{MARIOBOX}: DES APIS {PLUMBER} À TOUTE ÉPREUVE

RENCONTRE R 2023 - AVIGNON

ANTOINE LANGUILLAUME

DESSINE MOI UNE API

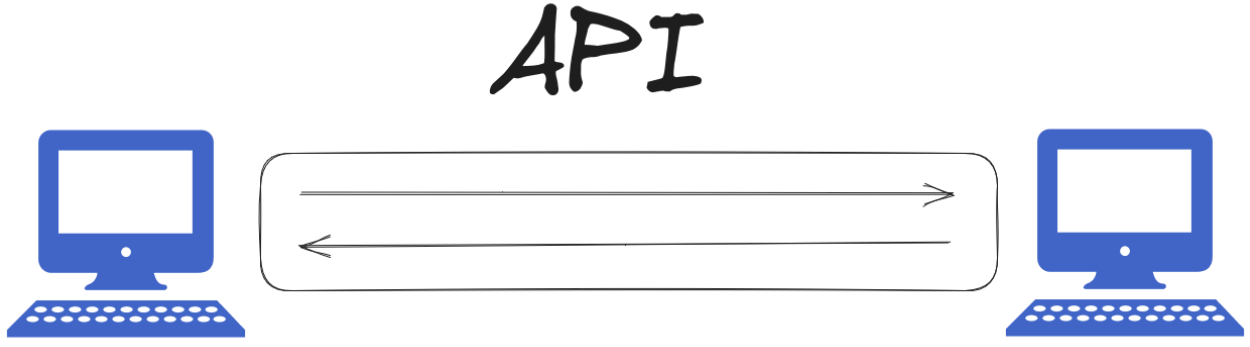
API == Application Programming Interface

RÉPUBLIQUE FRANÇAISE
Liberté
Égalité
Fraternité

api.gouv.fr

Ce qui n'est vraiment pas très clair

Une API, permet à un ordinateur de demander une information à un autre ordinateur, par internet.



OK MAIS ÇA RESSEMBLE À QUOI ?

Simplement à une url HTTP

```
https://calendrier.api.gouv.fr/jours-feries/metropole/2024
```

Tous les jours fériés de 2024

```
https://api.punkapi.com/v2/beers?food=banana
```

Toutes les bières Brewdog qui s'allient bien avec la banane

```
https://api.github.com/users/thinkr-open
```

Toutes les infos publiques sur le compte github de ThinkR

APIS: LINGUA FRANCA INTERNETI

Côté client il nous suffit de requêter l'url avec le language de notre choix.

```r
# R
httr2::request("https://calendrier.api.gouv.fr/jours-feries/metropole/2024")
```

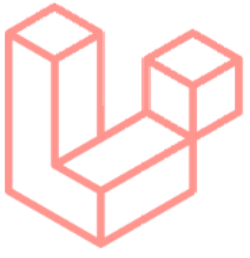
```javascript
// JavaScript
fetch("https://calendrier.api.gouv.fr/jours-feries/metropole/2024)
```

```python
# Python
requests.get("https://calendrier.api.gouv.fr/jours-feries/metropole/2024")
```

```go
// Go
resp, err := http.Get("https://calendrier.api.gouv.fr/jours-feries/metropole/2024")
```

Peu importe le language ayant servi à l'implémentation côté serveur.

FastAPI



Et pourquoi pas R ?

IL N'A PAS FALLU ATTENDRE 2023 POUR S'EN RENDRE COMPTE

- {opencpu}
- {RestRServe}
- {ambiorix}
- {plumber}

LE VILAIN PETIT {PLUMBER}

{PLUMBER} C'EST COOL POUR :

L'INTEROPÉRABILITÉ

- Une API écrite en R peut s'intégrer dans un système écrit en Java, Python, TurboPascal…

LA SCALABILITÉ

- Pas de websocket
- Pas d'état n'importe quel client peut requêter n'importe quel serveur

{PLUMBER} C'EST COOL MAIS...

```
#* Echo back the input
#* @param msg The message to echo
#* @get /echo
function(msg="") {
  list(msg = paste0("The message is: '", msg, "'"))
}
```

Yet Another: #, #', #* ...

