

# shinyMobile: : CHEAT SHEET



## Basics

shinyMobile is a template to design mobile ready and desktop Shiny Apps. shinyMobile is leveraging the progressive web app capability, thereby providing a standalone/fullscreen support.

```
install.package("shinyMobile")
Remotes::install_github("RinteRface/shinyMobile")
```

You can setup custom desktop icons and splashscreen to produce a native app feeling

## 3 TEMPLATES

shinyMobile has 3 predefined templates:

- **f7SingleLayout**(..., navbar, toolbar = NULL, panels = NULL, appbar = NULL, statusbar = f7Statusbar()): one page layout
- **f7TabLayout**(..., navbar, panels = NULL, appbar = NULL, statusbar = f7Statusbar()): tab layout
- **f7SplitLayout**(..., navbar, sidebar, toolbar = NULL, panels = NULL, appbar = NULL, statusbar = f7Statusbar()): ipad specific layout with sidebar panel and main panel

**f7Page** is the main wrapper function

```
library(shiny)
library(shinyMobile)
ui <- f7Page()
server <- function(input, output){}
shinyApp(ui = ui, server = server)
```

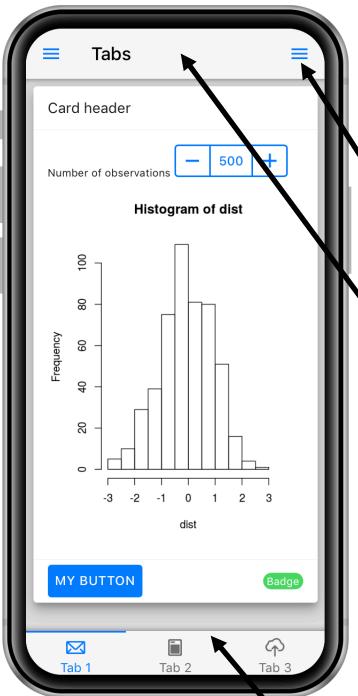
## BOOKMARK YOUR APP:

Host it on [shinyapps.io](https://shinyapps.io), RStudio Connect ([www.rstudio.com/products/connect/](https://www.rstudio.com/products/connect/)) or Build your own Shiny Server ([www.rstudio.com/products/shiny-server/](https://www.rstudio.com/products/shiny-server/))

1. Visit your app using your favorite web browser (Chrome, Firefox, ...)
2. Depending on the web browser, select the add to home screen feature
3. Change the name
4. Click on ok
5. Enjoy!

## Layout Examples: Simple and Tabs (1)

### TAB LAYOUT

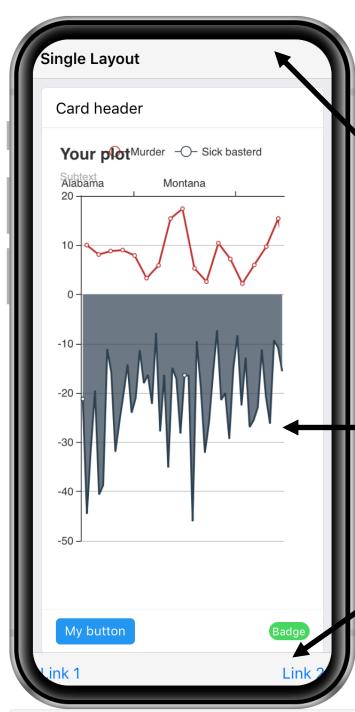


**f7TabLayout** is probably the most versatile template since it adapts to many situations. By specifying an id to **f7Tabs**, you may recover the currently selected tab in input\$Id

```
library(shiny)
library(shinyMobile)
```

```
shiny::shinyApp(
  ui = f7Page(
    title = "Tab Layout",
    f7TabLayout(
      # panels are not mandatory. These are similar to sidebars
      panels = tagList(
        f7Panel(side = "left", theme = "light", effect = "cover", ...),
        f7Panel(side = "right", theme = "dark", effect = "reveal", ...),
        ),
        navbar = f7Navbar(
          title = "Tabs",
          # enable both panels
          left_panel = TRUE,
          right_panel = TRUE
        ),
        # f7Tabs is a special toolbar with included navigation
        f7Tabs(
          animated = TRUE,
          id = "tabs",
          f7Tab(
            tabName = "Tab 1",
            icon = f7Icon("email"),
            active = TRUE,
            # Tab 1 content
            ...
            ),
            # Other tabs
            ...
          )
        ),
        server = function(input, output) {}
      )
    )
  )
)
```

