



# Functional Reactive Programming

:

*Делаем жизнь проще*

( Функциональное )



Can't embed values when interpreting  
code-in-interpreter (embedded-lisp  
(c s)  
format s "Can't embed code when interpreting. Code:  
run emit-element-body (processor tag body)  
(when (block-element-p tag)

**( Плюсы**

( Отсутствие состояния )

( Лень )

( Частичное применение и карринг ))

**Декларативный подход**

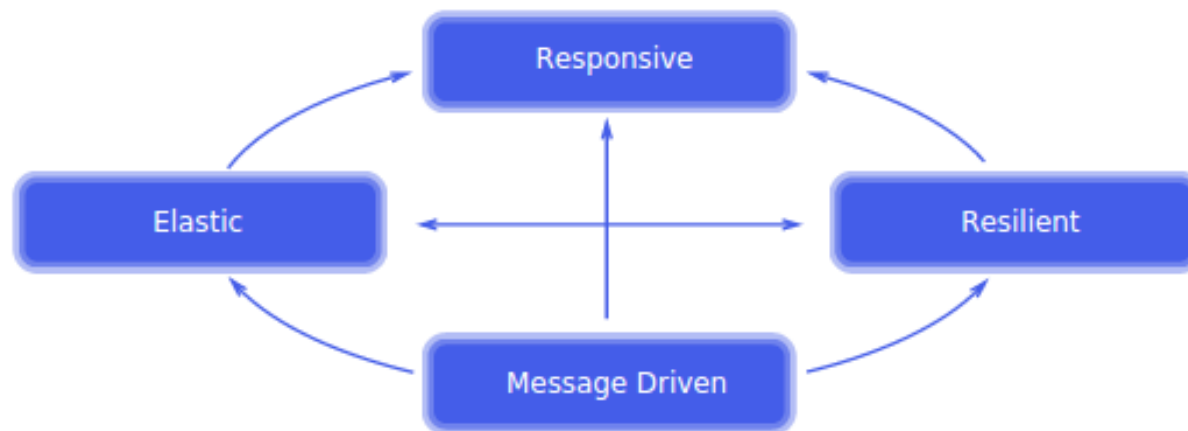
( **Минусы** :в не функциональной среде (

( Производительность )

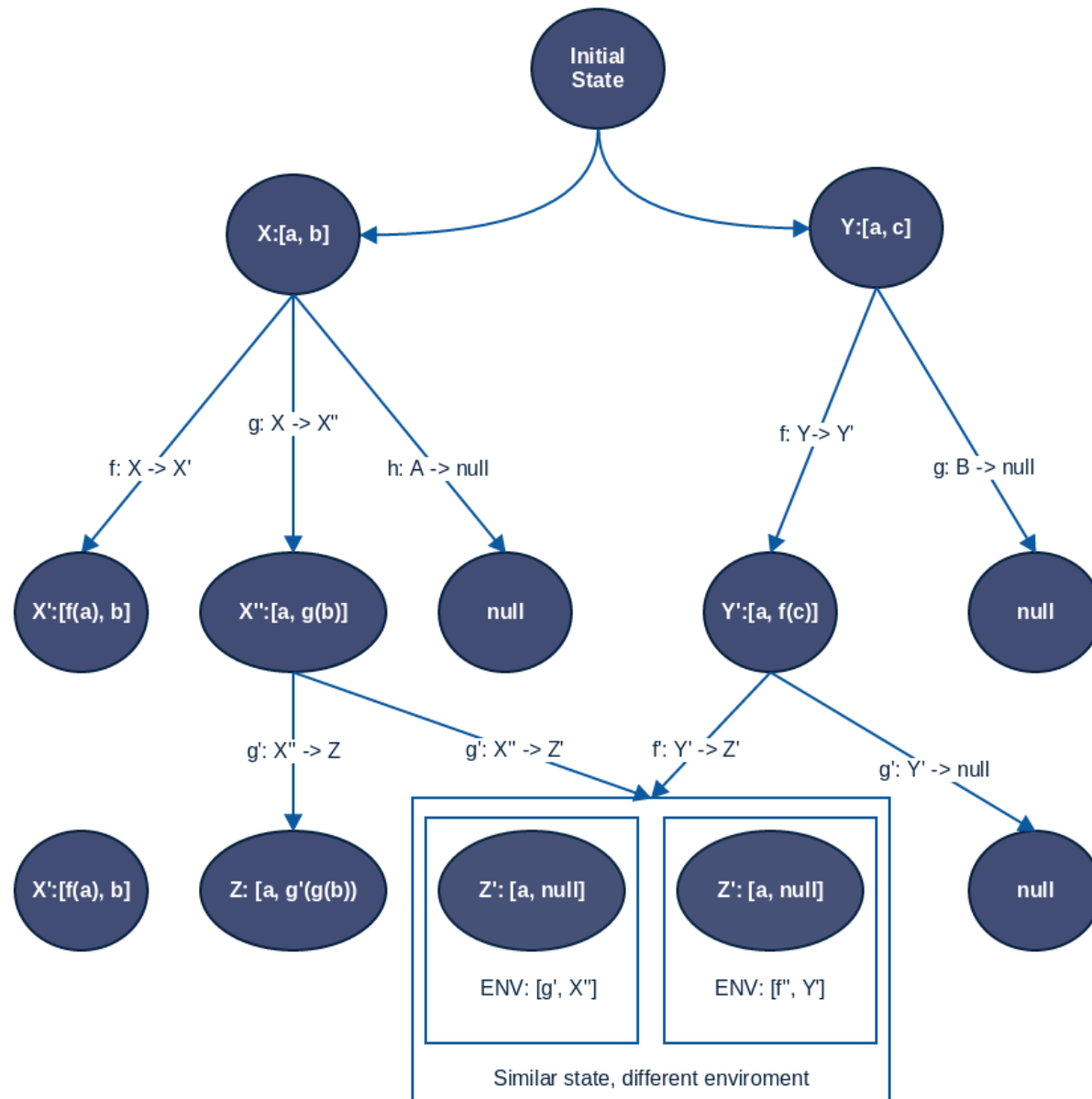
( Комьюнити ))

