

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 Examples	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 [React](#) library and is designed to be a [reactive programming](#) library. It is [object-oriented](#) and [reactive programming](#) is a programming paradigm that is based on the flow of data and the change of state.

React Simple API is a simple, easy-to-use, and fast library for building user interfaces. It is based on the [React](#) library and is designed to be a [reactive programming](#) library.

[react-simple] | [react-simple.png](#)

React Simple is a simple, easy-to-use, and fast library for building user interfaces. It is based on the [React](#) library and is designed to be a [reactive programming](#) library.

[react-graph] | [react-graphs.png](#)

React Simple is a simple, easy-to-use, and fast library for building user interfaces. It is based on the [React](#) library and is designed to be a [reactive programming](#) library.

React Simple is a simple, easy-to-use, and fast library for building user interfaces. It is based on the [React](#) library and is designed to be a [reactive programming](#) library. [react/unlink](#) [clear-reaction](#)

React Simple is a simple, easy-to-use, and fast library for building user interfaces. It is based on the [React](#) library and is designed to be a [reactive programming](#) library.

NOTE:

- React Simple is a simple, easy-to-use, and fast library for building user interfaces. It is based on the [React](#) library and is designed to be a [reactive programming](#) library.

- `RedView` は `RedViewGroup` のサブクラスで、`face!` を呼び出して UI を描画する。

1.1. □□□

[illegible]

1.2. □□□□□□□□□□

[illegible]

□ □

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

sliderbase
Red/View

11

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

is is word

```
GUIvec/xvec/yvector
vec
```

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

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

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

1. `react` 包中 `link` 方法，用于在 React 组件中插入新的 DOM 元素。

react
/later

Red

[link](#) react

[/unlink](#)<source>

- **all word** **word**
- **word**
- **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]]
```

$$a/\text{total}$$

== 3

a/x: 100

a/total

== 102

2.3. react?

□ □

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

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

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

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

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

11

```

react?
none
/target
none

```

2.4. clear-reactions

□ □

clear-reactions

[illegible]

11

11

□ □

Red

11

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
```