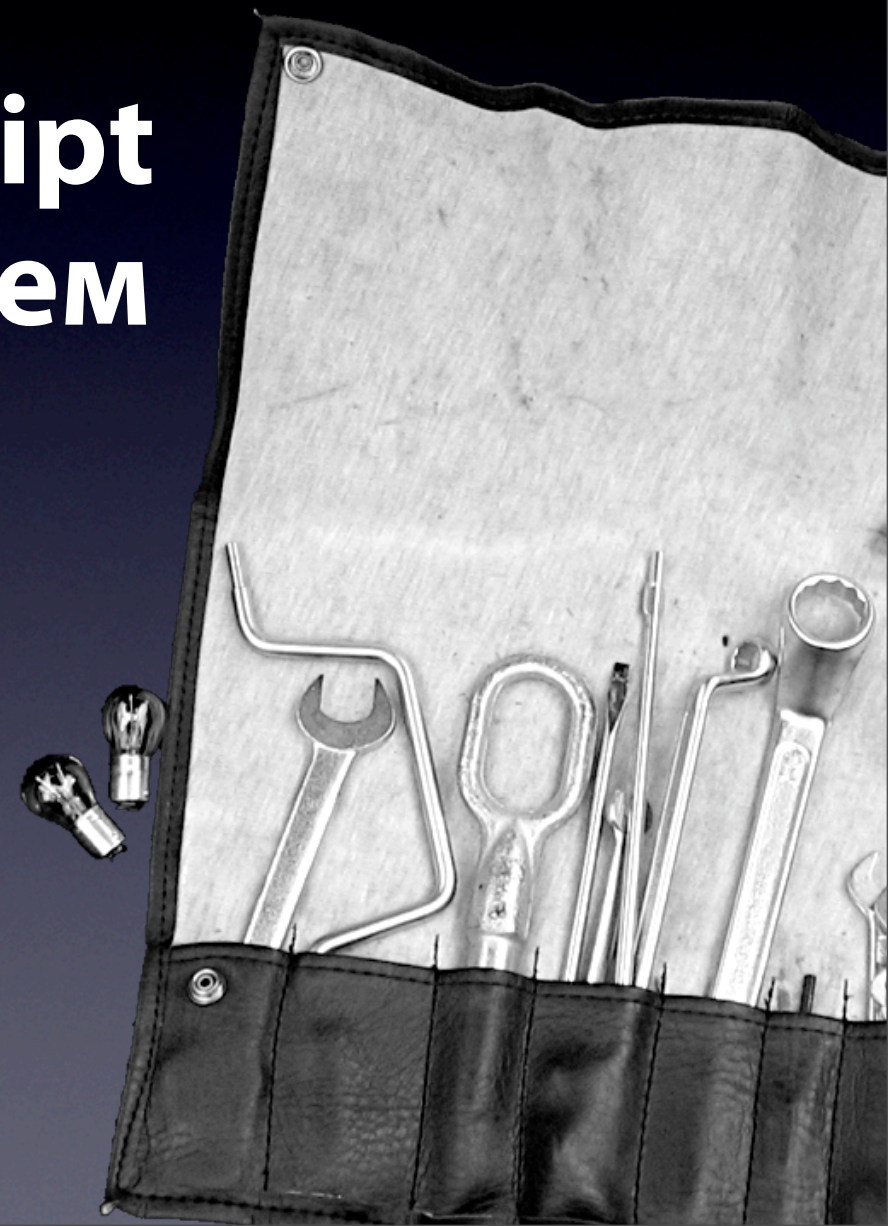
JavaScript для насыщенных пользовательских интерфейсов.

Обзор фреймворков, скорость разработки, особенности поддержки.

РОССИЙСКИЕ ИНТЕРНЕТ ТЕХНОЛОГИИ

Что такое JavaScript фреймворк и зачем он нужен?



Prototype, JQuery, YUI, Mochikit, Mootools, what_ever?...