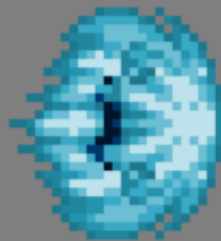
*( Реактивное )*

*www.reactivemanifesto.org*



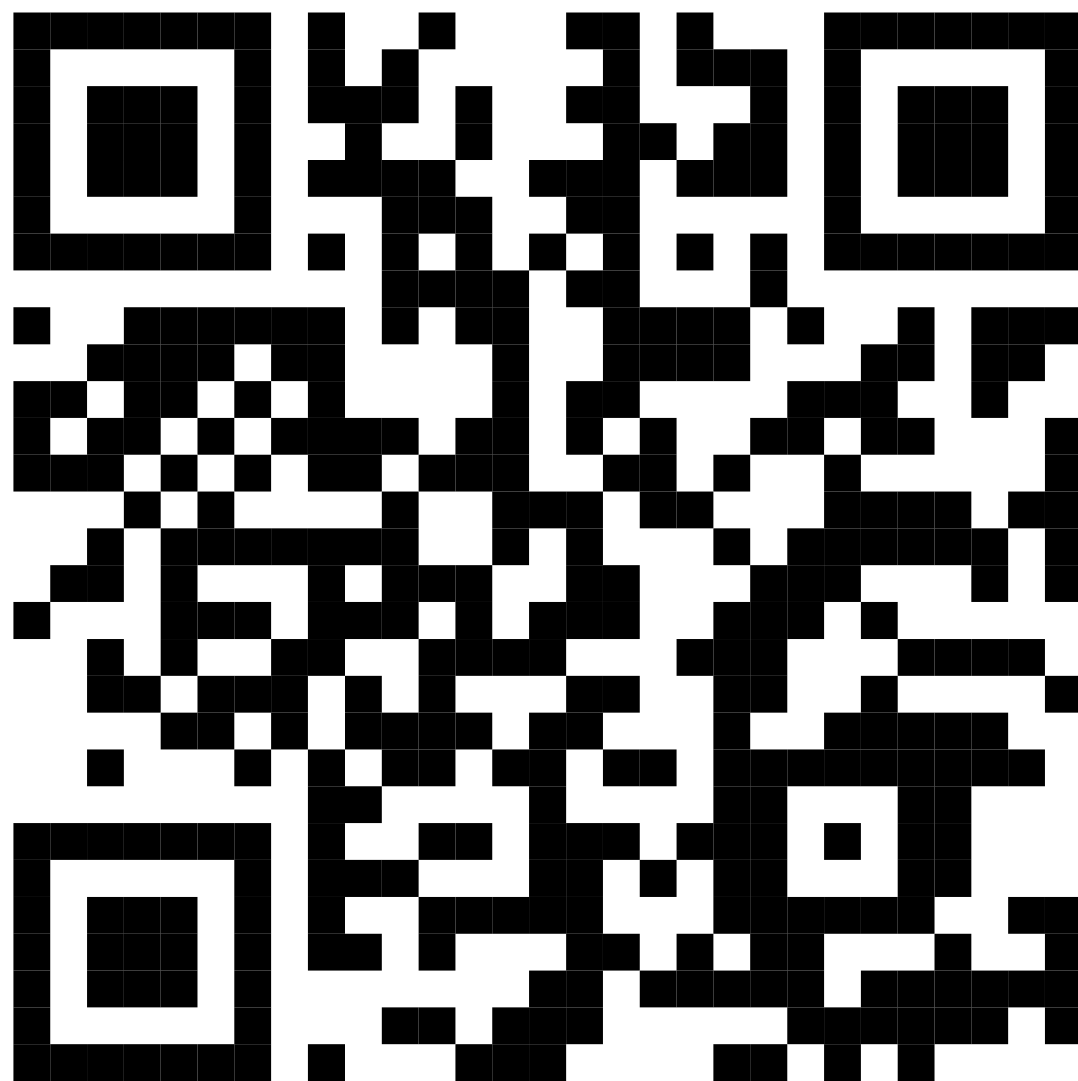
*Асинхронность + Дерево состояний = Непредсказуемость*





