

# 整体架构

CSS3 Transform & Transition

多点触控API

滑块Puzzle Demo

# 摆脱“这是一个网页”的设计思维

# UI框架

## jQuery Mobile

## Sencha Touch

## jQTouch



...

# 用MVC理念开发javascript前端应用

基于服务器端的MVC框架  
(Ruby on Rails, Django, CodeIgniter, Yii...)

vs.

基于客户端的框架  
(Backbone.js, Ember.js, Spine.js...)



后台只是API  
界面的生成转移到终端

根据应用的重量级  
选择适合的架构

# Backbone.js

router  
models  
views  
templates

# 前端模版语言

John Resig's Mirco Template  
ejs - embedded javascript

`{{ Mustache }}`  
Handlebars

Jade  
Haml

# 模块化

YUI3  
Require.js  
Sea.js

自定义编译脚本

# Require.js简单例子

```
//定义一个模块 one.js
define(
    ["libs/underscore", "libs/backbone"],
    function (_, Backbone) {
        ...
        return {
            ...
            //此模块的API
        };
    }
);

//在主文件里调用此模块
require(
    ["one"],
    function (one) {
        ...
    }
);
```

或者你也可以什么都不用。

# CSS3 Transform & Transition

# CSS3 Transform & Transition

transform: *function*

rotate(ndeg)  
rotateX(ndeg)  
rotateY(ndeg)  
rotateZ(ndeg)

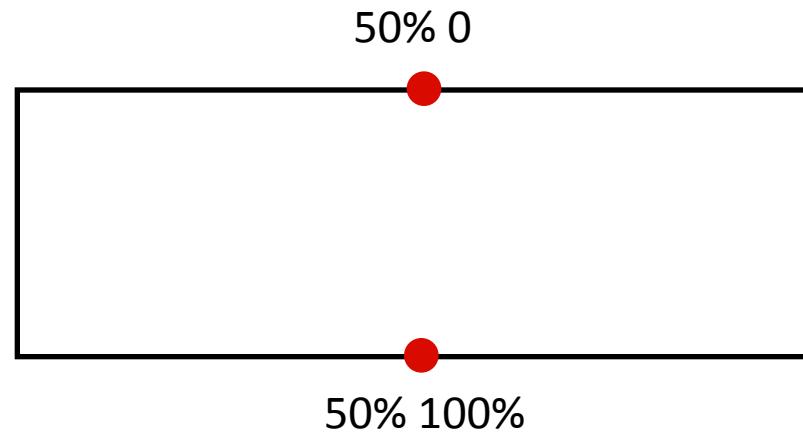
translate(xpx, ypx)  
translateX(xpx)  
translateY(ypx)  
translate3d(xpx, ypx, zpx)

scale(n)  
scaleX(n)  
scaleY(n)  
skew(n)  
skewX(n)  
skewY(n)

matrix(a, c, b, d, tx, ty)

# CSS3 Transform & Transition

## transform-origin



# CSS3 Transform & Transition

transition: *property duration timing-function delay*

例子：

```
transition: all 1s linear; /* 用all属性的时候要谨慎 */
```

```
transition: color .3s ease-in-out;
```

```
transition: transform 0s cubic-bezier(0, 0, 0.2, 1);
```

# CSS3 Transform & Transition

## transition easing function

linear

ease

ease-in

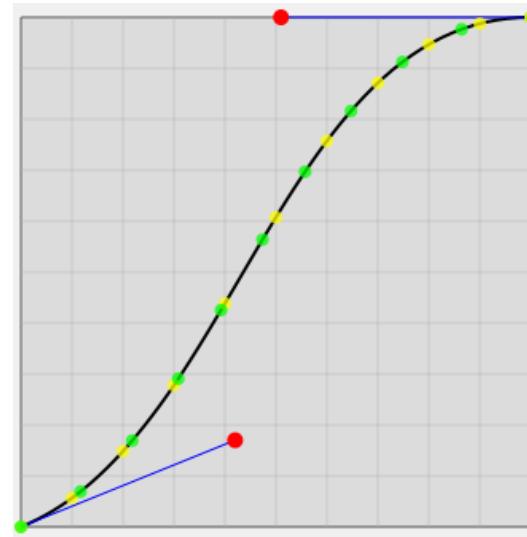
ease-out

ease-in-out

cubic-bezier(x1, y1, x2, y2)

