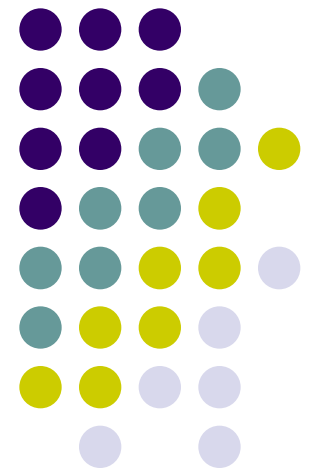


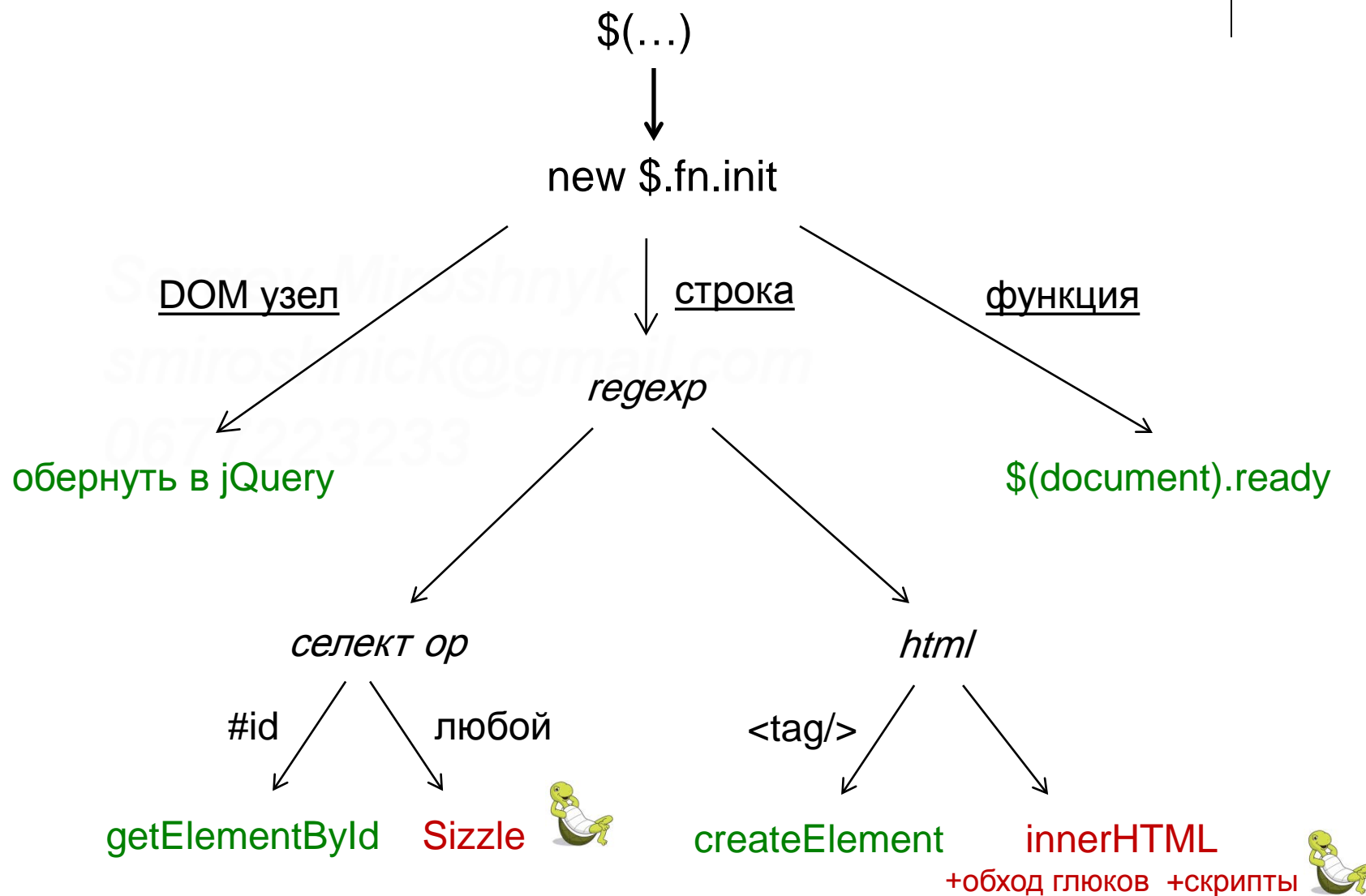
Особенности jQuery





\$(...) изнутри

10 \$ === jQuery





Селекторы

10 CSS3-селекторы

- Обычный селектор
 - `$(“p.myStyle”)`
- Прямые потомки
 - `$(“ul.class1 > li”)`
 - Не затронет вложенных списков
- Условия на атрибуты
 - `$(“td[id^=user-]”)`
 - id начинается на “user-” :
 - `user-name`
 - `user-address`
- ...



:Фильтры

- ⑩ По позиции
 - `$('td:odd > p:first')`
- ⑩ По содержанию
 - `$('td:empty')`
 - `$('p:contains(“text”)')`
- ⑩ Фильтры в javascript
 - `$('td').not(this).get(0)`
 - уточнение найденных результатов
- ⑩ Дополнительные фильтры
 - `$('p:animated, input:focus')`
 - `$('.info:hidden')`
 - Свои :фильтры
 - <http://javascript.ru/examples/sizzle/newfilter.html>

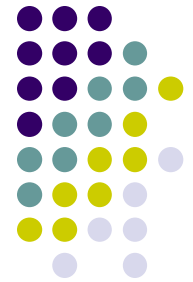


Стек селекторов

10 Пример

```
$( 'form' )  
  .find( '.title' ).html( 'Welcome' ).end()  
  .find( '[name=info]' ).focus( function() {  
    $( this ).select()  
  })
```

Serгей Смирнов
smiroshnick@gmail.com
0677223233



DOM

10 Стандартный пакет манипуляций

- Почти всегда есть действие в обе стороны:
 - `a.append/prepend(b) → b.appendTo/prependTo(a)`
 - `insertAfter/insertBefore → after/before`
 - `replaceAll → replaceWith`
- Дополнительно
 - `wrap, wrapAll, wrapInner`

CSS



10 Получить/установить свойства

- Примеры

- `$(this).css("color","red")`



- `$(".info").css({'background-color': '#123'})`