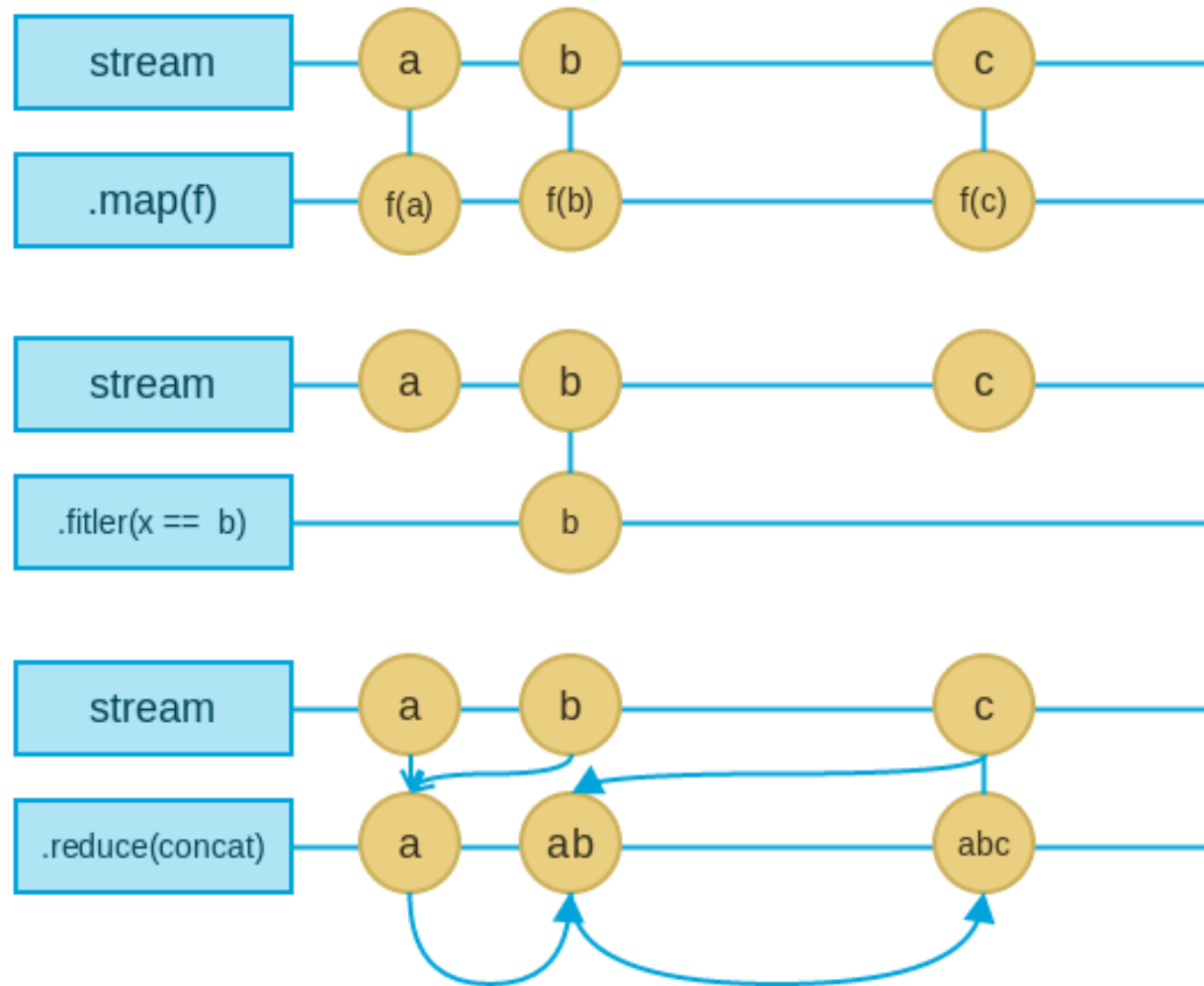
```
function doSomething(params){
  $.get(url, function(result){
    setTimeout(function(){
      startAsyncProcess(function(){
        $.post(url, function(response){
          if(response.good){
            setStateasGoodResponse(function(){
              console.log('Hooray!')
            });
          }
        });
      });
    });
  });
}
```



