Package 'recogito'

June 17, 2021

Title Interactive Annotation of Text and Images

Version 0.1.1

Description Annotate text with entities and the relations between them. Annotate areas of interest in images with your labels.
Providing 'htmlwidgets' bindings to the 'recogito' <https://github.com/recogito/recogito-js> and 'annotorious' <https://github.com/recogito/annotorious> libraries.

License BSD_3_clause + file LICENSE

Encoding UTF-8

URL https://github.com/DIGI-VUB/recogito

RoxygenNote 7.1.1

Suggests shiny, magick, opencv

Imports htmlwidgets, htmltools, jsonlite

NeedsCompilation no

Author Jan Wijffels [aut, cre, cph],
Vrije Universiteit Brussel - DIGI: Brussels Platform for Digital Humanities [cph],
Pelagios Network [cph] (code in inst/htmlwidgets/lib)

Maintainer Jan Wijffels <jan.wijffels@vub.be>

Repository CRAN

Date/Publication 2021-06-17 08:20:02 UTC

R topics documented:

annotorious *Annotate images with areas of interest*

Description

Functionality to label areas in images

Usage

```
annotorious(
  inputId = "annotations",
  src,
  tags = c("Cat", "Dog", "Person", "Other"),
  width = NULL,
  height = NULL,
  elementId = NULL,
  dependencies = NULL
)
```

Arguments

inputId	The input slot that will be used to access the value
src	character string with the image/url to annotate
tags	character vector of possible labels you want to use
width	passed on to createWidget
height	passed on to createWidget
elementId	passed on to createWidget
dependencies	passed on to createWidget

Value

An object of class htmlwidget as returned by createWidget that will intelligently print itself into HTML in a variety of contexts including the R console, within R Markdown documents, and within Shiny output bindings.

See Also

annotorious-shiny

Examples

```
url <- "https://www.w3schools.com/images/picture.jpg"
annotorious(src = url)
url <- paste("https://nl.wikipedia.org/wiki/",
             "Hoofdpagina#/media/Bestand:Pamphlet_dutch_tulipomania_1637.jpg")
url <- paste("https://upload.wikimedia.org/",
```

```
             "wikipedia/commons/a/a0/Pamphlet_dutch_tulipomania_1637.jpg",
             sep = "")
annotorious(src = url)
annotorious(src = url, tags = c("Image", "Text", "Other"))
```

---

annotorious-shiny *Shiny bindings for annotorious*

Description

Output and render functions for using annotorious within Shiny applications and interactive Rmd documents.

Usage

```
annotoriousOutput(outputId, width = "100%", height = "400px")

renderAnnotorious(expr, env = parent.frame(), quoted = FALSE)
```

Arguments

outputId	output variable to read from
width, height	Must be a valid CSS unit (like '100%', '400px', 'auto') or a number, which will be coerced to a string and have 'px' appended.
expr	An expression that generates a annotorious
env	The environment in which to evaluate expr.
quoted	Is expr a quoted expression (with quote())? This is useful if you want to save an expression in a variable.

Value

An output element for use in a Shiny user interface.
Consisting of a toggle button to switch between rectangle / polygon mode (id: outputId-toggle) and the html-widget (id: outputId) which contains an image (id: outputId-img)

Examples

```
if(interactive() && require(shiny)){
library(shiny)
library(recogito)
url <- paste("https://upload.wikimedia.org/",
             "wikipedia/commons/a/a0/Pamphlet_dutch_tulipomania_1637.jpg",
             sep = "")
ui <- fluidPage(annotoriousOutput(outputId = "anno"),
                tags$hr(),
                tags$h3("Results"),
                verbatimTextOutput(outputId = "annotation_result"))
```

```
server <- function(input, output) {
  output$anno <- renderAnnotorious({
    annotorious("annotations", tags = c("IMAGE", "TEXT"), src = url)
  })
  output$annotation_result <- renderPrint({
    read_annotorious(input$annotations)
  })
}
shinyApp(ui, server)

}


annotoriousOutput(outputId = "anno")
```

`read_annotorious` *Parse annotorious annotations*

Description

Parse annotorious annotations

Usage

```
read_annotorious(x, src = character())
```

Arguments

`x`	a character string with json as returned by the htmlwidget
`src`	a character string with the image src which was used in `x`

Value

a data.frame with annotations with columns: id, type, label, comment, x, y, width, height, polygon and an attribute `src` with the provided `src`