### SINGLE LAYOUT

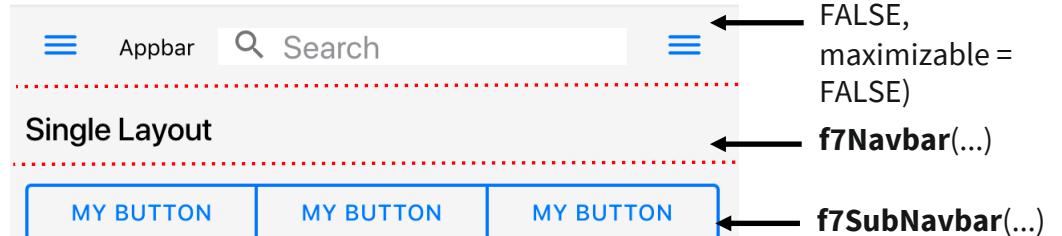


**f7SingleLayout** is well suited for simple prototype apps, with only one view.

```
library(shiny)
library(shinyMobile)
```

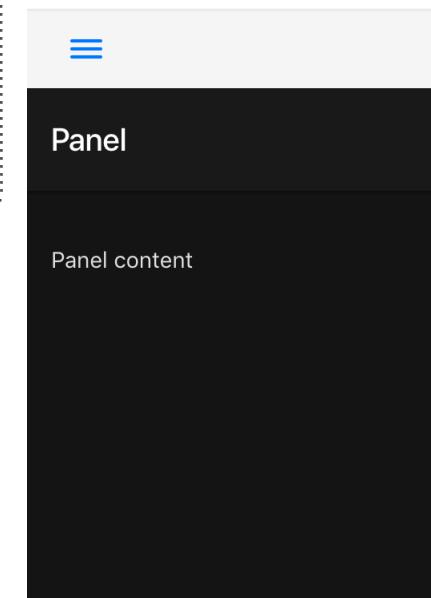
```
ui <- f7Page(
  title = "My app",
  f7SingleLayout(
    # navbar is mandatory
    navbar = f7Navbar(
      title = "Single Layout",
      hairline = FALSE,
      shadow = TRUE
    ),
    # main content
    ...
    # toolbar
    toolbar = f7Toolbar(
      position = "bottom",
      # toolbar content
      ...
    )
  )
)
server <- function(input, output){}
shinyApp(ui = ui, server = server)
```

### COMMON LAYOUT ELEMENTS



**f7Appbar**(...),  
left\_panel = FALSE,  
right\_panel = FALSE,  
maximizable = FALSE)  
**f7Navbar**(...)  
**f7SubNavbar**(...)

**f7TabLayout** and **f7SingleLayout** have an extra slot for an appbar. **f7Appbar** is an extra navigation bar!



**f7Panel**(  
...,  
inputId = NULL,  
title = NULL,  
side = c("left", "right"),  
theme = c("dark", "light"),  
effect = c("reveal", "cover"),  
resizable = FALSE  
)

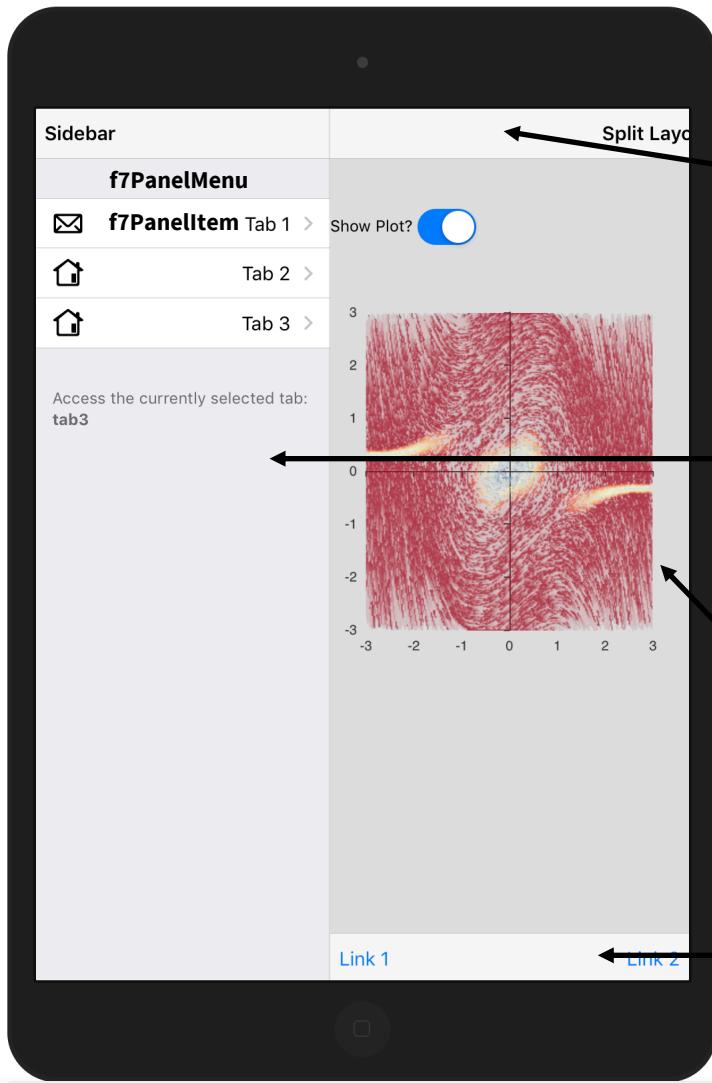
### ONLY FOR SPLIT LAYOUT

**f7PanelMenu**(..., id = NULL)  
**f7PanelItem**(title, tabName, icon = NULL, active = FALSE)

When used in a **split layout** context, input\$<inputId> returns the currently selected panel item. These items may be passed in the ...