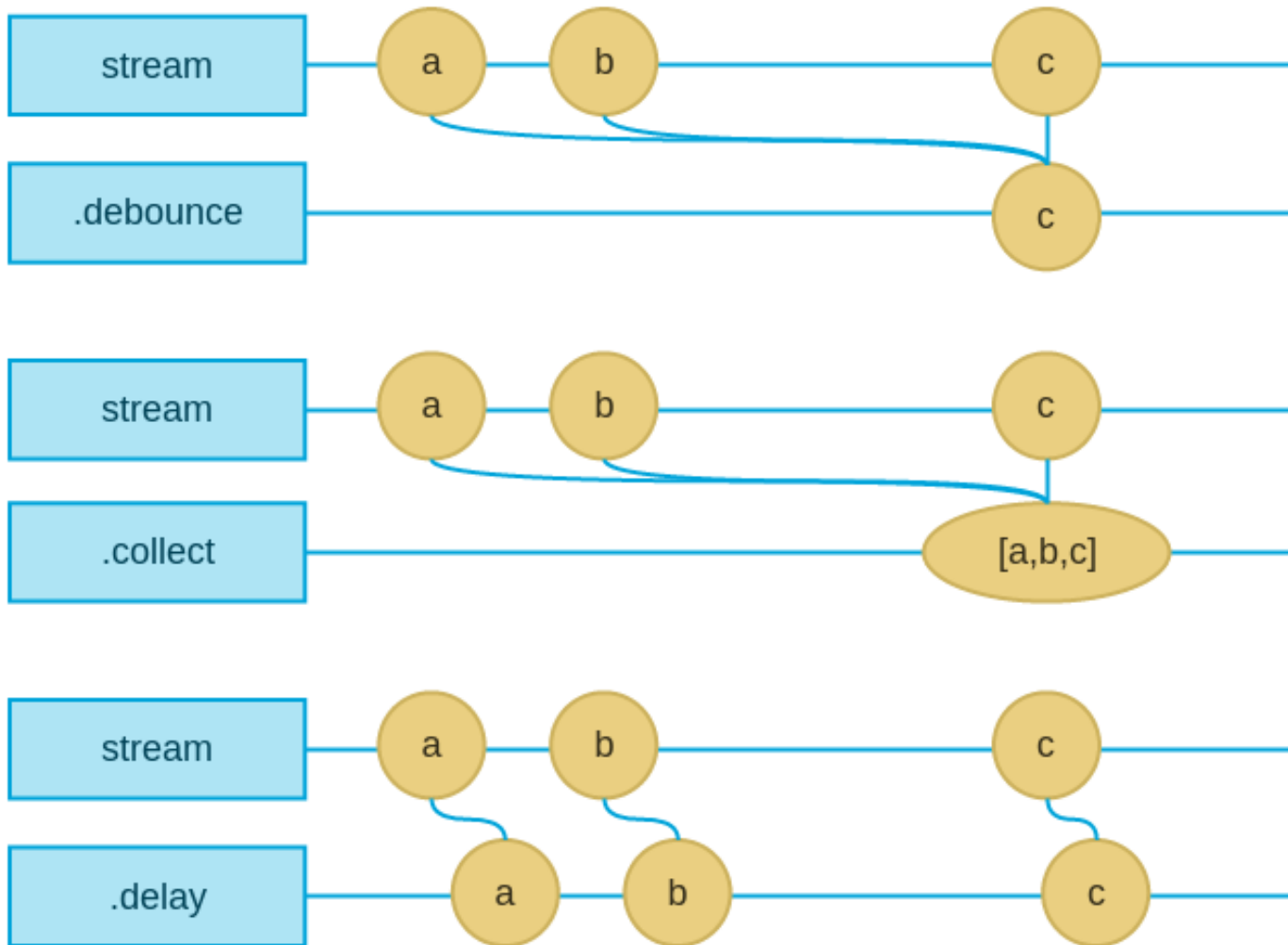


**[www.zefirka.github.io/MoscowJS](http://www.zefirka.github.io/MoscowJS)**

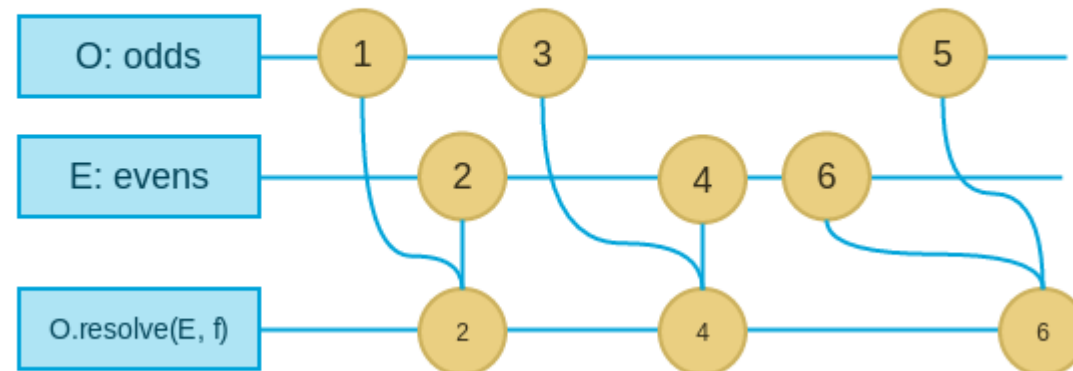