Задача всех фреймворков — сделать жизнь разработчика легче.

JavaScript фреймворк =

отсутствующая в языке JavaScript
стандартная библиотека классов

```
var treeView = document.getElementById('treemenu');
if(!treeView) return;

var aMenus = treeView.getElementsByTagName('LI');

for (var i = 0; i < aMenus.length; i++) {
    var mclass = aMenus[i].className;
    if (mclass.indexOf('treenode') > -1) {
        var submenu = aMenus[i].childNodes;
        for (var j = 0; j < submenu.length; j++) {
            if (submenu[j].tagName == 'A') {
                // Ну давайте уже сделаем что-нибудь полезное!
            }
        }
    }
}
```

VS.

```
$$("ul#treemenu li.treenode a").each( function(item) {
    // делаем что-нибудь полезное
});
```

JavaScript фреймворк =

простая кросс-браузерная разработка

```
function createXMLHttpRequest() {
    if (typeof XMLHttpRequest != "undefined") {
        return new XMLHttpRequest();
    } else if (typeof ActiveXObject != "undefined") {
        return new ActiveXObject("Microsoft.XMLHTTP");
    } else {
        throw new Error("XMLHttpRequest not supported");
    }
}
```

VS.

```
new Ajax.Request();
```

Классификация JavaScript фреймворков

Первый уровень

Photo by andreika

Base2 Дина Эдвардса

 http://dean.edwards.name/weblog/2007/03/yet-another/

- Selectors API
- кросс-браузерность, заплатки для ошибок в браузерах
- минимальный набор «приятностей» — работа с CSS-классами

Второй уровень

Photo by philipmak

Prototype, JQuery, ...

Функциональность I уровня +

- «Синтаксический сахар» для работы с массивами, строками, enumerables и многим другим
- Простой кросс-браузерный Ajax
- Навигация по DOM
- ООП (классы, наследование и т. д.)

Третий уровень

Photo by trehala

Yahoo! UI Library, OpenRico, MochiKit, Script.aculo.us

Функциональность I и II уровней +

- Визуальные эффекты
- Библиотека готовых компонентов (виджетов)
- API для создания собственных компонентов

Есть и другие...