- изменение стиля через `element.style`

- получение стиля через `computedStyle`

- `computedStyle` капризен при получении данных

- точное название: `margin` `marginLeft`

- [Демо](#)

- reflow



CSS Hooks

10 CSS Hooks

- Кросс-браузерная поддержка любых свойств CSS
 - встроенные свойства: opacity, width, height
 - + поддержка анимации

```
$.cssHooks.borderRadius = {  
  get: function( elem, computed, extra ) { // получить свойство  
    // computed = true - при вызове из $.css  
    // computed = false при вызове из $.style  
  },  
  set: function( elem, value ) { /* установить свойство */ }  
}  
  
// если к свойству НЕ надо добавлять px  
$.cssNumber.borderRadius = true;  
  
// анимировать свойство  
$.fx.step.borderRadius = function( fx ) { ... }
```

10 плагин jquery-cssHooks

- Демо



Утечка памяти

10 Демо

What's wrong?

Sergey Miroshnyk
smiroshnick@gmail.com
0677223233



jQuery.data

10 Привязка данных к элементу

- [Документация](#)

10 Примеры применения

строка \Rightarrow

```
$('#article div').data('author', 'Vasya')  
...  
var author = $(elem).data('author')
```

функция \Rightarrow

```
function Widget(elem) {  
    var self = this  
  
    $(elem).data('getWidget', function(){ return self })  
}  
...  
var widget = $(elem).data('getWidget')()
```

объект \Rightarrow

```
$('#input').data('validator', new EmailValidator())  
...  
$(elem).data('validator').validate()
```



jQuery.data

Как?