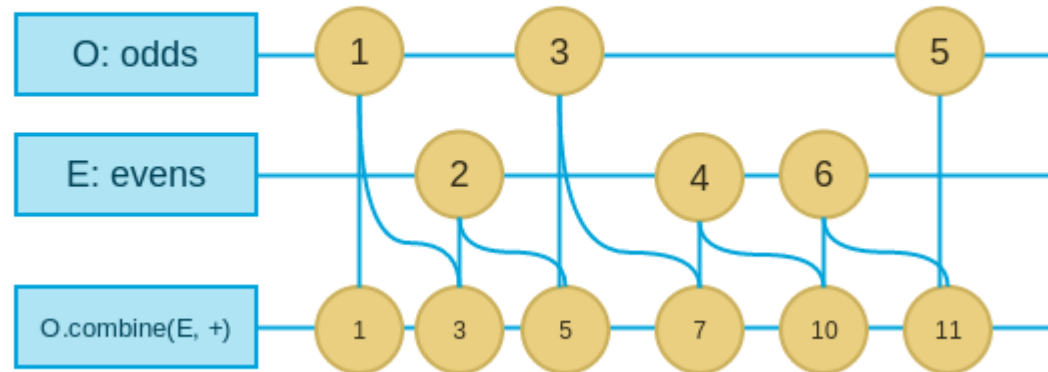
*( map / filter / reduce )*



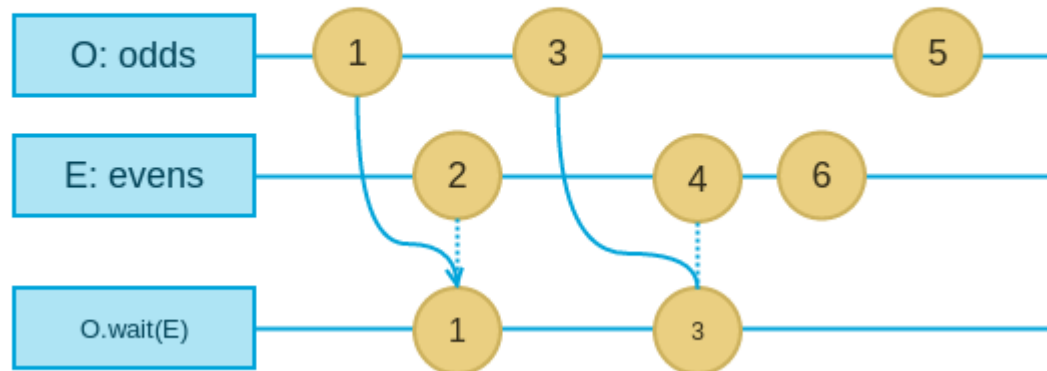
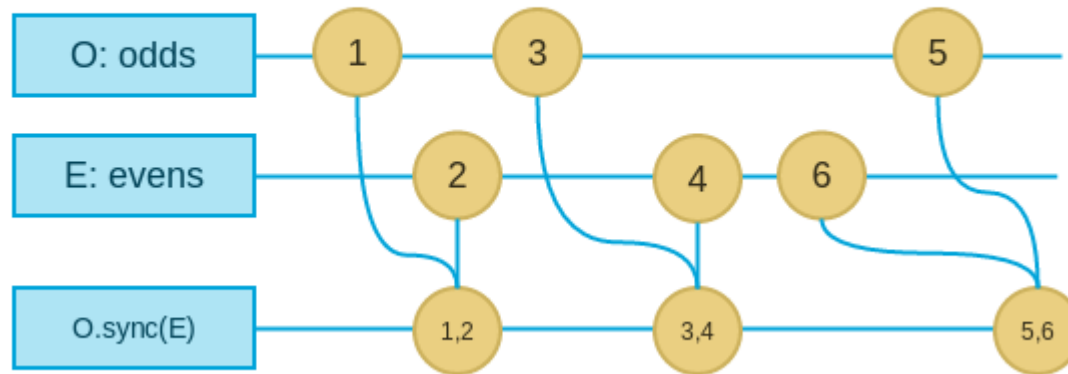
# ( Timing )