Четвертый уровень

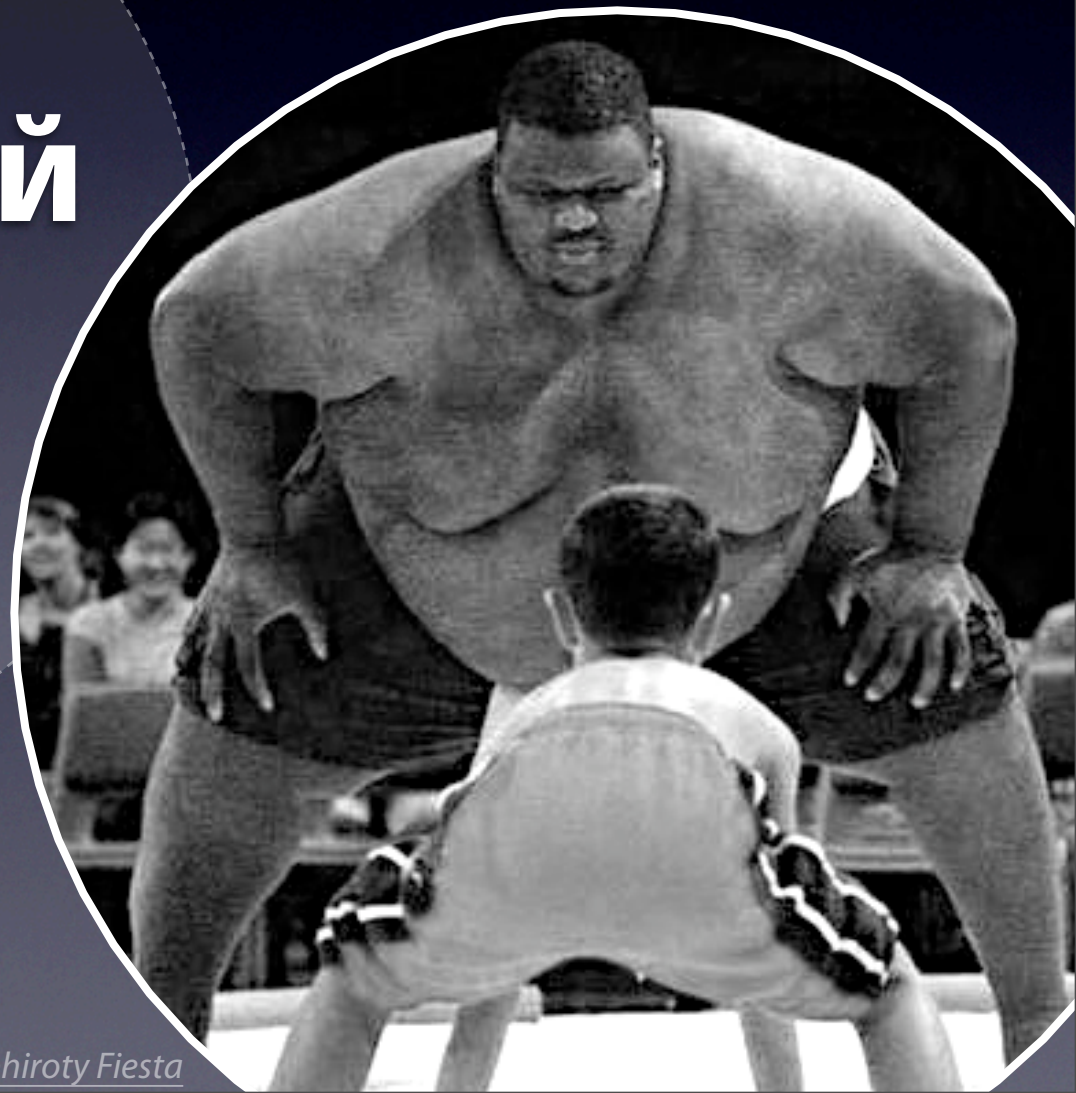