## Layout Example: Split (2)

### SPLIT LAYOUT

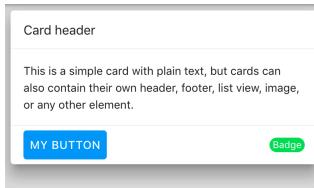


**f7SplitLayout** is a tablet/desktop specific layout.

By specifying an id to **f7PanelMenu**, you may recover the currently selected tab in input\$id. Importantly, each **f7PanelItem** tabName must have a corresponding **f7Item** in the body content!

## Layout: effects

### SHADOW



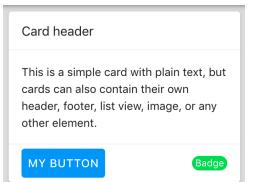
**f7Shadow(tag, intensity, hover = FALSE, pressed = FALSE)**

### BEFORE



fade 3s

### AFTER



### SKELETON (LOADING OVERLAY)

**f7Skeleton(tag, effect = "fade", duration = 2)**

## Layout: typography

**f7Align(tag, side = c("left", "center", "right", "justify"))**

**f7Float(tag, side = c("left", "right"))**

**f7Margin(tag, side = NULL)**

**f7Padding(tag, side = NULL)**

Note: f7Align only work on text tags like p, h1, ...

## UI Configuration

shinyMobile has a dedicated configuration function allowing to fine tune the behavior of layout components, the global theme and skin.

```
f7Page(
  init = f7Init(
    skin = c("ios", "md", "auto", "aurora"),
    theme = c("dark", "light"),
    filled = FALSE,
    color = NULL,
    tapHold = TRUE,
    iosTouchRipple = FALSE,
    iosCenterTitle = TRUE,
    iosTranslucentBars = FALSE,
    hideNavOnPageScroll = TRUE,
    hideTabsOnPageScroll = FALSE
  )
)
```

The tapHold feature is unique to shinyMobile. A tapHold event corresponds to a long press on a button for instance. This allows to handle another kind of click, since on mobile, right click is not possible.

## Layout: block

**f7Block(..., hairlines = TRUE, strong = FALSE, inset = FALSE, tablet = FALSE)**

**f7BlockTitle(title = NULL, size = NULL)**

**f7BlockHeader(text = NULL)**

**f7BlockFooter(text = NULL)**

### STRONG

A medium title

Header

Here comes paragraph within content block. Donec et nulla auctor massa pharetra adipiscing ut sit amet sem. Suspendisse molestie velit vitae mattis tincidunt. Ut sit amet quam mollis, vulputate turpis vel, sagittis felis.

Footer

### STRONG + INSET

Header

Here comes paragraph within content block. Donec et nulla auctor massa pharetra adipiscing ut sit amet sem. Suspendisse molestie velit vitae mattis tincidunt. Ut sit amet quam mollis, vulputate turpis vel, sagittis felis.

Footer

# UI: Inputs

shinyMobile has brand new inputs for Shiny! Moreover each input has its own design depending on the currently selected skin (iOS, material or desktop)

## SLIDERS

```
f7Slider(inputId, label, min,
max, value, step = NULL,
scale = FALSE, vertical = FALSE)
```

Number of observations



Range values



Number of observati...



Number of observations



## TOGGLE

```
f7Toggle(inputId, label, checked = FALSE, color = NULL)
```

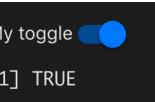
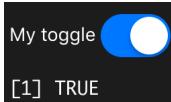
**LIGHT**



**MD**



**DARK**



## STEPPER

```
f7Stepper(inputId, label, min, max, value, step = 1,
fill = FALSE, rounded = FALSE, raised = FALSE, size = NULL,
color = NULL, wraps = FALSE, autorepeat = TRUE,
manual = TRUE )
```

My stepper - 4 +

[1] 4

My stepper 2 - 4 +

[1] 4

simple stepper

Stepper with raised, fill and rounded set to TRUE

## TEXT

```
f7Text(inputId, label, value = "", placeholder = NULL)
```

```
f7Password(inputId, label, value = "", placeholder = NULL)
```

**iOS**

Your text  
some text ×

**MD**

Your text  
some text ×

## CHECKBOXES/GROUPS

```
f7checkBox(inputId, label, value = FALSE)
```

**iOS**

Checkbox  
[1] TRUE

**MD**

Checkbox  
[1] TRUE

```
f7checkBoxGroup(inputId, label, choices = NULL,
selected = NULL)
```

