
Basic-Ruby-Style-Guide

Semi colons ;

Avoid them unless writing multiple statements per line

Example:

```
1      # Good
2      def needs_cursor?; true; end
3
4      # Bad
5      name = gets.chomp;
6      puts("hi #{name}");
7
8      # Even worse - uses inconsistently
9      name = gets.chomp
10     puts("hi #{name}");
```

Variables

Names

Must be in the snake_case format.

Example:

```
1      #Good
2      name = "paul"
3      first_name = "paul"
4
5      #Bad
6      Name = "paul"
7      FamilyName = "Sarda"
8      Family_Name = "Sarda"
```

Operators

Must have a space on both sides of the operator always.

Example:

```
1      # Good
2      paul_is_cool = true
3
4      # Bad
5      paul_is_not_cool =false
6      paul_is_not_cool=false
7      paul_is_not_cool= false
8
9      # Good
10     ten = 5 + 5
11
12     # Bad
13     ten = 5+ 5
```

functions

Names

Function names should use snake_case.

Examples:

```
1      # Good
2      def return_paul()
3          return "paul"
4      end
5
6      # Bad - PascalCase
7
8      def returnPaul()
9          return "paul"
10     end
11
12     # Also Bad - camelCase
13     def ReturnPaul()
14         return "paul"
15     end
```

Parentheses (round brackets)

Calls

Use parentheses whenever there is more one or more arguments. Or always use it.

Examples:

```
1      # Good
2      puts("Hi my name is paul") # needed
3      name = gets.chomp() # not needed no arguments
4
5      puts("Hi my name is paul")
6      name = gets.chomp()
7
8      # Bad
9      # inconsistent
10     name = gets.chomp()
11     last_name = gets.chomp
12
13     # Missing parentheses
14     puts "Hi my name is paul"
```

No spaces between (and the first argument. No spaces between) and the last argument.

Examples:

```
1      # Good
2      puts("paul is so cool")
3
4      # Bad
5      puts( "paul is not cool" )
```

Definitions

Add parentheses if there is one or more arguments.

Example:

```
1      def sum(a, b) # needed
2          return (a + b)
3      end
4
```

```
5   def return_paul # no brackets needed
6       return "paul"
7   end
```

Return

The return keyword must always be used if a function returns a value.

Example:

```
1   # Good
2   def return_paul()
3       return "Paul"
4   end
5
6   # Good because the function doesn't need to return anything
7   def print_paul()
8       puts("Paul")
9   end
10
11  # Bad
12  # missing return
13  def return_paul
14      "paul"
15  end
```

If

Don't use Parentheses or use them not both

(use is preferred) Example:

```
1   # Good
2   if name === "paul"
3       puts("Woah you are so cool")
4   end
5
6   if course == "BA-CS"
7       puts("Good choice")
8   end
9
```

```
10   # Good
11   if (name === "paul")
12     puts("Woah you are so cool")
13   end
14
15   if (course == "BA-CS")
16     puts("Good choice")
17   end
18
19   #Bad becuase inconsistent
20   if (name === "paul")
21     puts("Woah you are so cool")
22   end
23
24   if course == "BA-CS"
25     puts("Good choice")
26   end
```

Always have a space between if and the condition.

Example:

```
1   # Good
2   if (name === "paul")
3     puts("Woah you are so cool")
4   end
5
6   # Bad
7   if(name === "paul")
8     puts("Woah you are so cool")
9   end
```

Case

Each when should be the same level of indentation as case

Example:

```
1   # Good
2   case name
3   when "paul"
4     puts("Old name");
```

```
5   when "andrew"
6       puts("Okay name");
7   when "john"
8       puts("Great name");
9   else
10      puts("bad name");
11  end
12
13  # Bad
14  case name
15      when "paul"
16          puts("Old name");
17      when "andrew"
18          puts("Okay name");
19      when "john"
20          puts("Great name");
21      else
22          puts("bad name");
23  end
24
25  case name
26      when "paul"
27          puts("Old name");
28      when "andrew"
29          puts("Okay name");
30      when "john"
31          puts("Great name");
32      else
33          puts("bad name");
34  end
```

Loops

While

Same condition rules as if.

Example:

```
1   while (i <= 50) do
2       puts("value of i is #{i}");
```

```
3         i += 1;
4     end
```

For

Example:

```
1     for i in 1..50 do
2         puts("value of i is #{i}");
3     end
```

Array Example:

```
1     dice_rolls = [1, 3, 3, 1];
2
3     for roll in dice_rolls do
4         puts("#{roll}");
5     end
```

Indentation

Everything between the end and what the end is ending must be indented

Examples:

```
1     # Good
2     def its_paul()
3         puts("REALLY?")
4     end
5
6     def main()
7         name = "paul"
8
9         if name === "paul"
10             its_paul()
11         end
12     end
13
14     # BAD
15         def its_paul
```

```
16     puts("REALLY?")
17     end
18
19     def main
20     name = "paul"
21
22     if name === "paul"
23         its_paul
24     end
25     end
26
27     # Also BAD
28
29     def its_paul
30     puts("REALLY?")
31     end
32
33     def main
34     name = "paul"
35
36     if name === "paul"
37     its_paul
38     end
39     end
40
41     # Also BAD
42
43     def its_paul
44     puts("REALLY?")
45     end
46
47     def main
48     name = "paul"
49
50     if name === "paul"
51     its_paul
52     end
53     end
```

Structures / Records / Classes

Name

Must be in PascalCase meaning that every word should be capitalised and no spaces.

Members must be in snake_case

Example:

```
1  class Person
2      attr_accessor :name, :birth_year
3  end
4
5  class EvilTeacher
6      attr_accessor :subject, :year_started
7  end
```

Array

All normal naming rules apply

Example:

```
1  numbers = [];
2
3  numbers << 10;
4
5  puts(numbers[0]);
```