Basic**:

```
def method_name(parameter)
  # code
end
```

- **Default Parameters**:

```
def method_name(parameter = default_value)
  # code
end
```

- **Return Value**: `return value`

Blocks

- **Single-line**: `do_something { |param| # code }`

- **Multi-line**:

```
do_something do |param|
  # code
end
```

Lambdas and Procs

- **Lambda**:

```
my_lambda = ->(param) { # code }
```

- **Proc**:

```
my_proc = Proc.new { |param| # code }
```

---

6. Classes and Objects

Defining Classes

- **Basic**:

```
class MyClass
  def initialize(parameter)
    @instance_variable = parameter
  end
end
```

Instance Methods

- **Defining**:

```
def instance_method
  # code
end
```

- **Calling**: `object.instance_method`

Class Methods

- **Defining**:

```
def self.class_method
  # code
end
```

- **Calling**: `MyClass.class_method`

Accessors

- **Getter**: `attr_reader :variable`
- **Setter**: `attr_writer :variable`
- **Both**: `attr_accessor :variable`

Inheritance

- **Basic**:

```
class ChildClass < ParentClass
  # code
end
```

Modules

- **Include**: `include ModuleName`
- **Extend**: `extend ModuleName`

---

7. Modules and Mixins

Defining Modules

- **Basic**:

```
module MyModule
  def my_method
    # code
  end
end
```

Including Modules

- **In Class**:

```
class MyClass
  include MyModule
end
```

Extending Modules

- **In Object**:

```
object.extend(MyModule)
```

---

8. Collections

Arrays
- **Creation**: `array = [1, 2, 3]`

- **Accessing**: `array[0]`, `array[-1]`

- **Methods**: `push`, `pop`, `shift`, `unshift`, `map`, `select`, `reject`, `reduce`

Hashes
- **Creation**: `hash = { key: "value" }`

- **Accessing**: `hash[:key]`

- **Methods**: `keys`, `values`, `each`, `merge`, `delete`

Ranges
- **Creation**: `range = 1..10`

- **Methods**: `to_a`, `include?`, `each`

---

9. File Handling

Reading Files
- **Basic**:

```
File.read("filename.txt")
```

- **Line by Line**:

```
File.foreach("filename.txt") do |line|
  # code
end
```

Writing Files
- **Basic**:

```
File.write("filename.txt", "content")
```

- **Append**:

```
File.open("filename.txt", "a") do |file|
  file.write("content")
end
```

File Operations

- **Exists?**: `File.exist?("filename.txt")`

- **Delete**: `File.delete("filename.txt")`

---

10. Error Handling

Basic Structure

- **Begin-Rescue**:

```
begin
  # code
rescue StandardError => e
  # error handling
end
```

Custom Exceptions

- **Defining**:

```
class MyError < StandardError; end
```

- **Raising**: `raise MyError, "message"`

Ensure

- **Always Execute**:

```
begin
  # code
rescue StandardError => e
  # error handling
ensure
  # always execute
end
```

---

11. Metaprogramming

Dynamic Methods

- **Define Method**:

```
define_method(:method_name) do |param|
  # code
end
```

Method Missing

- **Override**:

```
def method_missing(method_name, *args)
  # code
end
```

Singleton Methods

- **Defining**:

```
def object.method_name
  # code
end
```

Class Macros

- **Attribute Methods**: `attr_accessor`, `attr_reader`, `attr_writer`

---

12. Ruby Gems

Installing Gems

- **Basic**: `gem install gem_name`

Bundler

- **Gemfile**:

```
source 'https://rubygems.org'
gem 'rails'
```

- **Install**: `bundle install`

Gem Commands

- **List**: `gem list`

- **Uninstall**: `gem uninstall gem_name`

---

13. Tips and Tricks

Shortcuts

- **Multi-line String**: `<<-HEREDOC ... HEREDOC`

- **Symbol Array**: `%i(symbol1 symbol2)`

- **String Array**: `%w(word1 word2)`

Debugging

- **Pry**: `require 'pry'; binding.pry`

- **Rails Console**: `rails console`

Performance

- **Benchmark**:

```
require 'benchmark'
Benchmark.bm do |x|
  x.report { # code }
end
```

Documentation

- **RDoc**: `rdoc filename.rb`

- **YARD**: `yard doc filename.rb`

This cheat sheet provides a comprehensive overview of Ruby, covering essential syntax, data types, control structures, methods, classes, modules, collections, file handling, error handling, metaprogramming, Ruby gems, and useful tips and tricks. Use this as a quick reference to master Ruby programming.

By Ahmed Baheeg Khorshid

ver 1.0