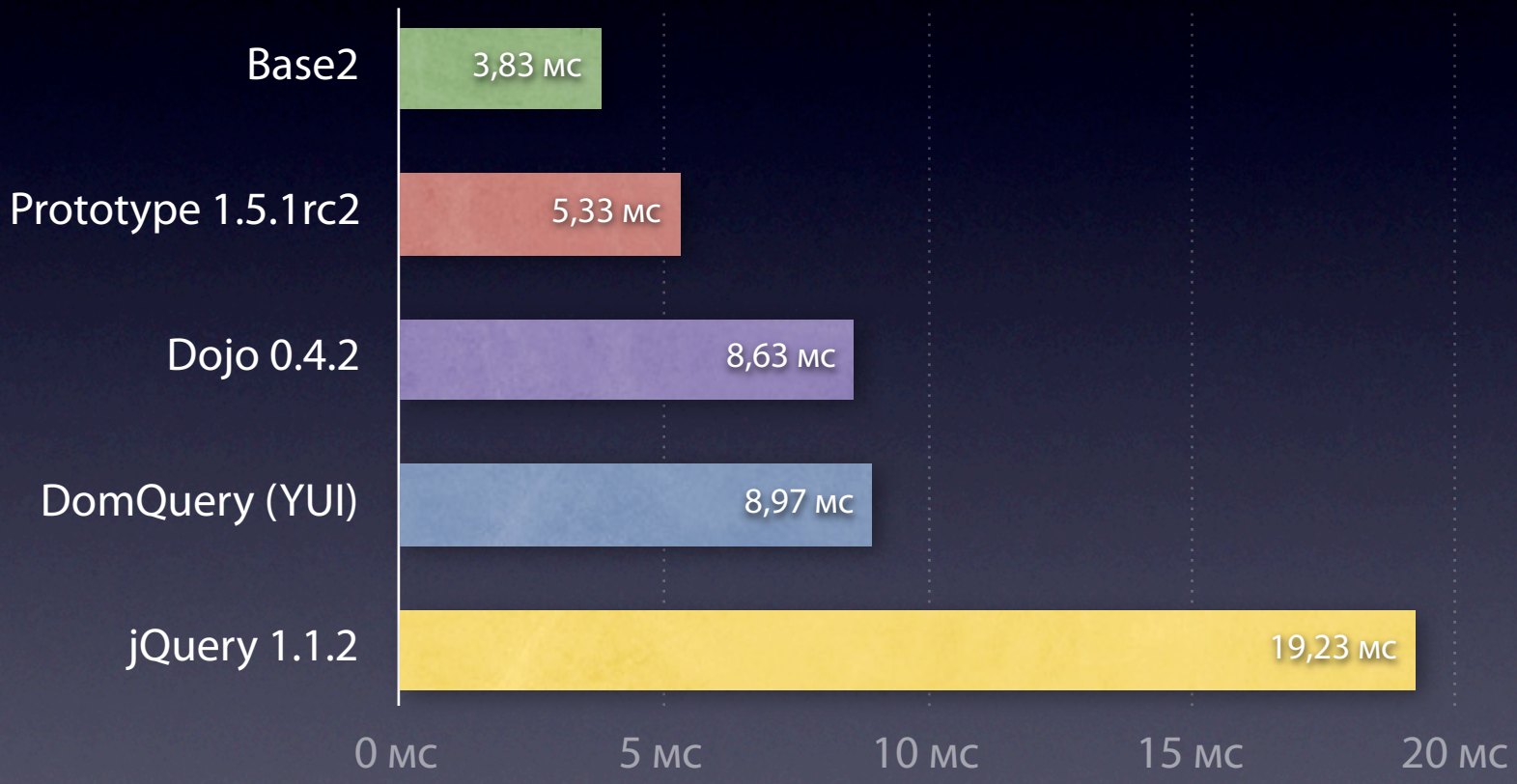


