

# 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 a **Red** library for **object-oriented reactive programming**. It is a **Red** library for **object-oriented reactive programming**.

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

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

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

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

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

React Simple is a simple, easy-to-use, and fast library for building user interfaces. **react/unlink** **clear-reaction**

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

NOTE:

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

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

### 1.1. □□□

項目	内容
プロジェクト概要	プロジェクトの目的、範囲、スケジュール、関係者
背景	プロジェクトの背景、目的、意義
プロジェクトの目的	プロジェクトの目的、目標、成果
プロジェクトの範囲	プロジェクトの範囲、対象範囲、除外範囲
プロジェクトのスケジュール	プロジェクトのスケジュール、開始日、終了日、マイルストーン
関係者	プロジェクトの関係者、ステークホルダー、役割
リスク	プロジェクトのリスク、リスク要因、リスク評価
結論	プロジェクトの結論、成果、今後の展望

## 1.2. □□□□□□□□□□

[illegible]

00

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

sliderbaseRed/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

+

[illegible]

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

word word  
Excel

NOTE:total

react/link

```

;--  []
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

```

import React from 'react';
import { Link } from 'react-router-dom';
import './App.css';

function App() {
  return (
    

# React Router



Home
About
Contact


  );
}

export default App;

```

react  
/later

Red

[link](#) `react` `react`

[/unlink](#)<source>

- **all word**
- 
- 

[/unlink](#)

## 2.2. is

11

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

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

```
<code> : 1000000000000000000000000block!00
```

11

`is` 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

00

clear-reactions



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
```