( *Combining* )



# ( Composing )



*( Программирование )*



( **RxJS** (Microsoft ReactiveExtensions) [github.com/Reactive-Extensions/RxJS](https://github.com/Reactive-Extensions/RxJS) )



( **Bacon** (Juha Paananen @raimohanska) [github.com/baconjs/baconjs](https://github.com/baconjs/baconjs) )



( **Kefir** (Roman Pominov @rpominov) [github.com/pezadi/kefir](https://github.com/pezadi/kefir) )



( **Warden.js** (Trdat Mkrtchyan @zefirsky) [github.com/zefirka/Warden.js](https://github.com/zefirka/Warden.js) )



( **ProAct** (Nickolay Tzvetinov ) [github.com/proactjs/proactjs](https://github.com/proactjs/proactjs) )

## Без FRP

```
var timer;

button.click(function(){
  clearTimeout(timer);
  setTimeout(function(){
    doSomething();
  }, 1000)
});
```

## C FRP

```
button.stream('click').debounce(1000).on(doSomething)
```



## Без FRP

```
$window.on('scroll', function() {  
  var scrollTop = $window.scrollTop(),  
      newFlag = scrollTop >= scrollDist;  
  
  if (flag !== newFlag) {  
  
    if (flag = newFlag) {  
      target.addClass('b-header--fixed');  
      animateHeaderControls();  
    } else {  
      target.removeClass('b-header--fixed');  
      animateHeaderControls(false);  
    }  
  }  
});
```

## C FRP

```
function animateHeaderControls(hide){  
  target[hide ? 'removeClass' : 'addClass']('b-header--fixed');  
  ...  
}  
  
scrolls  
  .map(moreThanScrollDist)  
  .listen(animateHeaderControls)
```

( *Middlewares* )

## FRP + Transducers

```
var transducer = _t.compose(  
  _t.filter(isActivePlayer),  
  _t.pluck('score'),  
  _t.map(add));  
  
var summaryScores = getPlayerData(url).transduce(transducer);
```

## FRP + Ramda

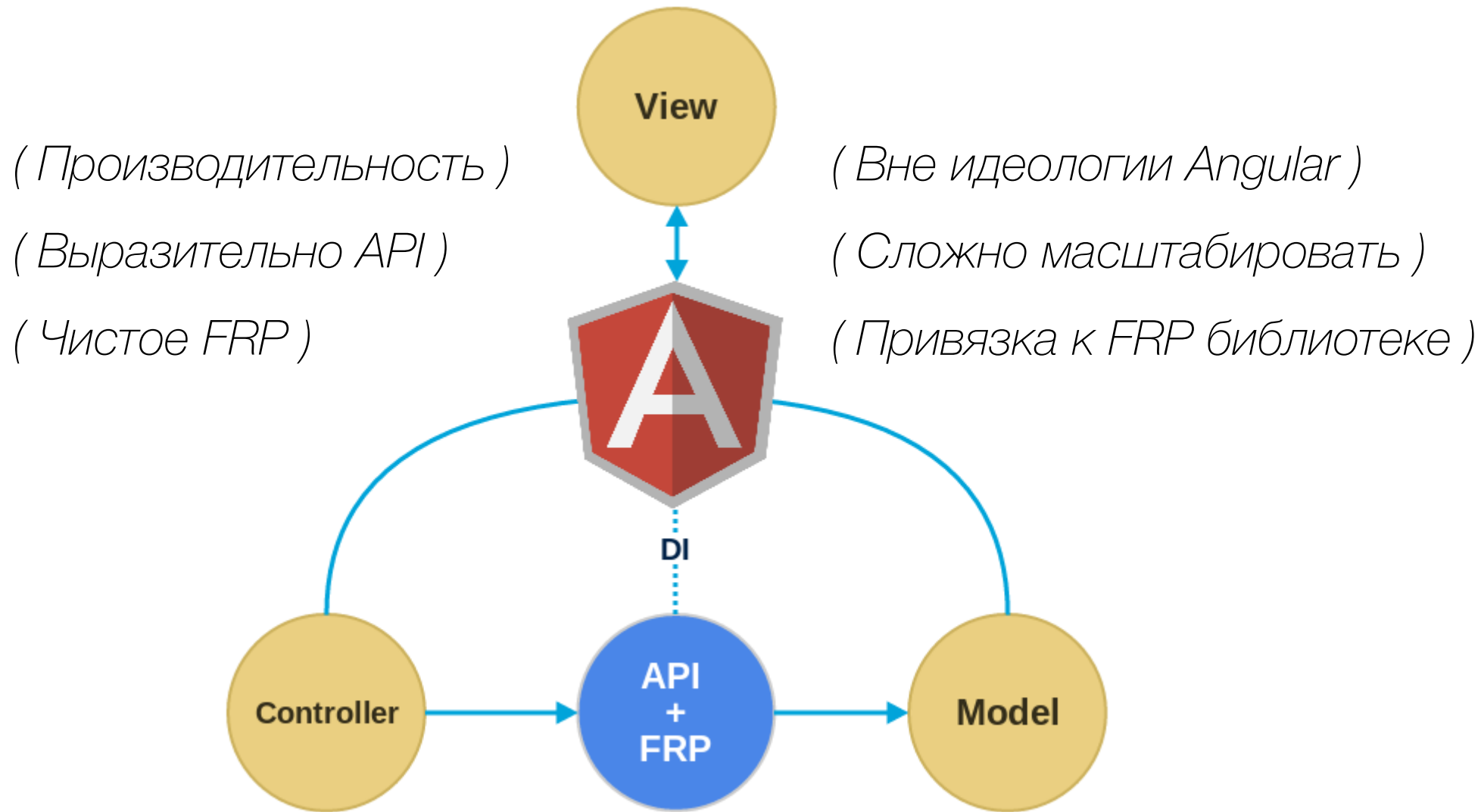
```
var summaryScores = getPlayerData(url)  
  .map(R.map(R.add, R.pluck('score', R.filter(isPlayerActive))))
```