**iOS**

mpg  
 cyl  
 disp

**MD**

mpg  
 cyl  
 disp

## RADIO

```
f7Radio(inputId, label, choices = NULL, selected = NULL)
```

**iOS**

banana ▼  
apple  
peach

**MD**

banana ○  
apple ●  
peach ○

## ACTION BUTTON

```
f7Button(inputId = NULL, label = NULL, src = NULL,
color = NULL, fill = TRUE, outline = FALSE, shadow =
FALSE, rounded = FALSE, size = NULL)
```

**ACTION BUT...**

**MY BUTTON**

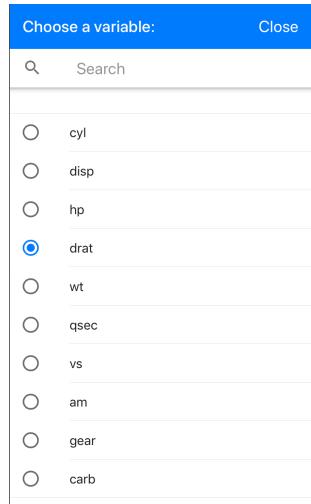
shinyMobile tip: **f7Button** may be grouped in an **f7Segment**.

## SMART SELECT

```
f7SmartSelect(inputId, label, choices, selected =
NULL, type = c("sheet", "popup", "popover"),
smart = TRUE, multiple = FALSE )
```

Choose a variable: (1) drat >

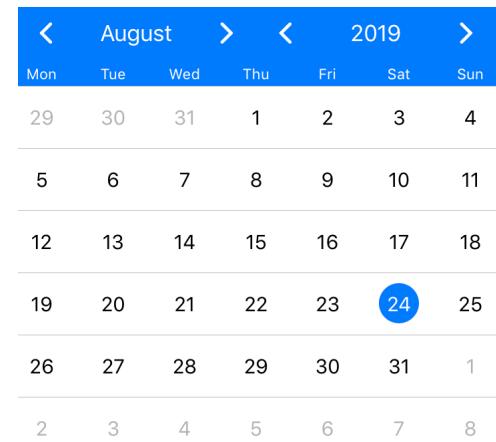
**MD**



(2)

## DATE PICKER

```
f7DatePicker(inputId, label, value = NULL, min = NULL,
max = NULL, format = "yyyy-mm-dd")
```

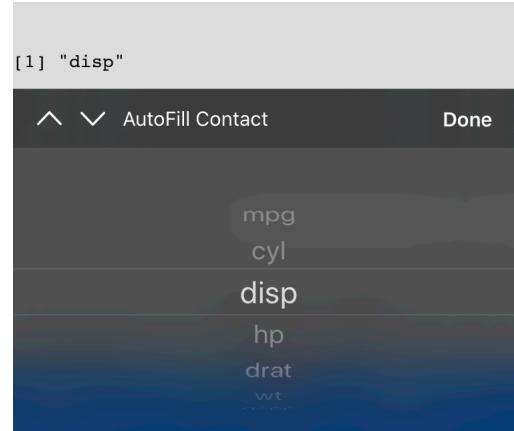


## SELECT

```
f7Select(inputId, label, choices)
```

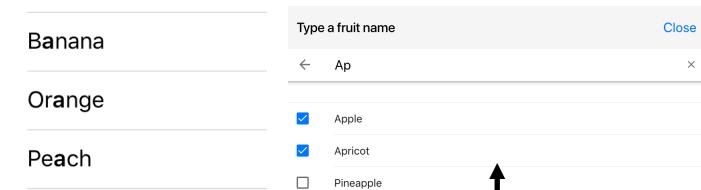
Choose a variable:

disp



## AUTOCOMPLETE

```
f7AutoComplete(inputId, label,
placeholder = NULL, value = choices[1],
choices, typeahead = TRUE,
expandInput = TRUE, type = c("popup",
"page", "dropdown"),
dropdownPlaceholderText = NULL,
multiple = FALSE )
```

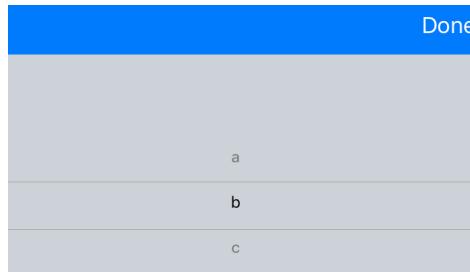


Autocomplete input in a popup with multiple enable

## PICKER

```
f7Picker(inputId, label, placeholder = NULL,
value = choices[1], choices)
```

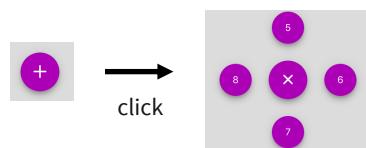
**iOS**



## ACTION BUTTON (FABS)

```
f7Fabs(..., position = c("right-top", "right-center",
"right-bottom", "left-top", "left-center",
"left-bottom", "center-center",
"center-top", "center-bottom"),
color = NULL, extended = FALSE, label = NULL,
sideOpen = c("left", "right", "top", "bottom",
"center"), morph = FALSE, morphTarget =
NULL )
```

```
f7Fab(inputId, label, width = NULL, ..., flag = NULL)
```



shinyMobile tip: **f7Fabs** contain **f7Fab**, an improved action button.



