• Open source
• Google
• Simple
• Fast enough
HUGO

A Fast & Modern Static Website Engine

https://bookdown.org/yihui/blogdown/
RStudio Connect

https://www.rstudio.com/products/connect/
Example
package main

import "fmt"

func fahrenheit(celcius float64) float64 {
    return celcius * 1.8 + 32
}

func main() {
    var freezing float64 = fahrenheit(0.0)
    boiling := fahrenheit(100.0)

    fmt.Printf("Water freezes at %4.2f F\n", freezing)
    fmt.Printf("Water boils at %4.2f F\n", boiling)
}
package main

import "fmt"

func fahrenheit(celcius float64) float64 {
    return celcius * 1.8 + 32
}

func main() {
    var freezing float64 = fahrenheit(0.0)
    boiling := fahrenheit(100.0)

    fmt.Printf("Water freezes at %4.2f F\n", freezing)
    fmt.Printf("Water boils at %4.2f F\n", boiling)
}
Water freezes at 32.00 F
Water boils at 212.00 F
research
2018

gofast

2017

Go Slices
Using Go strings in R
Go packages in R packages
Calling go from R

purrple.cat/tags/go/
bit.ly/ergorigin
# install_github("rstats-go/rencontresr2018/fahrenheit")

```r
library(fahrenheit)
fahrenheit(100)
[1] 212

fahrenheit(37)
[1] 98.6

fahrenheit(0)
[1] 32

> 
```
Thread
Robin, Julien, and 2 others

romainfrancois 🌊  Jul 1st at 11:08 AM
in #questions

qqun a une api pour recuperer la temperature actuelle d'une ville ?

4 replies

Robin 4 days ago
Pas vraiment une API, mais ça c'est top :
https://github.com/chubin/wttr.in

GitHub
chubin/wttr.in
wttr.in - 🌠 The right way to check the weather

<table>
<thead>
<tr>
<th></th>
<th>Morning</th>
<th>Noon</th>
<th>Evening</th>
<th>Night</th>
</tr>
</thead>
<tbody>
<tr>
<td>Thu 05 Jul</td>
<td>Partly cloudy</td>
<td>23-24 °C, 6-7 km/h</td>
<td>Light rain showers, 9-12 km/h</td>
<td>Partly cloudy</td>
</tr>
<tr>
<td>19 °C</td>
<td>16 km</td>
<td>0.3 mm, 31%</td>
<td>17 km, 0.3 mm, 61%</td>
<td>20 °C</td>
</tr>
<tr>
<td></td>
<td>7-9 km/h</td>
<td></td>
<td></td>
<td>Partly cloudy</td>
</tr>
<tr>
<td></td>
<td>15 km</td>
<td></td>
<td></td>
<td>22 °C</td>
</tr>
<tr>
<td></td>
<td>0.0 mm, 0%</td>
<td></td>
<td></td>
<td>14-26 km/h</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>Morning</th>
<th>Noon</th>
<th>Evening</th>
<th>Night</th>
</tr>
</thead>
<tbody>
<tr>
<td>Fri 06 Jul</td>
<td>Partly cloudy</td>
<td>24-25 °C, 8-9 km/h</td>
<td>Partly cloudy, 13-19 km/h</td>
<td>Partly cloudy</td>
</tr>
<tr>
<td>20 °C</td>
<td>17 km</td>
<td>15 km, 0.0 mm, 0%</td>
<td>16 km, 0.0 mm, 0%</td>
<td>22 °C</td>
</tr>
<tr>
<td></td>
<td>7-9 km/h</td>
<td></td>
<td></td>
<td>Patchy rain</td>
</tr>
<tr>
<td></td>
<td>18 km</td>
<td></td>
<td></td>
<td>21 °C</td>
</tr>
<tr>
<td></td>
<td>0.0 mm, 0%</td>
<td></td>
<td></td>
<td>11-26 km/h</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>Morning</th>
<th>Noon</th>
<th>Evening</th>
<th>Night</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sat 07 Jul</td>
<td>Partly cloudy</td>
<td>26-27 °C, 9-10 km/h</td>
<td>Light rain showers, 12-16 km/h</td>
<td>Patchy rain</td>
</tr>
<tr>
<td>22-23 °C</td>
<td>16 km</td>
<td>13 km, 0.0 mm, 0%</td>
<td>15 km, 0.5 mm, 70%</td>
<td>21 °C</td>
</tr>
<tr>
<td></td>
<td>8-10 km/h</td>
<td></td>
<td></td>
<td>11-21 km/h</td>
</tr>
<tr>
<td></td>
<td>18 km</td>
<td></td>
<td></td>
<td>12 °C</td>
</tr>
</tbody>
</table>
library(rvest)
library(glue)
library(fahrenheit)
library(tibble)
library(dplyr)