⑩ `$(elem).data('author', 'Vasya')`

1. Создается свойство элемента

```
elem[jQuery.expando] = ++uuid
```

↑
спецключ

↑
уникальный номер

2. Данные - в jQuery.cache

```
jQuery.cache[uuid] = {  
  'author': 'Vasya'  
}
```

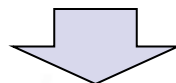


jQuery.data

Зачем?

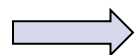
→ Все данные хранятся в `jQuery.cache[номер элемента]`

→ Элемент хранит ссылку только на `uuid` данных



⑩ Единая точка доступа к данным из элемента

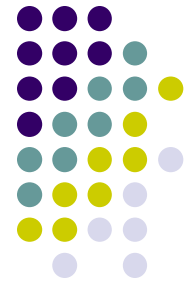
~~elem.foo = "..."~~
~~elem.bar = "..."~~



`$.cache[elem uuid]`

jQuery.data

Зачем?



- ⑩ Привязка ваших данных известна библиотеке
 - Можно использовать в библиотечных функциях
- ⑩ Уменьшает вероятность утечек памяти
 - Не создается ссылка DOM -> JS

Sergey Smiroshnik
smiroshnick@gmail.com
0677223233



jQuery.data

Дополнительные возможности

- ⑩ Событие setData
 - При установке \$.data
 - [Демо setData](#)
- ⑩ Поддержка невложенных пространств имен
 - [Демо setDataNS](#)
- ⑩ Это событие не всплывает



jQuery.data

Дополнительные возможности

- ⑩ Событие `getData`
 - При получении `$.data`
 - [Демо getData](#)
 - Можно изменить результат
- ⑩ Поддержка невложенных пространств имен
 - [Демо getDataNS](#)
- ⑩ Не всплывает



HTML5 data

- ⑩ <http://api.jquery.com/data/>
 - 1.4.3
 - HTML 5 data- attributes
 - [демо](#)
 - ТОЛЬКО ЧТЕНИЕ
 - обязателен валидный JSON
 - ~~{name: "John"}~~
 - ~~{'name': 'John'}~~
 - {"name": "John"} !



События

⑩ bind / one

- Несколько событий сразу

```
$elem.bind('mouseenter mouseleave',  
function() {  
    $(this).toggleClass('entered')  
})
```

- Объект с событиями

```
$elem.bind({  
    click: function() {  
        // обработка onclick  
    },  
    mouseenter: function() {  
        // обработка onmouseenter  
    }  
})
```

- click, keypress ... – сокращенный bind

⑩ unbind – не обязательно хранить обработчик

- `$elem.unbind('click')`



События

10 trigger

- свои и встроенные события
- всплывают

```
$elem.bind('eventName', function(event, p1, p2, ...) {  
    // ...  
})
```

...

```
$elem.trigger('eventName', [p1, p2, ...])
```

- Обязательны для правильной архитектуры UI
- Не нативные dispatchEvent / fireEvent
- [Демо trigger](#)



События

10 Привязка данных к обработчику

- Проблема

```
function addEvents(divs) {  
    for(var i=0; i<divs.length; i++) {  
        divs[i].innerHTML = i  
        divs[i].onclick = function(){ alert(i) }  
    }  
}
```

- Решение jQuery

```
function addEvents(divs) {  
    for(var i=0; i<divs.length; i++) {  
        divs[i].innerHTML = i  
  
        $(divs[i]).click({num: i}, function(event){  
            alert(event.data.num)  
        })  
    }  
}
```

- [Демо bindData](#)



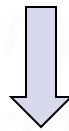
События

- ⑩ Кроссбраузерная обертка
 - *this*
 - *event.preventDefault() / stopPropagation()*
 - *Гарантирует порядок исполнения*
 - *event.stopImmediatePropagation()*
- ⑩ Передача результата по цепочке
 - [Демо](#)



