

10 raisons pour lesquelles j'adore R en tant que développeur Python

Joseph Barbier¹

Résumé

Le langage R possède un grand nombre de particularités dans son fonctionnement qui semblent banales ou sont peu connus, alors qu'elles sont en fait particulièrement agréables à utiliser, notamment du point de vue d'un développeur Python.

Dans cette présentation, j'exposerai une liste de 10 exemples de choses que R permet et qui me font adorer le langage R, et que je souhaite encourager les gens à utiliser davantage.

On parlera de méthodes S3, de la fonction `substitute()`, de l'opérateur pipe, de contrôle de l'état global, de lazy evaluation ou encore de retours invisibles. Pleins de choses qui me paraissaient bizarres en venant de Python, mais qui sont en fait géniales.

Mots-clefs (3 à 5) : Python - Syntaxe - Lazy - Environnement - Pipe

Développement

Créer ses propres opérateurs

```
`%within%` <- function(x, range) {  
  x >= range[1] & x <= range[2]  
}  
  
5 %within% c(1, 10)
```

```
[1] TRUE
```

```
12 %within% c(1, 10)
```

```
[1] FALSE
```

Surcharge d'opérateurs

¹Yellow Sunflower, joseph@ysunflower.com

```

# Crée un objet avec une unité
with_unit <- function(x, unit) {
  structure(x, unit = unit, class = "unit_val")
}

# Conversion interne pour harmoniser les unités
convert_to <- function(x, target_unit) {
  if (attr(x, "unit") == target_unit) {
    return(x)
  }

  value <- unclass(x)
  new_value <- switch(
    paste(attr(x, "unit"), target_unit, sep = "->"),
    "g->kg" = value / 1000,
    "kg->g" = value * 1000,
    stop("Conversion non supportée")
  )
  with_unit(new_value, target_unit)
}

# Surcharge de l'addition pour unit_val
`+.unit_val` <- function(a, b) {
  if (inherits(b, "unit_val")) {
    # Convert b vers l'unité de a
    b_conv <- convert_to(b, attr(a, "unit"))
    res <- unclass(a) + unclass(b_conv)
    with_unit(res, attr(a, "unit"))
  } else {
    res <- unclass(a) + b
    with_unit(res, attr(a, "unit"))
  }
}

# Surcharge de la soustraction pour unit_val
`-.unit_val` <- function(a, b) {
  if (inherits(b, "unit_val")) {
    b_conv <- convert_to(b, attr(a, "unit"))
    res <- unclass(a) - unclass(b_conv)
    with_unit(res, attr(a, "unit"))
  } else {
    res <- unclass(a) - b
    with_unit(res, attr(a, "unit"))
  }
}

x <- with_unit(5, "kg")
y <- with_unit(3000, "g")

```

```
z <- with_unit(2, "kg")
```

```
x + z
```

```
[1] 7  
attr(,"unit")  
[1] "kg"  
attr(,"class")  
[1] "unit_val"
```

```
x + y
```

```
[1] 8  
attr(,"unit")  
[1] "kg"  
attr(,"class")  
[1] "unit_val"
```

```
y - x
```

```
[1] -2000  
attr(,"unit")  
[1] "g"  
attr(,"class")  
[1] "unit_val"
```

Retours invisibles

```
foo <- function() {  
  invisible(NULL)  
}
```

La soumission doit contenir :

1. le titre de la présentation,
2. les noms des auteurs,
3. les contacts des auteurs, incluant l'institut et le courriel,
4. le résumé de l'intervention,
5. les mots-clefs,
6. le développement,

7. d'éventuelles références (D. Dupont et al [1]).

Bibliographie

[1] D. Dupont et al, « the title », *The journal*, n° nb, p. nb–nb, 2023.