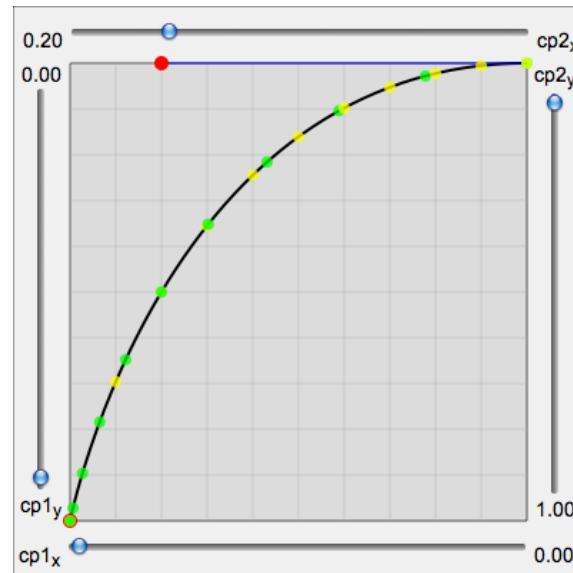
# CSS3 Transform & Transition

cubic-bezier(x1, y1, x2, y2)



# CSS3 Transform & Transition

```
.drag {  
transition: transform 0s cubic-bezier(0, 0, 0.2, 1);  
}
```



# CSS3 Transform & Transition

## 强制硬件加速

`translate3d(x, y, z)`

或者任何只适用于3D的属性，比如

- webkit-perspective
- webkit-backface-visibility

# HTML5 MultiTouch API

# HTML5 MultiTouch API

3种基本事件

touchstart  
touchmove  
touchend



mousedown  
mousemove  
mouseup

\* click事件是通用的

# HTML5 MultiTouch API

## 事件对象

```
$(document.body).on('touchstart', function (e) {  
    console.log(e.touches[0].pageX);  
});
```

e.touches => 一个包含touch对象的数组

e.touches.length => 当前触摸点的数量

e.touches[0] => 第一个触摸点

# HTML5 MultiTouch API

## 事件对象

e.touches  
e.targetTouches  
e.changedTouches

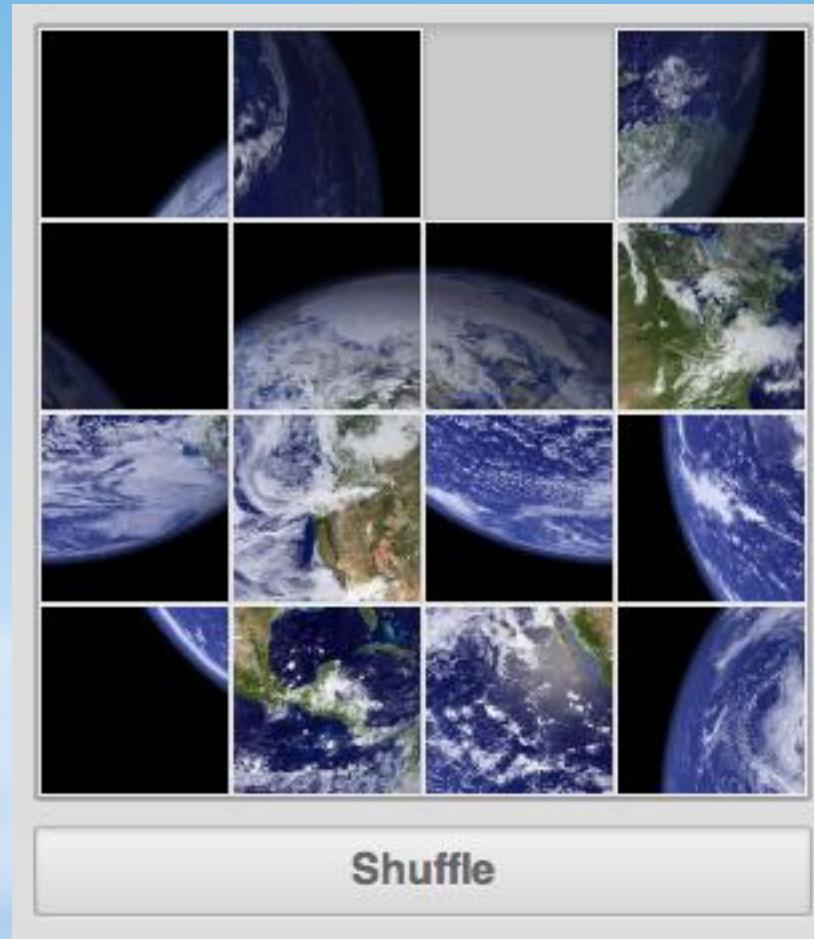
更多兼容性细节

<http://www.html5rocks.com/en/mobile/touch/>

# Puzzle Demo

<http://puzzle.youuxi.com>

<https://github.com/yyx990803/Puzzle-Demo>



# Zepto.js

作者: Thomas Fuchs  
(script.aculo.us作者, Prototype.js团队核心成员)

针对移动端轻量化的jQuery

# LESS

给CSS加上嵌套，变量，运算和Mixin  
让你写出更模块化，更易维护的CSS

支撑Twitter Bootstrap的核心技术

# Puzzle Demo

```
$(document).ready(function () {  
    PUZZLE.init(document.body);  
});
```

# Puzzle Demo

```
var PUZZLE = (function ($, window, undefined) {  
    //在此闭包中干活  
    ...  
  
    //对外暴露API  
    return {  
        init: function (container) {  
            //初始化代码  
        }  
    };  
})(Zepto, window);
```

# Puzzle Demo

//要用到的一些变量

```
var animating = false,  
    tileSize = 72, animationSpeed = 150,  
    setTimeout = window.setTimeout;
```

# Puzzle Demo

```
//一个简单的容纳HTML模版的对象
//一个模版其实就是一个返回HTML字符串的函数var templates = {
  board: ...,
  tile: function (id) {
    return '...';},
  shuffleButton: ...
};
