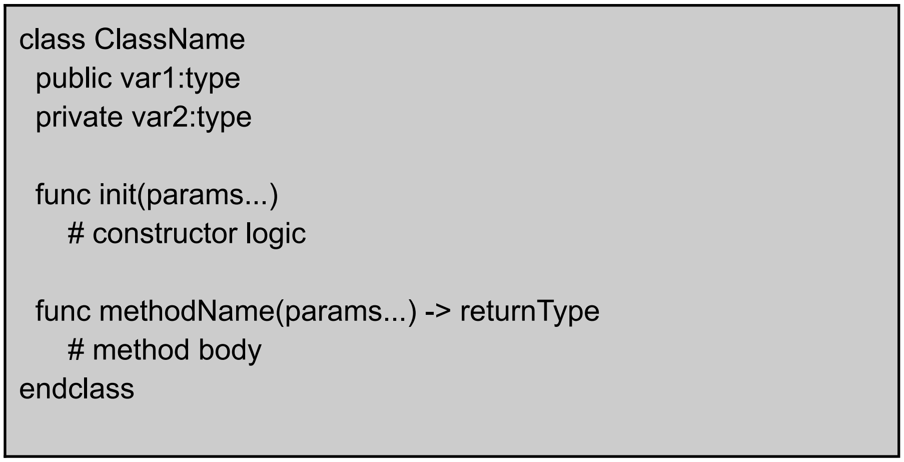
RCBScript — Object-Oriented Programming (OOP) Specification

This document defines the **Object-Oriented Features** of RCBScript as per the tasks.

---

1) Class Declaration Syntax

Syntax:



```
class ClassName
  public var1:type
  private var2:type

  func init(params...)
      # constructor logic

  func methodName(params...) -> returnType
      # method body
endclass
```

- **class** keyword starts the definition.
- Attributes can be **public** or **private**.
- **func init(...)** is treated as a constructor.
- Methods are declared inside the class.
- **endclass** closes the definition.

2) Object Creation and Destruction

Objects are created and destroyed using **make** and **discard**:

```
# Create object
p = make Person("Alice", 30)

# Method invocation
result(p.greet())

# Destroy object
discard p
```

- **make ClassName(args...)** allocates and initializes an object.
- **discard obj** deallocates the object and frees resources.

3) Records (Immutable Structures)

Records are lightweight, immutable structures for holding data.

Syntax:

```
record RecordName
  field1:type
  field2:type
endrecord

pt = Point(10, 20)
```

```
result("x = " + pt.x)
```

- Records do not support methods.
- Fields are immutable once set.

---

4) Example Programs

Example 1: Person Class

```
class Person
  public name:text
  private age:number

  func init(n:text, a:number)
    name = n
    age = a

  func greet() -> text
    reply "Hello, my name is " + name
endclass

p = make Person("Alice", 30)
result(p.greet())
discard p
```

Example 2: Hello Class

```
class Hello
  func sayHello(name:text) -> text
    reply "Hello, " + name + "!"
endclass
```

```
h = make Hello()
result(h.sayHello("RCBScript"))
discard h
```

Example 3: Record Example

```
record Point
  x:number
  y:number
endrecord

pt = Point(5, 10)
result("Coordinates: (" + pt.x + ", " + pt.y + ")")
```

---

5) Deliverables

- **OOP Specification Document** (this document).
- **Example Programs:**
  - `Person` class
  - `Hello` class
  - `Point` record

These serve as references for implementing object-oriented features in RCBScript.