temperature <- function(where = "Rennes"){  
  glue("https://wttr.in/{where}" ) %>%
    read_html() %>%  
    html_node("span:nth-child(3)" ) %>%
    html_text() %>%
    as.numeric()
}

temperature("Rennes" )  
  tibble(celcius = . )  
  mutate(fahrenheit = fahrenheit(celcius))

#> # A tibble: 1 x 2
#>  celcius    fahrenheit
#>    <dbl>        <dbl>
#> 1  19.0        66.2

#' Created on 2018-07-05 by the [reprex package](http://reprex.tidyverse.org) (v0.2.0).
package fahrenheit

func Fahrenheit(celcius float64) float64 {
    return celcius * 1.8 + 32
}
package main

import "C"

import "fahrenheit"

//export Fahrenheit
func Fahrenheit(x float64) float64 {
    return fahrenheit.Fahrenheit(x);
}

func main() {}
#include <R.h>
#include <Rinternals.h>
#include "_cgo_export.h"

SEXP _fahrenheit(SEXP x) {
    return Rf_ScalarReal(Fahrenheit(REAL(x)[0]);
}
#' @useDynLib fahrenheit
#' @export
fahrenheit <- function(celcius) {
  .Call("_fahrenheit",
        celcius,
        PACKAGE = "fahrenheit"
  )
}
.PHONY: go

CGO_CFLAGS  = "$(ALL_CPPFLAGS)"
CGO_LDFLAGS = "$(PKG_LIBS) $(SHLIB_LIBADD) $(LIBR)"
GOPATH     = $(CURDIR)/go

go: 
    CGO_CFLAGS=$(CGO_CFLAGS) CGO_LDFLAGS=$(CGO_LDFLAGS) \ 
    GOPATH=$(GOPATH) /usr/local/go/bin/go \ 
    build -o $(SHLIB) -x -buildmode=c-shared main
future
The problem

R is an amazing interpreted language, giving a flexible and agile foundation for Data Science. Efforts such as Rcpp and reticulate have established that it can be an advantage to pair R with another programming language. Sometimes for speed, sometimes to have alternative options of expression, sometimes to have access to existing libraries.

Go ([https://golang.org](https://golang.org)) is an open source programming language that makes it easy to build simple, reliable and efficient software. It is sometimes said to be the language C++ should have been, in particular if it did not carry a strong commitment to backwards compatibility to C and a taste for complexity.

Go is beautiful and simple, its standard library is one of the most impressive for a programming language. It comes with concurrency built in, which includes (but is not limited to) running code in parallel. The static site generator hugo, the
Dear Romain François,

Thank you for sharing your proposal with us. Unfortunately the ISC did not elect to fund your proposal as many proposals were submitted to the ISC and only a limited number could be funded. Although the committee agrees that a Go interface for R would be useful, we don't feel that this is of broad enough need to be considered an infrastructure project for the R community. The consortium does not generally fund individual package development work.

Regards,

Hadley Wickham
Chair, Infrastructure Steering Committee
R Consortium