RinteRface

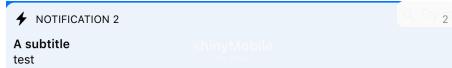
# Server: Alerts

shinyMobile has brand new notifications, popups, ... Except for the **f7Popup** and **f7Tooltip**, all items are generated on the server side

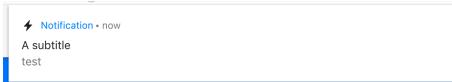
## NOTIFICATIONS

```
f7Notif( text, icon = NULL, title = NULL,  
titleRightText = NULL, subtitle = NULL,  
closeTimeout = 5000, closeButton = FALSE,  
closeOnClick = TRUE, swipeToClose = TRUE,  
session )
```

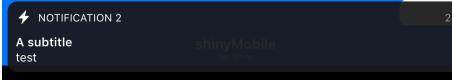
iOS



MD



iOS/DARK



## TOOLTIP

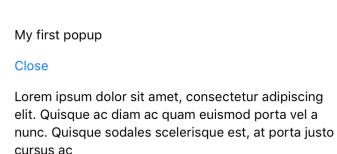
```
A tooltip!  
Hover on me
```

f7Tooltip(tag, text)

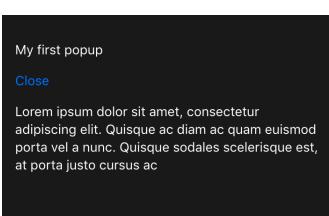
## POPUP

f7Popup(..., id, label = "Open", title)

LIGHT



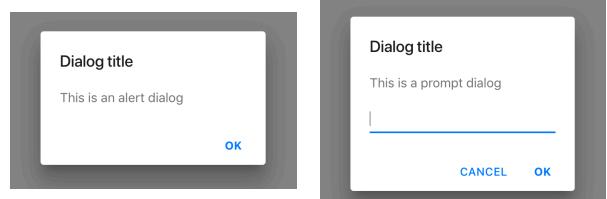
DARK



## DIALOG

```
f7Dialog( inputId = NULL, title = NULL, text, type =  
c("alert", "confirm", "prompt", "login"), session )
```

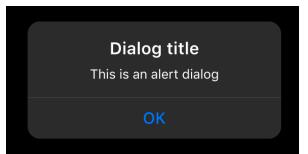
MD



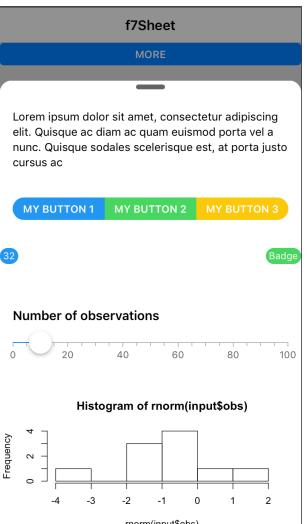
Simple dialog

Dialog with  
prompt input

iOS/DARK



## MODAL SHEET



```
f7Sheet( ..., hiddenItems, id,  
label = "Open", orientation =  
c("top", "bottom"),  
swipeToClose = FALSE,  
swipeToStep = FALSE,  
backdrop = FALSE,  
closeByOutsideClick = TRUE,  
swipeHandler = TRUE )
```

shinyMobile tip: **f7Sheet** may also open from the top. **swipeToStep** allows to keep the sheet slightly open instead of completely closing it.

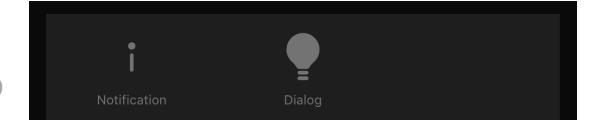
## ACTION SHEET

```
f7ActionSheet( id, session =  
shiny::getDefaultReactiveDomain(), grid =  
FALSE, buttons, icons = NULL )
```

iOS



MD



shinyMobile tip: in **f7ActionSheet** buttons must be provided in a dataframe, icons in a list.

