

React Simple

Table of Contents

1. Introduction	1
1.1. What is React Simple?	2
1.2. Why use React Simple?	2
1.3. How to use React Simple?	3
2. API	4
2.1. react	4
2.2. is	4
2.3. react?	5
2.4. clear-reactions	5
2.5. dump-reactions	6
3. React Simple Components	6
3.1. reactor!	6
3.2. deep-reactor!	6

1. Introduction

React Simple

0.6.0 Red is a simple, easy-to-use, and fast library for building user interfaces. It is based on the **object-oriented reactive programming** paradigm. It is designed to be a simple and easy-to-use library for building user interfaces.

React Simple API is a simple and easy-to-use library for building user interfaces.

[react-simple] | *react-simple.png*

React Simple is a simple and easy-to-use library for building user interfaces.

[react-graph] | *react-graphs.png*

React Simple is a simple and easy-to-use library for building user interfaces.

React Simple is a simple and easy-to-use library for building user interfaces. **react/unlink** and **clear-reaction** are the main methods for building user interfaces.

React Simple is a simple and easy-to-use library for building user interfaces.

NOTE:

- React Simple is a simple and easy-to-use library for building user interfaces.

- `Red` `FRP`
- `Red/View` `GUI` *face!*

1.1. □□□

[illegible]

1.2. □□□□□□□□□□

[illegible]

11

```
view [
  s: slider return
  b: base react [b/color/1: to integer! 255 * s/data]
]
```

sliderbase
Red/View

00

```
vec: make reactor! [x: 0 y: 10]
box: object [length: is [square-root (vec/x ** 2) + (vec/y ** 2)]]
```

is is word

GUI
vec/x
vec/y
vector
vec

11

```
a: make reactor! [x: 1 y: 2 total: is [x + y]]
```

$$y_{total} = \sum_{x=1}^n y_x$$

11

```
a: make reactor! [x: 1 y: 2]
total: is [a/x + a/y]
```

word word
Excel

NOTE: total

1.3. □□□□□□□□□□

react/link

1

```

;--
win: layout [
  size 400x500
  across
  style ball: base 30x30 transparent draw [fill-pen blue circle 15x15 14]
  ball ball ball ball ball ball ball b: ball loose
  do [b/draw/2: red]
]

follow: func [left right][left/offset/y: to integer! right/offset/y * 108%]

faces: win/pane
while [not tail? next faces][
  react/link :follow [faces/1 faces/2]
  faces: next faces
]
view win

```

face follow

2. API

2.1. react

00

```
react <code>  
react/unlink <code> <source>
```

```
react/link <func> <objects>
react/unlink <func> <source>
```

```
react/later <code>
```

```
<code>      : []  
<func>     : []  
<objects>  : []  
<source>   : ['allwordobject!block!']
```

```
Returns      : [] <code> [] <func>
```

00

React1
/link
path!

```

#####react#####
#####/later#####

```

Red

[link](#) `react`

[/unlink](#)<source>

- **all word**
-
-

[/unlink](#)

2.2. is

11

```
<word>: is <code>
```

```
<word> : [] [] [] [] [] [] [] [] [] [] word set-word! []
```

[illegible]

11

Dis word <code> 2

NOTE: Excel

1

```
a: make reactor! [x: 1 y: 2 total: is [x + y]]
```

a/total

$$== 3$$

a/x: 100

a/total

== 102

2.3. react?

11

```
react? <obj> <field>
```

```
react?/target <obj> <field>
```

```
<obj>      : []object![]
```

```
<field> : []word![]
```

```
Returns : block!function! none!
```

11

```

react?
none
/target
none

```

2.4. clear-reactions

11

clear-reactions

□ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □

11

11

[illegible]

Red

00

11

[illegible]

NOTE: $\frac{1}{2}$ is $\frac{1}{2}$

11

11

```

series

```

series

NOTE:

series

NOTE:deep-reactor!
series

```
r: make deep-reactor! [  
  x: [1 2 3]  
  y: [[a b] [c d]]  
  total: is [append copy x copy y]  
]  
append r/y/2 'e  
print mold r/total
```