Examples

```
url <- paste("https://upload.wikimedia.org/",
             "wikipedia/commons/a/a0/Pamphlet_dutch_tulipomania_1637.jpg",
             sep = "")
url <- system.file(package = "recogito", "examples", "Pamphlet_dutch_tulipomania_1637.jpg")
x <- '[
{
"type":"Annotation",
"body":[{"type":"TextualBody","value":"IMAGE","purpose":"tagging"}],
"target":{"selector":{
  "type":"FragmentSelector",
  "conformsTo":"http://www.w3.org/TR/media-frags/",
```

```
  "value":"xywh=pixel:41,249.5234375,371,245"}},
"@context":"http://www.w3.org/ns/anno.jsonld",
"id":"#58f0096c-4675-4ea8-9f38-bffce0887ab8"
},
{
"type":"Annotation",
"body":[{"type":"TextualBody","value":"TEXT","purpose":"tagging"}],
"target":{"selector":{
  "type":"FragmentSelector",
  "conformsTo":"http://www.w3.org/TR/media-frags/",
  "value":"xywh=pixel:46,5.523437976837158,371,239.99999952316284"}},
"@context":"http://www.w3.org/ns/anno.jsonld",
"id":"#50035dda-c62b-4f30-bf95-1879d60288a5"}]'
anno <- read_annotorious(x, src = url)
anno

if(require(magick)){
library(magick)
img  <- image_read(url)
area <- head(anno, n = 1)
image_crop(img, geometry_area(x = area$x, y = area$y,
                              width = area$width, height = area$height))
area <- subset(anno, type == "RECTANGLE")
allrectangles <- Map(
  x      = area$x,
  y      = area$y,
  width  = area$width,
  height = area$height,
  f = function(x, y, width, height){
    image_crop(img, geometry_area(x = x, y = y, width = width, height = height))
})
allrectangles <- do.call(c, allrectangles)
allrectangles
}


x <- '[
{
  "type":"Annotation",
  "body":[{"type":"TextualBody","value":"IMAGE","purpose":"tagging"}],
  "target":{"selector":{
    "type":"FragmentSelector",
    "conformsTo":"http://www.w3.org/TR/media-frags/",
    "value":"xywh=pixel:43,244.5234375,362,252"
    }},
 "@context":"http://www.w3.org/ns/anno.jsonld",
 "id":"#4eaa8788-0c7e-42d2-b004-4d66b57018a1"},
{
  "type":"Annotation",
  "body":[{"type":"TextualBody","value":"TEXT","purpose":"tagging"}],
  "target":{"selector":{
    "type":"SvgSelector",
    "value":"<svg>
```

```
    <polygon points=\\\"75,4 75,58 32,95 32,194 410,195 391,70 373,63 368,3 222,1.5\\\">
    </polygon></svg>"}},
  "@context":"http://www.w3.org/ns/anno.jsonld",
  "id":"#8bf0a557-c847-4a07-91bc-68a98c499615"}]'
x    <- gsub(x, pattern = "\n", replacement = "")
anno <- read_annotorious(x, src = url)
anno
anno$polygon

if(require(opencv)){
library(opencv)
img  <- ocv_read(url)
area <- subset(anno, type == "POLYGON")
ocv_polygon(img, pts = area$polygon[[1]])
}
```

`read_recogito` *Parse recogito annotations*

Description

Parse recogito annotations

Usage

```
read_recogito(x, text = character())
```

Arguments

`x`	a character string with json as returned by the htmlwidget
`text`	a character string with the text which was used in `x`

Value

a data.frame with annotations with columns: id, type, label, chunk_text, chunk_start, chunk_end, relation_from, relation_to, chunk_comment and an attribute `text` with the provided `text`

Examples

```
x <- '[
{
"type":"Annotation",
"body":[
   {"type":"TextualBody","value":"sdfsd","purpose":"commenting"},
   {"type":"TextualBody","value":"Person","purpose":"tagging"}],
"target":{"selector":[
  {"type":"TextQuoteSelector","exact":"ngenious hero"},
  {"type":"TextPositionSelector","start":42,"end":55}]},
"@context":"http://www.w3.org/ns/anno.jsonld",
```

```
"id":"#a4ea53d4-69f3-4392-a3dd-cbb7e9ad50cb"
},
{
"type":"Annotation",
"body":[{"type":"TextualBody","value":"Person","purpose":"tagging"},
  {"type":"TextualBody","value":"Location","purpose":"tagging"}],
"target":{"selector":[{"type":"TextQuoteSelector","exact":"far and"},
  {"type":"TextPositionSelector","start":70,"end":77}]},
"@context":"http://www.w3.org/ns/anno.jsonld",
"id":"#d7050196-2537-42bf-9d1b-a3f9e4c9fbc6"
}
]'
read_recogito(x)
```

