RealNode-API-for-reactive-programming

为响应式编程提供了若干实用的ES10类，基类为`RealNode`和`RealWorld`。

由于大多数的类有着极其复杂的属性和方法，此处先进行相关说明：

属性分级

类/实例的属性在使用频率和是否只读的方面被分为6级：

1. 常用只读属性
2. 常用可写属性
3. 谨慎只读属性
4. 谨慎可写属性
5. 隐藏只读属性
6. 隐藏可写属性

级别越高，稳定性越差，越不建议使用！

未提及的属性一般被认为是5-6级属性。级别为奇数的只读，偶数的可写。

方法分级

类/实例的方法在使用频率的方面被分为三级：

- 一 常用方法
- 二 谨慎方法
- 三 隐藏方法

级别越高，稳定性越差，越不建议使用！

未提及的方法一般被认为是三级方法。

尽可能不要使用三级方法，因为这些方法针对性（功能性）极强，不适合也没必要在日常中使用。

各种类的使用指南

- 事件循环类`RealWorld`
- 响应式类`RealNode`
- 对象响应式类`RealGroup`

`RealWorld`

这是一个事件循环类，基于`setInterval()`函数实现。对该类的一个实例而言，每过一段固定时间将会调用二级实例方法`_mainFn()`。

构造函数 `new RealWorld(timeSep,...fnList)`

`timeSep`（可选）应为一个数值，否则默认为10，单位为毫秒。

`...fnList`（可选）应为`Function`类型，但不建议使用。

1级属性

- `timeSep` 实例属性，`Number`类型。

2级属性

- `paused` 实例属性。若为真值，则会暂停该实例的运行，否则恢复该实例的运行。
- `intervalFn` 实例属性，应为`Function`类型。会在二级实例方法`_mainFn()`被调用时执行，若报错，则会被清除。
- `ifFn` 实例属性，应为`Function`类型。会在二级实例方法`_mainFn()`被调用时执行，若报错，则会被清除。若执行的返回值是真值，则会被清除并尝试执行2级实例属性`soFn`。
- `soFn` 实例属性，应为`Function`类型。会在二级实例方法`_mainFn()`被调用且2级实例属性`ifFn`被执行并返回真值时执行，执行后会被立即清除。

3级属性

- `fnList` 实例属性，`Array`类型。会在二级实例方法`_mainFn()`调用时执行`fnList.pop()`并执行其返回值。

4级属性

- `onload` 静态属性，应为`Promise`类型。浏览器环境下网页文档准备就绪时兑现。

一级方法

- `destroy()` 实例方法，返回`undefined`。永久停止实例的运行，原理是使用`clearInterval()`函数。
- `setTimeSep()` 实例方法，返回`Boolean`类型。接收一个参数作为新的时间间隔，更改成功则返回`true`，否则反之。
- `then()` 实例方法，返回实例本身。接收一个参数`fn`，若`fn`是为`Function`类型，则插入到3级实例属性`fnList`的第一位。
- `onceIf()` 静态方法，返回`Promise`类型。接收一个参数`ifFn`，必须为`Function`类型。当`ifFn`被执行并返回真值时，`onceIf()`方法返回的承诺将会被兑现。

二级方法

- `_mainFn()` 实例方法，返回`undefined`。每过一段固定时间将会被调用。

RealNode

这是一个响应式类，基于Promise类的微任务队列实现。对该类的一个实例而言，可以存储一个值，并在变更存储的值时会产生响应。

构造函数 new RealNode(config,tryRealNode,...relativeRNs)

config（可选）应为一个对象，根据config的属性决定某些行为。

- get（可选）对2级实例属性get进行赋值。
- set（可选）对2级实例属性set进行赋值。
- react（可选）对2级实例属性react进行赋值。
- id（可选）初始化1级实例属性id时作为description。
- info（可选）对4级实例属性info进行赋值。
- value（可选）对2级实例属性value进行赋值。

tryRealNode（可选）不建议使用。若为真值，当变更存储的值时将会尝试对新值中嵌套的RealNode实例进行解析。

...relativeRNs（可选）应为RealNode类型或Symbol类型，但不建议使用。将会调用一级实例方法relate()，参数为...relativeRNs。