События изнутри

```
$(elem).click(function(e) { alert("Hi") })  
      .click(function(e) { alert("I am clicked") })  
      .mouseover(function(e) { alert("Mouse over!") })
```





```
$(elem).data("events") = {  
  'click' : [  
    function(e) { alert("Hi!") },  
    function(e) { alert("I am clicked") }  
  ],  
  'mouseover' : [  
    function(e) { alert("Mouse over!") }  
  ]  
}
```



События и DOM

10 Влияние на манипуляции с DOM

- `elem.remove/html/empty` 
- в 1.4 detach
 - Можно очистить элемент позже
 - ...или не очищать вообще
- `elem.clone(true)` 



Namespaced Events

why?

⑩ Как назначить/удалить конкретный обработчик?

1. Можно сохранять указатель на функцию

```
var handler = function(){ ... }  
$('.class').bind('click', handler)  
...  
$('.class').unbind('click', handler)
```

Минусы: неудобно таскать за собой переменную

2. Можно удалять все обработчики

```
$('.class').bind('click', function(){ ... })  
$('.class').unbind('click')
```

Минусы: не интегрируется с другими обработчиками

`$(document).click ??`



Namespaced Events

10 Решение

- Плагин/участок кода объявляет события в namespace

```
$('.class').bind('click.plugin', function(){} )
```

- Обработчик сработает на «своих» событиях

```
$('.class').trigger('click.plugin')
```

- Обработчик сработает на обычных событиях

```
$('.class').trigger('click')
```

- Обработчик не сработает на «чужих» событиях

```
$('.class').trigger('click.other')
```

- Удалить «свои» обработчики

```
$('.class').unbind('click.plugin')
```

```
$('.class').unbind('.plugin')
```




Namespaced Events

10 Несколько namespace

namespace не вложенные

```
$('.class').bind('click.a.b', function(){})  
$('.class').trigger('click.a')  
$('.class').unbind('click.b')
```

jQuery перебирает
все перечисленные namespace

10 Вызов события без namespace

```
$elem.trigger("click!")
```

Не вызовет
обработчики click.*

10 Don't use it.



Namespaced Events

10 Почитать

- http://docs.jquery.com/Namespaced_Events
- <http://longgoldenears.blogspot.com/2010/02/jquery-namespaced-events-exclusive.html>
- <http://www.learningjquery.com/2007/09/namespace-your-events>

Sergey Smirnov
smirnick@gmail.com
0677223233



Делегирование событий

10 live/die

- Обработка всплывающих событий

- Удобная

- Быстрая

Обычный DOM элемент
На нем будет внутренний обработчик liveHandler

- Объявление

```
$('.clickable', container).live('click',  
  function(e) { /* обработчик события */ }  
);
```



Делегирование событий

⑩ live/die

```
$('.clickable', container).live('click',  
    function(e) { /* обработчик события */  
    })
```

- При клике:
 1. Событие всплывает до container
 2. Сработает встроенный обработчик liveHandler
 1. Найдёт `$(e.target).closest('.clickable', container)`
 2. Запустит обработчик события на найденном элементе
- Побочный эффект:
 - Вместо `event.stopPropagation()` → `return false`



Контекст

- 10 Второй аргумент `$(selector, context)`

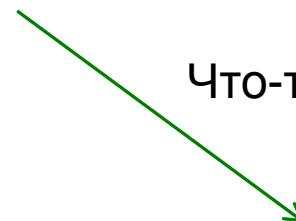
Узел DOM ?



Становится context



Что-то другое?