`recogito` *Annotate text with tags and relations*

Description

Functionality to tag text with entities and relations between these

Usage

```
recogito(
  inputId = "annotations",
  text,
  type = c("relations", "tags"),
  tags = c("Location", "Person", "Place", "Other"),
  mode = c("html", "pre"),
  width = NULL,
  height = NULL,
  elementId = NULL,
  dependencies = NULL
)
```

Arguments

`inputId`	The input slot that will be used to access the value
`text`	character string with the text to annotate
`type`	either 'relations' or 'tags' in order to label relations between tags or only plain tags
`tags`	character vector of possible tags
`mode`	either 'html' or 'pre'
`width`	passed on to `createWidget`
`height`	passed on to `createWidget`
`elementId`	passed on to `createWidget`
`dependencies`	passed on to `createWidget`

Value

An object of class htmlwidget as returned by `createWidget` that will intelligently print itself into HTML in a variety of contexts including the R console, within R Markdown documents, and within Shiny output bindings.

See Also

`recogito-shiny`

Examples

```
txt <- "Josh went to the bakery in Brussels.\nWhat an adventure!"
x   <- recogito(inputId = "annotations", txt)
x
x   <- recogito(inputId = "annotations", txt, type = "tags",
                tags = c("LOCATION", "TIME", "PERSON"))
x
txt <- "Lorem ipsum dolor sit amet consectetur adipiscing elit Quisque tellus urna
  placerat in tortor ac imperdiet sollicitudin mi Integer vel dolor mollis feugiat
  sem eu porttitor elit Sed aliquam urna sed placerat euismod In risus sem ornare
  nec malesuada eu ornare quis dui Nunc finibus fermentum sollicitudin Fusce vel
  imperdiet mi ac faucibus leo Cras massa massa ultricies et justo vitae molestie
  auctor turpis Vestibulum euismod porta risus euismod dapibus Nullam facilisis
  ipsum sed est tempor et aliquam sapien auctor Aliquam velit ligula convallis a
  dui id varius bibendum quam Cras malesuada nec justo sed
  aliquet Fusce urna magna malesuada"
x   <- recogito(inputId = "annotations", txt)
x
x   <- recogito(inputId = "annotations", txt, type = "tags",
                tags = c("LOCATION", "TIME", "PERSON", "OTHER"))
x

##
## Color the tags by specifying CSS - they should have .tag-{TAGLABEL}
##
library(htmltools)
cat(readLines(system.file(package = "recogito", "examples", "example.css")), sep = "\n")
tagsetcss <- htmlDependency(name = "mytagset", version = "1.0",
                            src = system.file(package = "recogito", "examples"),
                            stylesheet = "example.css")
x   <- recogito(inputId = "annotations", txt,
                tags = c("LOCATION", "TIME", "PERSON", "OTHER"),
                dependencies = tagsetcss)
x
```

`recogito-shiny` *Shiny bindings for recogito*

Description

Output and render functions for using recogito within Shiny applications and interactive Rmd documents.

Usage

```
recogitoOutput(outputId, width = "100%", height = "400px")

renderRecogito(expr, env = parent.frame(), quoted = FALSE)

recogitotagsonlyOutput(outputId, width = "100%", height = "400px")

renderRecogitotagsonly(expr, env = parent.frame(), quoted = FALSE)
```

Arguments

outputId	output variable to read from
width, height	Must be a valid CSS unit (like '100%', '400px', 'auto') or a number, which will be coerced to a string and have 'px' appended.
expr	An expression that generates a recogito
env	The environment in which to evaluate expr.
quoted	Is expr a quoted expression (with quote())? This is useful if you want to save an expression in a variable.

Value

An output element for use in a Shiny user interface.
Consisting of a div of class plaintext which contains an optional toggle button to switch between annotation / relation mode (id: outputId-toggle) and the html-widget (id: outputId)

Examples

```
if(interactive() && require(shiny)){

library(shiny)
library(recogito)
txt <- "Josh went to the bakery in Brussels.\nWhat an adventure!"
ui <- fluidPage(recogitoOutput(outputId = "annotation_text"),
                tags$hr(),
                tags$h3("Results"),
                verbatimTextOutput(outputId = "annotation_result"))
server <- function(input, output) {
  output$annotation_text <- renderRecogitotagsonly({
    recogito("annotations", text = txt, tags = c("LOCATION", "TIME", "PERSON"))
  })
  output$annotation_result <- renderPrint({
    if(length(input$annotations) > 0){
      x <- read_recogito(input$annotations)
      x
```

```
    }
  })
}
shinyApp(ui, server)

}

recogitoOutput(outputId = "annotation_text")
recogitotagsonlyOutput(outputId = "annotation_text")
```

Index