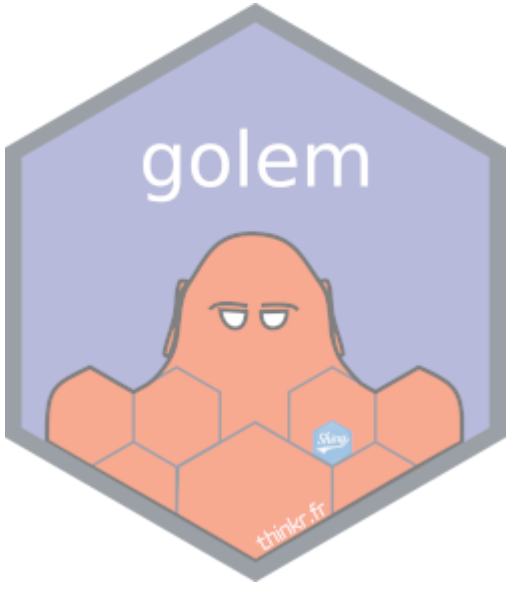
UN NOUVEL ÉPISODE DE PACKAGITE AIGÜE

TOUT EST PACKAGE

Un package c'est :

- de la documentation
- des tests
- une gestion intégrée des dépendances

PLUSIEURS CRISES DE META-PACKAGITE AIGÜE PAR LE PASSÉ



AINSI NAQUIT {MARIOBOX}

EN ROUTE !

1. Installer {mariobox}

```
pak::pak("thinkr-open/mariobox")
# ou
remotes::install_github("thinkr-open/mariobox")
```

2. Créer une nouvelle API packagée

```
mariobox::create_mariobox(
 path = "mon.api"
)
```

LA STRUCTURE DE BASE

```
> fs::dir_tree("mon.api/")
mon.api/
├── DESCRIPTION
├── NAMESPACE
├── R
│   ├── get_health.R
│   └── run_plumber.R
├── dev
│   └── run_dev.R
├── inst
│   └── mariobox.yml # Le coeur d'une API {mariobox}
├── man
│   ├── get_health.Rd
│   └── run_api.Rd
└── tests
    ├── testthat
    │   ├── test-health.R
    │   └── test-run_plumber.R
    └── testthat.R
```

LE COEUR D'UNE API {MARIOBOX}

```
$ cat inst/mariobox.yml
metadata:
  title: mariobox API
handles:
  health_get:
    methods: GET
    path: /health
    handler: get_health
```

TESTER {MON.API}

```
> source("dev/run_dev.R", echo = TRUE)
[..]
Running plumber API at http://127.0.0.1:19483
Running swagger Docs at http://127.0.0.1:19483/__docs__/
```

- Depuis un terminal

```
> httr::GET("http://127.0.0.1:19483/health")
Response [http://127.0.0.1:19483/health]
  Date: 2023-06-19 10:39
  Status: 200
  Content-Type: application/json
  Size: 6 B

> httr::GET("http://127.0.0.1:19483/health") |> httr::content()
[[1]]
[1] "ok"
```

- Depuis votre navigateur: `http://127.0.0.1:19483/__docs__/`

AJOUTER UN ENDPOINT

```
mariobox::add_endpoint(
  "text",
  method = "GET"
)
```

Va :

- Mettre à jour `/inst/mariobox.yml`
- Créer un fichier `R/get_text.R`
- Créer un test `tests/testthat/test-get_text.R`

LA MÉTHODE {MARIOBOX}

```
#' GET text
#'
#' @param req,res HTTP objects
#'
#' @export
get_text <- function(req, res) {
  mariobox::mario_log(
    method = "GET",
    name = "text"
  )
  get_text_f()
}

#' GET text internal
#'
#' @noRd
get_text_f <- function() {
  return("Coucou !")
}
```

METHOD_NAME()

- get_text()
- **HTTP**

METHOD_NAME_F()

- get_text_f()
- **logique métier**

“Separation of concerns”

Séparation des préoccupations

RE-TESTER {MON.API}

```
> source("dev/run_dev.R", echo = TRUE)
[..]
Running plumber API at http://127.0.0.1:19483
Running swagger Docs at http://127.0.0.1:19483/__docs__/
```

- Depuis un terminal

```
> httr::GET("http://127.0.0.1:19483/text")
Response [http://127.0.0.1:19483/text]
  Date: 2023-06-19 10:39
  Status: 200
  Content-Type: application/json
  Size: 6 B

> httr::GET("http://127.0.0.1:19483/text") |> httr::content()
[[1]]
[1] "Coucou !"
```

EN BREF

DES APIS AVEC MARIOBOX C'EST...

- Une méthode scalable pour diffuser le résultats de vos calculs R
- Sans avoir à réfléchir à comment structurer votre code
- Ni à apprendre de nouvelle syntaxe
- et qui peut s'intégrer dans n'imoprte quel SI

https://github.com/ThinkR-open/mariobox

MERCI !