Rewrite запроса

```
$( 'a' , $( 'div' )[0] ). context
```

```
$( 'a' , $( 'div' )) → $( 'div a' )
```

```
$( 'a' , $( 'div' )[0] ). context
```



Делегирование событий

10 Синтаксис

- С контекстом

`$('#id')[0]`
↙
`$('.clickable', domElem).live('click', handler)`

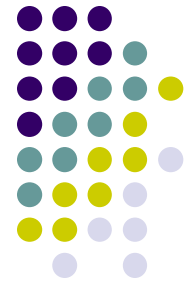
или

`$('.clickable', $(domElem)).live('click', handler)`

- Без контекста

`$('#id')`
`$('.clickable', '#id').live(...)`

С контекстом	Без контекста
liveHandler на <u>domElem</u>	liveHandler на <u>document</u>
Селектор <code>.clickable</code>	Селектор <code>#id .clickable</code>



Делегирование событий

⑩ Delegate / undelegate

- Удобный shortcut для live

```
$("#table").delegate("td", "hover", function(){  
    $(this).toggleClass("hover")  
})
```

- Повесит liveHandler именно на table

Semya Miroshnik
smiroshnick@gmail.com
0677223233



focusin / focusout

- ⑩ Обычные события `focus/blur` не всплывают
 - нельзя использовать event delegation

- ⑩ `focusin/focusout` – всплывающие `focus/blur`
 - IE: родные события
 - Другие браузеры :
 - `addEventListener(..., true)`
 - Отлов на фазе capturing



mouseenter / mouseleave

- ⑩ Кроссбраузерная поддержка
 - Фильтр `mouseover/mouseout`
- ⑩ Событие `hover`
 - `hover` ⇔ `"mouseenter mouseleave"`
 - `$elem.hover(f1, f2)`

smiroshnick@gmail.com

0677223233



Фреймворк для событий

⑩ special

- [multiclick](#)
- mouse gestures
- mobile devices
- [timedKeyPress](#)

• Альтернатива: [jQuery Throttle](#)

```
$("#search").keydown(  
    $.debounce( function(){ /* код */ }, 250, null, true)  
);
```



Анимация

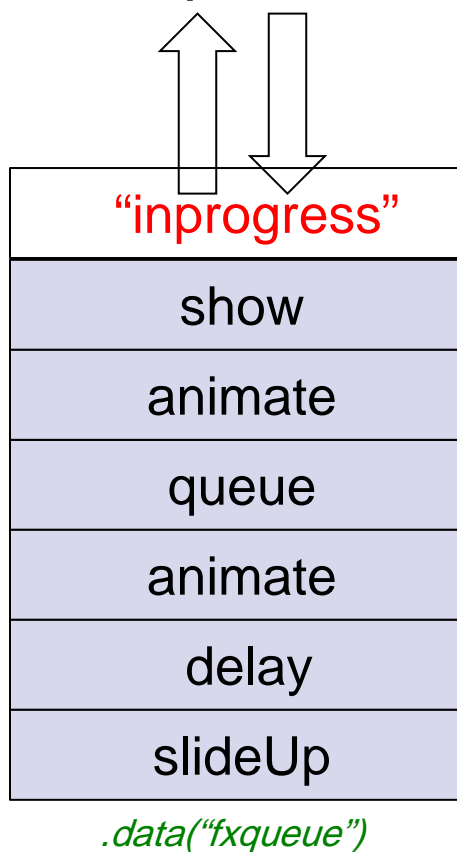
10 Анимация CSS-свойств

- Пример
- Управление процессом анимации
 - Easing
- Отображение изменений
 - step(now, fx)
 - `fx = { prop: "width", elem: ...}`
 - Демо
 - Color Transitions
- Остановка
 - Stop



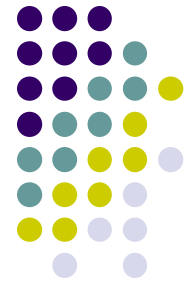
Очередь: анимация изнутри

10 Очередь для элемента



```
$("div")
  .show("slow")
  .animate({left: '+=200'}, 2000)
  .queue(function(next) {
    $(this).addClass("newcolor")
    next()
  })
  .animate({left: '-=200'}, 500)
  .delay(700)
  .slideUp()
```

Очередь fx: авто-dequeue [Демо](#)