## FRP + ClojureScript + Ramda

```
(def summary-scores (cell (  
  R/map R/add (  
    R/pluck 'score' (  
      R/filter isPlayerActive ( getPlayerData url ))))))
```

*( Интеграция )*

# *( Angular: случай 1 )*



## ( Angular: случай 2 )

( Адаптер FRP to Angular )

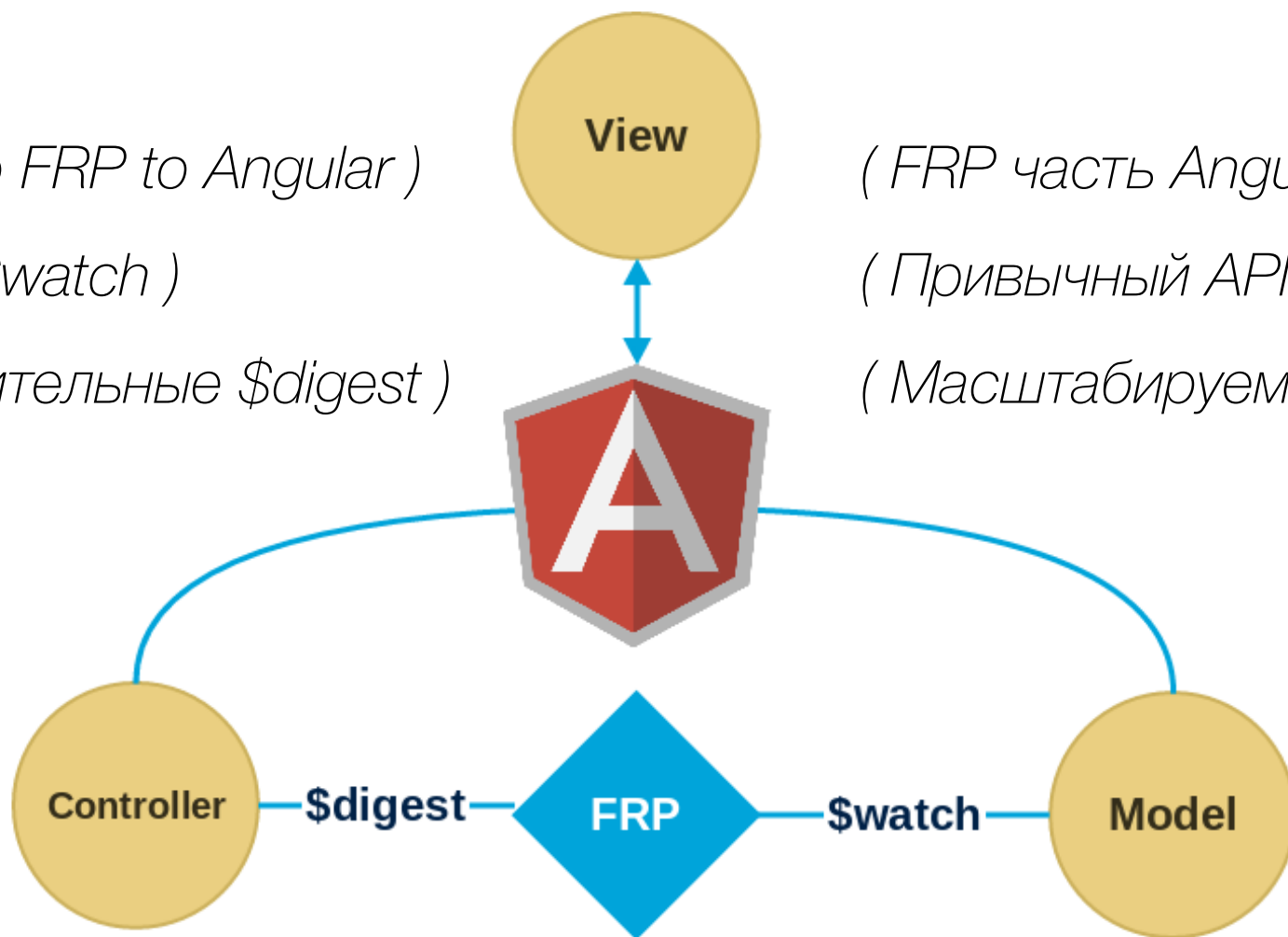
( Много \$watch )