## TOAST

```
f7Toast( session, text, position =  
c("bottom", "top", "center"), closeButton =  
TRUE, closeButtonText = "close",  
closeButtonColor = "red", closeTimeout =  
3000, icon = NULL )
```

I am a toast. Eat me! CLOSE

# Server: Update

## INPUTS

updateF7AutoComplete(session, inputId, value = NULL)

updateF7Checkbox(session, inputId, label = NULL, value = NULL)

updateF7Fab(session, inputId, label = NULL)

updateF7Picker(session, inputId, value = NULL, choices = NULL)

updateF7Slider(session, inputId, min = NULL, max = NULL, value = NULL, scale = FALSE)

updateF7Stepper(session, inputId, min = NULL, max = NULL, step = NULL, fill = NULL, rounded = NULL, raised = NULL, size = NULL, color = NULL, wraps = NULL, autorepeat = NULL, manual = NULL)

updateF7Text(session, inputId, label = NULL, value = NULL, placeholder = NULL)

updateF7Toggle(session, inputId, checked = NULL, color = NULL)

## OTHER ELEMENTS

shinyMobile tip: shinyMobile elements like tabs, panels, sheet, accordion, cards, gauges, progress have an associated input to trigger events on the server side. There are function to update these inputs!

updateF7Accordion(inputId, selected = NULL, session)

updateF7Card(id, session)

updateF7Gauge(session, id, value)

updateF7Panel(inputId, session)

updateF7Progress(session, id, value)

updateF7Sheet(inputId, session)

updateF7Tabs(session, id, selected = NULL)

# Server: tapHold

shinyMobile tip: to trigger a taphold event (long press) on a tag, use the **f7TapHold** function

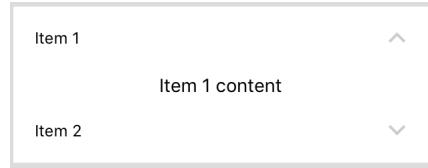
f7TapHold(target, callback, session)

# UI: Widgets

shinyMobile has brand new widgets for Shiny! Some of them have their own design depending on the currently selected skin (iOS, material or desktop)

## ACCORDIONS

**f7Accordion**(..., inputId = NULL, multiCollapse = FALSE)  
**f7AccordionItem**(..., title = NULL, open = FALSE)

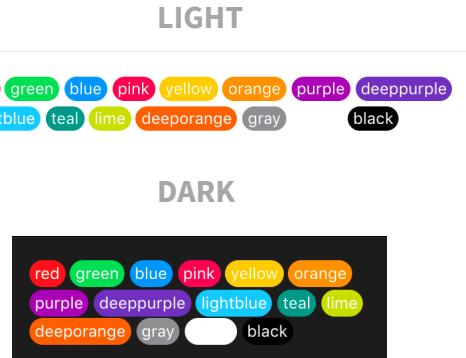


If inputId is passed, one may retrieve the currently selected item as follows

- input\$<inputId>\$state is TRUE if open, else FALSE
- input\$<inputId>\$value is NULL if closed else contain the **f7AccordionItem** value

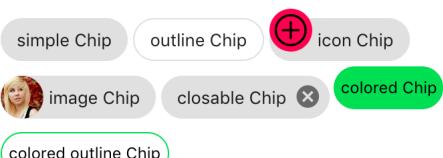
## BADGES

**f7Badge**(..., color = NULL)



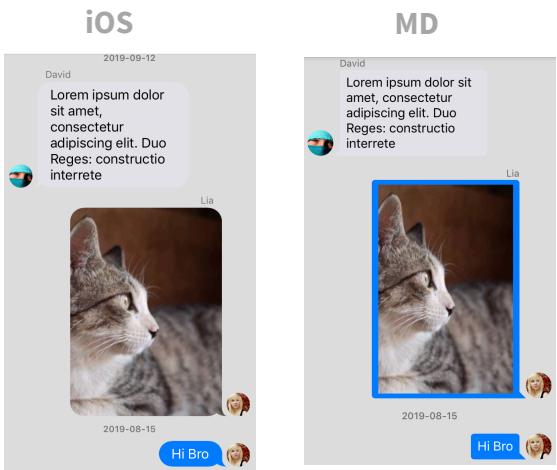
## CHIPS

**f7Chip**( label = NULL, img = NULL, icon = NULL, outline = FALSE, status = NULL, icon\_status = NULL, closable = FALSE )



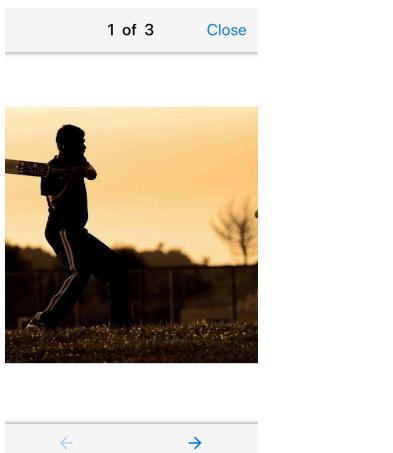
## MESSAGES

**f7Messages**(..., id)  
**f7Message**( content, src = NULL, author = NULL, date = NULL, state = c("sent", "received"), type = c("text", "img") )