Очередь: анимация изнутри

- ⑩ hide не знает об анимации

```
$( "div" )  
  .show( "slow" )  
  .animate( { left: '+=200' }, 2000 )  
  .queue( function ( next ) {  
    $( this ).addClass( "newcolor" )  
    next ()  
  })  
  .animate( { left: '-=200' }, 500 )  
  .delay( 700 )  
  .slideUp()  
  .hide() => .hide( 0 )
```



Параллельные анимации

- 10 Обычно вызовы на элементе встают в очередь:

```
$( "div" )  
  .animate({left: '+=200'}, 2000)  
  .animate({height: '100%'}, 3000)
```

- 10 Откатывается, т.к. начинает с left = 0

- 10 Для параллельного запуска:

```
$( "div" )  
  .animate({left: '+=200'}, {queue: false, duration: 2000})  
  .animate({opacity: '0'}, {queue: false, duration: 3000})
```

- 10 [Демо](#)

AJAX



10 Deferred \$.ajax

- Синхронные каллбэки
 - .done(cb)
 - .fail(cb)
 - .notify(cb)
 - .then(done, fail, notify)
- Асинхронные каллбэки
 - .pipe(done, fail, notify)



\$.Callbacks

Внутри Deferred/очередей

- ⑩ Интерфейс: `add/remove + fire(value)`
- ⑩ Виды (можно сочетать):
 - Простой
 - Memory
 - запускает новые обработчики после `fire`
 - Once
 - запускается только один раз
 - Unique
 - один обработчик можно добавить только один раз
 - stopOnFalse
 - `return false` останавливает работу
- ⑩ Почитать

AJAX



10 Глобальные настройки

- `$.ajaxSetup`
 - `timeout`: по умолчанию
 - `error`: обработчик ошибок по умолчанию
 - `preventCaching`: `true`

Sergey Smirnov
smiroshnick@gmail.com
0677223233



Префильтры, транспорты

10 Фикстуры

- Отделяют клиентскую разработку от серверной
- [Демо \(см. исходник\)](#)
- [Еще пример использования префильтров](#)

Sergey Smiroshnick
smiroshnick@gmail.com
0677223233

Своя сборка



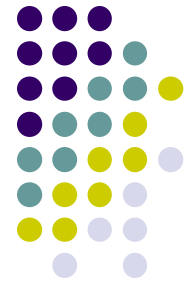
1. Git clone <http://github.com/jquery/jquery.git>
2. Node.js
3. make

- ⑩ Свои компоненты в обертке jQuery
 - сборка js “все в одном” для ленивых



Дополнительно

- ⑩ Служебные функции
 - [\\$\(something\).each](#) -> `$.each(arr/obj, function)`
 - `callback.call(value, i, value)`
 - [map](#)
 - [grep](#)
 - `callback(elems[i], i)`
 - ...



jQuery.proxy

10 Виды связывания для ООП

- Раннее связывание

- `var func = $.proxy(func, _this)`
- `var method = $.proxy(obj, 'method')`
 \Leftrightarrow `$.proxy(obj.method, obj)`

- Позднее связывание

```
function bindLate(fixThis, funcName)
  return function() {
    return fixThis[funcName].apply(fixThis, arguments)
  }
}
bindLate(obj, 'method')
```

[Демо proxy VS позднее связывание](#)



Плагины

⑩ Пример

- <http://javascript.ru/examples/jquery/plugin.html>

⑩ Почему оно работает?

```
jQuery = function( selector, context ) {  
    // The jQuery object is actually just the init constructor 'enhanced'  
    return new jQuery.fn.init( selector, context );  
}  
...  
init: function( selector, context ) {  
    // Make sure that a selection was provided  
    selector = selector || document;  
    // Handle $(DOMElement)  
    if ( selector.nodeType ) {  
        this[0] = selector;  
        this.length = 1;  
        this.context = selector;  
        return this;  
    }  
    ...  
  
jQuery.fn.init.prototype = jQuery.fn
```