Photo by Sephiroty Fiesta

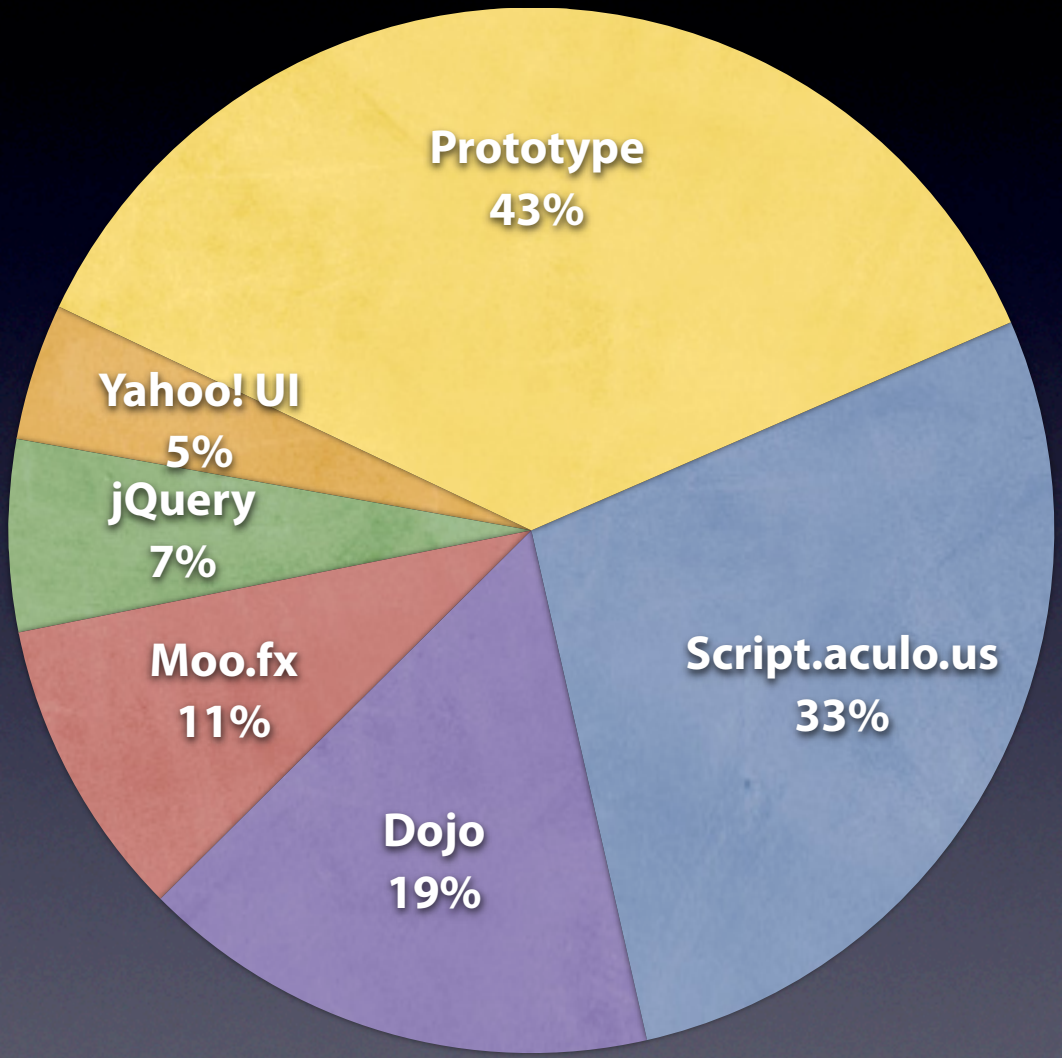
Сравнение основных JavaScript фреймворков

Скорость работы с селекторами



на основе данных *http://www.andrewdupont.net/test/double-dollar/*

Популярность фреймворков



на основе данных Ajaxian.com

http://ajaxian.com/archives/ajaxiancom-2006-survey-results

Поддержка основных функций и объем

	DOM	CSS	Ajax	JSON	FX	Widgets	Размер (КБ)
Base2	•	•	—	—	—	—	20
Prototype	•	•	•	•	—	—	94
jQuery	•	•	•	—	•	—	55
MooTools	•	•	•	•	•	•	36
Dojo	•	•	•	•	•	•	148
MochiKit	•	•	•	•	•	•	113
Yahoo! UI	•	•	•	•	•	•	684

Структура и функциональность типичного JavaScript фреймворка

1. Вспомогательные методы

Вспомогательные методы *— это набор методов и/или алиасов для сокращения записи и удобства вызова.*

$, $$, $A, $R, $F, $H и некоторые другие — возвращают ноды, массивы, хэши, элементы форм, и т. д. в виде, пригодном для дальнейшего использования с другими функциями Prototype.

2. Работа с CSS

- получение массива элементов по селектору,
- работа с атрибутом class.

```
Element.getElementsBySelector(element, "#treeview ul li
span") -> [HTMLElement...]
```

```
Element.setStyle(element, {color: blue, textDecoration:
underline})
```

3. Работа с событиями

- кросс-браузерное добавление/удаление обработчиков событий,
- поиск элемента, на котором произошло событие.