1级属性

- id 实例属性，Symbol类型。当实例的引用不小心丢失时，可以通过一级静态方法search()尝试找回。

2级属性

- get 实例属性，应为Function类型，且能够返回一个值。
- set 实例属性，读取值为二级实例方法realSet()，写入值应为Function类型，应接收2级实例属性get的执行返回值并返回Boolean类型。所赋的值将在变更存储的值时被执行。
- react 实例属性，应为Function类型。
- value 实例属性。读取值为2级实例属性get的执行返回值，写入时将执行2级实例属性set，若返回真值，则会执行2级实例属性react和调用一级实例方法notify()。
- display 实例属性，读取值为Boolean类型，默认是true。若写入真值，将能够接收到其他实例的广播通知，否则反之且无法被一级静态方法search()查询。

4级属性

- eventLoop 静态属性，必须是RealWorld实例。

6级属性

- relativeRNs 实例属性，必须是Array实例，且每个元素都必须为Symbol类型，即RealNode实例的1级实例属性id。

一级方法

- `notify()` 实例方法，返回`undefined`。将根据6级实例属性`relativeRNs`查询`RealNode`实例并依次生成微任务，将依次执行2级实例属性`react`和调用一级实例方法`notify()`。
- `relate()` 实例方法，返回`RealNode`实例或`undefined`。接收若干`RealNode`实例或`Symbol`类型作为参数，并尝试返回最后一个`RealNode`实例。
- `unrelate()` 实例方法，返回`Boolean`类型。接收若干`RealNode`实例或`Symbol`类型作为参数。
- `search()` 静态方法，返回`RealNode`实例或`undefined`。接收一个参数`id`，应为`Symbol`类型。
- `justNow()` 静态方法，返回`Promise`类型。接收一个参数`fn`，应为`Function`类型，在生成的一个微任务中执行后兑现返回值。
- `afterNow()` 静态方法，返回`Promise`类型。接收一个参数`fn`，应为`Function`类型，在生成的一个宏任务中执行后兑现返回值。

二级方法

- `realSet()` 实例方法，返回`Boolean`类型。执行时接收四个参数`value`、`react`、`notify`、`noSelf`，将根据6级实例属性`relativeRNs`查询`RealNode`实例并依次生成微任务，将依次执行2级实例属性`react`和调用一级实例方法`notify()`。
- `time()` 静态方法，返回`Promise`类型。接收一个参数`promise`，若为`Function`类型则执行，若为`Promise`类型则等待兑现，最终返回值将兑现`{time: Number,value: any | Error}`。

RealGroup

> 继承`RealNode`

这是针对对象的响应式类，是`RealNode`类的子类。对该类的一个实例而言，可以代理一个对象，并在代理变更对象的键值对时会产生响应。

构造函数 `new RealGroup({id,info,self})`

`self`（可选）默认是一个`null`为原型的空对象。必须是一个对象，否则会报错！（注意：根据相同对象创建的`RealGroup`实例是同一个实例！）

`id`（可选）初始化1级实例属性`id`时作为`description`。

`info`（可选）对4级实例属性`info`进行赋值。

1级属性

- `proxy` 实例属性，`Proxy`类型。对该属性的读写操作将完全转移到构造实例时的`self`对象上。当执行该属性时，将返回构造实例时的`self`对象。
- `get` 实例属性，返回三级实例方法`protoGet()`。执行时：若没有参数，则返回构造实例时的`self`对象的浅拷贝；接收一个参数`keyOrkeyObj`，若是一个对象，则返回一个`null`为原型的相同结构的对象，否则返回对应键的值。

- `set` 实例属性，返回三级实例方法`realSet()`。执行时接收三个参数`value`、`notify`、`noSelf`，`value`必须是对象，不能读取其原型链上的属性。
- `react` 实例属性，返回三级实例方法`protoReact()`。

3级属性

- `listenerMap` 实例属性，`Map`类型。键为`String`类型或`Function`类型，值为`Array`类型，所有元素为`Function`类型。

一级方法

- `keys()` 实例方法，返回`Array`实例，每个元素为`String`类型或`Symbol`类型。接收一个参数`all`，若为真值，则返回值包括`Symbol`类型和不可枚举的键。
- `addSetterListener()` 实例方法，返回`undefined`。接收两个参数`ifKeyOrFn`和`listener`，`ifKeyOrFn`必须为`String`类型或返回`Boolean`类型的`Function`类型，`listener`必须为`Function`类型。
- `getByFilter()` 实例方法，返回一个`null`为原型的含对应键值对的对象。接收一个参数`filterFn`，必须为`Function`类型，根据筛选出的键返回一个`null`为原型的含对应键值对的对象。

敬请期待后续更新