( Принудительные \$digest )

( FRP часть Angular )

( Привычный API )

( Масштабируемость )



# API

```
api.http = Warden.makeStream(function(emit){
  var $http = app.inject('$http');

  this.get = function(url, data){
    $http.get(url, data).then(emit);
    return api.http;
  }

}, api).bus();
```

# Controller

```
function someCtrl($scope){
  $scope.users = [];

  api.get('api:users')
    .map('.users')
    .bindTo($scope, 'users')
}
```

# View

```
<ul>
  <li ng-repeat="user in users">
    ...
  </li>
</ul>
```

# Controller

```
function someCtrl($scope){
  $scope.users = [];
  $scope.query = '';

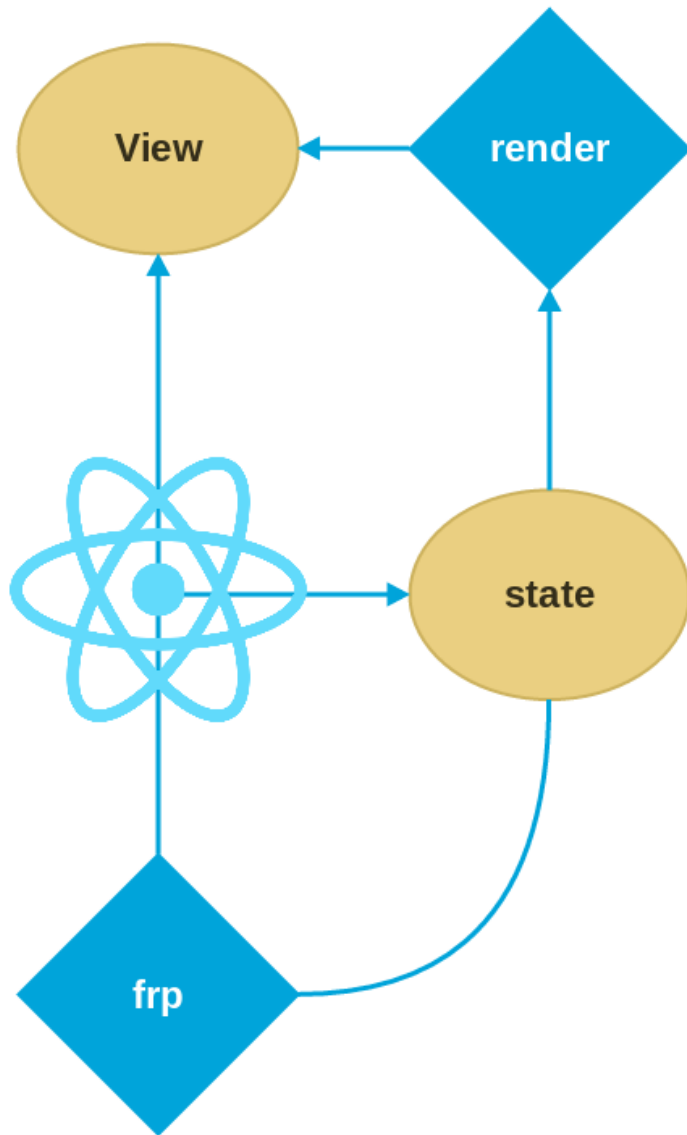
  $scope.$stream('query')
    .debounce(500)
    .map('.newValue')
    .filter(notEmpty)
    .onValue(function(query){
      $http.get('/search', { query : query })
        .bindTo($scope, 'users');
    });
}
```

# View

```
<input type='text' ng-model="query" value={{query}}>
<ul>
  <li ng-repeat="user in users">
    {{user}}
  </li>
</ul>
```



# ( React )



( Изменение состояния –  
результат вычислений )

( Render – это просто рендер )

( Идеально встраивается во FLUX )

( Можно, наконец, понять,  
что такое монада )