HTML

```
<form id="signinForm" method="post" action="/auth/signin">
…
</form>
```

JavaScript

```
Event.observe(window, 'load', function() {
  Event.observe('signinForm', 'submit', checkForm);
});
```

4. Ajax

- создание и выполнение Ajax-запросов,
- работа с колбэками,
- обновление данных в HTML контейнерах.

```
var url = encodeURIComponent('http://www.google.com/search?q=Prototype');

new Ajax.Request(url, {
  method: 'get',
  onSuccess: function(transport) {
    var notice = $('notice');
    if (transport.responseText.match(/href="http:\/\/prototypejs.org/))
      notice.update('Yeah! You are in the Top 10!').setStyle({ background:
'#dfd' });
  }
});
```

5. DOM

- навигация по DOM,
- управление видимостью элементов,
- работа с размерами и позиционированием,
- создание и добавление новых элементов.

```
$(element).down(1).next('li', 2).hide();

$('error-message', 'welcome-message').invoke('toggle');
```

6. Формы

- блокировка-разблокировка полей,
- установка фокуса,
- сериализация полей,
- auto-complete для элементов форм.

```
focusFirstElement($('myform'));

$('person-example').serialize(); ->
'username=leonya&age=24&hobbies=coding&hobbies=biking'
```

7. Строки

- различные преобразования (capitalize, dasherize, camelize),
- поиск подстрок,
- эскейпинг HTML,
- вырезание разметки и скриптов,
- парсинг строки запроса и многое другое...

```
'background-color'.camelize(); -> 'backgroundColor'

'<div class="article">This is an article</div>'.escapeHTML();
-> "&lt;div class="article"&gt;This is an article&lt;/div&gt;"

'section=blog&id=45'.toQueryParams();
 -> {section: 'blog', id: '45'}
```

8. Enumerables (коллекции)

Enumerables *— это суперкласс для работы с коллекциями однотипных данных (массивы, хэши, строки, другие объекты)*

- создание, заполнение и поиск по коллекциям,
- различные преобразование коллекций,
- создание производных коллекций из коллекций.

```
['Hitch', "Hiker's", 'Guide', 'To', 'The', 'Galaxy'].collect
(function(s) {
 return s.charAt(0).toUpperCase();
}).join('')
// -> 'HHGTTG'

$R(1, 10).findAll(function(n) { return 0 == n % 2; })
// -> [2, 4, 6, 8, 10]
```

А нужен ли вообще фреймворк?

- Я просто хочу попробовать на этом маленьком сайте
- Мне хочется показывать картинки в красивых попапах, поэтому мне нужен фреймворк
- Перепишу-ка я свои 5 функций на фреймворке...

Фреймворк НЕ нужен!

- Большой проект с большим количеством функциональности
- Функциональность будет постоянно расти
- В команде будет несколько человек

Фреймворк пригодится!

Критерии выбора фреймворка для конкретного проекта

Насколько планируемая функциональность реализуема во фреймворке?

- Начните с ТЗ
- Все фреймворки имеют плюсы и минусы
- Выбирайте наименьшее зло
- Думайте наперед

В проекте уже используется фреймворк?

- Скорее всего, выбор уже сделан до этого...
- Менять фреймворк — себе дороже
- Но если все-таки менять?
- Пишите Unit-тесты!
- Оцените затраты на портацию кода
- Пишите портируемый код

А документация есть?

- Плохо документированный фреймворк — практически бесполезен
- Достаточно ли времени на обучение?
- Документируйте свой код

Не забудьте про серверную часть!

- Серверная часть может не позволить использовать некоторые функции фреймворка (ASP.NET & Script.aculo.us)
- Некоторые серверные фреймворки могут быть связаны с JavaScript фреймворками (Ruby on Rails & Prototype)

Стоит ли разрабатывать свой фреймворк?

- Масса свободного времени?
- У вас есть лишние разработчики?
- Вам кажется, что существующие фреймворки не подходят?
- Знаете, как сделать лучше?

Не изобретайте!
Улучшайте!

- Займитесь самообразованием
- Найдите проекты или начните свой собственный
- Гуглите, Яндексите, ищите ;)
- Поддерживайте open-source

Спасибо за внимание!

Кстати, мы с удовольствием ответим на любые вопросы по данному докладу.

Леонид Хачатуров
leonidkhachaturov@gmail.com

Андрей Оконечников
andrej.okonetschnikow@gmail.com