```

# Puzzle Demo

```
//Utility function
//返回一个可以用在$.css()里面的CSS对象var cssTransform = function
(x, y) {
    var translation = 'translate(' + x + 'px,' + y + 'px)';
    return {
        '-webkit-transform': translation,
        '-moz-transform': translation,
        '-o-transform': translation,
        '-ms-transform': translation,
        'transform': translation
    };
}
```

# Puzzle Demo

```
//滑块对象var Tile = function (id, board) {  
    this.id = id;  
    this.board = board;  
    this.render();};  
  
Tile.prototype = {  
    render: function () { ... },  
    update: function (position) { ... },  
    getMovableDirection: function () { ... },  
    getNeighbor: function (direction) { ... }  
};
```

# Puzzle Demo

```
//滑块对象的render()方法
render: function () {
  this.el = $(templates.tile(this.id));
}
```

# Puzzle Demo

```
//滑块对象的update()方法 update: function (position) {this.position = position;  
this.x = (position % 4) * tileSize;  
this.y = ~~(position / 4) * tileSize;  
this.el  
    .data('position', position)  
.css(cssTransform(this.x, this.y));}
```

# Puzzle Demo

```
//底板对象var Board = function () {  
    this.buildTiles();  
    this.shuffle();  
    this.render();  
    this.update();};  
  
Board.prototype = {  
    buildTiles: ...,  
    shuffle: ...,  
    render: ...,  
    initEvents: ...,  
    update: ...,  
    ...  
};
```

# Puzzle Demo

```
//在底板对象上委托事件处理
initEvents: function () {
    var board = this,
        touch = {};
    board.el.delegate('.tile', 'touchstart mousedown', function (e) {
        ...
    })
    .delegate('.tile', 'touchmove mousemove', function (e) {
        ...
    })
    .delegate('.tile', 'touchend mouseup', function (e) {
        ...
    });
}
```

# 欢迎加入百度开发者中



<http://developer.baidu.com>