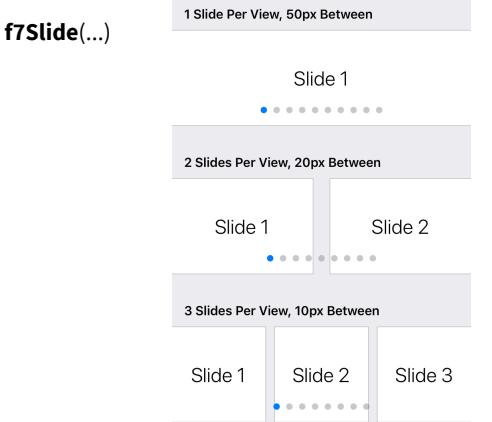
## PHOTO BROWSER

**f7PhotoBrowser**( id, label, photos, theme = c("light", "dark"), type = c("popup", "standalone", "page") )



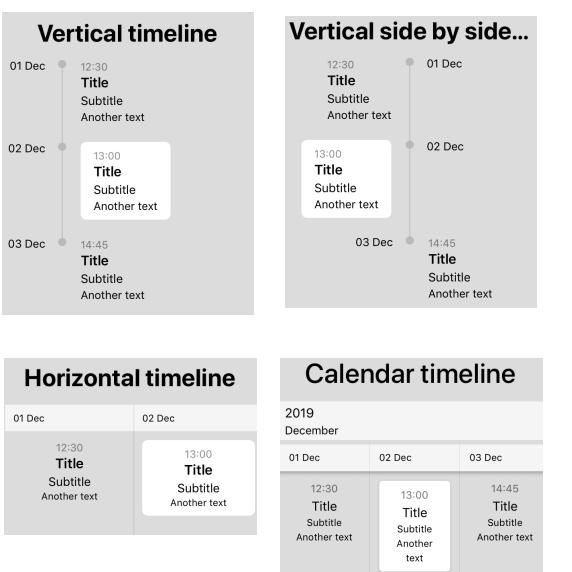
## SWIPE SLIDER

**f7Swiper**( ..., id, spaceBetween = 50, slidePerView = "auto", centered = TRUE, speed = 400 )



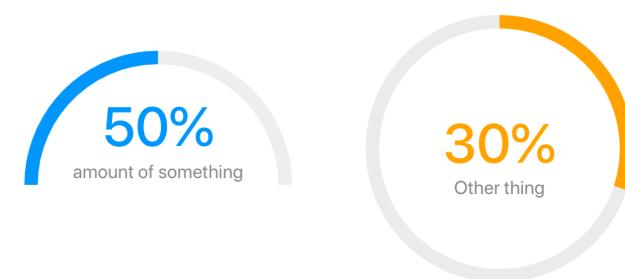
## TIMELINES (Examples with iOS design)

**f7Timeline**( ..., sides = FALSE, horizontal = FALSE, calendar = FALSE, year = NULL, month = NULL )  
**f7TimelineItem**( ..., date = NULL, card = FALSE, time = NULL, title = NULL, subtitle = NULL, side = NULL )



## GAUGES

**f7Gauge**( id, type = NULL, value = NULL, size = NULL, bgColor = NULL, borderBgColor = NULL, borderColor = NULL, borderWidth = NULL, valueText = NULL, valueTextColor = NULL, valueFontSize = NULL, valueFontWeight = NULL, labelText = NULL, labelTextColor = NULL, labelFontSize = NULL, labelFontWeight = NULL )



## PROGRESS BARS

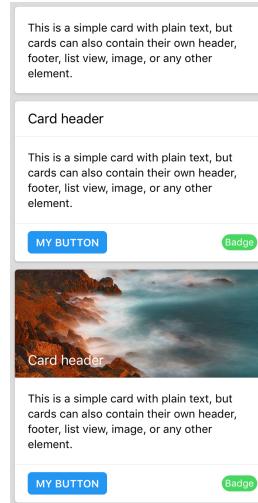
**f7Progress**(id, value, color)  
**f7ProgressInf**(color = NULL)



# UI: Cards

## CARDS

**f7Card**( ..., img = NULL, title = NULL, footer = NULL, outline = FALSE, height = NULL )



Card without header nor footer

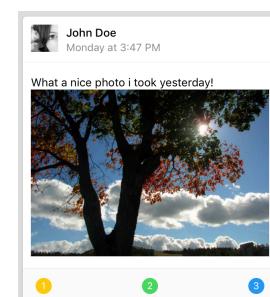
Card with header and footer

Card with header, footer and header image

shinyMobile tip: Don't forget that you may apply shadow effects to cards with **f7Shadow**!

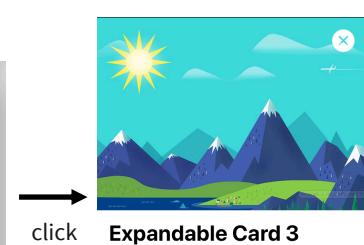
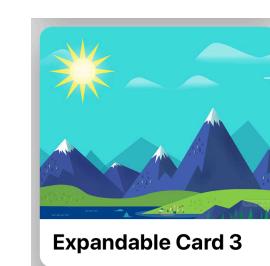
## SOCIAL CARDS

**f7SocialCard**(..., author\_img = NULL, author = NULL, date = NULL, footer = NULL)



## EXPANDABLE CARDS

**f7ExpandableCard**( ..., id = NULL, title = NULL, subtitle = NULL, color = NULL, img = NULL, fullBackground = FALSE )



Framework7 - is a free and open source HTML mobile framework to develop hybrid mobile apps or web apps with iOS or Android native look and feel. It is also an indispensable prototyping tools to show working app prototype as soon as possible in case you need to.

# UI: Lists

shinyMobile lists are a major element of the template

**f7List(..., mode = NULL, inset = FALSE)**

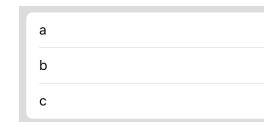
**f7ListGroup(..., title)**

**f7ListItem( ..., title = NULL, subtitle = NULL, header = NULL, footer = NULL, url = NULL, media = NULL, right = NULL )**

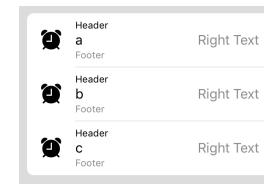
## SIMPLE LISTS

Below, we set inset to TRUE to apply margins around the list. We only show for iOS design. Material is similar

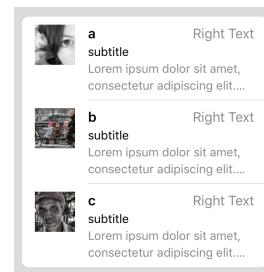
iOS



Simple list items



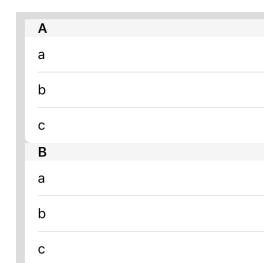
List items, with header, footer, right content and media (icon here)



List in "media" mode with list items containing title, subtitle, right content and media (images here)



List items with an url



List in "contact" mode with list group containing list items.

## LIST INDEX

This widget is useful in case you want to sort items by name and quickly navigate them with a scroll handle.

**f7ListIndex(..., id)**

**f7ListGroup(..., title)**

**f7ListItem( ..., title = NULL, subtitle = NULL, header = NULL, footer = NULL, url = NULL, media = NULL, right = NULL )**

## LIST UTILS: SEARCHBAR

shinyMobile has a searchbar that allows to browse in a list or page containing multiple lists. The searchbar must be included in **f7Navbar** or **f7Appbar**.

**f7Searchbar(id = NULL, placeholder = "Search", expandable = FALSE, inline = FALSE)**

**f7SearchbarTrigger(targetId)**

**f7Found(tag)**

**f7NotFound(tag)**

**f7HideOnSearch(tag)**

There are 3 ways of using the searchbar:

- Call **f7Searchbar** anywhere in **f7Navbar** or **f7Subnavbar**
- Use a **f7SearchbarTrigger** where id points to the searchbar unique id and put the searchbar anywhere
- Put the **f7Searchbar** in the **f7Appbar**. The inline parameter must be TRUE

```
f7SingleLayout(
  navbar = f7Navbar(
    title = "f7Searchbar",
    f7SearchbarTrigger(targetId = "search1"),
    subNavbar = f7SubNavbar(
      f7Searchbar(id = "search1")
    )
  ),
  f7Block(
    "This block will be hidden on search.
     Lorem ipsum dolor sit amet, consectetur adipisicing elit."
  ) %>% f7HideOnSearch() # to hide on search,
  f7List(
    lapply(seq_along(cars), function(i) {
      f7ListItem(cars[i])
    })
  ) %>% f7Found(), # to show on search

  f7Block(
    p("Nothing found")
  ) %>% f7NotFound() # to show if nothing is found
)
```

# Progressive Web App

shinyMobile is first designed to develop mobile ready apps! However, there is no sense opening a mobile app in a web browser like chrome or safari.

## MANIFEST

Progressive web apps (PWA) have the ability to display as standalone apps, even being available offline (the later not available for shinyMobile). Creating a manifest allow us to pass useful information such as the start url, the display mode, ...

**create\_manifest(path, name = "My App", shortName = "My App", description = "What it does!", lang = "en-US", startUrl, display = c("minimal-ui", "standalone", "fullscreen", "browser"), icons )**

This function creates a www folder containing a manifest.json file as well as 2 subdirectories that will contain app icons and splashscreens. While splashscreens are automatically created in android, they must be generated for iOS. Warning: this function does not design icons for you.

```
{
  "name": "My App",
  "short_name": "My App",
  "description": "My App",
  "lang": "en-US",
  "start_url": "https://www.google.com/",
  "display": "standalone",
  "icons": [
    {
      "src": "icons/128x128.png",
      "sizes": "128x128",
      "type": "image/png"
    },
    {
      "src": "icons/144x144.png",
      "sizes": "144x144",
      "type": "image/png"
    }
  ]
}
```

